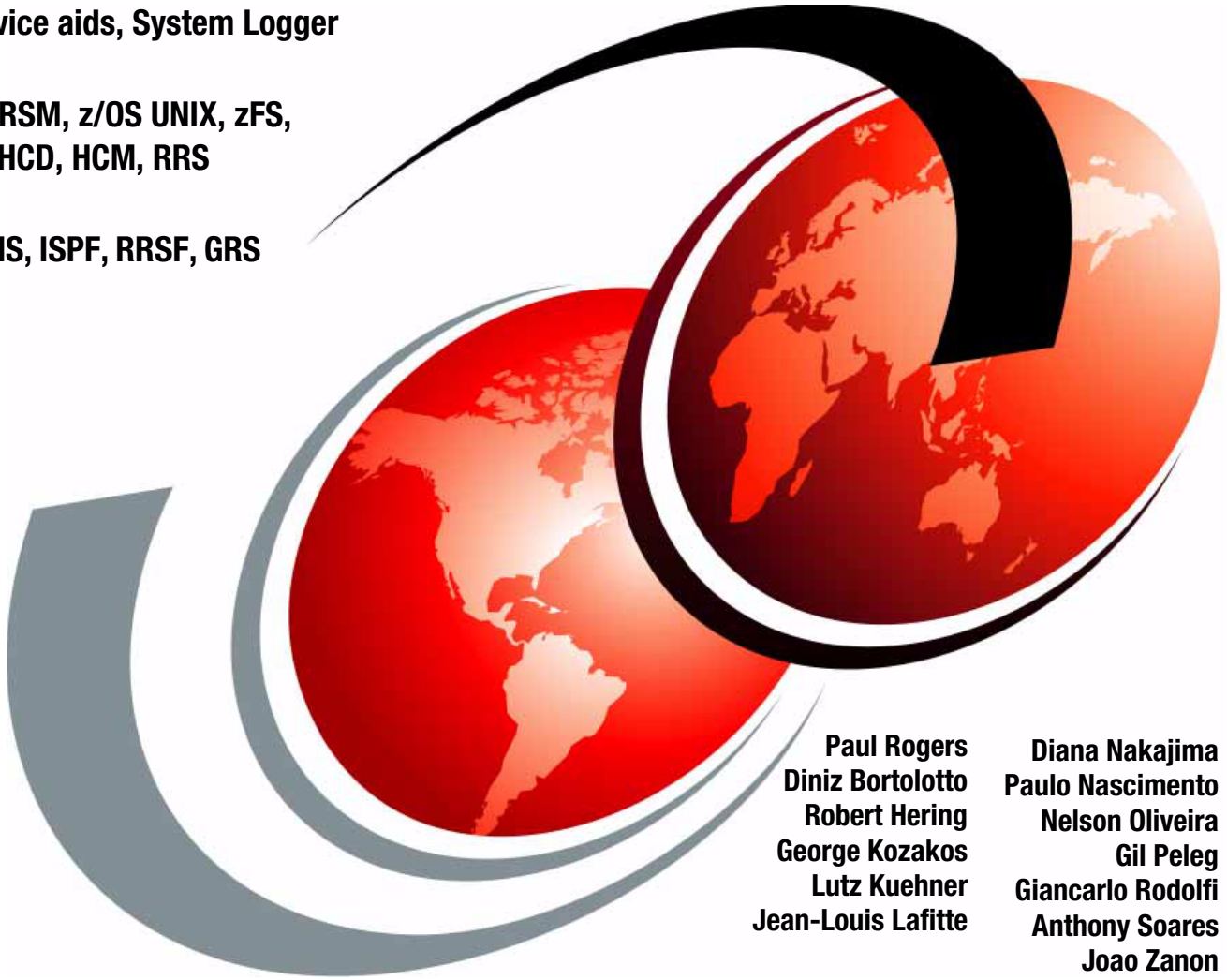
IBM

z/OS Version 1 Release 13 Implementation

JES2, JES3, SDSF, RMF, z/OSMF, EAV, BCPii, Service aids, System Logger

Consoles, RSM, z/OS UNIX, zFS, CIM, PFA, HCD, HCM, RRS

XCF, DFSMS, ISPF, RRSF, GRS



Paul Rogers
Diniz Bortolotto
Robert Hering
George Kozakos
Lutz Kuehner
Jean-Louis Lafitte
Diana Nakajima
Paulo Nascimento
Nelson Oliveira
Gil Peleg
Giancarlo Rodolfi
Anthony Soares
Joao Zanon

Redbooks

ibm.com/redbooks

International Technical Support Organization

z/OS Version 1 Release 13 Implementation

March 2012

SG24-7946-00

Note: Before using this information and the product it supports, read the information in “Notices” on page xvii.

First Edition (March 2012)

This edition applies to Version 1 Release 13 modification 0 of IBM z/OS (product number 5694-A01) and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Notices xvii
Trademarks xviii

Preface xix
The team who wrote this book xix
Now you can become a published author, too! xxi
Comments welcome. xxi
Stay connected to IBM Redbooks xxi

Chapter 1. IBM z/OS Version 1 Release 13 overview 1
1.1 Base z/OS enhancements 2
1.1.1 System logger 4
1.1.2 Predictive Failure Analysis 4
1.1.3 GRS enhancements 5
1.1.4 Health checks 5
1.1.5 XES enhancements 6
1.1.6 Capacity Provisioning 6
1.1.7 Resource Management Facility enhancements 7
1.1.8 BCPii enhancements 7
1.1.9 Common Information Model enhancements 8
1.1.10 ISPF enhancements 8
1.1.11 RRS enhancements 9
1.1.12 XCF enhancements 9
1.1.13 Language Environment enhancements 10
1.1.14 Hardware Configuration Definition - HCM enhancements 10
1.2 JES2 enhancements 11
1.3 System Display and Search Facility enhancements 11
1.4 JES3 enhancements 12
1.5 zFS and z/OS UNIX enhancements 12
1.6 DFSMS - DFSMSdfp enhancements 14
1.6.1 DFSMS EAV volumes 18
1.6.2 DFSMShsm enhancements 18
1.6.3 DFSMSdss enhancements 19
1.6.4 DFSMSrmm enhancements 20
1.6.5 DFSMSoam enhancements 21
1.7 z/OSMF 21
1.8 z/OS Communications Server 23
1.9 Infoprint Server 24
1.10 Security enhancements. 25

Chapter 2. z/OS V1R13 installation 31
2.1 z/OS V1R13 installation 32
2.2 Planning for z/OS V1R13 installation 32
2.3 Installing z/OS V1R13 33
2.3.1 Installing with ServerPac. 33
2.3.2 Installing with CBPDO. 34
2.4 Driving system requirements for z/OS V1R13. 34
2.4.1 Driving system components requirements 34
2.4.2 Customized Offerings Driver V3 (5751-COD) 37

2.5 Elements changed in z/OS V1R13 38
2.5.1 JES2, JES3, and SDSF 38
2.5.2 Cryptographic Services 39
2.6 Preventative service planning (PSP) buckets 39
2.7 SMP/E V3R6 overview 40
2.7.1 GIMADR service routine for z/OSMF 40
2.7.2 Multitasking using GIMDDALC SYSPRINT allocation 41
2.7.3 Adding SAF checks to SMP/E processing 41
2.7.4 Retention of HOLDDATA 42
2.7.5 Additional SMP/E enhancements in z/OS V1R13 42
2.8 Coexistence, migration, and fallback considerations 47
2.8.1 Using SMP/E for coexistence 47
2.9 Migration actions for z/OS V1R11 and z/OS V1R12 49
2.9.1 Updating CONSOLxx PARMLIB members 50
2.9.2 Stand-alone dump data set allocations 50
2.9.3 Exploiting WARNUND as a new IEASYSxx parameter 51
2.9.4 Adjusting CON= system parameter 51
2.9.5 Making sysplex=filesys available on all R11 and R12 systems 53
2.9.6 New DASD space requirements for zFS 54
2.9.7 Copying cloned file systems to a compatibility mode aggregate 56
2.9.8 zFS multifile system aggregates to zFS compatibility mode aggregates 56
2.9.9 Making sysplex=filesys available on all R11 and R12 systems 57
2.9.10 zFS verify virtual storage usage 58
2.9.11 Deleting NOIMBED and NOREPLICAT LISTCAT command attributes 60
2.9.12 Updating procedures using IEBDSCPY alias name to access IEBCOPY 60
2.10 BCP migration actions after the first IPL of z/OS V1R13 61
2.10.1 Disabling DFS client initialization 61
2.11 Deleted elements in z/OS V1R13 61

Chapter 3. Managing volume backups with fast replication 63
3.1 Automatic refresh UCB function 64
3.2 Refreshing the UCB on remote systems 64
3.3 Enabling the REFUCB function 64
3.3.1 REFUCB function examples 67
3.3.2 Using the ICKDSF REFORMAT NEWVTOC command 68

Chapter 4. XCF enhancements 71
4.1 Large fixed CTRACE buffer size 72
4.2 CF structure enhancements 72
4.2.1 Support for greater than 1024 CF structures 72
4.2.2 Support for greater than 32 CF structure connectors 73
4.3 DISPLAY XCF,SYSPLEX enhancement 77
4.4 CF not suitable for allocation preference 79
4.4.1 Examples of messages with new inserts 81
4.5 Controlling CF structure ALTER processing - OA34579 84

Chapter 5. Console service enhancements 89
5.1 CMDS command extensions 90
5.1.1 Abnormally ending a command currently executing in z/OS V1R12 90
5.1.2 Using the new FORCE option with CMDS command in z/OS V1R13 91
5.1.3 CMDS command example 93
5.1.4 Controlling the use of the FORCE operand 95
5.2 Distributed mode as default console operating mode 95
5.3 Message Flood Automation constraint 96

5.4 SETMF command 100

Chapter 6. DFSMSdfp, DFSMSoam, and DFSMShsm 103
6.1 New PDSE diagnostic commands 105
6.1.1 PDSE diagnostic aids 105
6.2 IEBPDSE utility to validate a PDSE data set 106
6.2.1 Installation considerations. 108
6.3 IEBCOPY enhancements 108
6.3.1 IEBCOPY performance improvement 108
6.3.2 IEBCOPY APF removal considerations 108
6.3.3 Installation considerations. 109
6.4 New catalog parmlib member support 109
6.4.1 IGGCATxx parmlib member 109
6.4.2 SYS1.PARMLIB(IEASYSxx) member changes 112
6.4.3 Installation considerations. 113
6.4.4 Changing the maximum number of catalogs and tasks in CAS 114
6.4.5 MODIFY CATALOG command 114
6.5 DASDM dynamic exits 115
6.5.1 Installation considerations. 116
6.6 New option for the LISTCAT LVL command 116
6.6.1 Installation considerations. 117
6.7 OPEN/CLOSE/EOV enhancements 117
6.7.1 New DEVSUPxx parmlib member option 118
6.8 New OCE diagnostic data 119
6.9 QSAM MULTSDN calculation with concatenation. 119
6.10 Free tape volumes at end of volume 120
6.10.1 Installation considerations. 121
6.11 XTIOT support for subsystem DCBs and ACBs 122
6.12 OAM file system support 122
6.12.1 Installation considerations. 123
6.13 OAM usability and reliability enhancements 125
6.13.1 Wildcard usage with the F OAM,S,STORGRP command. 126
6.13.2 Extend object expiration beyond 27 years 126
6.13.3 SGMAXTAPESTORETASKS and SGMAXTAPERETRIEVETASKS 127
6.13.4 Improved media migration. 127
6.13.5 Enhanced OAM messages for specific DB2 errors. 128
6.13.6 SMF counter scalability 128
6.13.7 CTICBR00 parmlib member 128
6.13.8 RECYCLE candidates display enhancement 129
6.13.9 Changed operator commands 129
6.14 CDS backup improvements for DFSMShsm 129
6.14.1 Non-intrusive journal backup with z/OS V1R13 130
6.14.2 Installation considerations for z/OS V1R13. 130
6.15 ARCCAT release. 132
6.16 Space management performance improvements 133
6.16.1 Replace interval migration with on-demand migration 133
6.16.2 Auto Migrate support. 135
6.16.3 Installation considerations. 136
6.17 SMS configuration change refresh 136
6.18 Small data set packing performance (SDSP) improvements 137
6.18.1 Moving the point of invocation for the ARCMMEXT 138
6.19 DFSMShsm serviceability and usability enhancements 138
6.20 PDA trace during DFSMShsm startup 139

6.21 Fast replication ARC1809I messages . . . 139
6.22 RELEASE RECALL(DASD) command . . . 140
6.23 QUERY CRQ . . . 140
6.24 ONLYIF enhancements . . . 141
6.25 ARC0570I patch . . . 142
6.26 AUDIT COPYPOOL . . . 143
6.27 Change messages from I to E . . . 144
6.28 Change FR(DSR) default to NONE . . . 144

Chapter 7. ISPF enhancements . . . 147
7.1 z/OS UNIX Directory List enhancements . . . 148
7.1.1 ISPF support for ACLs in z/O V1R13 . . . 148
7.2 New data set allocation command for ISPF Data Set List . . . 151
7.3 Display data set create jobname and stepname . . . 154
7.4 Editor support for line command macros . . . 157
7.4.1 Installation considerations . . . 161
7.5 INFO command to display PDS member extended statistics . . . 162

Chapter 8. DFSMSrmm enhancements . . . 165
8.1 Retention methods for tapes . . . 166
8.1.1 Specifying a retention method . . . 166
8.1.2 Retaining source data sets managed by the EXPDT retention method . . . 168
8.1.3 Subcommands for RETENTIONMETHOD parameters . . . 169
8.1.4 Benefiting from the EXPDT retention method . . . 170
8.1.5 Installation considerations and coexistence . . . 172
8.1.6 EDG_EXIT100 retention method support . . . 174
8.2 SEARCHDATASET extensions . . . 174
8.3 TVEXTPURGE extra days . . . 179
8.4 Expiry by data set information . . . 182
8.5 Exclude data sets from VRSEL processing . . . 184
8.5.1 Retaining source data sets managed by the VRSEL retention method . . . 186
8.5.2 DFSMSrmm dynamic exit services . . . 189
8.6 Data set attribute COPYFROM function . . . 193
8.6.1 Using the CHANGEDATASET COPYFROM subcommand . . . 193
8.7 Support RETPD (93000) . . . 196
8.8 Selective volume movement . . . 197
8.9 Last change details . . . 198
8.10 VRS last reference date . . . 200
8.11 Display navigation enhancements . . . 203

Chapter 9. Establishing RACF security for RRSF TCP/IP connections . . . 207
9.1 RACF security for RRSF TCP/IP connections . . . 208
9.2 RRSF concepts . . . 208
9.2.1 RRSF functions . . . 209
9.3 Setting up RRSF with TCP/IP . . . 210
9.3.1 Changing the protocol for a connection . . . 210
9.3.2 Defining the RRSFDATA class with RACF . . . 211
9.3.3 Creating an IRROPTxx parmlib member . . . 211
9.3.4 Setting up security for TCP/IP communication . . . 212
9.3.5 Setting up the TCPIP environment . . . 212

Chapter 10. GRS enhancements . . . 217
10.1 Latch Identity for D GRS,C . . . 218
10.1.1 Display GRS command syntax . . . 218

10.1.2 Examples of D GRS output 218
10.2 New ISGQUERY request types 221
10.2.1 ISGQUERY REQINFO=LATCHECA 222
10.2.2 ISGQUERY REQINFO=USAGESTATS 223
10.3 New authorized QNAMEs 224
10.3.1 Managing authorized qname lists 225
10.4 Diagnostics for GQSCAN 227
10.5 IPCS enhancements 228
10.5.1 Examples of GRSTRACE reports 228

Chapter 11. BCP supervisor, contents supervisor, and RSM updates 231
11.1 SSRB above 2 GB 232
11.2 SRB enclave join and leave services 232
11.2.1 z/OS V1R13 enhancements - SRB enclave join and leave services 233
11.2.2 Installation considerations and software requirements 234
11.3 Join and leave services for subtasks 234
11.4 RMODE 64 Phase 1 236
11.4.1 Infrastructure changes to support 16 bytes PSW 236
11.4.2 Changes to system trace records 239
11.5 HiperDispatch=yes for the z196 241
11.6 WARNUND system parameter 242
11.7 RMTR control with local lock option 242
11.8 Hardware Instrumentation Services miscellaneous update 243
11.9 LLA use old suffix on restart 243
11.10 CSVDLPAU message for RC=8 244
11.11 Additional EXAA information 244
11.12 64-bit subspace support 244

Chapter 12. Improved channel recovery 247
12.1 Improved channel recovery 248
12.2 Enabling the IOS channel recovery function 249
12.2.1 IECIOSxx parmlib member specifications 250
12.2.2 IEASYSxx parmlib member 252
12.2.3 SETIOS RECOVERY command 252
12.3 Displaying the current IOS recovery options 254
12.4 New messages about path-related errors 254
12.5 Displaying the reason why the path is offline 256
12.6 Restoring after the path-related error has been corrected 257

Chapter 13. Service aids enhancements 259
13.1 z/OS Problem Documentation Upload Utility 260
13.1.1 Work data sets and work data sets guidelines 260
13.1.2 PDDU compression and encryption 261
13.1.3 JCL statements for PDUU 262
13.1.4 JCL examples 263
13.1.5 Using an NRTRC data set 265
13.2 IPCS SYSTRACE subcommand enhancement 266
13.3 IPCS subcommand DOCPU 268
13.3.1 Data selection parameters for DOCPU 268

Chapter 14. System logger - SMF 271
14.1 Monitoring for log stream offload data set allocations 272
14.1.1 IXGCNFxx parmlib member statement for logger monitoring 272
14.2 RELATIVEDATE parameter update 275

14.2.1 RELATIVEDATE parameter with z/OS V1R13 275
14.2.2 Keeping the log stream archive process similar to SMF MANx 276
14.2.3 DUMP, ARCHIVE, and DELETE options with the LSNAME parameter 277
14.3 Stop reading before end of log stream support 279
14.3.1 SMARTENDPOINT processing. 279
14.3.2 SMARTEPOVER processing 280

Chapter 15. z/OS UNIX System Services 283
15.1 zFS installation changes for z/OS V1R13 284
15.2 zFS sysplex support with releases prior to z/OS V1R13. 284
15.2.1 File system ownership. 285
15.2.2 Sysplex-unaware read-write file system 285
15.2.3 z/OS V1R11 sysplex-aware read-write file system 286
15.2.4 zFS sysplex-aware on a file system basis 288
15.3 z/OS V1R13 Direct I/O sysplex-aware read-write file system 289
15.4 zFS internal restart 290
15.4.1 zFS internal restart processing with z/OS V1R13. 291
15.4.2 Forcing a zFS internal restart 292
15.4.3 Results of the zFS restart 295
15.5 zFS automatic re-enablement of disabled aggregates 296
15.5.1 Automatic re-enablement of disabled aggregates considerations 297
15.6 Migration considerations 298
15.6.1 zFS health check for sysplex=filesys 299
15.6.2 DASD space with z/OS V1R13 299
15.6.3 Changes in zFS installation 301
15.6.4 Changes in IOEPRMxx configuration options 301
15.6.5 zFS ownership versus z/OS UNIX ownership of file systems 302
15.7 zFS statement of direction 303
15.8 Performance comparisons with sysplex-sharing 304
15.8.1 Setup and description of the test environments 304
15.8.2 Test results 307
15.9 Add symbolic links to shared root filesystem 308
15.9.1 Using the new support 308
15.9.2 CBPDO considerations 309
15.9.3 Migration considerations 310
15.9.4 Installation considerations. 311
15.10 Lost message detection 312
15.11 Provide script utility to log session 312
15.11.1 Using the script command. 312
15.11.2 Usage considerations 313
15.12 Enhancement for the D OMVS,W command 314
15.12.1 Showing file latch activity 314
15.12.2 Output filtering. 316
15.12.3 Using an assembler program 317
15.13 RACF support for z/OS UNIX user mounts 317
15.13.1 Mount security requirements. 317
15.13.2 Mount and unmount processing 319
15.13.3 Setup for the new support 319
15.13.4 Authorizing a nonprivileged user mount 320
15.13.5 Additional mount and unmount enhancements. 324
15.13.6 z/OS UNIX System Services file system access. 327
15.14 IPv4 IP_PKTINFO support for AF_INET UDP sockets 328
15.14.1 Using the new interface 328

15.14.2 IPv4 IP_PKTINFO external interface 329
15.15 Option on vi to edit ASCII files 329
15.15.1 Using the enhancements 330
15.15.2 Example of using the new support 331

Chapter 16. z/OS UNIX-related applications 333
16.1 Windows 7 support for DFS/SMB 334
16.2 Ported Tools sudo utility 334
16.2.1 Benefits of using sudo 334
16.2.2 More information about sudo 335
16.2.3 Using sudo 335
16.2.4 Examples of using sudo 336
16.2.5 Comparing several z/OS UNIX authority interfaces 338
16.2.6 Installation considerations regarding security 338
16.2.7 Additional sudo command examples 340
16.2.8 Migration and coexistence considerations 340
16.3 Hookless debug support for dbx 342
16.3.1 Using the hookless debug enhancement 343
16.3.2 List instructions from the program 343

Chapter 17. Resource Recovery Services 345
17.1 Automated shutdown 346

Chapter 18. IBM z/OS Management Facility 349
18.1 IBM z/OS Management Facility (z/OSMF) 350
18.1.1 Introduction to the new faces for z/OS 350
18.1.2 z/OS ease-of-use enhancements 350
18.1.3 The “new face” of z/OS 352
18.2 Introduction to z/OSMF V1R13 354
18.3 z/OSMF administrator with z/OS V1R13 356
18.3.1 SAF Authorization Mode 357
18.3.2 z/OSMF administrator user ID and group ID 360
18.4 z/OSMF system management tasks with z/OS V1R12 361
18.4.1 z/OSMF new system management tasks with z/OS V1R13 362
18.5 Installation considerations for z/OSMF V1R13 363
18.5.1 Customization requirements 363
18.5.2 Active log medium 366
18.6 Migrating to z/OSMF V1R13 366
18.7 Installing and customizing z/OSMF 366
18.7.1 z/OSMF plug-ins 367
18.8 Software category and Deployment task 368
18.8.1 Software instance 369
18.8.2 Deployment task 371
18.8.3 Deployment software 372
18.8.4 Deployment Checklist page 373
18.8.5 z/OSMF deployment topology 374
18.8.6 Software deployment summary 375
18.9 ISPF task with z/OSMF 376
18.9.1 Logon to ISPF 376
18.10 DASD Management task 382
18.10.1 Reserve storage pools 383
18.10.2 Defining pool storage group SMA attributes 384
18.11 Capacity Provisioning 385
18.11.1 z/OSMF Capacity Provisioning task 386

18.12 z/OSMF base enhancements 390
18.12.1 Application Linking Manager task 390
18.12.2 z/OSMF Application Linking Manager interface 392
18.12.3 REST API for job management 394
18.13 Workload Management and z/OSMF 397
18.13.1 z/OSMF Workload Management V1R13 new features 397
18.14 Incident Log enhancements 401
18.14.1 Installation considerations for the Incident Log 406
18.15 Resource Monitoring task 407
18.16 Configuration Assistant for z/OS Communications Server 408
18.16.1 Configuration Assistant primary panel 409
18.16.2 Support configuration for new IPSec enhancements 410
18.16.3 Support for reusable rules for IP security 411
18.16.4 Support for configuration of multiple z/OS releases 412
18.17 Getting started with z/OSMF V1R13 412
18.17.1 Customizing the Welcome panel for guest users 413
18.17.2 z/OSMF in a monoplex or sysplex 417

Chapter 19. z/OS Hardware Configuration Definition and Hardware Configuration Manager 419
19.1 HCD and HCM enhancements with z/OS V1R13 420
19.2 HCD support for new hardware 420
19.2.1 Support of 32 subchannels per CIB CF link 422
19.2.2 Internal Queued Direct I/O Extensions for IEDN support 424
19.2.3 Support of IQDX in HCM 428
19.3 Support of z/VM V6R2 432
19.3.1 Support of configuration packages in z/VM 432
19.3.2 Support of z/VM Single System Image 434
19.4 Quality enhancements 435
19.4.1 Miscellaneous enhancements 440

Chapter 20. C language 441
20.1 Improved Metal C optimization 442
20.1.1 z/OS V1R13 enhancements 442
20.1.2 IPA compiler option 443
20.1.3 Interprocedural analysis 444
20.2 New hardware built-ins 446
20.3 Multiply and Add for hexadecimal types 447
20.4 Informational messages in z/OS UNIX System Services 448
20.5 Metal C: function property information 448
20.6 Metal C DSA support 448
20.7 Metal C argument parsing 449
20.8 C++: template depth 450
20.9 Compatibility support 450
20.9.1 Text following #endif 450
20.9.2 Function attribute gnu_inline 451
20.9.3 Function attribute used 451
20.9.4 Function attribute malloc 452
20.9.5 Temporary lifetime extension 452
20.9.6 Binding rvalue to non-const reference 453
20.9.7 Intrinsic complex types 453
20.9.8 Addressable labels 454
20.9.9 Trailing return type 455
20.10 New debugging APIs 455

20.11 Debugging inline procedures 456

Chapter 21. Storage management enhancements 457
21.1 64-bit storage overview 458
21.1.1 64-bit common virtual storage 458
21.1.2 Local system area 459
21.1.3 z/OS V1R13 virtual storage 460
21.2 64-bit subspace support 460
21.2.1 Subspace mode 462
21.2.2 64-bit subspace 463
21.3 Large page enhancements 465
21.3.1 Large page support for the nucleus 465
21.3.2 Large page coalesce support starting with z/OS V1R12 467
21.4 RSM component trace 469

Chapter 22. Common Information Model 471
22.1 Introduction to Common Information Model 472
22.1.1 CIM cross-platform management 472
22.1.2 CIM components and dependencies 473
22.1.3 CIM server and z/OSMF 475
22.1.4 CIM Server overview 476
22.1.5 CIM client-to-CIM Server access 479
22.1.6 CIM enablement 480
22.1.7 CIM client API for Java 481
22.1.8 CIM client/server implementation 482
22.1.9 zIIP support 483
22.2 Enhancements with z/OS V1R13 483
22.2.1 SMI-S support enhancement 484
22.3 CIM customization and setup 486
22.3.1 Required parmlib updates 487

Chapter 23. Predictive Failure Analysis 489
23.1 Predictive Failure Analysis overview 490
23.1.1 PFA-detected system failures 490
23.2 How PFA chooses address spaces to track 491
23.3 Efficient PFA usage 492
23.3.1 Reduce the number of false positives 492
23.3.2 Eliminate jobs causing false positives 493
23.3.3 Automate the PFA IBM Health Checker for z/OS exceptions 493
23.3.4 PFA detection of damaged systems 494
23.3.5 Supervised learning support 495
23.4 PFA infrastructure 496
23.4.1 PFA parameters for health checks 496
23.4.2 Differences between PFA checks and other remote health checks 497
23.5 PFA and IBM Health Checker for z/OS 499
23.5.1 PFA_COMMON_STORAGE_USAGE health check 499
23.5.2 LOGREC arrival rate check 501
23.5.3 Frames and slots usage check 501
23.5.4 Message arrival rate check 501
23.5.5 SMF arrival rate check 502
23.5.6 Miscellaneous improvements 503
23.6 PFA enhancements with z/OS V1R13 503
23.6.1 PFA ENQUEUE REQUEST RATE check 504
23.6.2 PFA_JES_SPOOL_USAGE check for JES2 505

23.6.3 Checking track group usage 506
23.6.4 Define DASD storage for Predictive Failure Analysis 507
23.7 Runtime Diagnostics and z/OS V1R13 507
23.7.1 Using Runtime Diagnostics 508
23.7.2 Runtime Diagnostics command examples and messages 510
23.7.3 PFA integration with Runtime Diagnostics 518
23.8 Using PFA with commands and SDSF 522
23.8.1 SDSF support for PFA 526
23.9 Installing PFA 529
23.9.1 Migration considerations for PFA with z/OS V1R12 531
23.9.2 PFA supports one or multiple ini files 534
23.10 Serviceability information 534
23.10.1 Serviceability improvements starting with z/OS V1R12 535
23.10.2 Migration actions to z/OS V1R12 and beyond 536
23.10.3 Runtime Diagnostics setup 537

Chapter 24. Extended address volume 539
24.1 Extended address volume overview 540
24.1.1 3390 Model A 540
24.1.2 EAV basic definitions 541
24.1.3 EAV terminology 541
24.2 EAV key design points 542
24.2.1 Track-managed space 543
24.2.2 Cylinder-managed space 543
24.2.3 Multicylinder unit 543
24.2.4 DASD track address format 544
24.2.5 Extended address volume attributes 547
24.2.6 EAS-eligible data sets 548
24.2.7 EAS non-eligible data sets 549
24.2.8 New format DSCBs for EAVs 550
24.2.9 EATTR data set attribute 553
24.2.10 Other miscellaneous support 554
24.3 z/OS V1R12 enhancements in DFSMSdfp related to EAV 554
24.3.1 IGGCATxx parmlib member 555
24.3.2 IEASYSxx parmlib member 555
24.3.3 IGGCATxx new statements and parameters 556
24.3.4 MODIFY CATALOG command 557
24.3.5 Catalog address space considerations 558
24.3.6 The intersection of SYSCATxx, LOADxx, and IGGCATxx 559
24.3.7 Installation considerations 560
24.3.8 Alias number constraint relief 560
24.3.9 Catalog VVDS expansion 562
24.3.10 VVDS changes in z/OS V1R13 564
24.3.11 EAS-eligible data sets 566
24.3.12 Review of FTP existing support (z/OS V1R11) 567
24.4 z/OS V1R13 enhancements to EAV support 568
24.4.1 Enhanced FTP support for EAV with z/OS V1R13 568
24.4.2 FTP support for large-format data sets 569
24.4.3 FTP z/OS V1R13 570
24.4.4 LOCSite subcommand 571
24.4.5 SIte subcommand 572
24.4.6 LOCSTat subcommand 573
24.4.7 STAtus subcommand 574

24.4.8 Creating a directory on the remote host 575
24.4.9 Allocating an EAS-eligible PDS. 576
24.4.10 Allocating an EAS-eligible PDSE 577
24.4.11 Using the HELP subcommand 578
24.4.12 Diagnosis and messages 579
24.4.13 FTP client installation considerations 579

Chapter 25. Base Control Program internal interface 581
25.1 Overview of BCPii 582
25.1.1 Using BCPii. 583
25.1.2 BCPii address space. 584
25.1.3 BCPii diagnostics 586
25.2 New in z/OS V1R13 586
25.2.1 BCPii services. 587
25.2.2 BCPii callable services 589
25.2.3 BCPii internal interfaces 591
25.3 Installing BCPii 592
25.3.1 Configuring the local Support Element to support BCPii. 593
25.3.2 Authorize an application to use BCPii. 599
25.3.3 Configuring the BCPii address space 602
25.3.4 Setting up the event notification mechanism for z/OS UNIX callers 602
25.4 BCPii dependencies 604
25.4.1 z/OS BCPii versus TSA BCPii 604
25.4.2 Hardware dependencies. 604
25.5 Exploiters of BCPii 605
25.5.1 Current z/OS system BCPii exploiters 605
25.6 BCP internal interface programming. 606
25.6.1 A Metal C-example 606
25.6.2 BCPii as a set of new z/OS commands 607
25.6.3 Potential benefits of a new command set 609

Chapter 26. Capacity Provisioning 611
26.1 Capacity Provisioning background 612
26.2 Capacity Provisioning Domain 615
26.3 Workload condition 618
26.4 Capacity increments starting with z/OS V1R13 621
26.5 Recurring time conditions starting with z/OS V1R13. 623
26.6 Filtered workload report 625
26.7 Windows 7 support 625
26.8 SNMP removal 625
26.9 Capacity Provisioning hardware requirements 626
26.9.1 Capacity Provisioning communications. 628
26.10 Capacity Provisioning software requirements 629
26.10.1 Preparing the Provisioning Manager. 629
26.10.2 Migration and coexistence 629
26.10.3 Coexistence of Control Center and policy versions 630
26.10.4 Installation of z/OS V1R13 Control Center 630
26.11 Supported LPAR and z/OS environments. 631
26.12 Capacity Provisioning Manager summary. 632

Chapter 27. System SSL enhancements. 633
27.1 ECC certification and key agreement 634
27.2 ECC support for Transport Layer Security (TLS) 634
27.2.1 Cipher suites. 634

27.3 ECC key reference by ICSF key label . . . 636
27.4 RACF and hardware ECC support for RACDCERT . . . 636
27.5 Prerequisites, migration, and coexistence considerations . . . 639
27.6 System SSL ECC example . . . 639
27.6.1 Checking the environment . . . 640
27.6.2 Creating the certificates using the gskkyman utility . . . 642
27.6.3 Signing a certificate using the command line . . . 646
27.6.4 Creating certificates using the RACF RACDERT command . . . 651
27.6.5 Application support . . . 654

Chapter 28. UNICODE support . . . 657
28.1 New Information APIs . . . 658
28.2 BiDirectonial Phase II Enhancements . . . 659

Chapter 29. Language Environment enhancements . . . 665
29.1 CEEPIPI multi-main and user word . . . 666
29.2 LE/C BSAM > 64 K tracks (binary and text) . . . 667
29.3 LE C-RTL I/O ABEND Recovery Part 1 . . . 668
29.4 Metal C qsort() . . . 669

Chapter 30. SDSF enhancements . . . 671
30.1 Enhancements for JES2 . . . 672
30.1.1 Spool migration support . . . 672
30.1.2 Job RC display . . . 672
30.1.3 JES2 $EJ,STEP command support . . . 673
30.1.4 Spin line action in the JDS panel . . . 673
30.1.5 Sysplex-wide view in more panels . . . 674
30.2 Enhancements for JES3 . . . 674
30.2.1 Printer (PR) display . . . 675
30.2.2 Punch (PU) and Reader (RDR) displays . . . 675
30.2.3 Line (LI) display . . . 676
30.2.4 Node (NO) display . . . 677
30.2.5 Initiator (INIT) display . . . 677
30.2.6 Job Zero (J0) display . . . 678
30.3 New panels for both JES2 and JES3 . . . 678
30.3.1 Network Server (NS) display . . . 679
30.3.2 Network Connection (NC) display . . . 679
30.4 Using XCF for sysplex-wide data . . . 680
30.4.1 Enabling SDSF server XCF communication . . . 681
30.4.2 Using z/OS V1R12 compatibility mode . . . 685
30.5 OPERLOG colors . . . 686
30.6 Reading the OPERLOG from SDSF/REXX . . . 688

Chapter 31. JES2 enhancements . . . 691
31.1 Spool migration . . . 692
31.1.1 Managing the migration . . . 693
31.1.2 Spool migration examples . . . 695
31.1.3 $DSPL command processing considerations . . . 698
31.1.4 Extending spool data sets . . . 699
31.1.5 Command support . . . 700
31.2 Naming spool data sets . . . 701
31.3 SPIN for any data set . . . 702
31.3.1 SPIN= JCL parameter operator command . . . 703
31.3.2 JCL SYSOUT DD statement . . . 703

31.4 JCL instream data sets in procedures . . . 705
31.5 Job completion code support . . . 705
31.5.1 JOBRC parameter . . . 706
31.6 Device and property SSI 82 . . . 706
31.7 Using the network resource monitor . . . 707
31.8 Evict job on step boundary . . . 708
31.9 Migration and coexistence considerations . . . 708

Chapter 32. JES3 enhancements . . . 711
32.1 Adding spool volumes without a warm start . . . 712
32.1.1 Adding a spool volume using a hot start with refresh . . . 712
32.1.2 Adding a spool volume using the *MODIFY CONFIG command . . . 714
32.2 Performing automatic flush after a local main failure . . . 716

Chapter 33. Resource Management Facility enhancements . . . 719
33.1 WLM reporting enhancements . . . 720
33.1.1 WLMGL postprocessing report . . . 720
33.1.2 Supervisor promotion . . . 723
33.1.3 Response time distribution for velocity and discretionary goals . . . 724
33.1.4 Resource group capacity values . . . 726
33.2 GRS and supervisor delay monitoring enhancements . . . 726
33.2.1 Contents and division of the SDELAY report . . . 726
33.3 Redesign of the Postprocessor Paging Activity report . . . 732
33.3.1 Purpose and uses of the information given on the report . . . 733
33.3.2 New contents of the report . . . 733
33.4 Enhanced RMF Monitor II OPT Settings report . . . 734
33.4.1 Purpose and content of the report . . . 735
33.5 Integrated ensemble performance monitoring . . . 736
33.5.1 Interactions and dependencies . . . 739
33.5.2 Installation considerations . . . 739
33.6 RMF data portal and z/OSMF integration . . . 740
33.6.1 z/OSMF Resource Monitoring plug-in . . . 742
33.6.2 System Status task overview . . . 744
33.7 z/VM guest support . . . 745

Chapter 34. z/OS Batch Runtime . . . 747
34.1 Batch runtime - from punch cards to Java batch . . . 748
34.1.1 Differences between OLTP and batch processing . . . 749
34.2 Batch launcher and toolkit for Java applications . . . 752
34.2.1 z/OS V1R13 enhancements . . . 752
34.3 Java and COBOL interoperability . . . 752
34.3.1 Resource Recovery Services . . . 755
34.3.2 Java COBOL with DB2 interoperability . . . 757
34.3.3 Usage and invocation . . . 757
34.3.4 Interactions and dependencies . . . 760
34.3.5 Installation considerations . . . 760
34.3.6 Benefits of Java batch on z/OS . . . 760
34.4 JES2 batch modernization . . . 761
34.4.1 JCL instream data sets . . . 761
34.4.2 Support for JOBRC (job return code) . . . 761
34.4.3 Evict job on a step boundary . . . 763

Appendix A. Setting up WebSphere OEM, z/OSMF, CIM, and Capacity Provisioning 765
A.1 Setting up WebSphere Application Server OEM . . . 766

A.2 Setting up z/OSMF V1R13 785
A.3 Setting up CIM 798
A.4 Setting up Capacity Provisioning 800
A.4.1 Defining runtime data sets 801
A.4.2 Adapting Provisioning Manager parameters. 801
A.4.3 Creating the started task procedure 803
A.4.4 Providing APF authorization 804
A.4.5 Securing the runtime system 805
A.4.6 Defining security for hardware access 806
A.4.7 Defining security for the Control Center user 807
A.4.8 Securing the observed systems 807
A.4.9 Setting up Automatic Restart Manager. 807
A.4.10 Preparing the connection to the CIM server 807
A.4.11 Preparing the connection to the Provisioning Manager 808

Appendix B. zFS commands 809
B.1 Using zFS commands. 810

Appendix C. BCPii Metal C example 823
C.1 C code 824
C.2 JCL to compile, assemble, and link edit 842
C.2.1 C compiler options 844
C.3 BCPii example as a System REXX 844
C.4 Sample C code for BCPii services 845

Related publications 887
IBM Redbooks publications 887
Other publications 887
Online resources 889
Help from IBM 890

Index 891

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

AIX®
BladeCenter®
BookManager®
CICS®
DB2®
Domino®
DS8000®
ESCON®
eServer™
FICON®
FlashCopy®
GDDM®
GDPS®
HiperSockets™
HyperSwap®
IBM®
IMS™
Language Environment®
Lotus®
MQSeries®
MVS™
NetView®
OMEGAMON®
OS/390®
Parallel Sysplex®
PR/SM™
ProductPac®
pureScale®
RACF®
Rational®
Redbooks®
Redbooks (logo) ®
Resource Measurement Facility™
RETAIN®
RMF™
System p®
System x®
System z10®
System z9®
System z®
System/390®
SystemPac®
Tivoli®
VTAM®
WebSphere®
xSeries®
z/Architecture®
z/OS®
z/VM®
z/VSE®
z10™
z9®
zEnterprise™
zSeries®

The following terms are trademarks of other companies:

Intel, Pentium, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Microsoft, Windows NT, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java, and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Cell Broadband Engine and Cell/B.E. are trademarks of Sony Computer Entertainment, Inc., in the United States, other countries, or both and is used under license therefrom.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Preface

This IBM® Redbooks® publication provides information about installation and migration changes to be aware of if you are responsible for migrating systems from IBM z/OS® V1R10, z/OS V1R11, and z/OS V1R12 to z/OS V1R13. It also highlights actions that are needed to prepare for the installation of z/OS V1R12, including ensuring driving system and target system requirements are met and coexistence requirements are satisfied. There is a special focus on identifying new migration actions that must be performed for selected elements when migrating to z/OS V1R13.

The book addresses the following topics:

- z/OS V1R13 overview, z/OS V1R13 installation, managing volume backups with fast replication, XCF enhancements, console service enhancements
- DFSMSdfp, DFSMSoam, DFSMShsm, ISPF enhancements, DFSMSrmm enhancements, establishing IBM RACF® security for RRSF TCP/IP connections
- GRS enhancements, BCP supervisor, contents supervisor and RSM updates, improved channel recovery, Service Aids enhancements, System Logger - SMF
- z/OS UNIX System Services, z/OS UNIX-related applications, RRS, z/OS Management Facility, z/OS HCD and HCM, C language
- Storage management enhancements, Common Information Model, Predictive Failure Analysis, extended address volume, BCPii, Capacity Provisioning
- System SSL enhancements, UNICODE, IBM Language Environment®, SDSF enhancements, JES2 enhancements, JES3 enhancements, IBM RMF™ enhancements
- IBM WebSphere® Application Server OEM, z/OSMF, CIM, and Capacity Provisioning setups
- BCPii Metal C example

The team who wrote this book

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, Poughkeepsie Center.

Paul Rogers is a Consulting IT Specialist at the International Technical Support Organization, Poughkeepsie Center. He writes extensively and teaches IBM classes worldwide on various aspects of z/OS, z/OS UNIX, JES3, and Infoprint Server. Before joining the ITSO 21 years ago, Paul worked in the IBM Installation Support Center (ISC) in Greenford, England for eight years, providing IBM OS/390® and JES support for IBM EMEA. He also worked in the Washington Systems Center for three years, and has been with IBM for more than 44 years.

Diniz Bortolotto is a systems programmer at Banco do Brasil, a government bank in Brazil. He has seven years of experience in the field of mainframe technology. Diniz's areas of expertise include z/OS, DFSMS, Storage Management, and Tape Libraries Architecture, Implementation and Administration.

Robert Hering is a systems programmer and technical support coordinator at Banrisul in Brazil. He has 34 years of experience in the field of mainframe technology including z/OS, and has a technical background in computer networks. His areas of expertise include IBM

CICS®, assembler programming, and computer performance evaluation. Robert has written extensively about SMF.

George Kozakos is a Senior IT Specialist with IBM Australia. He has more than 22 years of experience in IBM MVS™ system programming. His areas of expertise include Server Time Protocol and IBM GDPS®. George holds degrees in Computing Science and Pure Mathematics.

Lutz Kuehner is a Senior Client Technical Specialist with the System Technologie Group sales organization, IBM Germany, providing technical System z® support for financial market clients. He has 25 years of experience in the field of mainframe technology, and his areas of expertise include z/OS, z/OS UNIX and high availability topics. Lutz has written extensively about z/OS-related topics and has contributed to several IBM Redbooks.

Jean-Louis Lafitte is a Senior Computing Machinery expert at GATE Informatic SA, an IBM Premier Business Partner in Switzerland. He has more than 40 years of experience with IBM Large Enterprise Systems, and has worked on several parallel machines (IBM RP3, CM2, KSR1, IBM SP1, SP2 and BG/L, P). Since 1984, he has been associated with IBM Parallel Sysplex® and its derivatives such as IBM DB2® pureScale®. More recently, he worked on designing hybrid structures exploiting Cell/B.E. Jean-Louis holds a Ph.D. in Theoretical Computer Science, as well as several patents in IBM System/390® hardware virtualizer architecture. He is a member of ACM and IEEE.

Diana Nakajima works in IBM Sales and Distribution in Brazil, where she is a Technical Sales Specialist working with clients of large IBM zSeries® systems.

Paulo Nascimento is a Support Coordinator with Eletrobras Eletronuclear, which is a nuclear plant in Brazil. He has 32 years of experience in the field of mainframe technology, and his areas of expertise include z/OS, DB2, SAP DB, RACF, and z/OS Workload Manager. Paulo has participated in several z/OS Explores meetings with IBM.

Nelson Oliveira is a Senior IT Specialist with IBM Brazil. He has 20 years of experience in the field of mainframe technology. His areas of expertise include z/OS, JES2, sysplex, high availability, and GDPS. Nelson is responsible for supporting the Bradesco bank in Brazil in his areas of expertise.

Gil Peleg is a systems programmer and manager of ServFrame, a mainframe consulting and training company in Israel and an IBM Business Partner. He has 13 years of experience in mainframe system programming and in teaching mainframe-related courses. His areas of expertise include Parallel Sysplex, GDPS, WLM, and JES3. Gil holds a degree in Computer Science.

Giancarlo Rodolfi is a System z Technical Sales Specialist in Brazil. He has 25 years of experience in the zSeries field. He has written extensively about the z/OS Communication Server, security and z/OS.

Anthony Soares is a systems programmer at Banco do Brasil, a government bank in Brazil. He has nine years of experience in the field of mainframe technology. His areas of expertise include z/OS and RACF. Anthony holds a degree in System Analysis from Universidade Federal do Acre (UFAC), and a graduate degree in Systems Engineering from Escola Aberta do Brasil (ESAB).

Joao Zanon is a systems programmer at Banco do Brasil, a government bank in Brazil. He has seven years of experience in mainframe systems, including IBM z/VM®, z/Linux, and z/OS. His areas of expertise include Parallel Sysplex, WLM, and RMF. Joao holds a degree in Telecommunications Engineering from the Brazilian federal university, Universidade de Brasilia (UNB).

Thanks to the following people for their contributions to this project:

Robert Haimowitz
International Technical Support Organization, Poughkeepsie Center

Richard Conway
International Technical Support Organization, Poughkeepsie Center

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

  ibm.com/redbooks
- Send your comments in an email to:

  redbooks@us.ibm.com
- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

Stay connected to IBM Redbooks

- Find us on Facebook:

  http://www.facebook.com/IBMRedbooks
- Follow us on Twitter:

  http://twitter.com/ibmredbooks
- Look for us on LinkedIn:

  http://www.linkedin.com/groups?home=&gid=2130806

- Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

  https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm

- Stay current on recent Redbooks publications with RSS Feeds:

  http://www.redbooks.ibm.com/rss.html

1

IBM z/OS Version 1 Release 13 overview

IBM z/OS (program number 5694-A01) is an operating system designed to meet the on-demand challenges of the business world. z/OS delivers the highest qualities of service for enterprise transactions and data, and extends these qualities to new applications using the latest software technologies. The following list describes several z/OS highlights:

- The 64-bit IBM z/Architecture® implemented by z/OS eliminates bottlenecks associated with the lack of addressable memory. The 64-bit real (central) storage support eliminates expanded storage, helps to eliminate paging, and may allow you to consolidate your current systems into fewer logical partitions (LPARs) or to a single native image.
- The Intelligent Resource Director (IRD) expands the capabilities of Workload Manager (WLM) by enabling resources to be dynamically managed across LPARs based on workload priorities.
- A Workload License Charges pricing model offers you flexibility in how your software product licenses are managed and charged.
- IBM HiperSockets™ provides high-speed, low-latency TCP/IP data communication across LPARs within the same System z server. HiperSockets acts like a TCP/IP network within the server, eliminating the need to use I/O subsystem operations and the need to traverse an external network to communicate between LPARs in the same server.

This chapter describes the functional enhancements to z/OS for Version 1 Release 13.

1.1 Base z/OS enhancements

The Base Control Program (BCP) provides essential operating system services. The BCP includes the I/O configuration program (IOCP), the Workload Manager (WLM), System Management Facilities (SMF), the z/OS UNIX System Services (z/OS UNIX) kernel, the program management binder, and other components.

RMODE 64 phase 1

In z/OS V1R13, z/OS supports programs running in 64-bit storage, provided that they meet certain restrictions. This is intended to provide virtual storage constraint relief to applications, particularly those that imbed code in data areas for performance reasons.

Improved channel recovery

In z/OS V1R13, the I/O Supervisor (IOS) has improvements to I/O error recovery. IOS is now designed to track path-related errors automatically to remove failing paths from all affected devices for the affected control unit. This new capability is designed to reduce the time it takes the system to recover from path-related errors and help prevent system performance problems that can occur when a significant amount of time is spent in repetitive channel error recovery.

Runtime Diagnostics enhancements

In z/OS V1R13, Runtime Diagnostics provides more information intended to help you determine why a system is not running normally. It is extended to check for latch contention, and for delays that affect z/OS UNIX System Services tasks. Additionally, in z/OS V1R13 the Runtime Diagnostics address space is started during IPL to make it available sooner during system operation.

CMDS command extensions

In z/OS V1R13, command processing improvements have been implemented. The **CMDS** command, which among other options can be used to terminate the processing of a particular command, has a new FORCE option. As with other FORCE commands and keywords, the FORCE option will allow you to specify that a command be terminated. However, the effects of forcing command termination are not always predictable. This option can be used only when there is no other option but to IPL.

In addition, new support allows command processors to specify whether the **CMDS** command should terminate its commands without the use of FORCE. A security profile in the OPERCMDS class can be used to limit the use of FORCE to authorized users. This new function helps improve system availability by making it clear which commands are intended to support termination at any time and which are not.

Make distributed mode the default console services operating mode

In z/OS V1R13, the default operating mode for consoles processing is changed to distributed mode.

z/OS JCL instream data sets

In z/OS V1R13, several batch enhancements are new, as described here:

- There is new support for in-stream data sets to be used within JCL procedures and for include statements. This will improve usability of JCL procedures, for example, by making it possible to put utility control statements in the same member as the rest of the procedure.

- There is support to allow jobs for which journaling is used to be stopped after a currently-running step has finished, and then held for restart in the following step. This is intended to allow less-disruptive system shutdowns.
- There is new support for job return codes. This support allows you to specify that the job return code be set to the highest return code encountered by any step, the last step, or a specified step in the job. This will help make it simpler to interpret the results of job execution.
- There is support for a new UNALLOC parameter for the SPIN keyword on the DD statement, to allow you to specify that any output data set be spun off at an interval you choose. This is intended to help improve availability for long-running address spaces.

BCP scheduler JOBRC support

In z/OS V1R13, job-level return code support is implemented. A new JOBRC keyword on the JOB statement is designed to allow you to specify whether the job's return code should be set to the highest return code of any step that was run, the return code of the last step, or the return code of a specific step. This new support can make it simpler to determine whether a job succeeded or failed by looking in a single place for the result.

Batch modernization container: Java COBOL infrastructure

In z/OS V1R13, a batch modernization initiative is designed to introduce a new platform-neutral programming model called the z/OS Batch Runtime, which is intended to allow batch code written to this new standard to run natively on z/OS.

z/OS Batch Runtime is intended to provide the framework for Java-to-COBOL interoperability, for transactional updates to DB2, and for sharing database connections between Java and COBOL. Java and DB2 support is provided with IBM 31-bit SDK for z/OS, Java Technology Edition, Version 6.0.1 (5655-R31), and DB2 9 or later with PTFs.

Reuse connections when using PDSEs

In z/OS V1R13, the Binder is changed to support processing more than 64 K PDSE members in a single binder dialog. This will remove the prior limit of approximately 64 K PDSE members per IEBCOPY operation to address the problem described by APAR OA11043.

Binder to delete unnamed sections in a load module

In z/OS V1R13, the Program Management Binder supports new operations for unnamed ($PRIVATE) sections in load modules and program objects. This new design will allow you to specify that all unnamed sections be deleted, or that specific unnamed sections be named (using CHANGE statements) or replaced.

These new functions are intended to make it easier to maintain load modules and program objects in place, and avoid unnecessary growth and virtual storage use due to an accumulation of unnamed sections.

IEBCOPY performance improvements

In z/OS V1R13, enhancements for the IEBCOPY utility are intended to improve performance when copying a partitioned data set (PDS) to another PDS. In addition, IEBCOPY code base uses 24-bit addressing, which inhibits use of virtual storage. Enhancements are also included to allocate track image buffers in 31-bit storage to provide more efficient usage of virtual storage. For improved ease of use of IEBCOPY, the current requirement for APF authorization will be removed in z/OS V1R13.

Message flood automation constraint

In z/OS V1R13, message flood automation processing is changed to increase the limit of message IDs from 50 to 1024 and allowing up to 128 address spaces to be tracked per system, and to allow the default message set to be identified in a parmlib member. This is intended to increase the scope of message flood automation, improve its usability, and help improve system availability.

A service to avoid DB2 recreating the JVM environment

In z/OS V1R13, support now allows tasks in an enclave that have subtasks to leave and rejoin the enclave. In prior releases, enclave tasks with subtasks are not allowed to leave an enclave.

Any subtasks created by a task that has joined an enclave are automatically added to the same enclave. This new support will allow a task to leave an enclave along with its subtasks.

Unicode improves INFO API to provide control character information

In z/OS V1R13, the Unicode conversion information service provides additional information about Coded Character Set Identifiers (CCSIDs), including information to identify substitution, newline, line feed, Carriage return, end-of-file and space character codes.

BiDi phase 2 support

In z/OS V1R13, Unicode Services has implemented improved bidirectional character support for applications that process scripts such as those for the Arabic and Hebrew languages. These languages, among others, are written and read from right to left but often contain strings that read from left to right. This new function is designed to support such nested bidirectional (bidi) strings. Also, samples are supplied that show how to use these extended bidirectional services, and a sample object file is supplied that you can include with C applications to make it easier to use z/OS Unicode Services functions defined by The Open Group's Bidi interface.

1.1.1 System logger

System logger is an MVS component that allows an application to log data from a sysplex. You can log data from one system or from multiple systems across the sysplex.

IFASMFDL to stop reading before end of logstream

In z/OS V1R13, the SMF dump program used for processing SMF log streams (IFASMFDL) is enhanced to reduce the time it takes to extract the data. This new function is designed to allow you to specify a new SMARTENDPOINT keyword when running the utility, to limit the amount of data read from the log stream being processed.

Lift restriction on IFASMFDL ARCHIVE/DELETE processing

In z/OS V1R13, IFASMFDL allows you to specify that an entire SMF log stream be archived or deleted. The new ARCHIVE and DELETE functions will transparently support processing all SMF data that exists in a log stream when the program begins to process it. This is intended to allow you to easily migrate SMF data management processes based on archiving the entire content of SYS1.MAN SMF data sets to use SMF log streams.

1.1.2 Predictive Failure Analysis

Soft failures are abnormal yet allowable behaviors that can slowly lead to the degradation of the operating system. To help eliminate soft failures, z/OS has developed Predictive Failure

Analysis (PFA). PFA is designed to predict if a soft failure will occur sometime in the future and to identify the cause, while keeping the base operating system components stateless.

PFA is intended to detect abnormal behavior early enough to allow you to correct the problem before it affects your business. PFA uses remote checks from IBM Health Checker for z/OS to collect data about your installation. Next, PFA uses machine learning to analyze this historical data to identify abnormal behavior. It warns you by issuing an exception message when a system trend might cause a problem. To help you correct the problem, it identifies a list of potential issues.

Integration of PFA and RTD to detect hung address spaces

In z/OS V1R13, PFA now monitors JES2 spool utilization and enqueue activity for persistent address spaces to predict potential problems. Also, when PFA detects a rate that is too low for SMF arrival rates, message arrival rates, or enqueue request rates, it invokes Runtime Diagnostics automatically. When Runtime Diagnostics indicates there is a problem, PFA issues a Health Check exception and includes that information. This new function is intended to help you quickly diagnose system problems and problems with persistent address spaces.

1.1.3 GRS enhancements

z/OS provides multiple ways of providing serialized access to data on a single or multiple systems, but global resource serialization (GRS) is a fundamental way for programs to get the control they need and ensure the integrity of resources in a multisystem environment.

Because global resource serialization is automatically part of your system and is present during z/OS initialization, it provides the application programming interfaces that are used by the applications on your system.

Latch Identity support for D GRS,C

In z/OS V1R11, GRS added support for latch obtainers to identify their latches to make it easier to determine who held them when reading messages from the **DISPLAY GRS,ANALYZE** command troubleshooting problems from dumps.

In z/OS V1R13, GRS adds the same latch identity information to the output of the **DISPLAY GRS,CONTENTION** command.

1.1.4 Health checks

The objective of IBM Health Checker for z/OS is to identify potential problems before they impact your availability or, in worst cases, cause outages. It checks the current active z/OS and sysplex settings and definitions for a system and compares the values to those suggested by IBM or defined by you. It is not meant to be a diagnostic or monitoring tool. It is intended to be a continuously running preventive that finds potential problems.

Note: IBM Health Checker for z/OS produces output in the form of detailed messages to let you know of both potential problems and suggested actions to take. These messages simply inform you of potential problems so that you can take action on your installation.

These messages do not mean that IBM Health Checker for z/OS has found problems that you need to report to IBM.

Provide precise check scheduling control

In z/OS V1R13, the Health Checker framework is enhanced to allow you to specify when health checks should be run for the first time, and how often they should be repeated using a new SYNCVAL keyword in the HZSPRMxx parmlib member or on the Health Checker **MODIFY** command. This can help you schedule checks to run at more predictable times.

Enhanced Check Exception Control

In z/OS V1R13, extensions to the Health Checker framework allow better control over check exception messages by allowing a health check to decide when exceptions messages are deleted, by allowing checks to specify the severity of an exception dynamically as system conditions change, and by allowing greater exception intervals to help avoid exception flooding. These changes are intended to allow programmers to write advanced health checks with improved usability.

zFS Health Check

In z/OS V1R13, a new migration health check is implemented for zFS. This function is also made available with PTFs for z/OS V1R11 and z/OS V1R12. This function warns you when the zFS configuration option is not set to sysplex=filesys as recommended.

1.1.5 XES enhancements

Cross-system extended services (XES) uses one or more coupling facilities to satisfy client requirements for:

- Data sharing across the systems in a sysplex
- Maintaining the integrity and consistency of shared data
- Maintaining the availability of a sysplex

“CF not suitable for allocation” preference list reasons

In z/OS V1R13, information about why a coupling facility is not suitable for allocation of a particular structure is added to XES messages. This will make it easier to determine why an allocation failed.

1.1.6 Capacity Provisioning

z/OS Capacity Provisioning helps you manage the CP, zAAP, and zIIP capacity of IBM System z10® servers that are running one or more z/OS systems. Based on On/Off CoD, temporary capacity may be activated and deactivated with a policy you define. z/OS Capacity Provisioning simplifies the monitoring of critical workloads, and its automation features can help to activate additional resources faster than manual operation.

Capacity Provisioning requirement

In z/OS V1R13, the Capacity Provisioning Manager supports the specification of capacity increments for both provisioning and deprovisioning actions, and allows you to specify different quantities for obtaining the first capacity increment and subsequent increments. This will help you add the right amount of capacity more quickly, with fewer activation actions.

Note: In z/OS V1R13, support is planned for Microsoft Windows 7 (both 32-bit and 64-bit versions).

1.1.7 Resource Management Facility enhancements

Resource Management Facility (RMF) issues reports about performance problems as they occur so that your installation can take action before the problems become critical.

An installation can use RMF to:

- Determine that your system is running smoothly
- Detect system bottlenecks caused by contention for resources
- Evaluate the service your installation provides to different groups of users
- Identify the workload delayed and the reason for the delay
- Monitor system failures, system stalls, and failures of selected applications

RMF: GRS reporting enhancements (Stage 2)

In z/OS V1R13, RMF provides additional system suspend lock, enqueue, and latch contention information in a new Postprocessor Serialization Report, and also in new SMF Type 72 subtype 5 records. This is expected to help make it easier to respond to serialization-related performance issues, including latch contention.

RMF: WLM reporting enhancements

In z/OS V1R13, RMF uses new WLM services to provide response time distribution information about all service classes for which velocity goals are set, in addition to those for which response time goals are set, in the Workload Activity Report. This is intended to give you a better view of response time distribution across all WLM service classes.

Windows 7 support

In z/OS V1R13, support is added for Microsoft Windows 7 (both 32-bit and 64-bit versions) in RMF Performance Monitoring.

RMF integrated ensemble performance monitoring

With z/OS V1 R13 and z/OSMF V1R13, RMF plans to provide new CIM-based performance data gatherers for Linux on System z, Linux on System x®, and AIX® systems to provide a consistent monitoring solution for IBM zEnterprise™ ensembles.

Along with the Resource Monitoring plug-in for the z/OS Management Facility, which was first made available with z/OSMF V1R12, this is intended to display performance metrics from those platforms and combine them with z/OS metrics in common graphic views.

1.1.8 BCPii enhancements

The Base Control Program Internal Interface (BCPii) function allows authorized applications to query, change, and perform basic operational procedures against the installed System z hardware base. As a base BCP component, the new BCPii address space allows authorized z/OS applications to access the local support element and other support elements in the HMC network to perform HMC-like functions.

BCPii hardware support

In z/OS V1R13, the BCPii allows authorized programs to perform functions related to CPC Image User groups on IBM zEnterprise and IBM System z10 servers with a minimum microcode level (MCL). This new function allows available CPC Image user groups on a particular CPC and their member images to be listed, connected to, and disconnected from groups; to support queries for group attributes; and to support issuing most HWICMD BCPii commands to all members of a specified group.

1.1.9 Common Information Model enhancements

Common Information Model (CIM) is a standard data model for describing and accessing systems management data in heterogeneous environments. It allows system administrators to write applications that measure system resources in a network with different operating systems and hardware. To enable z/OS for cross-platform management, a subset of resources and metrics of a z/OS system are mapped into the CIM standard data model.

CIM: SMI-S storage HBA enhancements

The Storage Management Initiative Specification (SMI-S) was developed by members of the Storage Networking Industry Association (SNIA). It defines “an interface for the secure, extensible, and interoperable management of a distributed and heterogeneous storage system.” The SMI-S specifications define the various domains of storage management in the form of CIM profiles and sub-profiles.

In z/OS V1R12, CIM added support for the Storage HBA and Host Discovered Resources (HDR) profiles. In z/OS V1R13, CIM is designed to add support for CIM Indications to the Storage HBA and SB Multipath Management profiles. These extensions are intended to help form a basis for multiplatform storage management tools.

CIM indication Service Enhancements

In z/OS V1R13, CIM provides sequence identifiers in the indications profile. This is designed to allow unsuccessful deliveries to be retried by the CIM Server; to allow lost and duplicate deliveries to be detected by a WBEM listener; and to allow a listener to reorder any indications that arrive out of order. This new function can provide better reliability and robustness for event processing in CIM.

CIM standards currency

In z/OS V1R13, the CIM Server is upgraded to version 2.11 of the OpenPegasus CIM Server. Also, the CIM Servers Schema repository is updated to CIM Schema version 2.25. This is intended to keep the z/OS CIM Server and schema current with the CIM standard from The Open Group and DMTF, and to allow z/OS enterprise management applications to manage z/OS systems.

1.1.10 ISPF enhancements

ISPF is a development tool set for the z/OS operating system. Since 1975, MVS programmers have used ISPF for host-based application development productivity. ISPF forms the basis of many TSO and CMS applications and also provides extensive programmer-oriented facilities.

ISPF editor support for line command macros

In z/OS V1R13, ISPF provides support for line command-level Edit macros, in addition to the existing Edit macro support. This new design is intended to allow you to write macros to be used as line commands, in addition to those you might have already written for use as Edit primary or initial processing commands.

Support for new data set allocation commands from ISPF Data Set List

In z/OS V1R13, ISPF provides a new **AL** line command on the Data Set List panel (option 3.4) command that you can use to allocate a new data set using a specified data set as a model for the new data set's attributes.

ISPF: z/OS UNIX Directory List enhancements

In z/OS V1R13, ISPF adds a new option to the z/OS UNIX Directory List panel to allow authorized users to update access control lists (ACLs) from within ISPF.

INFO command to display PDS member extended statistics

In z/OS V1R11, ISPF provided extended statistics support for partitioned data set members with more than 32,756 lines. In z/OS V1R13, display supports extended statistics with a new INFO command to display extended line counts when the Extended PDS statistics function has been enabled.

Display data set create jobname and stepname

In z/OS V1R13, ISPF provides support for the job and step names stored by the system for data sets that are eligible to reside in the extended addressing space (EAS) on an extended address volume (EAV) in the Data Set List utility (option 3.4), and in the DSLIST and LMDLIST services. This is intended to allow you to find the creating job and step names easily.

New IPCS subcommand DOCPU to perform tasks on specified CPUs

As z/OS supports more and more CPUs, problem determination becomes more complex. In z/OS V1R13, a new IPCS subcommand, DOCPU, is created to repeat the requested tasks on specified CPUs when diagnosing problems using a stand-alone dump.

1.1.11 RRS enhancements

Many computer resources are so critical to a company's work that the integrity of these resources must be guaranteed. If changes to the data in the resources are corrupted by a hardware or software failure, human error, or a catastrophe, the computer must be able to restore the data. These critical resources are called protected resources or, sometimes, recoverable resources.

Resource recovery is the protection of the resources. Resource recovery consists of the protocols and program interfaces that allow an application program to make consistent changes to multiple protected resources.

RRS queue optimization

In z/OS V1R13, RRS improves performance when there is a large number of outstanding units of work in a large log sharing environment.

1.1.12 XCF enhancements

Cross-system coupling (XCF) services allow multiple instances of an application or subsystem, running on different systems in a sysplex, to share status information and communicate with each other.

Simplified XCF communications services

In z/OS V1R13, XCF provides a simplified set of interfaces for passing messages within a Parallel Sysplex. New services are designed to allow a server to be established to process messages and for messages to be sent across the sysplex without first joining an XCF group. This is intended to make it easier to benefit from XCF services for applications that need relatively simple messaging functions.

1.1.13 Language Environment enhancements

Language Environment provides a common run-time environment for IBM versions of certain high-level languages (HLLs), namely C, C++, COBOL, Fortran, and PL/I, in which you can run existing applications written in previous versions of these languages as well as in the current versions. Prior to Language Environment, each of the HLLs had to provide a separate run-time environment.

LE/C BSAM >64 K tracks (binary and text)

In z/OS V1R8, Language Environment began to support large nonextended format sequential (DSNTYPE=LARGE) data sets when accessed using QSAM (noseek). In z/OS V1R12, this function was extended to provide BSAM (seek) support for record I/O.

In z/OS V1R13, BSAM (seek) for binary and text I/O for sequential data sets is implemented. This new support is intended to let your Language Environment C/C++ applications take full advantage of nonextended format sequential data sets using Language Environment applications.

Provide qsort function within Metal C

In z/OS V1R13, Language Environment adds the qsort() function to the z/OS Metal C Runtime Library. This function allows an array to be sorted using a function you supply. It is intended to relieve Metal C programmers from having to write sort routines with similar capabilities.

LE C-RTL I/O ABEND recovery

In z/OS V1R13, Language Environment supports recovery from all I/O abend conditions for C/C++ programs and return to C/C++ programs indicating that an I/O error has occurred, rather than issuing an abend. This is intended to provide a more predictable recovery environment for C programs when I/O errors are encountered.

CEEPIPI multiple main and user word

In z/OS V1R13, Language Environment adds support for initializing multiple CEEPIPI main environments under one task control block (TCB) and to provide access to a user word for each environment both within and outside it. This is intended to help you migrate Preinitialization Compatibility Interface (PICI) environments to CEEPIPI.

1.1.14 Hardware Configuration Definition - HCM enhancements

Hardware Configuration Definition (HCD) provides an interactive interface that allows you to define the hardware configuration for both a processor's channel subsystems and the operating system running on the processor.

The z/OS and z/VM Hardware Configuration Manager (HCM) is a PC-based client/server interface to HCD that combines the logical and physical aspects of hardware configuration management.

HCD/HCM: HW/SW currency support

In z/OS V1R13, HCM supports Microsoft Windows 7 Professional Edition (both 32-bit and 64-bit).

1.2 JES2 enhancements

MVS uses a job entry subsystem (JES) to receive jobs into the operating system, schedule them for processing by MVS, and control their output processing. JES2 is descended from Houston automatic spooling priority (HASP).

HASP is defined as a computer program that provides supplementary job management, data management, and task management functions such as scheduling, control of job flow, and spooling. HASP remains within JES2 as the prefix of most module names and the prefix of all messages sent by JES2 to the operator.

JES2 spool migration

In z/OS V1R13, JES2 support allows you to discontinue the use of a spool volume dynamically. A new **$MIG SPOOL** command allows you to specify that either a new spool data set of equal or greater size on a different volume is to be used to replace an existing spool data set, or that another existing spool data set with sufficient contiguous space is to be used to replace an existing spool data set. This new function is intended to help you improve availability when removing spool volumes from a JES2 system or MAS.

1.3 System Display and Search Facility enhancements

System Display and Search Facility (SDSF) is a utility that allows you to monitor, control, and view the output of jobs in the system. After submitting a job, it is common to use SDSF to review the output for successful completion or to review and correct JCL errors. SDSF allows you to display printed output held in the JES spool area. Much of the printed output sent to JES by batch jobs and other jobs is never printed. Instead, it is inspected using SDSF and deleted or used as needed.

New SDSF support (JES2 and JES3)

In z/OS V1R13, SDSF adds new support to remove the requirement for IBM MQSeries® in JES2 environments after all systems in a MAS are running z/OS V1R13 JES2. In this release, SDSF implements for JES3 all the applicable functions that are supported for JES2.

For JES2, new planned support includes JES network server and network connections displays. For JES3, new support includes displays for initiators, output, held information, punches, readers, JES network server, and JES Network Connections.

JES3 functions require that the JES3 global and local systems are all running z/OS V1R13 JES3. The corresponding SDSF Java classes are updated to support the new displays and actions. These changes are intended to provide system management improvements.

SDSF EAV support for output data sets

In z/OS V1AR13, SDSF supports extended format sequential (DSNTYPE=LARGE) print files, and print files that are placed in the extended addressing space (EAS) of an extended address volume (EAV).

SDSF Java API caching support

In z/OS V1R12, a Java API was implemented for SDSF to allow Java applications access to the data available through SDSF panels. That API is a set of classes. Each class corresponds to a panel, and each instance represents a row, with methods to perform operations similar to action characters and overtypes, and support for filtering.

In z/OS V1R13, there is support for returning a subset of the requested data. For example, if a set of filters matches a large number of SDSF instances that are normally all returned in a list, then a sublist can be requested in a similar manner where the caller can specify relative starting and ending indexes to limit the number of instances to be returned. Additionally, support is planned for paging through active SDSF data while maintaining a relative index that can be repositioned as needed. This new support is intended to allow Java applications to request smaller amounts of data from SDSF.

SDSF REXX operlog support

In z/OS V1R13, SDSF REXX has support for reading the sysplex-wide operations log (OPERLOG), in addition to the single system SYSLOG. The support for OPERLOG is planned to be very similar to that for SYSLOG, allowing records to be selected by start time and date, and providing the ability to specify a maximum number of records to be returned. A Java application is also planned to allow access to OPERLOG. These enhancements are intended to make it easier for you to perform complex repetitive functions programatically using SDSF. Additionally, the OPERLOG display will be designed to show messages in the same colors in which they would be displayed on a console.

Eliminate MQ requirement for sysplex requests - SDSF

In z/OS V1R13, SDSF provides sysplex-scope displays, including those for the system log (SYSLOG), devices, and initiators, without the use of MQSeries after all systems in the sysplex are at the z/OS V1R13 SDSF level. This new function is designed to use SDSF server address spaces and new XCF services, and to help simplify SDSF setup and operation.

1.4 JES3 enhancements

With the z/OS MVS JES3 system, resource management and workflow management are shared between MVS and its Job Entry Subsystem 3 (JES3) component. Generally speaking, JES3 performs resource management and workflow management before and after job execution. MVS performs resource and workflow management during job execution.

JES3 spool add

In z/OS V1R13, JES3 supports adding spool volumes dynamically by using new operands on the **`*F CONFIG`** command, or by using a JES3 hot start with refresh. This is intended to help improve availability by removing the requirement for a complex-wide IPL for adding spool volumes.

1.5 zFS and z/OS UNIX enhancements

The z/OS Distributed File Service zSeries File System (zFS) is a z/OS UNIX System Services (z/OS UNIX) file system that can be used in addition to the hierarchical file system (HFS). zFS file systems contain files and directories that can be accessed with z/OS UNIX application programming interfaces (APIs). These file systems can support access control lists (ACLs). zFS file systems can be mounted into the z/OS UNIX hierarchy along with other local (or remote) file system types (for example, HFS, TFS, AUTOMNT, and NFS).

The z/OS UNIX System Services element is a UNIX operating environment, implemented within the z/OS operating system. It is also known as z/OS UNIX. The z/OS support enables

two open systems interfaces on the z/OS operating system: an application programming interface (API) and an interactive shell interface.

zFS automatic takeover of disabled aggregates

In z/OS V1R13, zFS can automatically recover disabled aggregates when possible in both single-system and in sysplex environments when multiple systems are running in zFS sysplex-aware mode. This is intended to eliminate the need to recover the file system manually before applications close and reopen the files to regain access to them.

zFS refresh

In z/OS V1R13, zFS will maintain existing connections to file systems while recovering from internal errors when possible. This is intended to provide less-disruptive recovery from most internal zFS problems, and designed to allow applications with open files to retry file system operations successfully after zFS recovery has been completed.

zFS client direct I/O

In z/OS V1R13, zFS processing has been redesigned to allow all members of a sysplex to perform zFS file system read and write I/O operations. This is expected to yield substantial performance gains for systems that would not have been zFS owning systems in the prior design, without performance impacts to systems that would have been zFS owning systems.

Mount/unmount granularity to prevent/warn mount overlay on a subtree

In z/OS V1R13, support in z/OS UNIX System Services is for user-level file system mounts. This provides support for limiting overall user mounts, the ability to allow these functions for specific users and groups of users, the ability to restrict which mount points a user may use, and the ability to allow user mounts only at empty mount points. Also, improved warning and failure messages are planned for some mount and unmount operations. The ability to move these mounts from systems programmers or administrators directly to users can improve usability and flexibility, and make it easier to use z/OS UNIX.

Enhance DISPLAY command showing locking information for z/OS UNIX

In z/OS V1R13, the **DISPLAY OMVS,WAITERS** command display is enhanced to show a table for file latch activity. Similar to the table for file system latches, it is also enhanced to show information about the holders, waiters, latches, file device numbers, file inode numbers, latch set identifiers, file names, and owning file systems. Additionally, filtering options are implemented for the **DISPLAY OMVS,WAITERS** command. This new function is intended to make it easier to diagnose file system-related latch contention problems.

Provide IPV4 pktinfo support

In z/OS V1R13, z/OS UNIX System Services has enhanced support by providing the capability for IPv4 UDP datagram reply packets to flow on the same interface where the request arrived when a server system has multiple home addresses with multiple routes back to the client or is using a DVIPA. This support, designed to be similar to the existing support for IPv6, is intended to allow applications to require that the response to a request be restricted to the same IPv4 address from which the request was received.

Define symbolic links for read-only filesystem for cron, mail, and uucp

This effort is intended to eliminate the manual step that clients need to perform for the mail, cron, and uucp utilities when running in a read-only root environment.

dbx hookless debug support

In prior releases, the z/OS dbx debugger for C/C++ programs requires that the compiler insert Execute (EX) instructions (known as “hooks”) so that the debugger can gain control during program execution to display information about the program and the data it processes.

In z/OS V1R13, dbx provides support for debugging programs compiled without hooks, in addition to those compiled with hooks. This support is intended to allow you to debug programs whose sizes and performance characteristics are more closely aligned with production programs.

dbx namespace phase 1 (alias and declare)

In z/OS V1R13, dbx provides support for the C++ namespace directive. Specifically, namespace members introduced into a block using the following syntax can now be directly accessed by dbx when the program is stopped within the block, as follows:

```
using namespace_name :: namespace_member;

(dbx) print namespace_member
```

Note that dbx also provides support for namespace aliases and allows the dbx user to access namespace members by fully qualifying the member names, as follows:

```
(dbx) print namespace_name::namespace_member
```

Shell and Utilities Add option on vi to edit ASCII files

In z/OS V1R13, z/OS UNIX System Services provides enhancements to the vi and ex editors to allow you to edit untagged text files and have them treated as if they contained text data using a codeset you specify. This new support allows you to override the built-in auto conversion function, making it easier to edit ASCII-encoded files under z/OS UNIX.

Provide script utility in z/OS UNIX Shells and Utilities

In z/OS V1R13, z/OS UNIX System Services provides a script command you can use to record the output of a shell session. For example, a script command could be used to create a session log file for auditing, or another output file intended to be processed later.

SMB server support for Windows 7 clients

In z/OS V1R13, the DFS SMB Server supports clients running both the 32-bit and 64-bit versions of Microsoft Windows 7 Professional, Microsoft Windows 7 Enterprise, and Microsoft Windows 7 Ultimate.

NFS Server Windows 7 support

In z/OS V1R13, NFS supports Microsoft Windows 7 with Open Text NFS Client or Open Text NFS Server installed.

1.6 DFSMS - DFSMSdfp enhancements

DFSMS is an exclusive element of the z/OS operating system. DFSMS is a software suite that automatically manages data from creation to expiration. The following elements comprise DFSMS:

- DFSMSdfp, a base element of z/OS

  DFSMSdfp provides a storage management subsystem (SMS) that allows storage administrators to control the use of storage. The storage management subsystem

provides storage groups, storage classes, management classes, and data classes that control the allocation parameters and management attributes of data sets. DFSMShsm performs space management and availability management of each data set as directed by the management class attributes of that data set. In addition, the storage group controls the allocation of the data set when DFSMShsm returns the data set to level 0 (L0) storage.

- DFSMSdss, an optional feature of z/OS
- DFSMShsm, an optional feature of z/OS
- DFSMSrmm, an optional feature of z/OS
- DFSMStvs, an optional feature of z/OS

Catalog VVDS expansion

In z/OS V1R13, the maximum usable size of the VSAM volume data set (VVDS) is increased, which will increase the architectural limit for the maximum number of SMS-managed and VSAM data sets that can reside on a single volume by a factor of 16. For most data set types, this is expected to be an increase from hundreds of thousands of data sets to millions or tens of millions of data sets per volume. It is intended to allow the number of data sets per volume to scale with extended address volume (EAV) sizes.

Catalog alias number constraint relief

In z/OS V1R13, catalog processing is enhanced to increase the number of aliases that can be defined for a user catalog. For example, if your master catalog is defined with the default record size, the maximum will be increased from approximately 3,500 single-level aliases per user catalog to approximately 250,000 or more, depending on alias sizes.

SDM RAS enhancements

In z/OS V1R13, the DFSMS System Data Mover (SDM) component has added new keywords to the ANTMINxx parmlib member. It also provides support in the **MODIFY ANTMAIN** command to help you tune Concurrent Copy operations during periods of high update activity.

Additionally, a new status filter option for the **XQUERY** command helps you to identify volumes for which performance might be causing application impacts. The PQUERY function of ANTRQST is changed to have new support to provide linkage adapter information between the primary and secondary storage controllers, and the **CQUERY** TSO/E command provides additional information about device connectivity.

XTIOT support for subsystem DCBs

In z/OS V1R13, DFSMSdfp is planned to add support to Open/Close/End of Volume to allow subsystems to use BAM DCBs and ACBs with extended TIOTs (XTIOT). This can help provide virtual storage constraint relief for address spaces that allocate a large number of data sets.

DADSM/CVAF/ device support availability

In z/OS V1R13, the DADSM and CVAF components of DFSMSdfp support concurrent service. These components allow you to dynamically update their programs without IPL. This is intended to help improve system and application availability.

In prior releases, the CATALOG, LLA, VLF, z/OS UNIX RESOLVER, TCP/IP, DFSMSrmm, and TN3270 address spaces were marked reusable. In z/OS V1R13, the DEVMAN address space is planned to be marked reusable so that restarting it does not subtract from the system's maximum number of address spaces or from the system's reserve of nonrestartable address spaces when REUSASID(YES) is specified in DIAGxx parmlib member. These changes are intended to help you improve system availability.

DFSMS OCE FREE=EOV JCL parameter

In z/OS V1R13, the system supports a new FREE=EOV keyword on the JCL DD statement to allow you to specify that each volume of a multivolume tape data set be made available for other input processing after the processing for that volume is finished. This is intended to allow overlapped processing for multivolume data sets, which can speed batch processing.

DFSMS ISMF: New sort capability for batch jobs

In z/OS V1R13, DFSMSdfp adds function to ISMF to allow you to sort saved volume lists by column and display space information in GB units. It also supports a new display for pool storage groups. This new function will make ISMF easier to use.

LISTCAT support of catalog CSI

In z/OS V1R13, Access Method Services (IDCAMS) supports a new option for the **LISTCAT LEVEL** command. This new option is designed to allow you to specify whether related component names be listed when a data set entry is listed based on the pattern specified by LEVEL. For example, if a cluster name is listed, the new option will allow you to specify whether or not the DATA and INDEX entries are also listed.

IDCAMS DELETE UCAT WTOR

In z/OS V1R13, DFSMSdfp catalog processing and the IDCAMS utility are changed to issue an operator message that requires a response before allowing a user catalog to be deleted when RECOVERY is specified. This new function is designed to be enabled using new operands of the **MODIFY CATALOG** command. It is intended to help prevent inadvertent deletion of user catalogs in batch jobs using IDCAMS.

New catalog parmlib member (IGCCATxx)

In z/OS V1R13, a new parmlib member, IGGCATxx, allows you to specify a number of catalog system parameters. A new CATALOG parameter in the IEASYSxx parmlib members also allows you to specify one or more IGGCATxx parmlib members, in which you can specify the maximum number of catalog address space (CAS) user service tasks; a threshold value for how full a catalog can be made before a warning message is issued; whether functions that can be controlled using the ENABLE and DISABLE keywords of the **MODIFY CATALOG** command should be active; and the amount of primary and secondary space to be allocated for implicitly-defined VSAM volume data sets (VVDSs). This new function is intended to make it easier to maintain those catalog parameters that are not needed very early during the IPL process.

DFSMS PDSE: refresh command

In z/OS V1R13, PDSE supports two new commands to simplify the identification and recovery from some PDSE problems by allowing you to display all users of a specified PDSE, and to discard stale pages from the PDSE directory cache.

SMS: New parmlib parameter and command

In z/OS V1R13, SMS now allows users the option to turn OFF the feature that enables SMS to externalize DELETE/RENAME errors through a new parameter in the IGDSMSxx parmlib member of SYS1.PARMLIB. This is introduced to control DELETE and RENAME messages to JOBLOG and the Hardcopy Log. The default for this parameter is that DELETE/RENAME error messages will be externalized by SMS. A **SETSMS** command is also provided to allow the operator to turn this parameter ON or OFF between IPLs.

DADSM/CVAF/ device support simplification

In z/OS V1R13, the Direct Access Device Storage Manager (DADSM) component provides dynamic exit support for both the preprocessing exit (IGGPRE00) and the postprocessing exit (IGGPOST0). In addition to providing the ability to change exits without interrupting the operation of the system, dynamic exits provide the ability to run multiple exit routines in an order you specify without having to integrate exits from multiple sources and vendors.

In z/OS V1R13, DFSMS provides a generalized device support health check exit routine which will check for tape library errors during system IPL. The check routine for this health check will be "DMO_TAPE_LIBRARY_INIT_ERRORS". It will report the specifics on devices that had initialization errors during a system IPL along with an explanation of the errors and a suggested remedy for each.

Unauthorized XTIOT and DSAB above

Now z/OS V1R13 provides support for unauthorized programs to use extended task I/O tables (XTIOTs) when a captured UCB is not requested. This new function is designed to allow all programs to allocate more data sets than can be supported by TIOTs below 16 MB, and to take advantage of data set access blocks (DSABs) above 16 MB.

64-bit subspace support

In z/OS V1R13, storage management is changed to allow tasks using subspaces to access 64-bit private and 64-bit shared virtual storage without the overhead of a branch in subspace group (BSG) instruction. This is intended to help provide virtual storage constraint relief by making it easier for applications to exploit 64-bit storage and to use system services that use 64-bit storage.

Externalize OCE detected abend and reason codes

In z/OS V1R13, DFSMSdfp allows you to use a new keyword in a DEVSUPxx parmlib member to specify that descriptive text, in addition to abend codes and return codes, be provided for many Open, Close, and End of Volume errors. This will make it easier to determine the reason for these errors quickly without having to look up the messages and return codes.

SMS: Provide volume space statistics through IGDCNS call or by ISMF

The space statistics for any given volume may not be accurate if no update has been done to that volume. In z/OS V1R13, SMS will acquire the current space information for the requested volumes that do not have updated space statistics. The updated information will then be available to callers such as IGDCNS or ISMF.

DFSMS OCE: RAS enhancements

In z/OS V1R13, DFSMSdfp supports key 8 callers of OPEN to pass a DCBE address in key 9 storage, in the same way a key 9 DCB address may be provided. This is intended to simplify the programming interface for key 8 callers of OPEN. Also, during OPEN processing, DCBE control blocks are validated. In z/OS V1R13, OPEN processing is designed to add why the reason validation failed to SMF Type 14 and Type 15 records.

MULTSDN changes for “QSAM-like” concatenation

In z/OS V1R13, DFSMSdfp changes recalculate the buffer size needed for concatenated data sets for each data set in the concatenation for data sets accessed using QSAM. This is intended to avoid out-of-storage conditions that can arise for concatenated data sets having different block sizes when MULTSDN is specified in the data control block extension (DCBE).

SMS: RAS enhancements

In z/OS V1R13, DFSMSdfp SMS processing determines whether or not the SMS configuration data set (CDS) has the REUSE attribute. If it does not, it is changed from NOREUSE to REUSE automatically during activation.

Improve tape label performance for ISO/ANSI V4 labels

In z/OS V1R13, DFSMSdfp Open/Close/End of Volume processing is changed to reduce tape movement for tapes having ISO/ANSI Version 4 labels. This is intended to improve tape processing performance without requiring any application changes.

DFSMS RLS: BMF Enhancement

In z/OS V1R13, VSAM record-level sharing (RLS) supports a new storage class (STORCLAS) attribute you can use to specify whether VSAM RLS data is retained for a period of time after a data set has been closed. This is intended to free the system's resources for VSAM RLS resources quickly for data sets that are not intended to be reopened shortly after being closed, so that additional data for other data sets can be buffered quickly. IDCAMS DCOLLECT will be designed to include information about this new attribute in storage class (type SC) records. Additional enhancements for VSAM RLS buffer management algorithms to improve processing of “aged” buffers. These enhancements are expected to help improve performance when processing large VSAM RLS data sets.

1.6.1 DFSMS EAV volumes

z/OS V1R10 introduced extended address volume (EAV), which allowed DASD storage volumes to be larger than 65,520 cylinders. The space above the first 65,520 cylinders is referred to as cylinder-managed space. Tracks in cylinder-managed space use extended addressing space (EAS) techniques to access these tracks. Data sets that are able to use cylinder-managed space are referred to as being EAS-eligible. EAV support has been enhanced for both z/OS V1R11 and V1R12.

Enhanced FTP support for extended address volumes

With z/OS V1R13, extended address volumes allow for files to be stored above the 2 Gig line of a volume. This support builds on previous extended address volume support, which only supported placing SMS managed non-VSAM physical sequential extended format data sets into the extended address volume. This support is now added for the following data set types:

- Physical sequential basic
- Large format data sets
- PDS and PDSE data sets
- GDG data sets

This allows z/OS FTP to allocate and transfer to and from these data set types when they are in the extended address space of an extended address volume (above the 2 Gig line of the volume).

1.6.2 DFSMShsm enhancements

DFSMShsm is a functional component of the DFSMS family, which provides facilities for managing your storage devices. DFSMShsm is a DASD storage management and productivity tool for managing low-activity and inactive data. It relieves you from manual storage management tasks and improves DASD use by automatically managing both space and data availability in a storage hierarchy. DFSMShsm cooperates with the other products in the DFSMSdfp family to provide efficient and effective storage management.

HSM CDS backup improvements

In z/OS V1R13, DFSMShsm control data set (CDS) backup processing is enhanced when you specify that a point-in-time copy technique is to be used. This processing will be designed to begin the CDS backup function immediately instead of waiting for DFSMShsm requests to complete, and allow DFSMShsm to continue processing requests with only brief interruptions. This is intended to avoid suspending DFSMShsm request processing for the duration of CDS backups.

HSM: Space management performance

In z/OS V1R13, new DFSMShsm and DFSMSdfp function is implemented to allow you to specify that space management be performed when any volume in an out-migration storage group exceeds the utilization threshold, instead of waiting for Interval Migration processing. This new function, intended to allow you to replace Interval Migration processing, makes DFSMShsm space management more responsive while reducing the overhead that would otherwise be associated with Interval Migration processing. Also, improvements are planned for DFSMShsm processing of volume data set lists to decrease the time it takes to begin data movement after space management begins for a particular volume.

DFSMShsm: Small data set packing performance

In z/OS V1R13, DFSMShsm small data set packing (SDSP) processing is changed to improve performance.

DFSMShsm: RAS/usability items

z/OS V1AR13 includes a number of DFSMShsm usability enhancements, as follows:

- Support for BEGIN and END specifications and multiple host IDs for the ONLYIF keyword is implemented to allow you to specify groups of parameters related to one or more DFSMShsm hosts with a single ONLYIF keyword.
- A new **SETSYS** subcommand is implemented that you can use in place of the existing patch command to reduce the number of fast replication backup messages.
- A new subparameter for the **RELEASE RECALL** command is implemented that you can use to specify that DFSMShsm is to avoid recalling data sets from missing or faulty tapes, while releasing the hold on recalls from DASD.
- Additional information in the output from the **QUERY COMMONQUEUE(RECALL)** command is implemented to identify the host from which a recall originated so you can more easily cancel a recall request.
- A new patch is implemented that you can use to suppress DFSMShsm messages when no storage groups or copy pools are eligible to be processed for various space management, backup, and restore operations.

1.6.3 DFSMSdss enhancements

DFSMSdss is a direct access storage device (DASD) data and space management tool. DFSMSdss works on DASD volumes only in the z/OS environment.

Notify SysPlex when volume is copied or restored

In z/OS V1R13, the system is changed to update volume information across a Parallel Sysplex when DFSMSdss full-volume copy or restore operations and DFSMShsm Fast Replicate Backup processing complete successfully and the volume serial or VTOC location, or both, have been changed. When a new REFUCB keyword is specified in a DEVSUPxx parmlib member, this is intended to eliminate the requirement to issue **VARY** commands on

sharing systems in the sysplex when volume information has been updated by these functions.

1.6.4 DFSMSrmm enhancements

With DFSMSrmm, you can manage your removable media as one enterprise-wide library across systems and sysplexes. DFSMSrmm manages your installation's tape volumes and the data sets on those volumes. DFSMSrmm also manages the shelves where volumes reside in all locations, except in automated tape libraries.

DFSMSrmm manages all tape media, such as cartridge system tapes and 3420 reels, as well as other removable media you define to it. For example, DFSMSrmm can record the shelf location for optical disks and track their vital record status. It does not manage the objects on optical disks.

DFSMSrmm simplified monitoring and management

In z/OS V1R13, new DFSMSrmm and DFSMSdfp function is planned for the following areas:

- Allowing the system to automatically correct the volume list for multivolume tape data sets in many cases when a volume list does not include all necessary volumes or the volumes are specified out of order. This is intended to help you avoid problems when processing multivolume tape data sets without having to code a label anomaly exit.
- Allowing you to specify whether data sets are managed by expiration date or VRS policy when they are created. This will help simplify your retention policies, help to avoid batch VRS policy management, and allow you to determine how long a tape data set will be retained at the time it is created. Corresponding support is planned for the DFSMSrmm dialog, to show either the VRS retention date or the expiration date in data set and volume search results.
- DFSMSrmm provides support for tape copy applications that enables them to copy and restack tape data sets while retaining, and preventing incorrect settings for, data set attributes. Options also allow setting predictable retention periods for source data. This will simplify moving and copying tape data, particularly when implementing new tape technologies and replacing older media.
- An enhanced **SEARCHDATASET** command will allow a more efficient search of tape data set metadata based on date ranges, including relative values, SMS constructs, and catalog status. This will make it easier to identify data sets that meet those criteria.

DFSMSrmm: user requirements

In z/OS V1R13, DFSMSrmm provides more control over automatic inventory management-driven volume movement by allowing you to specify locations that are not eligible for automated movement during inventory management processing; for example, those that might otherwise be moved based on vital records specifications (VRS).

SMS: Increase retention period limit from 9999 to 93000

In z/OS V1R13, the data set retention period supported by JCL, TSO/E, DFSMShsm, and DFSMSrmm is increased from the current limit (up to 9999 days, or approximately 27 years) to 93,000 days, or approximately 254 years. This is intended to make it easier to retain data for longer periods of time.

1.6.5 DFSMSoam enhancements

The object access method (OAM) uses a class of data referred to as *objects*. An object is a named stream of bytes. The content, format, and structure of that byte stream are unknown to OAM. There are no restrictions on the data in an object. For example, an object can be a compressed scanned image or coded data. Objects are different from data sets handled by existing access methods.

DFSMS OAM: Usability and reliability enhancements

z/OS V1R13 includes several improvements for the OAM component of DFSMSdfp:

- Extending the maximum specific expiration date from the current maximum of 9999 days to 93,000 days (approximately 254 years), in addition to the existing support for NOLIMIT
- Adding wildcard support for the **`MODIFY OAM,START,STORGRP`** command to allow you to initiate OSMC storage group processing for multiple object and object backup storage groups in single commands
- Providing dynamic update capabilities to allow you to change the maximum number of tape drives OAM will allocate to a given object or object backup storage group without restarting OAM
- Enhancing the OAM media migration utility, MOVEVOL, to improve performance when moving objects from a source volume that contains a large number of OAM collections
- Shipping the OAM component trace member, CTICBR00, in the parmlib data set so that you can use parmlib concatenation to avoid having to copy it from the samplib data set to parmlib during migration to new releases of z/OS
- Enhancing SMF Type 85 records to add counter fields with higher maximum values, in addition to the existing fields in KB

DFSMSoam archiving filesystem support

The Object Access Method component of DFSMSdfp supports a storage hierarchy that includes disk, tape, and optical storage levels. In z/OS V1R13, OAM is adding support for file systems to the disk level for zSeries File Systems (zFS) and Network File Systems (NFS) file systems, in addition to the existing support for DB2-backed object storage. Supporting file systems for primary OAM object storage is planned to support storing, retrieving, deleting, movement to and from file systems to other levels in the OAM hierarchy. This support is intended to provide you new and more flexible ways to configure your OAM storage hierarchy.

1.7 z/OSMF

z/OSMF provides a framework for managing various aspects of a z/OS system through a web browser interface. By streamlining various traditional tasks and automating others, z/OSMF can help to simplify areas of system management and reduce the level of expertise needed for managing a system.

z/OSMF System Managed Storage

z/OSMF V1R13, along with z/OS V1R13, provided simplified storage administration. A new z/OSMF application makes it easier to manage pools of reserve storage volumes and add them to storage groups as needed. This application will help you take volumes from one or more reserve pools, initialize them, add them to storage groups, and update and activate a modified SMS configuration based on a policy you define.

A setup wizard to guide storage administrators in creating the policy, a graphical display to show the utilization status of your storage groups, and the ability to perform storage group management tasks from within the application are also implemented. This application is intended to reduce the time needed to manage storage groups and improve application reliability by helping you identify storage groups that are allocated above their utilization goals and making it easy to add storage to them quickly, while providing the flexibility necessary to continue to drive high storage utilization rates.

Problem Documentation Upload Utility a supported part of z/OS

In z/OS V1R13, a Problem Documentation Upload utility is added to z/OS, and will be intended to be used to transmit dumps to IBM. This utility is designed to break dumps into segments that can be transmitted in multiple data streams to help reduce data transfer time, and to support encryption. This utility is similar to the Problem Documentation Upload utility currently available for download, and is planned to have an alias entry point named MTFTPS for compatibility and to be called from the z/OSMF Incident Log function.

Note: The Problem Documentation Upload utility can currently be downloaded from:

```
http://www.ibm.com/support/docview.wss?uid=isg3T1011823&myns=z000&mynp=OCSWG
90&mync=E
```

z/OSMF SAF-based authorization (LI2082, LI2212)

z/OSMF V1R13 running on z/OS V1R13 now supports SAF-based security to manage user authorization and roles in addition to the current z/OSMF security implementation. This is intended to allow you to migrate to RACF-based or other SAF-based security for z/OSMF.

z/OSMF currency

In z/OSMF V1R13, there is support for the 64-bit versions of Microsoft Windows 7 and Internet Explorer 8. There is also support for Mozilla Firefox 3.5 and 3.6 on Microsoft Windows XP, Vista, and on both the 32-bit and 64-bit versions of Windows 7.

z/OSMF-Capacity Provisioning Manager

z/OSMF V1R13 running on z/OS V1R13 supports a Capacity Provisioning Manager application, which in turn is designed to support easier monitoring of Capacity Provisioning Manager (CPM) status.

Deployment manager - instrumentation - z/OSMF

A new software deployment function is implemented for z/OSMF V1R13, which will run on z/OS V1R13. The software deployment function provides the functions needed to create and deploy a copy, or clone, of an existing SMP/E-installed software image. This includes IBM software installed using ServerPac, CBPDO, or fee-based installation offerings, and ISV or client software.

The function helps to create and distribute copies of system software, including target libraries, distribution libraries, SMP/E zones, and related data sets you identify. Software deployment is designed as a z/OSMF application and is intended to make it easier to manage software images by simplifying and standardizing these deployment processes.

z/OSMF application linking

z/OSMF V1R13 is designed to allow better integration between tasks by allowing z/OSMF applications to link to other applications using a URL. This is intended to help simplify complex series of tasks by allowing one task to link to another in context.

Batch modernization - REST style submit API

In z/OSMF V1R13, a new RESTful API allows applications to submit batch jobs, check status, and retrieve results using the HTTP and HTTPS protocols. This is intended to help provide improved access to z/OS batch processing using industry-standard interfaces.

1.8 z/OS Communications Server

z/OS Communications Server provides a set of communications protocols that support peer-to-peer connectivity functions for both local and wide-area networks, including the most popular wide-area network, the Internet. z/OS Communications Server also provides performance enhancements that can benefit a variety of TCP/IP applications.

z/OS Communications Server provides both SNA and TCP/IP networking protocols for z/OS. The SNA protocols are provided by IBM VTAM® and include Subarea, Advanced Peer-to-Peer Networking, and High Performance Routing protocols.

CSSMTP extended long retry

The CSSMTP server provided by z/OS Communications server sends bulk e-mail from the JES spool. In z/OS V1R13, when mail servers fail to respond for a long period of time, the SMTP server will release memory and JES resources, while continuing to retry operations at longer intervals. This is intended to allow extended retry processing less overall system impact when mail servers are unresponsive.

System resolver autonomic quiescing of unresponsive name servers

The z/OS system resolver was enhanced in z/OS V1R12 to detect unresponsive name servers and warn the operator with messages. In z/OS V1R13 this support is taken a step further so that the system resolver will automatically stop using name servers that become unresponsive, and then automatically start using them again when they recover. This enhances network availability for processes that rely on name resolution services, because long time-out periods for unresponsive name servers are avoided.

SNA APPN and EE enhancements

These enhancements provide improved APPN routing resilience for Enterprise Extender firewall-friendly connectivity.

Configuration Assistant management of multiple releases

The IBM Configuration Assistant for z/OS Communication Server is enhanced so that a single instance of the Configuration Assistant can configure multiple releases of z/OS Communications Server. The z/OS V1R13 configuration assistant will be able to configure both z/OS V1R12 and z/OS V1R13 Communications Server. This allows you to configure all of the systems in a mixed-release environment from a single instance of the Configuration Assistant running under z/OSMF.

Configuration Assistant common configuration of multiple stacks

The IBM Configuration Assistant for z/OS Communications Server is enhanced to allow a policy rule to be defined one time for multiple stacks. This permits more efficient policy configuration for multiple systems, without having to individually define every policy rule for every stack.

FTP support for large format data sets

z/OS Communications Server adds support to FTP for DFSMS large format data sets. With this support FTP can allocate large format data sets that can contain more than 65,525 tracks, or more than 2 gigabytes of data, without requiring the use of SMS managed data sets.

Provide the d telnet command to display telnet servers

Communications Server provides a **`d telnet`** command to display the list of telnet servers, similar to the **`d tcpip`** command.

NMI enhancements - support for Resolver data

Communications Server will create a network management interface for the system resolver, which will support retrieval of the resolver configuration file and the contents of the global TCPIP.DATA file, if one is in use.

FTP support for password phrases

The FTP server provided with z/OS Communications Server is updated to accept and work with password phrases. This enables users of this server, including applications written to it, to benefit from the improved security provided by password phrases.

Expanded intrusion detection services

Communications Server intrusion detection technology is enhanced to add support for IPv6 traffic and additional attack types including data hiding and out-of-sequence packet denial-of-service attacks. This provides IPv6 intrusion detection security equivalent to IPv4 and helps you prevent certain error situations and denial of service attacks on Communications Server from causing system-wide storage constraint situations.

Password Phrase support for TN3270

The TN3270E server provided with z/OS Communications Server is updated to accept and work with password phrases on the solicitor panel. This enables installations that use the TN3270E solicitor panel to exploit the improved security provided by password phrases.

1.9 Infoprint Server

Infoprint Server is an optional feature of z/OS that uses z/OS UNIX System Services. This feature is the basis for a total print serving solution for the z/OS environment. It lets you consolidate your print workload from many servers onto a central z/OS print server.

Infoprint Server delivers improved efficiency and lower overall printing cost with the flexibility for high-volume, high-speed printing from anywhere in the network. With Infoprint Server, you can reduce the overall cost of printing while improving manageability, data retrievability, and usability.

Infoprint Server for z/OS support for a time-in-queue value

In z/OS V1R13, a number of usability enhancements are implemented for Infoprint Server for z/OS.

Infoprint Central

Infoprint Central adds the elapsed time to print jobs that have been in the queue to the display, and allows you to search for jobs that have been in the queue for a specified period.

Infoprint Server provides support to limit selection of work by lines

Infoprint Server for z/OS provides users with the option to exclude output from being selected by the number of lines that would potentially print.

Infoprint Server for z/OS enhanced sendmail function

In z/OS V1R13, a number of usability enhancements are implemented for Infoprint Server for z/OS:

- An enhanced sendmail function to use EMAIL headers when they are included in the data
- Ability to specify that text files be included in the bodies of emails rather than as attachments when using sendmail
- Ability to specify that email addresses contained in RFC 2822 headers be carried into the text of emails created by the sendmail function when they appear within the first 32 records of the data set

Infoprint Server support of a secondary JES2

In z/OS V1R13, Infoprint Server provides support for printing jobs managed using a secondary JES2 subsystem running under a primary JES2 subsystem. This will simplify printing for jobs run under a secondary JES2 by eliminating the need to route job output to the primary JES2 subsystem or to another JES2 or JES3 node for printing.

1.10 Security enhancements

IBM Security Server is a set of features in z/OS that provide security.

Security Server provisions include:

- Controlling the access of users (user ID and password) to the system
- Restricting the functions that an authorized user can perform on the systems' data files and programs

Many installations use a package called Security Server, which is commonly referred to by the name of its most well-known component, RACF. Resource Access Control Facility (RACF) is a component of Security Server. It controls access to all protected z/OS resources. RACF protects resources by granting access only to authorized users of the protected resources, and retains information about the users, resources, and access authorities in specific profiles.

The following list highlights the security components of z/OS that are collectively known as Security Server:

- Lightweight Directory Access Protocol (LDAP) Server

  The Lightweight Directory Access Protocol (LDAP) Server is based on a client/server model that provides client access to an LDAP server. An LDAP directory provides an easy way to maintain directory information in a central location for storage, update, retrieval, and exchange.

- z/OS Firewall Technologies

  z/OS Firewall Technologies is an IPV4 network security firewall program for z/OS. In essence, the z/OS firewall consists of traditional firewall functions and support for virtual private networks. The inclusion of a firewall means that the mainframe can be connected directly to the Internet if required without any intervening hardware, and can provide the

required levels of security to protect vital company data. With VPN technology, securely encrypted tunnels can be established through the Internet from a client to the mainframe.

- Network Authentication Service for z/OS

  Network Authentication Service for z/OS provides Kerberos security services.

- Enterprise Identity Mapping

  Enterprise Identity Mapping (EIM) offers a new approach to enable inexpensive solutions to easily manage multiple user registries and user identities in an enterprise.

- PKI Services

  PKI Services allows you to establish a public key infrastructure and serve as a certificate authority for your internal and external users, issuing and administering digital certificates in accordance with your own organization's policies.

- Resource Access Control Facility

  Resource Access Control Facility (RACF) is the primary component of the Security Server; it works closely with z/OS to protect vital resources.

Enhanced ITDS for z/OS TDBM back end and bulk load utility

In z/OS V1R13, the IBM Tivoli® Directory Server for z/OS(LDAP) DB2-based TDBM back end and bulk load utility are enhanced to support 64-bit addressing. This enhanced TDBM back end, when used with DB2 10 for z/OS (5605-DB2), is intended to improve scalability of IBM Tivoli Directory Server for z/OS for large LDAP deployments. In addition to this TDBM enhancement, a bulk load utility capable of executing in 64-bit addressing mode is planned to facilitate loading large LDAP directory databases.

Server support of paged and sorted search results

In z/OS V1AR13, the IBM Tivoli Directory Server element supports paged and server-side sorting of search results as described by RFC2696 and RFC2891. Paged sorting capability allows LDAP clients to specify that they should be passed a subset of search results (called a “page”) and successive pages one at a time, rather than receiving an entire set of results list.

Server-side sorting capability enables LDAP clients to receive sorted search results based on a list of criteria, where each criteria represents a sort key. For example, a client application might want to sort the list of employees at a particular work location by surname, common name, and telephone number. Rather than building two search lists, a client application can build a single search list for the server to use so the sorted list can be returned. This is intended to provide sorting capability for client applications that do not have available native sort functions, and it can help improve performance.

z/OS Network Authentication Service updated

In z/OS V1R13, Network Authentication Service (NAS) is updated to support the functions described by RFC4537, thereby helping to ensure the continued interoperability between z/OS and other industry standard Kerberos implementations. This RFC defines an encryption-type negotiation extension to the Kerberos protocol, to enable clients and servers to use stronger or different encryption mechanisms than are supported by the KDC. This is intended to help improve the security and interoperability of applications that use Kerberos and the GSS-API on z/OS and other platforms.

ITDS for z/OS SHA-2 password hashing

In z/OS V1R13, the IBM Tivoli Directory Server for z/OS (LDAP) supports SHA-2 hashing for user passwords stored in the LDBM, TDBM, and CDBM back ends. This is intended to help address the need for stronger hashing and cryptographic algorithms and enhance

interoperability with distributed IBM TDS, open LDAP, and other LDAP servers. It is also intended to meet the National Institute of Standards and Technology (NIST) policy for the use of hash functions. For more information, see:

http://csrc.nist.gov/groups/ST/hash/policy.html

This extension supports SHA-2 (SHA224, SHA256, SHA384, and SHA512) and salted SHA-2 (SSHA224, SSHA256, SSHA384, and SSHA512) hashing of user password attributes. This support is planned to use the persistent PKCS#11 token in ICSF to perform the hashing.

z/OS PKI Services and the optional use of DB2 for z/OS

In z/OS V1AR13, z/OS PKI services allows you to use DB2 for Object Storage and for the Issued Certificate List. The optional use of DB2 by z/OS PKI is designed to allow you to take advantage of the scalability of DB2 for large-scale certificate deployments, and also take advantage of DB2 designs for high availability, backup, and recovery.

z/OS PKI Services browser currency

In z/OS V1R13, z/OS PKI Services adds support to enable Mozilla-based web browsers on Windows and Linux platforms to use smart cards when generating certificates. It also enables Microsoft Internet Explorer 6, Internet Explorer 7, and Internet Explorer 8 to use an updated PKI application that includes its own ActiveX controls, which allows users to install renewed certificates.

z/OS PKI Services support for larger certificate revocation lists (CRLs)

In z/OS V1 R13, PKI services supports certificate revocation lists (CRLs) larger than 32 K (32,767) bytes in size. This is intended to help support CRL distribution point environments such as those using LDAP for large certificate hosting environments, and to improve the flexibility of z/OS PKI Services.

z/OS System SSL ECC certificate and key agreement

In z/OS V1R13, System SSL extends its Elliptical Curve Cryptography (ECC) support to enable the creation of X.509 V3 certificates using the ECDSA and ECDH algorithms. This is planned to enable you to create these certificates in key database files or ICSF PKCS#11 tokens, and to allow applications that use certificate support through the Certificate Management Services (CMS) API to create ECC-style certificates.

z/OS System SSL ECC key reference by ICSF key label (phase 2)

In z/OS V1 R13, System SSL supports Elliptical Curve Cryptography (ECC) certificates residing in SAF key rings with their private keys stored in the ICSF public key data set (PKDS). System SSL uses the private keys in secure digital signature generation operations available through Crypto Express3 Coprocessor (CEX3C) cards on IBM zEnterprise and IBM System z10 servers.

ITDS for z/OS group specific search limits

In z/OS VR13, the IBM Tivoli Directory Server (LDAP) element is extended to allow you to specify flexible search and time limits for LDAP groups. This new support is designed to enable LDAP administrators to balance LDAP server-enforced limits and the time needed by specific applications.

RACF RRSF TCP/IP support

In z/OS V1R13, RRSF is designed to support the use of TCP/IP connections, in addition to the current support for SNA Advanced Peer-to-Peer Communications (APPC). When used with TCP/IP, RRSF is designed to use Application-Transparent Transport Layer Security

(AT-TLS) to authenticate peer RRSF nodes and encrypt replication traffic. AT-TLS provides encryption algorithms thought to be stronger than those available using APPC. A sample rule that specifies the strongest available encryption method is provided. The use of TCP/IP is intended to help improve usability, simplify network configuration, and improve the security of RACF data shared between RACF nodes in the RRSF network.

RACF: Add new field to KERB segment of REALM class profiles

In z/OS V1R13, Network Authentication Service (NAS) supports checking IP addresses in tickets for Kerberos, as described by RFC4120. A new CHECKADDRS field in the KERB segment of the KERBDFLT profile in the REALM class allows you to specify whether address checking should be enabled or disabled.

RACF identity propagation ID mapping support

z/OS Security Server are SAF and RACF callable service have implemented changes that are intended to help exploiters more easily use the SAF and RACF identity propagation services. Identity propagation provides the capability for exploiting applications and subsystems of associating an auditing distributed user ID with a z/OS User ID. This helps to improve the accountability resource access requests on z/OS.

This enhancement enables an exploiter to decouple the building of a user's security context and mapping a distributed identity to a z/OS User ID into two steps to help to facilitate the use of the ID Propagation services into a wider range of applications. With today's focus on accountability and auditing, the identity propagation services provided by RACF are key in helping to you to meet your compliance goals. CICS is exploiting this to improve the accountability of transactions.

ITDS for z/OS Kerberos authentication interoperability

In z/OS V1R13, the IBM Tivoli Directory Server (ITDS) element is extended to enable Kerberos binds to be processed by Microsoft's Active Directory Server. This support is intended to improve the interoperability between z/OS applications that utilize the ITDS client services and Kerberos authentication in environments where Active Directory is being utilized.

z/OS System SSL ECC support for TLS

In z/OS V1R13, System SSL is extends its use of Elliptic Curve Cryptography (ECC) to enable TLS V1.0 and TLS V1.1 handshakes using ECC cipher suites and digital certificates during secure connection negotiations as described by RFC 4492.

RACF: Hardware ECC support for RACDCERT

In z/OS V1R13, RACF support is implemented for generating Elliptical Curve Cryptography (ECC) secure keys using the Crypto Express3 Cryptographic Coprocessors (CEX3C) available for zEnterprise servers. New keywords on the **RACDCERT** command are designed to allow you to specify that an ECC key be stored in the ICSF public key data set (PKDS). Corresponding hardware ECC key support is supported for PKI Services. This new support is intended to allow you to expand your use of certificates with ECC keys protected by hardware.

Network Address Translation traversal support

Internet Key Exchange Version 2 (IKEv2) is the latest version of the Internet Key Exchange (IKE) protocol specified by RFC 4306. It was added to Communications Server V1R12. In z/OS V1R13 Communications Server adds Network Address Translation (NAT) traversal support over IKEv2, for IPv4 only, to facilitate migration to IKEv2 by clients who require NAT functionality.

Advanced CRL support for IKE/NSSD

In z/OS V1R13, Advanced CRL support for IKE/NSSD is provided.

Enhanced Security for binding to DVIPAs

Previous to z/OS V1R13, SAF profiles can be used to control which user IDs can create and destroy VIPARANGE DIVPAs. The existing profile support covers all VIPARANGE DVIPAs.

In z/OS V1R13, this capability is extended to allow you to specify authorization for specific ranges of VIPARANGE DVIPAs, or for individual VIPARANGE DVIPA addresses.

ITDS for z/OS administration group support

In z/OS V1R13, the IBM Tivoli Directory Server for z/OS (LDAP) allows LDAP administrators to delegate LDAP administrative authority. This function is designed to allow the LDAP administrator to define an administrative group, add one or more distinguished names to that group, and assign one or more administrative roles to each user. This is intended to provide more flexibility in LDAP administration, help improve auditability, and help improve security by allowing for separation of duties and eliminating reasons for identity sharing.

2

z/OS V1R13 installation

This chapter describe changes in installation and migration to be aware of when you are responsible for migrating from z/OS V1R10, z/OS V1R11, or z/OS V1R12 to z/OS V1R13. It also describes the tasks needed to prepare for the installation of z/OS V1R13, including ensuring that driving system and target system requirements are met and that coexistence requirements are satisfied. There are new migration actions introduced in z/OS V1R13.

The chapter focuses on identifying new migration actions that must be performed for selected elements when migrating to z/OS V1R13, as follows:

- Planning for z/OS V1R13 installation
- Installing z/OS V1R13 with ServerPac or CBPDO
- Driving system requirements
- JES2, JES3, and SDSF considerations
- Element and feature changes
- Preventative service planning (PSP) buckets
- SMP/E V3R6 overview
- Coexistence, migration, and fallback considerations
- Migration actions for z/OS V1R11 and z/OS V1R12
- BCP migration actions before and after the first IPL

2.1 z/OS V1R13 installation

Several IBM packages are available for installing z/OS. Some are entitled with the product (as part of your z/OS license, at no additional charge). Other packages are available for an additional fee.

The program number for z/OS V1R13 is 5694-A01. Use this program number when ordering z/OS V1R13 ®.

z/OS V1R13 became generally available on September 30, 2011 through ServerPac, CBPDO and SystemPac. When ordering z/OS V1R13, ensure that you order all the optional features that you were licensed for in previous releases of z/OS.

Ordering for z/OS V1R12 CBPDO and ServerPac ended October 2011.

Note: z/OS must be IPLed in z/Architecture mode.

2.2 Planning for z/OS V1R13 installation

This section gives an overview of the tasks needed to prepare for the installation of z/OS V1R13. The tasks include:

- Consulting the documentation describing the installation and migration considerations.
- Reviewing the Preventative Service Planning (PSP) buckets for installation-related information.
- Ensuring that driving system and target system requirements are met.
- Installing the required coexistence and fallback service on z/OS V1R11 and z/OS V1R12 systems that will coexist with z/OS V1R13 systems.

Note: For an overview of z/OS, and to help you plan for the installation of z/OS V1R13, consult the following documentation for z/OS V1R13:

- *z/OS Migration*, GA22-7499
- *z/OS Planning for Installation*, GA22-7504

Before installing z/OS V1R13, review the Preventive Service Planning (PSP) Bucket for z/OS V1R13 (upgrade ZOSV1R13). This includes the subset ZOSGEN, which contains general information, and also includes subsets for the z/OS elements.

The z/OS V1R13 PSP bucket can contain installation-related information, such as changes to the z/OS Program Directory since the document was written.

Identifying server requirements

z/OS V1R13 runs on the following IBM servers:

- IBM System z10 Enterprise Class (z10 EC)
- IBM System z10 Business Class (z10 BC)
- IBM System z9® Enterprise Class (z9 EC), formerly IBM System z9 109 (z9-109)
- IBM System z9 Business Class (z9 BC)
- IBM eServer™ zSeries 990 (z990)

- IBM eServer zSeries 890 (z890)
- IBM eServer zSeries 900 (z900)
- IBM eServer zSeries 800 (z800)
- IBM eServer zSeries z196

VM guest considerations: z/OS can run as a guest of z/VM. The requirements are:

- For z9 EC or z9 BC servers, the z/VM PTFs for the following APARs must be installed:
  - On z/VM V5 (5741-A05): PTFs for APARs OA11650, VM63646, VM63721, VM63722, VM63740, VM63743, and VM63744.
  - On z/VM V4R4 (5739-A03): PTFs for APARs VM63124, VM63646, VM63721, VM63740, and VM63743. Also, if z/VM is using QDIO Guest LAN to deploy guest LANs/VSWITCHs using VLANs, the PTF for APAR VM64359 must be installed on the z/VM V4R4 system.
- For z890 or z990 servers, the z/VM PTF for APAR VM63124 must be installed. Also, if z/VM is at V4R4 using QDIO Guest LAN to deploy guest LANs/VSWITCHs using VLANs, the PTF for APAR VM64359 must be installed on the z/VM V4R4 system.
- There are no special requirements for z800 or z900 servers.

2.3 Installing z/OS V1R13

As mentioned, there are several IBM packages available for installing z/OS. Some packages are entitled with the product (as part of your z/OS license, at no additional charge), and others are available for an additional fee.

You might find that sharing system libraries or cloning an already installed z/OS system is faster and easier than installing z/OS with an IBM installation package such as ServerPac. You do not have to wait for the order to be processed and the package delivered, and you do not have to use the CustomPac Installation Dialog twice. Sharing the system libraries (logical SYSRES volume) may also save DASD and support costs because you only need to install service (or additional products) one time.

However, before sharing or cloning z/OS, you must have a license for each z/OS operating system that you run. If you do not have the appropriate license or licenses, you must contact IBM. Any sharing or cloning of z/OS without the appropriate licenses is not an authorized use of such programs.

2.3.1 Installing with ServerPac

ServerPac is an entitled software delivery package consisting of products and service for which IBM has performed the SMP/E installation steps and some of the post-SMP/E installation steps. To install the package on your system and complete the installation of the software it includes, you use the CustomPac Installation Dialog. Both z/OS and the products that run on it are available by way of ServerPac.

CustomPac dialog

The CustomPac Installation Dialog generates tailored installation jobs and saves detailed definitions of volume, catalog, and data set configurations, which can be tailored, saved, and merged to install subsequent ServerPacs. The CustomPac Installation Dialog is the same dialog that is used for all the CustomPac offerings, including SystemPac (dump-by-data-set

format), IBM ProductPac®, and RefreshPac. For more information about CustomPac fee offerings, see:

http://www.ibm.com/services/custompac

If you plan to install a ServerPac, review the ServerPac PSP bucket. In addition to reviewing the z/OS V1R13 PSP bucket, review the hardware PSP buckets that are applicable for your installation. *z/OS Migration*, GA22-7499, documents the hardware PSP buckets.

Installing z/OS V1R13 using ServerPac has the following documentation provided with the ServerPac order:

- *ServerPac: Using the Installation Dialog*, SA22-7815
- The custom-built installation guide “ServerPac: Installing Your Order” (there is no order number; it is custom-built to your order).

2.3.2 Installing with CBPDO

If you plan to install the CBPDO deliverable for z/OS V1R13, you must delete the elements that are withdrawn as of z/OS V1R13 from the target system, as described in the z/OS V1R13 Program Directory. The sample job CLNOS390 is provided to delete the FMIDs for the withdrawn elements. The z/OS V1R13 Program Directory provides instructions about running sample job CLNOS390. The obsolete libraries, paths, and DDDEFs for the deleted elements must be removed after the withdrawn elements have been deleted by running the CLNOS390 sample job.

z/OS Migration, GA22-7499, identifies the deleted libraries, paths and DDDEFs. If you plan to install z/OS V1R13 using CBPDO, review *z/OS Program Directory*, GI10-0670.

Additional documentation:

- *z/OS Introduction and Release Guide*, GA22-7502
- *z/OS Licensed Program Specifications*, GA22-7503
- *z/OS MVS Planning: Operations*, SA22-7601
- *z/OS MVS Initialization and Tuning Reference*, SA22-7592
- *z/OS UNIX System Services Planning*, GA22-7800

2.4 Driving system requirements for z/OS V1R13

The *driving* system is the system image (hardware and software) that you use to install the target system. The *target* system is the system software libraries and other data sets that you are installing. You log on to the driving system and run jobs there to create or update the target system. After the target system is built, it can be IPLed on the same hardware (same LPAR or same processor) or different hardware than that used for the driving system.

If your driving system will share resources with your target system after the target system has been IPLed, be sure to install applicable coexistence service on the driving system before you IPL the target system.

2.4.1 Driving system components requirements

Driving system requirements for installing z/OS by way of ServerPac or dump-by-data-set SystemPac are listed here:

Operating system Use any of the following:

- z/OS V1R11 or later.
- The Customized Offerings Driver V3 (5751-COD).

TSO/E session A TSO/E session on the IPLed system must be established using a locally-attached or network-attached terminal.

Proper authority Use the RACFDRV installation job as a sample of the security system definitions required so that you can perform the installation tasks.

Proper security To install the UNIX files, the following requirements must be met:

- The user ID you use must be a superuser (UID=0) or have read access to the BPX.SUPERUSER resource in the RACF FACILITY class.
- The user ID you use must have read access to FACILITY class resources BPX.FILEATTR.APF, BPX.FILEATTR.PROGCTL, and BPX.FILEATTR.SHARELIB (or BPX.FILEATTR.* if you choose to use a generic name for these resources). The commands to define these FACILITY class resources are in SYS1.SAMPLIB member BPXISEC1.
- Group IDs uucpg and TTY, and user ID uucp, must be defined in your security database. These IDs must contain OMVS segments with a GID value for each group and a UID value for the user ID. (For ease of use and manageability, define the names in uppercase.)

OMVS A/S active For ServerPac only (not SystemPac), an activated OMVS address space with z/OS UNIX kernel services operating in full function mode is required.

SMS active The Storage Management Subsystem (SMS) must be active to allocate z/OS UNIX file systems (HFS or zFS) and PDSE data sets, whether they are SMS-managed or non-SMS-managed. Also, the use of z/OS UNIX file systems (HFS or zFS) is supported only when SMS is active in at least a null configuration, even when the data sets are not SMS-managed.

LE The CustomPac Installation Dialog uses the Language Environment (LE) run-time library, SCEERUN. If SCEERUN is not in the link list on the driving system, you must edit the ServerPac installation jobs to add it to the JOBLIB or STEPLIB DD statements.

Do not specify the following Language Environment run-time options as nonoverrideable (NONOVR) in the CEEDOPT CSECT: ALL31, ANYHEAP, BELOWHEAP, DEPTHCONDLIMIT, ERRCOUNT, HEAP, HEAPCHK, HEAPPOOLS, INTERRUPT, LIBSTACK, PLITASKCOUNT, STACK, STORAGE, THREADHEAP, THREADSTACK.

CustomPac The CustomPac Installation Dialog is needed if you are installing a ServerPac or dump-by-data-set SystemPac for the first time. Install the CustomPac Installation Dialog on your driving system. See “ServerPac: Using the Installation Dialog or SystemPac: CustomPac Dialog Reference” for instructions. For subsequent orders you do not need to reinstall the dialog.

IBM ships dialog updates with each order. Check the PSP bucket for possible updates to the CustomPac Installation Dialog. For ServerPac,

the upgrade is ZOSV1R13 and the subset is SERVERPAC. For SystemPac dump-by-data-set orders, the upgrade is CUSTOMPAC and the subset is SYSPAC/DBD.

Service level

To install service on the target system that you are building, your driving system must minimally meet the driving system requirements for CBPDO Wave 1 and must have the current (latest) levels of the program management binder, SMP/E, and HLASM. The service jobs generated by the CustomPac Installation Dialog use the target system's (and therefore current) level of the program management binder, SMP/E, and HLASM.

If you choose to use your own jobs, model them after the jobs provided by ServerPac or dump-by-data-set SystemPac by adding STEPLIB DD statements to access MIGLIB (for the program management binder and SMP/E) and SASMMOD1 (for HLASM). Be sure that the SASMMOD1 and SYS1.MIGLIB data sets are APF authorized.

Another way to install service is from a copy of your target system.

SMP/E

For the SMP/E ++JAR support, if your ServerPac order contains any product that uses the ++JARUPD support introduced in SMP/E V3R2 (which is the SMP/E in z/OS V1R5), then your driving system requires IBM SDK for z/OS, Java 2 Technology Edition, V1 (5655-I56) at SDK 1.4 or later.

zFS

zFS must be configured properly. If you will use a zFS for installation, then you must be sure that the zFS has been installed and configured, as described in *z/OS Distributed File Service zSeries File System Administration*, SC24-5989.

Internet delivery

If you intend to receive your ServerPac order by way of the Internet, you need the following.

- SMP/E V3R5, which is in z/OS V1R10, V1R11, and V1R12.
- ICSF configured and active so that it can calculate SHA-1 hash values to verify the integrity of data being transmitted. If ICSF is not configured and active, SMP/E calculates the SHA-1 hash values using an SMP/E Java application class, provided that IBM SDK for z/OS, Java 2 Technology Edition, V1 (5655-I56) or later is installed.

  The ICSF method is likely to perform better than the SMP/E method.
- A download file system; your order is provided in a compressed format and is saved in a download file system. The size of this file system should be approximately twice the compressed size of your order to accommodate the order and workspace to process it.
- Firewall configuration; if your enterprise requires specific commands to allow the download of your order using FTP through a local firewall, you must identify these commands for later use in the CustomPac Installation Dialog, which manages the download of your order.
- The proper dialog level; if you are using a CustomPac Installation Dialog whose Package Version is less than 17.00.00, you must migrate the dialog to this level or later. You can determine if you have the correct dialog level by looking for the text “This dialog supports electronic delivery.” at the bottom of panel CPPPPOLI. If your dialog is not at the minimum level, follow the migration

scenarios and steps described in *ServerPac: Using the Installation Dialog*.

DASD storage requirements for z/OS V1R13

If you are migrating to z/OS V1R13 from z/OS V1R12 or you will have a different product set than your previous release, you may see increased need for DASD. How much more depends on what levels of products you are running.

Note: The DASD required for your z/OS system includes all elements, all features that support dynamic enablement, regardless of your order, and all unpriced features that you ordered.

This storage is in addition to the storage required by other products you might have installed. All sizes include 15% freespace to accommodate the installation of maintenance The total storage required for z/OS data sets is listed in the space tables in the z/OS Program Directory.

Table 2-1 lists the approximate total storage required for all the target data sets.

Table 2-1 Target data sets - approximate total storage required

	V1R11	V1R12	V1R13
Target	6400	5891	5868
DLIB	9200	8599	8941
File system	3100	3100	3100

2.4.2 Customized Offerings Driver V3 (5751-COD)

The Customized Offerings Driver V3 (5751-COD) is an entitled driving system you can use if the following situations exist:

1. You do not have an existing system to use as a driving system.
2. Or your existing system does not meet driving system requirements and you do not want to upgrade it to meet those requirements. This driver is a subset of a z/OS V1R11 system.

The Customized Offerings Driver is in DFSMSdss dump/restore format and supports 3390 triple-density or higher DASD devices. The Customized Offerings Driver requires a locally attached non-SNA terminal and a system console from the IBM (or equivalent) family of supported terminal types: 317x, 319x, 327x, or 348x. An IBM (or equivalent) supported tape drive is also required to restore the driver.

The Customized Offerings Driver is intended to run in single-system image and monoplex modes only. Its use in multisystem configurations is not supported. The Customized Offerings Driver is intended to be used only to install new levels of z/OS using ServerPac or CBPDO, and to install service on the new software until a copy (clone) of the new system can be made. The use of the Customized Offerings Driver for other purposes is not supported.

Note: IBM does not support the use of the Customized Offerings Driver to run any production workload.

The Customized Offerings Driver includes a hierarchical file system and the necessary function to use Communications Server (IP Services), Security Server, and the system-managed storage (SMS) facility of DFSMSdfp. These elements are not customized. However, existing environments can be connected to, and used from, the Customized Offerings Driver system.

2.5 Elements changed in z/OS V1R13

The following elements are changed in z/OS V1R13:

- BCP
  - BCP Unicode, Program Management Binder, BCP Capacity Provisioning
- C/C++ without Debug Tool
- Common Information Model (CIM)
- Communications Server
- Cryptographic Services
  - ICSF, PKI Services, System SSL
- DFSMS
- Distributed File Service (DFS) and zFS
- HCD and HCM
- IBM Tivoli Directory Server
- Infoprint Server
- Integrated Security Services Network Authentication Service
- JES2, JES3
- Language Environment
- Metal C Runtime Library
- NFS
- RMF
- Run-Time Library Extensions
- SDSF
- Security Server RACF
- SMP/E (equivalent to SMP/E V3.6)
- TSO/E
- z/OS Security Level 3
  - System SSL, Network Authentication Service, IBM Tivoli Directory Server
- z/OS UNIX System Services Application Services

2.5.1 JES2, JES3, and SDSF

Migrate to the JES2 or JES3 that comes comprehensively tested with z/OS V1R13 at the same time you migrate to the rest of z/OS V1R13, or as soon as possible thereafter. In this way, you benefit directly from the new functions in the z/OS V1R13 level of JES2 or JES3 and enable other elements and features to benefit from this level.

Using z/OS V1R13 with lower releases of JES2 and JES3

Because such a migration is not always practical, certain prior levels of JES2 and JES3 are supported with z/OS V1R13 so that you can stage your migration to z/OS V1R13 (that is, migrate your JES2 and JES3 later).

Allowable BCP, JES2, and SDSF combinations

With z/OS, the JES levels supported by a given release are the same as the JES levels that may coexist in the same multi-access spool (MAS) or multisystem complex with the JES

delivered in that z/OS release. In addition, starting with z/OS V1R9, the SDSF release level must be the same as the JES2 release level if using JES2 and SDSF.

Table 2-2 lists the JES2 and SDSF levels allowed with z/OSV1R13 BCP.

Table 2-2 JES2 and SDSF levels

BCP release	JES2 release allowed	SDSF release allowed
z/OSV1R13	z/OSV1R11	z/OSV1R11
	z/OSV1R12	z/OSV1R12
	z/OSV1R13	z/OSV1R13

Allowable BCP, JES3, and SDSF combinations

With z/OS V1R10 and later releases, SDSF is supported with JES3. Table 2-3 lists the JES2 and SDSF levels allowed with z/OSV1R13 BCP.

Table 2-3 JES3 and SDSF levels

BCP release	JES3 release allowed	SDSF release allowed
z/OSV1R13	z/OSV1R11	z/OSV1R11
	z/OSV1R12	z/OSV1R12
	z/OSV1R13	z/OSV1R13

2.5.2 Cryptographic Services

Cryptographic Services ISCF is changing in z/OS V1R13, and the ICSF FMID is HCR7780. The FMID HCR7780 is also available as a web deliverable known as Cryptographic support for z/OS V1R10-R12. It is available on the web and can downloaded here:

`http://www-03.ibm.com/systems/z/os/zos/downloads/`

2.6 Preventative service planning (PSP) buckets

z/OS and most products that run on it provide files containing information that becomes available after the product information is published. These files are kept on the IBM RETAIN® system and are also available using IBMLink. These files are called preventive service planning (PSP) “buckets”, or just “PSPs.”

PSP buckets are available to clients at:

`http://www14.software.ibm.com/webapp/set2/psp/srchBroker`

The PSP buckets are identified by an upgrade identifier, and specific parts of them are called subsets. Each upgrade contains information about a product. Subsets contain information about specific parts of a product. For example, the z/OS PSP bucket has subsets for the z/OS elements, such as BCP, JES2, ServerPac, and others.

Note: To simplify finding the appropriate PSP bucket, search the Technical Help Database for Mainframe Preventive Service Planning Buckets:

http://www14.software.ibm.com/webapp/set2/psp/srchBroker

Change index

At the beginning of each PSP bucket is a change index. For each subset, the change index identifies the date of the latest entries in each section. You can quickly determine whether there are new entries you must read by checking the change index.

Software upgrades

For software upgrades for ServerPac and CBPDO installations, refer to the z/OS Program Directory. For software upgrades for SystemPac installations, the upgrade is CUSTOMPAC and the subsets are SYSPAC/FVD (for full volume dump format) and SYSPAC/DBD (for dump-by-data-set format).

A software upgrade installs only system software and related data sets (such as distribution and target libraries, SMP/E CSI data sets, and sample libraries). It does not create the set of new operational data sets required to IPL (such as page data sets, system control files, and a master catalog). With a software upgrade, all operational data sets are assumed to already exist and to be usable by the new level of software installed.

The upgrade for the z/OS V1R13 PSP bucket is ZOSV1R13. z/OS uses descriptive element names instead of FMIDs for the subsets. This reduces the number of PSP buckets that must be reviewed, because most elements are composed of multiple FMIDs.

There are subsets in the ZOSV1R13 upgrade for general topics (ZOSGEN), and for the ServerPac deliverable (SERVERPAC) that should also be reviewed. All PSP upgrades and subset IDs for z/OS V1R13 are listed in the z/OS Program Directory. However, for the non-exclusive elements, the program product upgrade and subsets are used.

2.7 SMP/E V3R6 overview

The SMP/E element is enhanced in z/OS V1R13 and it is the same level of SMP/E in SMP/E V3R6 (program number 5655-G44).

Note: Effective 31 December 2010, the Enhanced PSP Tool (EPSPT), host compare program, and the associated extract files are removed from the IBM Technical Support web site:

http://www14.software.ibm.com/webapp/set2/psp/srchBroker)

The Enhanced PSP Tool's function is replaced by the addition of FIXCAT (fix category) information to Enhanced HOLDDATA and the REPORT MISSINGFIX function introduced in z/OS V1R10 SMP/E, which offers distinct advantages over the Enhanced PSP Tool. This SMP/E function is also available for all supported releases of z/OS in SMP/E for z/OS V3.5 (5655-G44), which you can order separately.

2.7.1 GIMADR service routine for z/OSMF

In support of z/OSMF, a new service routine called GIMADR has been created as an interface to the DFSMSdss ADRDSSU service.

z/OSMF creates a JCL job stream to clone a software instance; see Figure 2-1. This task includes three steps of the clone operation:

- Delete existing data sets.
- Copy the data sets for the software instance.
- Update zone entries as needed.

GIMADR is an interface to the DFSMSdss ADRDSSU service and manages the copying of data sets for the software instance. GIMADR is intended for use only by z/OSMF.

```
//job JOB.....
//step EXEC PGM=GIMADR,
// PARM='KEY=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx'
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
ADRDSSU control statements
/*
Where: KEY=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx is a hash value used by
GIMADR to ensure that the job step was generated by z/OSMF.
```

Figure 2-1 JCL job stream to clone a software instance

Important: SMP/E V3R6 is new with z/OS V1R13, and this level is required for the deployment manager under z/OSMF.

2.7.2 Multitasking using GIMDDALC SYSPRINT allocation

SMP/E processing is modified to support multitasking using the SYSPRINT definition in the GIMDDALC data set. To exploit utility multitasking in SMP/E, ensure that the ddname that is to contain the link edit utility output is defined with either a DDDEF entry that identifies a SYSOUT class, or with an entry in the SMPPARM GIMDDALC member that identifies a SYSOUT class. SMP/E's default ddname for utility output is SYSPRINT, but this can be changed using the PRINT subentry of the LKED UTILITY entry.

When multitasking, SMP/E will invoke multiple instances of the link edit utility at the same time, thereby decreasing the total time required to complete an ACCEPT, APPLY, LINK LMODS, or RESTORE command. Multitasking of link edit can occur when there are different target libraries and there are no dependencies on previous and subsequent link edits. If you do not define the print ddname using either a DDDEF entry or an SMPPARM GIMDDALC member, or if the print ddname definition identifies something other than SYSOUT class, or if you override the SYSPRINT DDDEF with a ddname in your JCL, then SMP/E will not multitask link edit utility operations.

2.7.3 Adding SAF checks to SMP/E processing

With IO11698, as an authorized program, SMP/E must ensure that any programs it calls that reside in an authorized library are called only in expected environments with expected parameters. SMP/E added SAF checks to its processing to ensure that only users with sufficient access authority are allowed to invoke certain SMP/E functions.

2.7.4 Retention of HOLDDATA

Prior to SMP/E V3R6, HOLDDATA was deleted from the global zone when the associated SYSMOD was deleted. Processing of the following commands has been modified with IO13643 to retain HOLDDATA when the associated SYSMOD is deleted from the global zone:

- After the SYSMOD is successfully accepted
- After the SYSMOD is successfully restored
- During REJECT PURGE and REJECT mass mode processing
- During REJECT NOFMID and REJECT select mode processing if the HOLDDATA operand is not specified

2.7.5 Additional SMP/E enhancements in z/OS V1R13

With z/OS V1R13, SMP/E has added the following enhancements to existing functions:

- A new ZONES operand is added.
- The **REPORT CROSSZONE** command.
- Comparing the SYSMODs installed in two zones.
- The **REPORT SYSMODS** command.

New ZONES operand

Your system may contain products that are packaged and installed separately, but which have service level or interface dependencies. For example, an interface error in one product may require a change to another product that used the interface. When this happens, a unique PTF is generated for each product. The relationship between the PTFs can be specified in a conditional requisite (++IF) modification control statement in the PTFs.

If you have completed the steps listed in “Specifying automatic cross-zone requisite checking” on page 80, SMP/E automatically checks the requisites during APPLY, ACCEPT, and RESTORE processing, whether the products are in the same zone or in separate zones.

However, if you have not completed these steps and the products are in separate zones, the requisites are not automatically checked in any other zones. In this case, to make sure these requisites are installed where they are needed, you must complete the following tasks:

1. Identify a set of zones to be checked for conditional requisites. These zones may be defined in the same global zone or in different global zones. You may also define groups of zones by creating ZONESET entries in the global zone. This is a new change with z/OS V1R13; see “Identifying zones to be processed” on page 43 for more information about this topic.

   Note: When defining a ZONESET, all the zones in the ZONESET must be defined by ZONEINDEX subentries in the same global zone as the ZONESET entry.

2. Run the **REPORT CROSSZONE** command to get a list of the SYSMODs that must be installed and the zones where they are needed.
3. Install the SYSMODs in the indicated zones.

Identifying zones to be processed

When identifying zones to be processed, you may group the zones by defining a global zone ZONESET entry. You may have one or more ZONESETs to describe groups of products that might have dependencies on each other or which may be defined in different global zones.

For example, assume you have a system that supports two products, ABC and XYZ, which have dependencies on one another. You could have one zone, BASEABC, for the base ABC function, and another zone, PRODABC, for a dependent function.

Commands to define ZONESETs

When you define a ZONESET, keep the following points in mind:

- Each zone in a ZONESET must also be defined in the global zone.
- Each zone in a ZONESET must be defined in the same global zone. They cannot be defined in global zones that are in different CSI data sets. If you have zones defined in two different global zones, and you want to use ZONESETs to identify the zones to the **REPORT CROSSZONE** command for processing, you must create a ZONESET in each global zone to contain the zones that are defined in each global zone.
- A zone can be part of more than one ZONESET.
- A ZONESET can contain both target and distribution zones.

The following examples of commands, shown in Figure 2-2 on page 43 and Figure 2-3 on page 44, are used to define the ZONESETs.

As shown in Figure 2-2 on page 43, assume there is a zone BASEXYZ for the base XYZ function, and another zone, PRODXYZ, for a dependent function. We also assume that all four zones are defined in the same global zone. The dependent functions are different versions of the same product, and they must be synchronized with each other and with their base functions. You can set up two ZONESETs to help keep these products at the same service level.

```
//UCL        JOB 'accounting info',MSGLEVEL=(1,1)
//UCL        EXEC SMPPROC
//SMPCNTL    DD *
SET      BDY(GLOBAL)        /* Set to global zone.   */.
UCLIN                       /*                       */.
ADD      ZONESET(ZONEA)     /* Create ZONESET ZONEA. */
         ZONE(BASEABC,      /* Include these zones.  */
              PRODABC,      /*                       */
              PRODXYZ)      /*                       */
         .
ADD      ZONESET(ZONEX)     /* Create ZONESET ZONEX. */
         ZONE(BASEXYZ,      /* Include these zones.  */
              PRODXYZ,      /*                       */
              PRODABC)      /*                       */
         .
ENDUCL                      /*                       */.
/*
```

Figure 2-2 Commands you can use to define the ZONESETs

As another example, as shown in Figure 2-3 on page 44, assume you have the same four zones, but here BASEABC and PRODABC are defined in one global zone and BASEXYZ and PRODXYZ are defined in a second global zone. Because each zone in a ZONESET must be

defined by a ZONEINDEX subentry in the same global zone as the ZONESET entry, the following ZONESETs may each be defined in their respective global zone.

```
//UCL       JOB 'accounting info',MSGLEVEL=(1,1)
//UCL1       EXEC SMPPROC
//SMPCNTL    DD *
SET           BDY(GLOBAL)     /* Set to global zone ABC */.
UCLIN                         /*                        */.
ADD           ZONESET(ZONEA)  /* Create ZONESET ZONEA   */
              ZONE(BASEABC,   /* Include these zones    */
                   PRODABC)   /*                        */
              .
ENDUCL                        /*                        */.
/*
//UCL2       EXEC SMPPROC
//SMPCNTL    DD *
SET           BDY(GLOBAL)     /* Set to global zone XYZ */.
UCLIN                 /*                                */.
ADD           ZONESET(ZONEX)  /* Create ZONESET ZONEX   */
              ZONE(BASEXYZ,   /* Include these zones    */
                   PRODXYZ    /*                        */
      .
EBDUCL                        /*
/*
```

Figure 2-3 Commands you can use to define the ZONESETs

Running the REPORT CROSSZONE command

The **REPORT CROSSZONE** command helps you to synchronize the service levels of different products when the products are installed in different zones. This command allows for the specification of target and DLIB zones that are defined either in the same global zone or in different global zones.

After you have identified the zones to be processed and created any desired ZONESET entries, you can run the **REPORT CROSSZONE** command to get a list of all the SYSMODs that must be installed and which zones need them. This list is the Cross-Zone Requisite SYSMOD report. It identifies which of the needed SYSMODs must be received and which SYSMODs caused the needed SYSMODs to be listed. In addition to the report, SMP/E writes commands to the SMPPUNCH data set. You can use them to install the SYSMODs in the appropriate zones. If all the zones being processed are of the same type, SMP/E determines the type of report to be generated. For example, identifying only target zones to be processed results in a Cross-Zone Requisite SYSMOD report for APPLY processing.

On the other hand, the set of zones to be processed can be mixed; that is, both target and distribution zones are identified. If this is the case, you need to specify which type of Cross-Zone report you want generated.

You can limit which SYSMODs SMP/E reports on by specifying any combination of these operands on the **REPORT CROSSZONE** command:

FORZONE Use this operand to list only SYSMODs needed in specific zones being processed.

FORFMID Use this operand to list only SYSMODs needed for specific FMIDs in the set of zones to be processed.

DLIBZONE or TARGETZONE Use this operand to tell SMP/E which zones you want a report on (if the zones being processed are mixed).

Notes: DLIBZONE and TARGETZONE are mutually exclusive operands. FORZONE is mutually exclusive with the ZONES operand.

As shown in Figure 2-2 on page 43, where all four zones are defined in the same global zone, assume that you applied a number of SYSMODs to the zones in ZONESET ZONEA. Some of these SYSMODs named conditional requisites. To find out which zones are affected by these requisites you can use the following commands, as shown in Figure 2-4 on page 45.

```
//REPORT JOB 'accounting info',MSGLEVEL=(1,1)
//REPORT EXEC SMPPROC
//SMPCNTL DD *
SET    BDY(GLOBAL)    /* Set to global zone. */.
REPORT CROSSZONE      /* Report on */
ZONESET(ZONEA)        /* ZONESET ZONEA. */.
/*
```

Figure 2-4 Commands to determine which zones are affected by requisites

As shown in Figure 2-3 on page 44, where the zones are defined in two separate global zones, assume that you applied a number of SYSMODs to the zones in ZONESET ZONEA defined in global zone 1 and also to PRODXYZ defined in global zone 2. Some of these SYSMODs named conditional requisites. To find out which zones are affected by these requisites you can use the following commands, as shown in Figure 2-5 on page 45.

```
//REPORT JOB 'accounting info',MSGLEVEL=(1,1)
//REPORT EXEC SMPPROC
//SMPCNTL DD                                        *
SET    BDY(GLOBAL) /*Set to global zone.             */.
REPORT CROSSZONE /* Report on */
       ZONES(ZONEA,(gzonedsn2,PRODXYZ)) /*
                        ZONESET ZONEA from
                        global zone 1 and PRODXYZ
                        from global zone 2           */.
/*
```

Figure 2-5 Commands to determine which zones are affected by requisites

Installing the SYSMODs

After you have the Cross-Zone Requisite SYSMOD report, you can install the missing SYSMODs in the zones where they are needed. Follow these steps:

1. Receive any SYSMODs that have not yet been received into the appropriate global zone.
2. Install the SYSMODs in the zones where they are needed. If you have the SMPPUNCH output from the **REPORT CROSSZONE** command, you can use that. If the zones were defined in different global zones, SMPCSI and SMPCNTL DD statements would have been generated in the SMPPUNCH output. These DD statements must be copied to your job before the SMPPUNCH can be used.
3. Rerun the **REPORT** command using the same set of zones to check the results of installing the new SYSMODs.
4. Receive and install any additional SYSMODs that are needed.

For the example in this chapter where all four zones are defined in the same global zone (Figure 2-2 on page 43), follow these steps for ZONESET ZONEA and ZONESET ZONEX, because both contain zone PRODXYZ. You can continue to run the **REPORT CROSSZONE** command and install SYSMODs until all the needed SYSMODs have been installed in both ZONESETs.

For the example in this chapter where the zones are defined in two different global zones (Figure 2-3 on page 44), follow these steps for ZONESET ZONEA and PRODXYZ and also for ZONESET ZONEX and PRODABC because both contain zone PRODXYZ. You can continue to run the **REPORT CROSSZONE** command and install SYSMODs until all the needed SYSMODs have been installed in the required zones.

Comparing the SYSMODs installed

The **REPORT SYSMODS** command can be used during the deployment of a new release or maintenance level to identify missing SYSMODs. This command compares the SYSMOD content of two target or DLIB zones. A SYSMOD Comparison Report is generated identifying the SYSMODs that exist in the input zone, but are not found in the comparison zone.

The **REPORT SYSMODS** command is modified to identify SYSTEM and USER HOLDs that must be resolved before the SYSMODs identified in the SYSMOD Comparison Report can be installed in the comparison zone.

You can run the **REPORT SYSMODS** command to get a list of all the SYSMODs that are installed in one zone and not in a second. An example of when you might want to use the **REPORT SYSMODS** command is shown in Figure 2-6 on page 46.

Assume you have two z/OS systems. The target zones that control these systems are TGZONE1 and TGZONE2, and they are serviced from the same global zone. You want to determine which SYSMODs are installed in TGZONE1 and are not installed in, but are applicable to, TGZONE2. You can use the following commands, as shown in Figure 2-6 on page 46.

```
SET BDY(GLOBAL)      /* process global zone     */.
REPORT SYSMODS       /* report on SYSMODs       */
INZONE(TGZONE1)      /* input zone TGZONE1      */
COMPAREDTO(TGZONE2)  /* comparison zone TGZONE2 */.
```

Figure 2-6 Comparing the SYSMODS installed

With z/OS V1R13, suppose we modify the example in Figure 2-6 on page 46 slightly and assume that TGZONE1 and TGZONE2 are serviced by different global zones. TGZONE1 is serviced by the global zone in SYS1.GLOBAL1.CSI and TGZONE2 is serviced by the global zone in SYS1.GLOBAL2.CSI. You can use the following commands, as shown in Figure 2-7 on page 46.

Because TGZONE1 and TGZONE2 are serviced by different global zones, an SMPCSI DD statement will be generated at the top of the SMPPUNCH output.This DD statement must be moved to your job before the SMPPUNCH output can be used.

```
SET BDY(GLOBAL)                              /* process global zone */.
REPORT SYSMODS                               /* report on SYSMODs */
       INZONE((TGZONE1)                      /* input zone TGTZONE1 */
       COMPAREDTO(SYS1.GLOBAL2.CSI,TGZONE2) /* comparison zone TGZONE2 */.
```

Figure 2-7 Comparing the SYSMODS installed

2.8 Coexistence, migration, and fallback considerations

Coexistence and fallback are important parts of planning for migration to the latest release. Coexistence of a z/OS V1R13 system with a z/OS V1R11 or z/OS V1R12 system is supported.

The required coexistence service must be installed on z/OS V1R11 and z/OS V1R12 systems that will coexist with z/OS V1R13 to enable the lower z/OS releases to tolerate changes in z/OS V1R13.

2.8.1 Using SMP/E for coexistence

You can use SMP/E V3R5 or SMP/E V3R6 to identify required coexistence PTFs that must be installed on z/OS V1R11 and z/OS V1R12 systems in preparation for migration to z/OS V1R13. SMP/E provides an automated method to identify required coexistence service for a z/OS system through the use of the **REPORT MISSINGFIX** command.

The REPORT MISSINGFIX command

Use the SMP/E **REPORT MISSINGFIX** command in conjunction with the FIXCAT type HOLDDATA, as described here:

1. Acquire and RECEIVE the latest HOLDDATA onto your z/OS V1R11 and z/OS V1R12 systems.

   Use your normal service acquisition portals, or download the HOLDDATA directly from:

   `http://service.software.ibm.com/holdata/390holddata.html`

2. Run the SMP/E **REPORT MISSINGFIX** command on your z/OS V1R11 and z/OS V1R12 systems, and specify a Fix Category (FIXCAT) value of:

   ```
   IBM.Coexistence.z/OS.V1R13
   ```

   The report will identify any missing coexistence and fallback PTFs for that system. For complete information about the **REPORT MISSINGFIX** command, see *SMP/E for z/OS Commands*, SA22-7771. Periodically, you might want to acquire the latest HOLDDATA and rerun the **REPORT MISSINGFIX** command to find out if there are any new coexistence and fallback PTFs.

Coexistence considerations

Coexistence occurs when two or more systems at different software levels share resources. The resources could be shared at the same time by different systems in a multisystem configuration, or they could be shared over a period of time by the same system in a single-system configuration.

z/OS systems can coexist with specific prior releases. This is important because it gives you the flexibility to migrate systems in a multisystem configuration using rolling IPLs, rather than requiring a systems-wide IPL. The way in which you make it possible for earlier-level systems to coexist with z/OS is to install coexistence service (PTFs) on the earlier-level systems. Complete the migration of all earlier-level coexisting systems as soon as you can.

Keep in mind that the objective of coexistence PTFs is to allow existing functions to continue to be used on the earlier-level systems when run in a mixed environment that contains later-level systems. Coexistence PTFs are not aimed at allowing new functions provided in later releases to work on earlier-level systems.

Examples of coexistence are two different JES releases sharing a spool, two different service levels of DFSMSdfp sharing catalogs, multiple levels of SMP/E processing SYSMODs packaged to exploit the latest enhancements, or an older level of the system using the updated system control files of a newer level (even if new function has been exploited in the newer level).

Figure 2-8 on page 48 illustrates coexistence releases with z/OS V1R13.

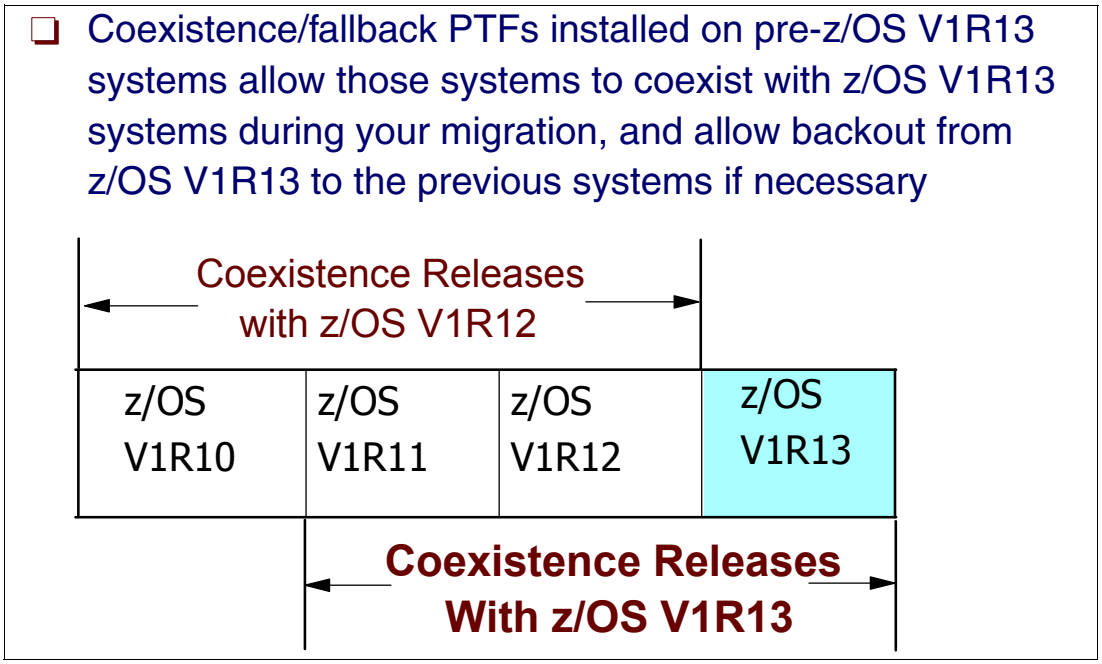


Figure 2-8 Coexistence releases with z/OS V1R13

Note: The term “coexistence” does not refer to z/OS residing on a single system along with IBM z/VSE®, VSE/ESA, or z/VM in an LPAR or as a VM guest.

Rolling IPLs

Rolling z/OS across a multisystem configuration is referred to as *rolling IPLs*. A rolling IPL is the IPL of one system at a time in a multisystem configuration. You can stage the IPLs over a few hours or a few weeks. The use of rolling IPLs allows you to migrate each z/OS system to a later release, one at a time, while allowing for continuous application availability. For example, data sharing applications offer continuous availability in a Parallel Sysplex configuration by treating each z/OS system as a resource for processing the workload.

The use of rolling IPLs allows z/OS systems running these applications to be IPLed one at a time, to migrate to a new release of z/OS, while the applications continue to be processed by the other z/OS systems that support the workload.

By using LPAR technology, you can use rolling IPLs to upgrade your systems without losing either availability or capacity.

You can use rolling IPLs when *both* of the following conditions are true:

- The release to which you are migrating is supported for coexistence, fallback, and migration with the releases running on the other systems.

- The appropriate coexistence PTFs have been installed on the other systems in the multisystem configuration.

Even when you are using applications that do not support data sharing, rolling IPLs often make it easier to schedule z/OS software upgrades. It can be very difficult to schedule a time when all applications running on all the systems in a multisystem configuration can be taken down to allow for a complex-wide or Parallel Sysplex-wide IPL. The use of rolling IPLs not only enables continuous availability from an end-user application point of view, but it also eliminates the work associated with migrating all z/OS systems in a multisystem configuration at the same time.

Fallback considerations

Fallback from a z/OS V1R13 system to a z/OS V1R11 or z/OS V1R12 system is supported. The terms *fallback* or *backout* mean a return to the prior level of a system. Fallback can be appropriate if you migrate to z/OS V1R13 and, during testing, encounter problems that can be resolved by backing out the new release. By applying fallback PTFs to the prior release system before you migrate, the prior release system can tolerate changes that were made by the new system during testing.

Fallback is relevant in all types of configurations, that is, single-system or multisystem, with or without resource sharing. As an example of fallback, consider a single system that shares data or data structures, such as user catalogs, as you shift the system image from production (on the earlier release) to test (on the new release) and back again (to the earlier release). The later-level test release might make changes that are incompatible with the earlier-level production release. Fallback PTFs on the earlier-level release can allow it to tolerate changes made by the later-level release.

As a general reminder, always plan to have a backout path when installing new software by identifying and installing any service required to support backout.

Fallback is at a system level, rather than an element or feature level, except for JES2 and JES3. That is, except for JES2 and JES3, you cannot back out an element or feature; you can only back out the entire product. JES2 and JES3 fallback can be done separately as long as the level of JES2 or JES3 is supported with the release of z/OS and any necessary fallback PTFs are installed.

Fallback and coexistence are alike in that the PTFs that ensure coexistence are the same ones that ensure fallback.

Note: Keep in mind that new functions can require that all systems be at z/OS V1R13 level before the new functions can be used. Therefore, be careful not to take advantage of new functions until you are fairly confident that you will not need to back out your z/OS V1R13 systems, because fallback maintenance is not available in these cases. Consult the appropriate element or feature documentation to determine the requirements for using a new function.

2.9 Migration actions for z/OS V1R11 and z/OS V1R12

You can perform the following migration actions to existing z/OS V1R11 and z/OS V1R12 systems to prepare for installing z/OS V1R13 in the sysplex.

2.9.1 Updating CONSOLxx PARMLIB members

Prior to z/OS V1R13, you could specify INIT, DEFAULT, HARDCOPY and CONSOLE keyword statements without using a blank delimiter. However, this can cause a problem if other keywords are misplaced or misspelled. For example, if INTIDS(Y) is misspelled as INITIDS(Y), the parser considers this an INIT statement. This could result in a console not being defined correctly, or even having a system with no consoles after initialization except the system console.

If you do not have a blank after the major keywords INIT, DEFAULT, and HARDCOPY, the default values will be used. In the case of the major keyword, CONSOLE, you will be left with only the system console if all of your CONSOLE statements do not end with a blank characters.

Required: If you do not have at least one blank after any of the four major keywords INIT, DEFAULT, HARDCOPY, and CONSOLE, you will receive a syntax error during CONSOLxx parmlib processing indicated by message IEA195I or message IEA196I, as follows:

```
IEA196I CONSOLM1 O3E0: NAME REQUIRED FOR CONSOLE.
IEA196I CONSOLM1 INIT: DUPLICATE SPECIFICATION IGNORED.
IEA196I CONSOLM1 O3E0: UNRECOGNIZED KEYWORD INITDS(Y) IGNORED.
IEA196I CONSOLM1 O3E0: UNRECOGNIZED KEYWORD INITDS(Y) IGNORED.
IEA195I CONSOLM1 LINE1: UNRECOGNIZED STATEMENT TYPE IGNORED.
IEA195I CONSOLM1 LINE1: UNRECOGNIZED STATEMENT TYPE IGNORED.
```

Migration action:

1. Examine your CONSOLxx parmlib member to verify that you have at least one blank after the major keywords.
2. If you do not have a blank, update your CONSOLxx parmlib member by entering one or more blanks between the major keyword statements and their associated keywords.

2.9.2 Stand-alone dump data set allocations

As an application grows in size and uses ever-greater amounts of storage, you need to evaluate whether the DASD space allocated for a stand-alone dump data set continues to be adequate. These migration actions are documented in *z/OS V1R13 Migration*, GA22-7499.

This migration action is recommended because of changes that have been made to stand-alone dump processing (that reorder dump records with the intent of recording more important data early). This task is particularly important if you deploy any LPARs with significantly more main storage than previously used. Consider the following actions:

- Use multivolume stand-alone dump data sets. Adjust the number of volumes and their separation to achieve tolerable stand-alone dump capture times.
- Use extended-format sequential data sets or large format sequential data sets. Copy their contents to an extended-format, compressed, striped data set using the IPCS **COPYDUMP** subcommand prior to analysis. Use the same or a larger striping factor than you used for your stand-alone dump data sets. Dump data sets to which stand-alone dump can write may not be compressed or striped, but both attributes are advantageous for the target of the copy operation.

Note: Starting with z/OS V1R12, stand-alone dump data sets can be placed in track-managed space as well as cylinder-managed space on Extended Address Volumes (EAVs).

- Use a large CISIZE and striping for IPCS dump directories, and use blocking, striping, and compression for the stand-alone dump data set. Very large stand-alone dumps might require that you define your directory with the extended addressing attribute, allowing it to hold more than 4 GB.

Note: Control interval sizes less than 24 K have been shown to be more vulnerable to fragmentation when used as IPCS dump directories, and IPCS performance can be degraded when such fragmentation occurs. In this background, warning message BLS21110I will be issued and you might recreate the DDIR by using the CLIST BLSCDDIR BCP migration actions to perform before first IPL.

2.9.3 Exploiting WARNUND as a new IEASYSxx parameter

Starting in z/OS V1R13, you can specify the WARNUND parameter in the IEASYSxx parmlib member to indicate that a warning message, IEA660I, be issued when undefined statements are encountered, rather than prompting for a correct statement. Using WARNUND can be particularly useful when specifying new parmlib options in IEASYSxx (such as the new IXGCNF and IGGCAT system parameters that are introduced in z/OS V1R13), and allowing these new IEASYSxx parmlib member specifications to be shared with pre-z/OS V1R13 systems.

The WARNUND parameter asks that the system warn on finding an undefined system parameter, instead of prompting for new system parameters. After the WARNUND parameter is found, it applies to all subsequent system parameter processing. If you want WARNUND processing for all IEASYSxx parmlib members, put WARNUND at the beginning of IEASYS00 because then it covers all subsequent system parameters within IEASYS00 and any other IEASYSxx members processed.

If you need WARNUND processing active for a specific parameter in some IEASYSxx parmlib member, you can place WARNUND just before that parameter. If you want WARNUND processing for your reply to IEA101A, you must specify it within the reply text (earlier within that reply than any undefined parameter). For example, you might reply “R 0,WARNUND,SYSP=01,NEWPARM=VALUE”. When a warning condition is detected, messages IEA321I and IEA660I are issued.

Migration: This function is rolled back to z/OS V1R12 and z/OS V1R11 with APAR OA35929.

1. Install the PTF for APAR OA35929 on all pre-z/OS V1R13 systems.
2. As you add new statements in the IEASYSxx parmlib member for functional exploitation and you want to share those modified IEASYSxx parmlib members with pre-z/OS V1R13 systems, add WARNUND to the beginning of IEASYS00 because that will cover updates in all IEASYSxx members.

2.9.4 Adjusting CON= system parameter

z/OS console support can be operated in one of two modes: shared mode and distributed mode.

Shared mode Shared mode is the name given to the way that z/OS console support has always operated in a sysplex environment. As sysplex environments have grown larger, various shortcomings in shared mode have been identified and are addressed by the distributed mode of operation.

Distributed mode Distributed mode is a mode of console operation that improves the operation of z/OS console support when running in a sysplex environment by:

- Potentially reducing the time that it takes to IPL a system
- Potentially reducing the time that it takes a system to join a sysplex
- Potentially reducing the scope of console-related hangs
- Reducing the possibility of console-related outages
- Allowing more MCS, SMCS, and subsystem consoles to be configured

Prior to z/OS V1R13, the default console operating mode was SHARED. Beginning with z/OS V1R13, the default console operating mode is changing from SHARED mode to DISTRIBUTED mode. SHARED mode will be removed in a future release. DISTRIBUTED mode is now the preferred mode of operation. Adjust this parameter in the CONSOLxx parmlib member to accommodate the default change for console operating mode.

The initial mode is specified on the CON= system parameter. Use the **D OPDATA,MODE** command to find the current mode, which is displayed in message CNZ9006I; see Figure 2-9 on page 52.

```
D OPDATA,MODE
 CNZ9006I 10.10.16 DISPLAY 0,MODE 713
 CURRENT: DISTRIBUTED
 SYSTEM   MIGRATION STATUS
 SC74     MIGRATION SUPPORTED
 SC75     MIGRATION SUPPORTED
 SYSPLEX ABLE TO MIGRATE: YES
```

Figure 2-9 D OPDATA,MODE command to display current console mode

Note: Distributed mode relieves the 99 MCS, SMCS and subsystem console per sysplex constraint by allowing up to 99 MCS, SMCS and subsystem consoles to be active per z/OS image. Additional flexibility is provided by allowing up to 250 MCS, SMCS and subsystem consoles to be defined per z/OS image (of which up to 99 may be concurrently active on that image).

The number of MCS, SMCS and subsystem consoles that may be concurrently active in a sysplex is now a function of the 99 active console limit of each image multiplied by the number of z/OS images in the sysplex.

Migration action

If there is no specification on the CON= system parameter, DISTRIBUTED mode is now the default.

If there is no specification on the CON= system parameter and SHARED mode is required, you have to explicitly request the SHARED mode on the CON= system parameter. This allows the system or systems to continue functioning in the same manner as they do today. Use the **SETCON MODE=SHARED** command to request SHARED mode.

Note: When you activate the OPERCMDS FACILITY class, you must have the CONTROL access authority to the profile when issuing the **SETCON MODE** command.

You can use the IBM Health Checker, CHECK(ZOSMIGV1R13_CNZ_CONS_OPER_MODE), to determine if your installation has explicitly identified your console service operating mode.

```
CHECK(IBMCNZ,CNZ_CONSOLE_OPERATING_MODE)
START TIME: 05/03/2011 07:08:59.538674
CHECK DATE: 20100101  CHECK SEVERITY: LOW

CNZHS0014I Console Services is running in the preferred operating mode
of Distributed.

END TIME: 05/03/2011 07:08:59.538915  STATUS: SUCCESSFUL
```

Figure 2-10 Health check for console mode, SHARED or DISTRIBUTED

2.9.5 Making sysplex=filesys available on all R11 and R12 systems

With z/OS V1R13, zFS only runs in sysplex=filesys mode. This requires that all sysplex members in the shared file system environment must run sysplex=filesys, including any z/OS V1R11 and z/OS V1R12 systems.

Migration action

In a shared file system environment, perform the following steps to ensure that sysplex=filesys is available on all zFS z/OS V1R11 and z/OS V1R12 systems:

1. Install the PTF for APAR OA32925 (UA55765) on all z/OS V1R11 and z/OS V1R12 systems and make it active on all systems through a rolling IPL. This provides the enhanced connect function required by zFS V1R13.
2. If you are currently running zFS sysplex=off, specify sysplex=filesys and make it active on all systems through a rolling IPL. If you are running sysplex=on, specify sysplex=filesys and sysplex_filesys_sharemode=rwshare and make it active on all systems through a rolling IPL.

To determine if you are running zFS sysplex=filesys, issue the **F ZFS,QUERY,LEVEL** command. In a shared file system environment, the last line of the response indicates if zFS is running sysplex=filesys, as shown in Figure 2-11 on page 53.

```
F ZFS,QUERY,LEVEL
  IOEZ00639I zFS kernel: z/OS    zSeries File System
  Version 01.13.00 Service Level z130224 - HZFS3D0.
  Created on Thu Mar 3 09:24:20 EST 2011.
  sysplex(filesys,rwshare) interface(4)
  IOEZ00025I zFS kernel: MODIFY command - QUERY,LEVEL completed successfully.
```

Figure 2-11 Command to determine sysplex=filesys mode is active

Health check for sysplex=filesys

Use the zFS migration check, ZOSMIGV1R13_ZFS_FILESYS, in the IBM Health Checker for z/OS to help determine if you are running zFS at the correct sysplex level. This check is automatically installed with APAR OA36514.

2.9.6 New DASD space requirements for zFS

zFS V1R13 may use more DASD storage for data than previous releases required. The amount of DASD storage depends on file sizes and on ACL and symbolic link usage. In general, the more small files in the file system, the more likely it is that a file system created or updated with zFS R13 will require more DASD storage than previous releases.

zFS always reads and writes data in 8 K blocks. However, in z/OS V1R13, zFS stores data either inline or in 8 K blocks. (Inline data is a file that is smaller than 53 bytes and is stored in the file's metadata.) Unlike in previous releases, zFS R13 no longer stores data in 1 K fragments. zFS R13 can read data stored in fragments; however, when the data is updated, it is moved into 8 K blocks. Previously, zFS could store data in 1 K fragments (contained in an 8 K block). This meant that multiple small files could be stored in a single 8 K block.

Because data is no longer stored in fragments, zFS R13 might need more DASD storage than was required in previous releases to store the same amount of data. More storage may also be needed if zFS R13 is in a mixed-release sysplex and becomes the zFS owning system of a file system. Consider the following scenarios:

1. If every file in the file system is 1 K or less, zFS R13 could require up to four times the DASD storage as was needed in previous releases.
2. Because HFS uses 4 K blocks to store data and zFS uses 8 K blocks, if every file in the file system were 4 K or less, zFS R13 could require up to twice as much DASD space to store these files.
3. If the file system contains 1000 files that are 1 K in size, zFS in R13 could take a maximum of 10 cylinders more than zFS in previous releases. Typically, however, any increase in the DASD storage used by zFS R13 will be negligible. For example, the z/OS V1R13 version root file system copied using zFS R13 takes approximately 2% more space than the same file system copied using zFS R11.

Note: zFS R13 packs multiple ACLs and symbolic links into an 8 K block which previous releases did not do. To minimize the chance of application failure due to running out of DASD storage in newly mounted file systems, the default value for the IOEFSPRM option aggrgrow is changed from Off to On.

Migration action

This action is required if you have not specified the zFS aggregate option in your IOEFSPRM configuration options file.

Note: The aggrgrow option specifies whether aggregates can be dynamically extended when they become full. As of z/OS V1R13, by default, a zFS read-write mounted file system mounted on a V1R13 system will attempt to dynamically extend when it runs out of space. The aggregate (that is, the VSAM Linear Data Set) must have a secondary allocation specified to be dynamically extended and there must be space on the volume or volumes. This global value can be overridden in the **`define_aggr`** option or the **`zfsadm attach`** command for multifile system aggregates and on the **`MOUNT`** command for compatibility mode aggregates.

With zFS V1R12, **Default value:** Off. With zFS V1R13, **Default option:** On.

For all zFS file systems, consider the following options:

1. If you have not specified the zFS aggrgrow option in your IOEFSPRM configuration options file, recognize that the default is changing in z/OS V1R13 from **aggrgrow=off** to **aggrgrow=on**. This means that by default, a zFS read-write mounted file system that is mounted on z/OS V1R13 will attempt to dynamically extend when it runs out of space if a secondary allocation size is specified and there is space on the volume or volumes.
2. If you do not want that default change and you want it to act as in prior releases, specify aggrgrow=off in your IOEFSPRM configuration options file so that it takes effect on the next IPL. You can dynamically change the aggrgrow option to off with the **`zfsadm config -aggrgrow off`** command. You can see your current value for aggrgow with the **`zfsadm configquery -aggrgrow`** command.

Space considerations for new zFS file systems

When creating new file systems, the following specification considerations can apply:

1. Increase the estimated size of a new zFS file system if you know that many files in the file system will be small.
2. Mount zFS read-write file systems and allow them to dynamically extend, because if additional DASD space is needed, applications will not fail because the file systems are out of storage.

   To do so, mount the file systems with the AGGRGROW mount option or use the default aggrgrow=on IOEFSPRM configuration option. The data set must have a non-zero secondary allocation size, and there must be space on the volume to allow dynamic extension.

Space considerations for existing file systems

For existing zFS file systems:

1. Use the scan for small files utility (zfsspace) to determine if an existing file system needs additional DASD storage. For a mounted zFS file system, the utility shows the number of small files (1K or less), whether a secondary allocation is specified, and whether aggrgrow=on is specified.

   You can determine how many files you have in a file system that are less than or equal to 1 K in size by using the following shell command:

   ```
   find <mountpoint> -size -3 -type f -xdev | wc -l
   ```

   The zfsspace utility can be downloaded from:

   `ftp://public.dhe.ibm.com/s390/zos/tools/zfsspace/zfsspace.txt`
2. If a file system has a secondary allocation size and is mounted with the AGGRGROW mount option, allow it to dynamically extend to minimize the potential failure due to lack of storage. If there are insufficient candidate volumes, also consider adding volumes by

using the **IDCAMS ALTER** command with the ADDVOLUMES option. Generally, after adding volumes, a remount samemode is required to have them take effect.

3. If a file system is not enabled to dynamically extend, consider explicitly growing the file system using the z/OS UNIX **zfsadm grow** command. This is especially important if the file system contains many small files that will be updated.
4. If you expect a file system to grow larger than 4 GB (about 5825 3390 cylinders) and it is not SMS-managed with extended addressability, you will need to copy it to an SMS-managed zFS data set with a data class that includes extended addressability.

   To do so, use the **pax** command. If a zFS aggregate is to be larger than 4 GB, it must be SMS-managed with a data class that includes extended addressability.

2.9.7 Copying cloned file systems to a compatibility mode aggregate

z/OS V1R13 is the last release that zFS will support cloning file systems. In anticipation of this removal of support, discontinue using zFS clone functions, such as the **zfsadm clone** and **zfsadm clonesys** commands. Also, discontinue mounting any zFS file system aggregates that contain a cloned (.bak) file system.

Note: The support for cloning file systems is withdrawn, and only zFS compatibility mode aggregates will be supported.

Migration action

Determine if cloned file systems (.bak) have been created or are in the process of being created on your system.

- Issue the **f zfs,query** command and review the contents of the FILE report. The Flg field in the report will indicate the status of the file system aggregate.

If your system contains cloned file systems, copy that data to a compatibility mode aggregate.

2.9.8 zFS multifile system aggregates to zFS compatibility mode aggregates

z/OS V1R13 is the last release of zFS support for multifile system aggregates. If you have data stored in zFS multifile system aggregates, you should copy the data from the zFS multi-file system aggregates into zFS compatibility mode aggregates.

Note: The support is withdrawn, and only zFS compatibility mode aggregates will be supported.

Migration action

Use one of the following methods to determine if you are using zFS multifile system aggregates.

- Use the IBM Health Checker for z/OS check as follows:

  Use check ZOSMIGV1R11_ZFS_RM_MULTIFS or check ZOSMIGREC_ZFS_RM_MULTIFS to help determine whether multifile system aggregates are attached on your system.
- Scan your zFS IOEFSPRM configuration options file for define_aggr statements.
- Scan your `/etc/rc` file for any **zfsadm attach** commands.

- Issue the **zfsadm aggrinfo** command to determine whether an aggregate is a multifile system aggregate; in the command response, COMP indicates compatibility mode and MULT indicates multifile system.

If you are using zFS multifile system aggregates, copy the data from each of those file systems into its own zFS compatibility mode aggregate.

2.9.9 Making sysplex=filesys available on all R11 and R12 systems

Ensure sysplex=filesys is available on all zFS V1R11 and V1R12 systems in a shared file system environment. In z/OS V1R13, zFS only runs in sysplex=filesys mode. This requires that all sysplex members in the shared file system environment must run sysplex=filesys, including any z/OS V1R11 and z/OS V1R12 systems.

Migration action

Install the PTF for APAR OA32925 on z/OS V1R11. If a problem occurs when zFS is running sysplex=filesys on a z/OS V1R11 or z/OS V1R12 system, you can perform the following steps:

1. Remove the sysplex specification from each system or specify sysplex=off on each system (this is equivalent to the default).
2. Perform a rolling IPL or restart zFS on each system.

Attention: This procedure cannot be done after zFS on the z/OS V1R13 system has joined the sysplex. If you try to start zFS on another z/OS V1R13 system after you have changed zFS to sysplex=off on the z/OS V1R11 or z/OS V1R12 system, zFS on z/OS V1R13 will not start. This happens because zFS on z/OS V1R13 requires all other systems be zFS sysplex=filesys.

Also, if you try to bring in zFS z/OS V1R13 when sysplex=filesys is not active on all systems, you will receive the following message:

```
IOEZ00721I Sysplex member sysname is not running sysplex=filesys
```

zFS on this initializing member will terminate, where sysname is the sysplex member that is not running sysplex=filesys.

Specifying zFS sysplex=filesys in a shared file system environment causes zFS to run sysplex-aware on a file system basis. This is the preferred mode for zFS in a shared file system environment.

Migration steps

Perform the following steps to ensure that sysplex=filesys is available on all zFS z/OS V1R11 and z/OS V1R12 systems in a shared file system environment.

1. Install the PTF for APAR OA32925 (UA55765) on all z/OS V1R11 systems, and make it active on all systems through a rolling IPL. This provides the enhanced connect function required by zFS V1R13.
2. If you are currently running zFS sysplex=off, specify sysplex=filesys and make it active on all systems through a rolling IPL. If you are running sysplex=on, specify sysplex=filesys and sysplex_filesys_sharemode=rwshare and make it active on all systems through a rolling IPL. The health check ZOSMIGV1R13_ZFS_FILESYS verifies that all z/OS V1R11 and z/OS V1R12 systems in the shared file system environment have specified sysplex=filesys before z/OS V1R13 is introduced.

Note: To determine if you are running zFS sysplex=filesys, issue the **F ZFS,QUERY,LEVEL** operator command. In a shared file system environment, the last line of the response indicates if zFS is running sysplex=filesys. In the following example, zFS is running sysplex=filesys.

```
f zfs,query,level
IOEZ00639I zFS kernel: z/OS    zSeries File System
Version 01.13.00 Service Level z130224 - HZFS3D0.
Created on Thu Mar 3 09:24:20 EST 2011.
sysplex(filesys,norwshare) interface(4)
IOEZ00025I zFS kernel: MODIFY command - QUERY,LEVEL completed successfully.
```

If you do not perform these steps on z/OS V1R11 or z/OS V1R2 systems, you will receive error messages when you try to bring up zFS on a z/OS V1R13 system.

2.9.10 zFS verify virtual storage usage

If your zFS virtual storage usage is close to the limit of the zFS address space, this additional virtual storage request at zFS initialization could cause zFS to fail to initialize and not come up, or zFS may come up but with insufficient remaining storage to handle zFS requests such as mount. In these cases, you would likely see zFS the warning message:

```
IOEZ00662I: ZFS is low on storage.
```

Migration action

Apply PTF UA55765 (zFS APAR OA33451) to z/OS V1R11. This fixes a performance problem that occurs because of too many storage obtains and releases in zFS. The resolution of the problem involves obtaining a new block of storage at zFS initialization. This storage obtain is for approximately 60 MB.

Check zFS storage usage by using the operator command **F ZFS,QUERY,STORAGE** command. If you compare the third line of data (`USS/External Storage Access Limit:`) to the fourth line (`Total Bytes Allocated (Stack+Heap+OS):`), you will be able to see how close zFS is to using its maximum storage. The Total Bytes Allocated should be less than the amount shown in the USS/External Storage Access Limit field.

Figure 2-12 on page 59 shows the **F ZFS,QUERY,STORAGE** command output.

```
F ZFS,QUERY,LEVEL
 IOEZ00438I Starting Query Command STORAGE.
                 zFS Primary Address Space Storage Usage
                 ---------------------------------------
 Total Storage Available to zFS: 1841299456 (1798144K) (1756M)
 Non-critical Storage Limit: 1820327936 (1777664K) (1736M)
 USS/External Storage Access Limit: 1778384896 (1736704K) (1696M)
 Total Bytes Allocated (Stack+Heap+OS): 368836608 (360192K) (351M)
 Heap Bytes Allocated: 318500680 (311035K) (303M)
 Heap Pieces Allocated: 665715
 Heap Allocation Requests: 1006551
 Heap Free Requests: 340836
            Heap Usage By Component

            Storage Usage By Component
            --------------------------
Bytes               No. of  No. of
Allocated   Pieces  Allocs  Frees   Component
----------  ------  ------  ------  ---------

    134820      41      41       0  Interface
   3926092    1054    1200     146  Media Manager I/O driver
      1828       5       5       0  Trace Facility
    473052       7       7       0  Message Service
    330980     720     726       6  Miscellaneous
     49664     111     163      52  Aggregate Management
    119252      40      44       4  Filesystem Management
     33920      24    6332    6308  Administration Command Handling
  16132280   65627   65636       9  Vnode Management
  16413124   33487   33643     156  Anode Management
         0       0       0       0  Directory Management
    914784    8238    8277      39  Log File Management
 137847088   32773   32780       7  Metadata Cache
    411760    2005    2005       0  Transaction Management
   5842700   98398   98402       4  Asynchronous I/O Component
     81092     263     263       0  Lock Facility
   1351916     695     695       0  Threading Services
  11870960  239243  239267      24  Cache Services
     43148       3 6226       6223  Configuration  parameters
processing
  14959944  148714  476402  327688  User File Cache
     61708     115     186      71  Storage Management
  63249396    1539    1541       2  XCF Services
     53952      16      65      49  Cross system attach validation
   2407152   20519   20525       6  Server Token Manager (STKM)
     14696      42      42       0  Server Token Cache (STKC)
  20722516   12013   12046      33  Client Token Cache (CTKC)
         0       0       0       0  Server Vnode Interface (SVI)
      4284      21      30       9  Name Space (NS)
   1048572       2       2       0  Directory storage

IOEZ00025I zFS kernel: MODIFY command - QUERY,STORAGE completed successfully.
```

Figure 2-12 F ZFS,QUERY,LEVEL command output

2.9.11 Deleting NOIMBED and NOREPLICAT LISTCAT command attributes

Prior to z/OS V1R13, output from the **IDCAMS LISTCAT** command displayed the NOIMBED and NOREPLICAT attributes. Starting with z/OS V1R13, these attributes are no longer included in the **LISTCAT** command output. This can affect programs that parse LISTCAT results.

Previously, support for creating data sets with IMBED or REPLICATE attributes on the **AMS DEFINE** command was removed. Starting with z/OS V1R13, the LISTCAT output no longer displays the default NOIMBED and NOREPLICAT attributes. This information is no longer needed. This can affect programs that parse LISTCAT results.

Note: For any cluster defined prior to 2001 with IMBED or REPLICATE attributes, those attributes are displayed in **IDCAMS LISTCAT** command output.

Migration action

Convert the programs that parse LISTCAT results to use Catalog Search Interface (CSI).

2.9.12 Updating procedures using IEBDSCPY alias name to access IEBCOPY

Prior to z/OS V1R13, the IEBCOPY utility was an APF-authorized program that had to run from an APF-authorized library. If another program called IEBCOPY, that program also had to be APF-authorized.

Starting in z/OS V1R13, the IEBCOPY utility is no longer APF-authorized and can be run from non-authorized libraries; callers of IEBCOPY no longer need to be APF-authorized. In addition to this authorization change, other performance improvements have also been made to IEBCOPY.

IEBCOPY is a data set library management utility that is used to perform many critical system build and maintenance activities. If this utility does not work correctly, then key library data sets could be rendered as unusable and the z/OS client would be left without a fallback method. A fallback copy of the z/OS V1R12 level of IEBCOPY is provided, named IEBCOPYO, for use if you experience problems with the updated version of IEBCOPY in z/OS V1R13. The old IEBCOPYO utility with its alias of IEBDSCPY is installed into SYS1.LINKLIB with authorization code one, AC(1).

Because the code in IEBCOPY is now different from the code in IEBCOPYO, do not copy and replace IEBCOPY with IEBCOPYO, due to maintenance issues. PTF updates will update the IEBCOPY and IEBCOPY load modules with changes that are targeted to the load modules that are shipped by IBM.

The IEBDSCPY alias name of IEBCOPY that exists in earlier versions of DFSMS is no longer an alias name in the z/OS DFSMS V1R13 IEBCOPY load module. The IEBDSCPY alias name continues to exist as the alias to the IEBCOPYO load module in z/OS V1R13, but will be eliminated as an alias of the IEBCOPY load module in future releases of DFSMS. If you currently access the IEBCOPY utility by using the IEBDSCPY alias name, you need to make the necessary change to use the IEBCOPY primary name.

Migration action

Programs or procedures that use the IEBDSCPY alias name to access IEBCOPY should be changed to use the IEBCOPY primary name. Programs that continue to use the alias name will connect to the old, pre-z/OSV1R13 version, currently called IEBCOPYO.

2.10 BCP migration actions after the first IPL of z/OS V1R13

Following are migrations actions that can made to an existing z/OS V1R11 and z/OS V1R12 system in preparation to installing z/OS V1R13 in the sysplex.

2.10.1 Disabling DFS client initialization

The DFS client (DFSCM) is a physical file system that is started during z/OS UNIX initialization based on a FILESYSTYPE statement in the BPXPRMxx parmlib member. Starting with z/OS V1R13, the DFS client function is removed.

Migration action

If your installation uses the DFS client, you must remove the following statement from the BPXPRMxx parmlib member to prevent the client from initializing:

```
FILESYSTYPE TYPE(DFSC)
ENTRYPOINT(IOECMINI)
PARM('ENVAR("_EUV_HOME=/opt/dfslocal/home/dfscm") />DD:IOEDFSD 2>&1')
ASNAME(DFSCM)
```

If this migration action is not performed before the first IPL of z/OS V1R13, you will receive the following error message:

```
IOEP12402E:  As of z/OS Version 1 Release 13, the DFS client function has been
removed.
```

z/OS UNIX will successfully initialize, but you will need to follow the guidance in the message to remove the entry and restart z/OS UNIX.

If you have not already done so, you should use the z/OS UNIX **pax** command to migrate any data in DCE DFS or Episode file systems to TFS, HFS, or zFS file systems. The recommended general procedure is as follows:

1. Set up a zFS file system to receive the data.
2. Copy your DCE DFS or Episode file system data to the zFS file system, using the z/OS UNIX pax command.
3. Set up a z/OS NFS server to allow data access from a remote z/OS UNIX system.

2.11 Deleted elements in z/OS V1R13

As of z/OS V1R13, the following elements have been withdrawn; these elements will be deleted by step DELZOSB in the CLNOS390 job:

- DCE (FMIDs HMB3190 and JMB319J)
- Integrated Security Services DCE (FMIDs HRSS190 and JRSS19J)
- z/OS UNIX System Services Connection Manager (FMIDs HCMG110 and JCMG1J0)
- z/OS UNIX System Services Process Manager (FMIDs HPMG110 and JPMG1J0)

Note: If elements are withdrawn and there are no superseding functions, normal SMP/E APPLY/ACCEPT processing will not delete the obsolete elements. In this case, you must run a delete job to remove them.

Sample JCL and instructions are provided in member CLNOS390 of SMPTLIB, 'prefix.HBB7780.F5' to remove the withdrawn elements that are not part of z/OS.

msys for Setup element

The msys for Setup element is withdrawn as of z/OS V1R12 and is not included in the product. Sample job CLNOS390 contains a sample delete SYSMOD, DELZOSA, to remove the msys for Setup FMIDs. When sample job CLNOS390 is run, SYSMOD DELZOSA will delete the obsolete FMIDs for msys for Setup.

Note: The msys for Setup element was withdrawn from z/OS V1R12. If you are migrating from z/OS V1R11, run the CLNOS390 job to remove msys for Setup element. After the CLNOS90 job runs, some three of the msys for Setup data sets will still contain code because they are shared with other z/OS V1R11 elements. If you are migrating from z/OS V1R11, delete the empty msys for Setup data sets and associated DDDEFs after z/OS V1R13 has been installed.

There are no new elements in z/OS V1R13.

3

Managing volume backups with fast replication

In z/OS V1R13, the system is changed to update volume information across a Parallel Sysplex after the successfully completion of DFSMSdss, DFSMShsm, or ICKDSF commands during which the volume serial or VTOC location, or both, have been changed. This enhancement is provided when the automatic REFUCB function of the Device Manager is enabled. It is intended to eliminate the requirement to issue **VARY** commands on sharing systems in the sysplex when volume information has been updated by these functions.

In this chapter, the following topics are discussed:

- Refreshing the UCB
- Enabling the REFUCB function

3.1 Automatic refresh UCB function

The automatic refresh UCB function causes the system to automatically update the UCB for an online volume that was the target of a DFSMSdss COPY or RESTORE operation or an ICKDSF REFORMAT NEWVTOC operation. Those operations may cause the volume serial and the VTOC location on the target volume to change. Before the volume can be accessed on any remote system, the UCB must be refreshed with the new values.

The automatic refreshing of the UCB eliminates the need to manually vary the volume offline and then online to update the UCB. To support the automatic refresh UCB function, DFSMSdss builds an ENF64 event to notify registered listeners that a volume has been the target of a copy or restore operation.

3.2 Refreshing the UCB on remote systems

The VTOC and the volume serial on a target volume may have changed as a result of one of the functions shown in Figure 3-1 on page 64. When these functions are invoked for a volume, this requires a refresh of the UCB.

DFSMSdss COPY or RESTORE - Following a COPY or RESTORE operation, the VTOC location or the volume serial on the target volume may change. Before this volume can be accessed on any remote system, the UCBs on the remote systems must be refreshed.
DFSMShsm fast replicate backup - The VTOC and the volume serial on the target volume may have changed as a result of this operation. Before the volume can be accessed on any remote system, the UCB must be refreshed.
An ICKDSF command - The **REFORMAT NEWVTOC** command moves and expands the VTOC into a new location and before the volume can be accessed on any remote system, the UCB must be refreshed.

Figure 3-1 Functions that require a refresh

REFUCB function

The system invokes the Device Manager REFUCB function on each system in the sysplex that has REFUCB enabled.

Before the volume can be accessed on any remote system, the UCB must be refreshed. The refresh occurs automatically if the volume is online and the Device Manager REFUCB function is enabled by using the **F DEVMAN** command.

A refresh of the UCB on any remote system is necessary and now the system will do this automatically if the volume serial or VTOC location, or both, have been changed successfully as result of the following operations.

3.3 Enabling the REFUCB function

You enable the REFUCB function through the DEVSUPxx parmlib member or the **F DEVMAN** command.

DEVSUPxx parmlib member

The DEVSUPxx parmlib member specifies the installation default for device support options. After an IPL, clients can use the system command **SET DEVSUP=XX** to activate the DEVSUPxx changes.

The parameter to activate or deactivate the REFUCB function is:

ENABLE/DISABLE This parameter enables or disables a particular feature. In this example the feature is REFUCB, as shown in Figure 3-2 on page 65.

```
REFUCB
   Enables or disables the automatic REFUCB function of the Device Manager:
   ENABLE(REFUCB)
      When the REFUCB feature is enabled, the system automatically updates the
      UCB when device support software detects that a DSS COPY, RESTORE, or
      ICKDSF REFORMAT NEWVTOC operation has changed either the volser or the
      VTOC location. In the case of a volser or VTOC location change, the
      system invokes the DEVMAN REFUCB service on each system in the sysplex
      that has REFUCB enabled.

      • If the device is ONLINE, REFUCB issues a VARY ONLINE,UNCONDITIONAL
        command, which updates both the volser and VTOC location in the UCB.

      • If the device is OFFLINE, no action is taken.

   DISABLE(REFUCB)
      When the REFUCB feature is disabled, the system does not refresh the UCB
      when a DSS COPY, RESTORE or ICKDSF REFORMAT NEWVTOC operation has changed
      either the volser or the VTOC location.

      Default value: for REFUCB is DISABLE(REFUCB).
```

Figure 3-2 REFUCB feature in DEVSUPxx

Figure 3-3 on page 65 shows an example of a DEVSUPxx parmlib member to enable the REFUCB function.

```
 BROWSE    SYS1.PARMLIB(DEVSUP51) - 01.00
 Command ===>
******************************** Top of Data ********
ENABLE(REFUCB)
******************************** Bottom of Data ******
```

Figure 3-3 DEVSUPxx parmlib member to enable REFUCB

Using the SET DEVSUP=xx command

After you create the DEVSUPxx parmlib member, you can update the IEASYSxx parmlib member and wait until the next IPL, or use the **SET DEVSUP=xx** command to enable the function.

DEVSUP=xx The two alphanumeric characters indicate the DEVSUPxx member of the logical parmlib that contains the parameters that the system is to use to set device-related controls.

For example, if you have created a DEVSUP51 parmlib member enabling the REFUCB feature, you can now use the command **SET DEVSUP=51** to activate the function, as shown in Figure 3-4 on page 66.

```
16:13:33.65 NARO     00000290  SET DEVSUP=51
16:13:33.67          00000290  IEE252I MEMBER DEVSUP51 FOUND IN SYS1.PARMLIB
16:13:33.67          00000090  IEA253I DEVSUP  REFUCB  FUNCTION IS ENABLED
16:13:33.67 NARO     00000090  IEE536I DEVSUP   VALUE 51 NOW IN EFFECT
```

Figure 3-4 Using the SET DEVSUP=xx command to enable REFUCB

Using the MODIFY DEVMAN command

Use the **F DEVMAN** command to communicate with the Device Manager address space to display information or to request a specified service.

Note: Use this command only at the direction of the systems programmer.

The syntax of the **F DEVMAN** command is shown in Figure 3-5 on page 66.

```
F DEVMAN,{DUMP}
         {REPORT}
         {RESTART}
         {END(taskid)}
         {ENABLE(feature) }
         {DISABLE(feature)}
         {?|HELP}
```

Figure 3-5 Syntax for the MODIFY DEVMAN command

Following is a brief description of the ENABLE/DISABLE(REFUCB) parameter:

ENABLE(REFUCB) This parameter enables the automatic REFUCB function of the Device Manager. When the REFUCB feature is enabled, the system automatically updates the UCB when device support software detects that an operation has changed either the volser or the VTOC location. The system invokes the DEVMAN REFUCB service on each system in the sysplex that has REFUCB enabled.

DISABLE(REFUCB) This parameter disables the REFUCB feature.

If you use the **F DEVMAN** command to enable the REFUCB function, the result will be similar to that shown in Figure 3-6 on page 66.

```
10:36:11.72 NARO     00000290  F DEVMAN,ENABLE(REFUCB)
10:36:11.73          00000090  DMO0012I DEVICE MANAGER REFUCB ENABLED
```

Figure 3-6 Using MODIFY DEVMAN command to enable REFUCB

The REPORT parameter of the **F DEVMAN, REPORT** command can be used to check whether the REFUCB function is enabled. An example is shown in Figure 3-7 on page 67.

```
11:58:00.51 NARO     00000290  F DEVMAN,REPORT
11:58:00.52          00000090  DMO0030I DEVICE MANAGER REPORT 997
                997  00000090  **** DEVMAN ************************
                997  00000090  * FMID:  HDZ1D10                    *
                997  00000090  * APARS: NONE                       *
                997  00000090  * OPTIONS:  REFUCB                  *
                997  00000090  * NO SUBTASKS ARE ACTIVE            *
                997  00000090  **** DEVMAN ************************
```

Figure 3-7 Using F DEVMAN,REPORT to check the REFUCB function

3.3.1 REFUCB function examples

The following simple examples illustrate the REFUCB function working to refresh the UCB. The test environment used was a small sysplex with only two members, and both systems running z/OS V1R13:

- SC74 - JES2 system
- SC75 - JES3 system

After a DFSMSdss COPY (with COPYVOLID)

If you submit a job to copy the content of the input DASD to the output DASD volume using **COPYVOLID** (which specifies that the volume serial number should also be copied), the Device Manager REFUCB service is invoked to automatically update the UCB on each system in the sysplex that has REFUCB enabled; see Figure 3-8 on page 67.

```
EDIT       NARO.DATA.JCL(DSSCOPYV) - 01.05               Columns 000
Command ===>                                               Scroll =
****** ***************************** Top of Data ********************
000100 //NARODSS   JOB (999,POK),'DSS TEST',CLASS=A,REGION=0M,
000200 //             MSGCLASS=T,TIME=10,MSGLEVEL=(1,1),NOTIFY=&SYSUID
000400 //STEP1    EXEC PGM=ADRDSSU
000500 //SYSPRINT DD   SYSOUT=*
000600 //DASDI    DD   UNIT=3390,VOL=SER=YY8143,DISP=SHR
000610 //DASDO    DD   UNIT=3390,VOL=SER=YY8142,DISP=OLD
000700 //SYSIN    DD   *
000800         COPY IDD(DASDI) -
000900              ODD(DASDO) -
001000              COPYVOLID -
001100              CAN
001200 /*
****** **************************** Bottom of Data ******************
```

Figure 3-8 DFSMSdss COPY job with COPYVOLID

If you access the JOBLOG you can find the Device Manager messages showing that the REFUCB function working just after the job has ended; see Figure 3-9 on page 68.

```
SC74 2011118 15:49:55.85 NARO     0290  D U,,,8142,1
SC74 2011118 15:49:55.85 NARO     0090  IEE457I 15.49.55 UNIT STATUS 8
                              880 0090  UNIT TYPE STATUS        VOLSER
                              880 0090  8142 3390 O             YY8142
SC74 2011118 15:50:13.16 NARO     0290  D U,VOL=YY8142
SC74 2011118 15:50:13.17 NARO     0090  IEE457I 15.50.13 UNIT STATUS 8
                              886 0090  UNIT TYPE STATUS        VOLSER
                              886 0090  8142 3390 O             YY8142
SC74 2011118 15:50:25.01 NARO     0290  D U,VOL=YY8143
SC74 2011118 15:50:25.01 NARO     0090  IEE457I 15.50.25 UNIT STATUS 8
                              888 0090  UNIT TYPE STATUS        VOLSER
                              888 0090  8143 3390 O             YY8143
SC75 2011118 15:51:00.55 JES3     0090  IAT6100 (JOB20589) JOB NARODSS
SC75 2011118 15:51:00.59 NARODSS  0090  IAT2000 JOB NARODSS  (JOB21457)
SC75 2011118 15:51:00.75 NARODSS  0290  -NARODSS  ENDED.
SC74 2011118 15:51:00.75          0290  DMO0061I 8142,YY8142,REFUCB STARTED
SC74 2011118 15:51:00.76 INTERNAL 0290  IEE302I 8142     ONLINE BY DMOAT002
SC74 2011118 15:51:00.76          0290  DMO0062I 8142,YY8143,REFUCB SUCCESSFUL
SC74 2011118 15:51:51.50 NARO     0290  D U,,,8142,1
SC74 2011118 15:51:51.51 NARO     0090  IEE457I 15.51.51 UNIT STATUS 9
                              917 0090  UNIT TYPE STATUS        VOLSER
                              917 0090  8142 3390 O             YY8143
SC74 2011118 15:52:00.44 NARO     0290  D U,VOL=YY8142
SC74 2011118 15:52:00.45 NARO     0090  IEE455I UNIT STATUS NO DEVICES
SC74 2011118 15:52:13.36 NARO     0290  D U,VOL=YY8143
SC74 2011118 15:52:13.37 NARO     0090  IEE457I 15.52.13 UNIT STATUS 9
                              921 0090  UNIT TYPE STATUS        VOLSER
                              921 0090  8142 3390 O             YY8143
```

Figure 3-9 REFUCB working after a DFSMSdss COPY

3.3.2 Using the ICKDSF REFORMAT NEWVTOC command

The VTOC information of the volume YY8142 before the submission of the ICKDSF job is shown in Figure 3-10 on page 68.

```
Volume . : YY8142
Command ===>

Unit . . : 3390

 Volume Data             VTOC Data              Free Space   Tracks
 Tracks . :   50,085     Tracks  . :        14  Size  . . :  50,068
 %Used  . :        0     %Used . . :         1  Largest . :  50,068
 Trks/Cyls:       15     Free DSCBS:       697  Free
                                                Extents . :       1
```

Figure 3-10 VTOC information before the ICKDSF REFORMAT

If you submit a job to move and expand the VTOC into a new location, the Device Manager REFUCB service is invoked to automatically update the UCB on each system in the sysplex that has REFUCB enabled; see Figure 3-11 on page 69.

```
EDIT       NARO.DATA.JCL(ICKDSF1) - 01.03                Columns 000
Command ===>                                               Scroll =
****** ***************************** Top of Data ********************
000100 //NAROICK   JOB (999,POK),'ICK TEST',CLASS=A,REGION=0M,
000200 //             MSGCLASS=T,TIME=10,MSGLEVEL=(1,1),NOTIFY=&SYSUID
000400 //STEP1    EXEC PGM=ICKDSF
000500 //SYSPRINT DD   SYSOUT=*
000700 //SYSIN    DD   *
000800         REFORMAT VERIFY(YY8142) UNITADDR(8142) NVTOC(ANY,105)
****** **************************** Bottom of Data ******************
```

Figure 3-11 ICKDSF REFORMAT job using NEWVTOC

If you access the JOBLOG you can find the Device Manager messages showing the REFUCB function working just after the job has ended; see Figure 3-12 on page 69.

```
SC74 2011119 09:24:42.07 JOB05089 0090  $HASP373 NAROICK  STARTED - INIT
SC74 2011119 09:24:42.13 JOB05089 0090  ICK091I   8142 NED=002105.000.IB
SC74 2011119 09:24:42.13 JOB05089 0090 *003 ICK003D REPLY U TO ALTER VOL
SC74 2011119 09:24:48.96 NARO     0290  D R,L
SC74 2011119 09:24:48.97 NARO     0090  IEE112I 09.24.48 PENDING REQUEST
                              531 0090  RM=1    IM=0     CEM=0     EM=0
                              531 0090  ID:R/K     T SYSNAME  MESSAGE TE
                              531 0090         003 R SC74     *003 ICK00
                              531 0090                        CONTENTS,
SC74 2011119 09:24:54.42 NARO     0290  R 003,U
SC74 2011119 09:24:54.42 JOB05089 0090  IEE600I REPLY TO 003 IS;U
SC74 2011119 09:24:55.19 JOB05089 0090  $HASP395 NAROICK  ENDED
SC75 2011119 09:24:55.19          0290  DMO0061I 8142,YY8142,REFUCB STARTED
SC75 2011119 09:24:55.21 INTERNAL 0290  IEE302I 8142     ONLINE BY DMOAT002
SC75 2011119 09:24:55.21          0290  DMO0062I 8142,YY8142,REFUCB SUCCESSFUL
```

Figure 3-12 REFUCB working after ICKDSF REFORMAT NEWVTOC

Note: If you receive an error message about a free space error when you try to access information about the VTOC shortly after a ICKDSF REFORMAT NEWVTOC, this is fixed at the first allocation of a data set on the target volume.

The VTOC information of the volume YY8142 on system SC75 just after the ICKDSF job has ended is shown in Figure 3-13 on page 70.

```
Volume . : YY8142
Command ===>

Unit . . : 3390

 Volume Data             VTOC Data              Free Space   Tracks
 Tracks . :   50,085     Tracks  . :       105  Size  . . :  49,979
 %Used  . :        0     %Used . . :         1  Largest . :  49,965
 Trks/Cyls:       15     Free DSCBS:     5,248  Free
                                                Extents . :       2
```

Figure 3-13 VTOC information after the ICKDSF REFORMAT

Device Manager messages

The following Device Manager messages are related to the REFUCB function.

DMO0061I message

```
DMO0061I dddd,vvvvvv, REFUCB STARTED
```

Explanation: Device support software has detected that the VOLSER or VTOC location for the volume has changed because of one of the following operations:

- DSS COPY
- RESTORE
- ICKDSF REFORMAT NEWVTOC

The Device Manager UCB update service has issued an **UNCONDITIONAL VARY ONLINE** command that will cause the VOLSER and VTOC location to be updated. The **UNCONDITIONAL VARY** command is only executed if the device is already ONLINE.

DMO0062I message

```
DMO0062I dddd,vvvvvv, REFUCB SUCCESSFUL
```

Explanation: Device support software has detected that the VOLSER or VTOC location for the volume has changed because of one of the following operations:

- DSS COPY
- RESTORE
- ICKDSF REFORMAT NEWVTOC

The **UNCONDITIONAL VARY ONLINE** command that was issued by the Device Manager was successful.

DMO0063E message

```
DMO0063I dddd,vvvvvv, REFUCB FAILED
```

Explanation: Device support software has detected that the VOLSER or VTOC location for the volume has changed because of one of the following operations:

- DSS COPY
- RESTORE
- ICKDSF REFORMAT NEWVTOC

The **UNCONDITIONAL VARY ONLINE** command that was issued by the Device Manager failed.

4

XCF enhancements

The cross-system coupling (XCF) services provide the following functions that a multisystem application or subsystem programmer can use:

- A way to define a collection of unique parts of a program, and a way for each part to identify the other parts so they can work together.
- A way for program parts to send messages to or receive messages from other parts on the same MVS system or on a different one, without regard for the I/O considerations involved. Messages can be sent without knowing specifically where the receiving part resides.
- A way to monitor the program parts that you as the programmer define to XCF. XCF maintains information about the parts you define, and provides notification of changes. These parts can be on the same MVS system or on different MVS systems.
- A way to design your program for high availability so that primary parts are on one system and backup parts are on another system. When the primary system fails, XCF notifies the backup parts on the other system and the backup parts can be designed to take over the function of the primary. The primary and backup parts can also be running in different address spaces on the same system. In this case, the parts running in the backup address space can be designed to take over when the primary address space fails.
- A way to allow batch jobs and started tasks to be restarted automatically. You can use the XCF recovery function, Automatic Restart Management (ARM), to design your application for high availability by allowing it to be restarted automatically when it fails or the system it is running on fails.

This chapter describes the following changes with z/OS V1R13:

- Large fixed CTRACE buffer size
- CF structure enhancements
  - Support for greater than 32 CF structure connectors
  - Support for greater than 1024 CF structures
- AutoAlter reclaim avoidance
- DISPLAY XCF,SYSPLEX enhancement
- “CF not suitable for allocation” preference list reasons
- Controlling CF structure ALTER processing - OA34579

4.1 Large fixed CTRACE buffer size

The default component buffer size, 1 MB for XCF and 168 KB for the XES connector, has been shown to be too small in certain circumstances to contain a complete correlation thread of CTRACE events. Buffer wrapping occurs prematurely and truncates trace thread correlation.

In z/OS V1R13 the following changes have been made:

- Larger default trace buffers are allocated at component initialization:
  - 4 MB for XCF
  - 336 KB for XES Connector
- The BUFSIZE keyword has been removed from CTIXCF00 and CTIXES00 parmlib members to facilitate “best practice” buffer sizes and avoid using down-level default.
- There is an increased “minimum tracing” base in XCF, focusing on major GROUP services and tracing events.
- Additional MODID filtering provides focused tracing at a module level.

These changes reduce the chances of “buffer wrapping” and the loss of critical correlation data. The need to recreate problems and capture diagnostics again due to trace buffers wrapping or because the needed trace option is not enabled is reduced. The MODID filtering provides trace event isolation for specific areas or a subset of processes within an XCF subcomponent.

Note: The CTIXCFxx and CTIXESxx parmlib members will override the default buffer size (BUFSIZE). To use the “best practice” default buffer sizes for V1R13, remove the BUFSIZE keyword.

4.2 CF structure enhancements

This section discusses the following CF structure-related enhancements:

- Support for greater than 1024 CF structures
- Support for greater than 32 CF structure connectors

Note: These enhancements are grouped together in this section because both of them are included in APAR OA32807 for z/OS V1R10 to z/OS V1R12.

4.2.1 Support for greater than 1024 CF structures

New function in z/OS V1R13 allows exploiters to increase the limit on the number of structures per CF from 1024 up to 2048 with CFLEVEL 17. There has been a constant increase in the supported number of CF structures ever since sysplex was first implemented. Originally only 64 CF structures were supported. As sysplex exploitation expanded, the number of CF structures supported increased to 256, 512, 1024, and now 2048.

This allows exploiters to utilize the increased capacity on provided by CFLEVEL 17.

4.2.2 Support for greater than 32 CF structure connectors

CFLEVEL 17 increased the number of list and lock CF structure connectors from 32 to 255 per structure. A sysplex can contain only 32 z/OS images but more than one exploiter instance (connector) can be defined on the same z/OS image to effectively utilize the available capacity of the image. This means the total number of connections to a structure can exceed 32.

A CFLEVEL 17 or higher CF will support 255 connectors for all structure types but there are z/OS limitations as follows:

- The z/OS lock table entry size limits the number of connectors to serialized lock structures to 247. The lock table entry size is 32 bytes, but 1 byte of the lock table entry is used as a global byte, thus leaving only 247 spare bits.
- The z/OS lock table entry size limits the number of connectors to serialized list structures to 127. The lock table entry size is 1 byte with the high order bit as a “transition bit” for transferring locks, thus leaving only a '7F'x number of unique connectors.

New function in z/OS V1R13 provides a new external on the existing IXLCONN interface to allow connectors to specify more than 32 connectors to list and lock structures. This allows more multisystem exploiter instances to utilize the capacity of the image.

IXLCONN changes - LIST structure

MAXCONN(xmaxconn | 32) is a new keyword that specifies the maximum number of users that can connect and use the list structure.

Note: For more information, see *z/OS MVS Programming: Sysplex Services Reference*, SA22-7618.

The actual number of connections supported by the structure is returned in the Connect Answer Area field, ConaCFacilityUserLimit. When the requested MAXCONNS attribute is between 0 and 32, inclusive, the value used will be 32. The default, if not specified, is 32.

To allocate a list structure that can support more than 32 connectors, use the MAXCONN keyword as follows:

- Ensure that the MAXCONN keyword is used by all connectors to the structure.
- Ensure that the structure is allocated in a coupling facility with CFLEVEL=17 or higher.

Specifying the MAXCONN keyword with any value indicates that the connection can support a user-id limit change that results from a system-managed process (for example, rebuild). XCF communicates the user-id limit change through the structure state change event. Specifying the MAXCONN keyword with a value greater than 32 indicates that the connector can understand events for subject connections with connection identifiers up to the value specified (for example, EeplExistingConnection or EeplNewConnection).

IXLCONN changes - LOCK structure

The new MAXCONN(xmaxconn) keyword specifies the maximum number of users that can connect and use the list structure. It is mutually exclusive to NUMUSERS(xnumusers).

Note: For more information, see *z/OS MVS Programming: Sysplex Services Reference*, SA22-7618.

NUMUSERS(xnumusers)| MAXCONN(xmaxconn) is a required set of mutually exclusive keywords that specify the maximum number of users that may connect and use the lock structure. The actual number of connections supported by the structure is returned in the Connect Answer Area field, ConaLockNumUsers.

Note: When the NUMUSERS keyword is used by any connector to a specific structure, the actual number of connections that will be supported by the structure will not go above 32. To allocate a structure that can support more than 32 connectors, use the MAXCONN keyword.

To allocate a lock structure that can support more than 32 connectors, use the MAXCONN=maxconns keyword as follows:

- Ensure that the MAXCONN keyword is used by all connectors to the structure.
- Ensure that the structure is allocated in a coupling facility with CFLEVEL=17 or higher.

Specifying the MAXCONN keyword with any value indicates that the connection can support a user-id limit change that results from a system-managed process (for example, rebuild). XCF communicates the user-id limit change through the structure state change event. Specifying the MAXCONN keyword with a value greater than 32 indicates that the connector can understand events for subject connections with connection identifiers up to the value specified (for example, EeplExistingConnection or EeplNewConnection).

When you migrate applications to use greater-than-32 connectors support in a rolling fashion, the application or installation can take the following steps to avoid failed connection attempts:

- Ensure that applications have internal structures coded to accept events for connection identifiers up to the planned value that is to be used on the MAXCONN keyword.
- Ensure that all instances of the application are updated to use the MAXCONN keyword. When all instances of the application have been upgraded to a level that specifies MAXCONN, the structure is eligible to accommodate more than 32 instances of the application.
- Ensure that the structure is reallocated, to take advantage of greater-than-32 connectors in a CFLEVEL 17 coupling facility. You can accomplish this through the rebuild process.

Message IXC360I changes

The maximum number of connections is displayed for both **D XCF** structure detail and **D XCF** structure connector detail commands.

Figure 4-1 shows the output to a **D XCF** structure detail command where:

- MAX CONNECTIONS is the maximum connections allowable for the structure from the NUMUSERS or MAXCONN specification of the first connector. It is bounded by:
  - The architected limits (255 for list, 247 for lock, 127 for serialized list).
  - The model-dependent user ID for the CF.
  - The number of CONNECT records for which the CFRM couple data set was formatted to support.
- # CONNECTIONS is the current number of connections for the structure.

```
D XCF,STR,STRNM=DBSVPLX4_LOCK1
IXC360I  10.50.54  DISPLAY XCF 194
STRNAME: DBSVPLX4_LOCK1
ALLOCATION TIME: 05/21/2010 08:47:52
 CFNAME         : X4CFH92
 COUPLING FACILITY: 002097.IBM.02.0000000BF715
                    PARTITION: 0F   CPCID: 00
 ACTUAL SIZE    : 125 M
           .
 SYSTEM-MANAGED PROCESS LEVEL: 8
 XCF GRPNAME    : IXCLO072
             .
 MAX CONNECTIONS: 23
 # CONNECTIONS  : 7
```

Figure 4-1 IXC360I for a Display XCF structure detail command

Figure 4-2 shows the output to a Display XCF structure connector detail command where:

- NUMUSERS: xx is the number of users specified by the connection on the IXLCONN request to a lock structure.
- MAXCONN(USER): xxxxx is the number of users specified by the connection. USER indicates that the MAXCONN was specified by the connector on the IXLCONN request to either a list or lock structure.
- MAXCONN(DEFAULT): xxxxx is the number of users specified by the connection. DEFAULT indicates that MAXCONN was not specified by the connector on the IXLCONN to a list structure.

```
D XCF,STR,STRNAME=ISGLOCK,CONNAME=ALL
IXC360I  10.50.54  DISPLAY XCF 194
STRNAME: ISGLOCK
          .
          .
 CONNECTION NAME  : ISGLOCK#SY1
           .
           .
     TERMLEVEL           : SYSTEM
      CRITICAL                : YES
      ALLOW ALTER      : NO
      MAXCONN(USER) : 48

D XCF,STR,STRNAME=TESTLIST,CONNAME=ALL
IXC360I  10.50.54  DISPLAY XCF 194
STRNAME: TESTLIST
          .
          .
 CONNECTION NAME  : TESTLIST#SY1
           .
           .
     TERMLEVEL                  : SYSTEM
      CRITICAL                       : YES
      ALLOW ALTER             : NO
      MAXCONN(DEFAULT) : 32
```

Figure 4-2 IXC360I for a Display XCF structure connector detail commands

Message IXC522I changes

There are two new inserts for message IXC522I related to greater than 32 connector support, as shown in Figure 4-3.

```
IXC522I SYSTEM-MANAGED DUPLEXING REBUILD FOR STRUCTURE strname IS BEING STOPPED
TO SWITCH TO THE NEW|OLD STRUCTURE DUE TO
   STRUCTURE INSTANCE DID NOT HAVE ANY AVAILABLE CONIDS
   CONNECTOR DID NOT ALLOW USER LIMIT CHANGES
```

Figure 4-3 Message IXC522I inserts

Where:

- `STRUCTURE INSTANCE DID NOT HAVE ANY AVAILABLE CONIDS`

  The duplexing rebuild was stopped because a structure instance did not have any available connection identifiers.

- `CONNECTOR DID NOT ALLOW USER LIMIT CHANGES`

  The duplexing rebuild was stopped because a connector connected and did not specify MAXCONN on the IXLCONN invocation.

Migration and coexistence considerations

Before upgrading to a z196 and coupling facility CFLEVEL = 17, you must install the following toleration APARs:

- OA33922 HBB7720 – HBB7730
- OA34044 HBB7740 – HBB7770

Before installing z/OS V1R13 or APAR OA32807 on any system in the sysplex you must install coexistence APARs on every system in the sysplex:

- OA34060 HBB7720 – HBB7740
- OA34061 HBB7750 – HBB7770

Important: The toleration APAR OA34061 has a HOLDERROR described by OA35961. However, the PTFs for OA35961 have a PREREQ of the PTFs for OA32807. This means that OA34061 must be applied without APAR OA35961, thus leaving systems exposed to an 8-byte storage fragmentation problem. Typically the growth is rather slow, but nevertheless undesirable. It is not possible to safely install the PTFs for both APAR OA34061 and APAR OA35961 around the sysplex at the same time. APAR OA35961 brings in APAR OA32807 through APAR OA34391, and APAR OA32807 cannot be installed at the same time as OA34061.

To safely move forward, apply the PTF for APAR OA34061 and apply the ++APAR for the appropriate z/OS release as noted in APAR OA35961. Then, in the ++APAR statement, change the ++APAR(xA35961) to ++APAR(AA35961). Using ++APAR(AA35961) will resolve the HOLDERROR(AA35961). After OA34061 and the ++APAR(AA35961) have been installed on all systems in the sysplex, a future SYSRES containing PTFs for APAR OA32807, APAR OA35961, and APAR OA34391 can be rolled out.

In summary, there are two choices regarding how to install the preconditioning APAR OA34061 to all systems before installing APAR OA32807:

1. Install APAR OA34061, bypassing the HOLDERROR for AA35961. This will leave the system exposed to the 8-byte storage fragmentation problem.
2. Install APAR OA34061 together with the ++APAR for AA35961.

APAR OA34060 has the same 8-byte storage fragmentation problem, but is not flagged as PE because HBB7720 to HBB7740 are out of support.

4.3 DISPLAY XCF,SYSPLEX enhancement

The **D XCF,SYSPLEX** command is a popular command used to display the status of the sysplex and systems within the sysplex. A significant amount of useful system and sysplex status information is kept by XCF, but not displayed in a central place.

In z/OS V1R13, the **D XCF,SYSPLEX** command now provides the output that was previously provided by the **D XCF,S,ALL** command. In addition, the **D XCF,SYSPLEX** command expands and reformats the output to fields such as node descriptor, z/OS release level, and join time.

This provides more information about systems in the sysplex consolidated into one command to enhance usability and diagnostic data capture.

Message changes

Message IXC335I has been removed and is replaced by new messages IXC336I and IXC337I.

- For a **D XCF** command, message IXC334I is issued as in prior releases.

- For the **D XCF,SYSPLEX** command, the new message IXC336I is issued instead of IXC334I.
- For **D XCF,SYSPLEX {,ALL | system name}** command, the new IXC337I message is issued instead of IXC335I.

In releases prior to z/OS V1R13, the **D XCF** command defaulted to **D XCF,SYS**.

With z/OS V1r13, there are changes to the following commands:

- The **D XCF** command is the same as prior releases.
- The **D XCF,SYS** command is the same as the **D XCF,ALL** command in prior releases.
- The **D XCF,SYS,ALL** command has new detail.

Displaying the difference with V1R12 and V1R13

The following figures show the various **D XCF** command outputs for z/OS V1R12 and z/OS V1R13.

- Figure 4-4 on page 78 shows the output for the **D XCF** command on z/OS V1R12 and z/OS V1R13.

```
D XCF on z/OS V1R12
IXC334I  01.51.16  DISPLAY XCF 254
   SYSPLEX PLEX75:     SC74              SC75

D XCF on z/OS V1R13
IXC334I  11.46.59  DISPLAY XCF 885
   SYSPLEX PLEX75:     SC74              SC75
```

Figure 4-4 D XCF display

- Figure 4-5 on page 78 shows the output for **D XCF,SYSTEM** command on z/OS V1R12 and z/OS V1R13.

```
D XCF,S on z/OS V1R12
IXC334I  01.55.14  DISPLAY XCF 260
   SYSPLEX PLEX75:     SC74              SC75

D XCF,S on z/OS V1R13
IXC336I  11.53.19  DISPLAY XCF 935
SYSPLEX PLEX75
SYSTEM   TYPE SERIAL LPAR STATUS TIME         SYSTEM STATUS
SC74     2817 3BD5   05   05/10/2011 11:53:19 ACTIVE         TM=STP
SC75     2097 DE50   2C   05/10/2011 11:53:16 ACTIVE         TM=STP

SYSTEM STATUS DETECTION PARTITIONING PROTOCOL CONNECTION EXCEPTIONS:
  SYSPLEX COUPLE DATA SET NOT FORMATTED FOR THE SSD PROTOCOL
```

Figure 4-5 D XCF,SYSPLEX display

- Figure 4-6 on page 79 shows the output for the **D XCF,SYSTEM,ALL** command on z/OS V1R12 and z/OS V1R13.

```
D XCF,S,ALL on z/OS V1R12
IXC335I  01.58.29  DISPLAY XCF 262
SYSPLEX PLEX75
SYSTEM   TYPE SERIAL LPAR STATUS TIME         SYSTEM STATUS
SC74     2817 3BD5   05   05/11/2011 01:58:28 ACTIVE         TM=STP
AAIS     2097 DE50   2C   05/11/2011 01:58:26 ACTIVE         TM=STP

D XCF,S,ALL on z/OS V1R13
IXC337I  11.57.32  DISPLAY XCF 957
SYSPLEX PLEX75           MODE: MULTISYSTEM-CAPABLE

 SYSTEM SC74          STATUS: ACTIVE
                      TIMING: STP CTNID: ITSOPOK
                STATUS TIME: 05/10/2011 11:57:32.042885
                  JOIN TIME: 04/29/2011 07:08:07.839841
              SYSTEM NUMBER: 010001DB
           SYSTEM IDENTIFIER: 3BD52817 050001DB
             NODE DESCRIPTOR: N/A
                              PARTITION: N/A    CPCID: N/A
                     RELEASE: N/A

 SYSTEM SC75          STATUS: ACTIVE
                      TIMING: STP CTNID: ITSOPOK
                STATUS TIME: 05/10/2011 11:57:31.082571
                  JOIN TIME: 04/29/2011 07:15:27.997632
              SYSTEM NUMBER: 020001DC
           SYSTEM IDENTIFIER: DE502097 2C0001DC
             NODE DESCRIPTOR: N/A
                              PARTITION: N/A    CPCID: N/A
                     RELEASE: N/A
```

Figure 4-6 D XCF,S,ALL output

Migration and coexistence considerations

Automation that references output from the following commands needs to be modified.

D XCF,SYSPLEX and **D XCF,SYSPLEX,ALL** or system name.

The following message changes also need to be considered:

- New message ICX336I is issued for the **D XCF,S** command.
- Message IXC335I is no longer issued for the **D XCF,S,ALL** command and is replaced by message IXC337I.

Note: Results from a **D XCF** command that is routed to all systems in a sysplex with mixed systems will have different message numbers for the z/OS V1R13 system and for systems that are not z/OS V1R13 systems.

4.4 CF not suitable for allocation preference

In the ongoing effort to make z/OS simpler to use, XCF in z/OS V1R13 provides client-usable explanations of the XCF reasons for making its CF structure allocation placement decisions.

The following messages have new inserts:

- Message IXL015I: Initial/rebuild structure allocation
  - Also has “CONNECTIVITY=” insert
- Message IXC347I: Reallocate/Reallocate test results
- Message IXC574I: Reallocate processing, system-managed rebuild processing, or duplexing feasibility

The new inserts are a sub-reason below the CF allocation reasons:

```
Preferred CF already selected
Preferred CF 2
```

XCF analyzes all available CFs from the preference list and ranks them according to weighted attributes that they have or do not have. If two CFs have equal attributes, then they are ranked by their order in the preference list. The insert shown for each CF shows why that CF was positioned below the previous one in the eligibility queue.

New message inserts and explanations

The following list shows the new inserts into existing messages, and their meaning for messages IXL015I, IXC347I, and IXC574I:

CONNECTIVITY REQUIREMENT MET BY PREFERRED CF:
At least one connector to the current (old) structure does not have connectivity to this coupling facility, and there is at least one coupling facility to which all connectors do have connectivity.

CFLEVEL REQUIREMENT MET BY PREFERRED CF
Another coupling facility was found with a more appropriate CF level.

FAILURE ISOLATION FOR DUPLEXING MET BY PREFERRED CF
This coupling facility is not failure isolated for duplexing, this is a duplexing rebuild, and there is at least one coupling facility that is failure isolated for duplexing.

SPACE REQUIREMENT MET BY PREFERRED CF
This coupling facility does not have enough free space to meet the requested structure size, and there is at least one coupling facility that does have enough free space to allocate the structure at the requested size.

SPACE AVAILABLE FOR MINIMUM SIZE IN PREFERRED CF
This coupling facility does not have enough free space to meet the minimum required structure size to allocate the new structure instance based on the current object counts, and there is at least one coupling facility that does have enough free space to allocate the structure at the minimum size.

SPACE AVAILABLE FOR CHANGED DATA IN PREFERRED CF
This coupling facility does not have enough free space to meet the minimum required structure size to allocate the new structure instance based on the current in-use and changed object counts, and there is at least one coupling facility that does have enough free space to allocate the structure with changed data only.

MORE SPACE AVAILABLE IN PREFERRED CF
This coupling facility does not have enough free space to allocate the structure at the requested size, and there is another coupling facility that also does not have enough free space but has more space than this one.

NON-VOLATILITY REQUIREMENT MET BY PREFERRED CF
This coupling facility is volatile, non-volatility was requested, and there is at least one coupling facility that is non-volatile.

FAILURE ISOLATION REQUIREMENT MET BY PREFERRED CF
This coupling facility is not failure isolated from all connectors, non-volatility was requested, and there is at least one coupling facility that is failure isolated from all connectors.

STAND-ALONE REQUIREMENT MET BY PREFERRED CF
This coupling facility is not stand-alone, non-volatility was requested, and there is at least one coupling facility that is stand-alone.

EXCLLIST REQUIREMENT FULLY MET BY PREFERRED CF
This coupling facility contains a structure from the EXCLLIST, and there is at least one coupling facility that does not contain any structures from the EXCLLIST.

EXCLLIST REQUIREMENT MET BY PREFERRED CF
This coupling facility contains a simplex structure from the EXCLLIST, and there is at least one coupling facility that contains only old or new structure instances from the EXCLLIST.

REMOTE FACILITY SPACE REQUIREMENT MET BY PREFERRED CF
System-managed duplexing rebuild is a possibility for the structure, this coupling facility is not connected by CF-to-CF links to any other coupling facilities in the PREFLIST that have adequate space to allocate the structure should a duplexing rebuild be started, and there is at least one coupling facility that is connected by CF-to-CF links to a coupling facility that has adequate space to allocate the structure.

REMOTE FACILITY REQUIREMENT MET BY PREFERRED CF
System-managed duplexing rebuild is a possibility for the structure, this coupling facility is not connected by CF-to-CF links to any other coupling facilities in the PREFLIST, and there is at least one coupling facility that is connected by CF-to-CF links to a remote facility, but the remote facility does not have adequate space to allocate the structure should a duplexing rebuild be started.

PREFERRED CF HIGHER IN PREFLIST
This coupling facility is lower in the PREFLIST than another coupling facility that is suitable for allocation.

ENFORCEORDER(YES) AND PREFERRED CF HIGHER IN PREFLIST
This coupling facility is lower in the PREFLIST than another coupling facility that is suitable for allocation, and since ENFORCEORDER(YES) was specified for the structure in the CFRM policy, XCF did not re-order the PREFLIST.

GREATER SFM WEIGHT CALCULATED FOR PREFERRED CF
This coupling facility has a lower SFM weight than another coupling facility that is suitable for allocation. For a structure without any active connectors, the SFM weight of each coupling facility is the sum of the SFM weights of all systems connected to that coupling facility. For a structure with active connectors, only systems with active connectors are used to determine the SFM weight of the coupling facility. Note that all systems are considered to have equal SFM weight if no SFM policy is active.

4.4.1 Examples of messages with new inserts

The following messages have examples of the new inserts.

Note: The new inserts are highlighted in bold text.

Message IXL015I

Explanation: A program attempted to connect to a coupling facility structure; see Figure 4-7. The IXLCONN returned allocation information to the program in the IXLYCONA. This information is recorded in the hardcopy log for debugging if the results are unexpected. The actual disposition of the IXLCONN invocation can be found by examining message IXL013I (for IXLCONN failures) or message IXL014I (for successful IXLCONN connects).

Note: An IXL message might have been issued prior to this message, providing additional information about where and why the system allocated the structure.

```
IXL015I strtype ALLOCATION INFORMATION FOR
STRUCTURE structure-name CONNECTOR NAME connector-name
CONNECTIVITY=connectivity
CFNAME ALLOCATION STATUS/FAILURE REASON
--------       ---------------------------------------
LF01           ALLOCATION NOT PERMITTED
               COUPLING FACILITY IS IN MAINTENANCE MODE
A              STRUCTURE ALLOCATED CC007B00
TESTCF         PREFERRED CF ALREADY SELECTED CC007B00
               PREFERRED CF HIGHER IN PREFLIST
LF02           PREFERRED CF ALREADY SELECTED CC007300
               EXCLLIST REQUIREMENT FULLY MET BY PREFERRED CF
SUPERSES       NO CONNECTIVITY 98007800
```

Figure 4-7 Message IXL015I with strtype=structure

Important: In the message text for message IXL015I, note the following points:

- `strtype` contains one of the following:

  ```
  STRUCTURE
  REBUILD NEW STRUCTURE
  ```

- `connectivity` can be

  ```
  SYSPLEX or DEFAULT
  ```

```
IXL015I strtype ALLOCATION INFORMATION FOR
STRUCTURE structure-name, CONNECTOR NAME connector-name,
CONNECTIVITY=connectivity
CFNAME         ALLOCATION STATUS/FAILURE REASON
--------       ---------------------------------------
LF01           STRUCTURE ALLOCATED CC005800
A              PREFERRED CF ALREADY SELECTED CC001800
               NON-VOLATILITY REQUIREMENT MET BY PREFERRED CF
LF02           PREFERRED CF ALREADY SELECTED 8C009800
               CFLEVEL REQUIREMENT MET BY PREFERRED CF
TESTCF         PREFERRED CF ALREADY SELECTED 4C001800
               CONNECTIVITY REQUIREMENT MET BY PREFERRED CF
SUPERSES       NO CONNECTIVITY 0800C800
```

Figure 4-8 Message IXL015I with strtype=rebuild new structure

Message IXC347I

In response to a **DISPLAY XCF,REALLOCATE** command, this message provides details of the most recent REALLOCATE process or what can be expected of the subsequent REALLOCATE processing; see Figure 4-9.

```
IXC347I hh.mm.ss DISPLAY XCF
      ...
STRNAME: structure-name
 DUPLEXED STRUCTURE ALLOCATED IN CF(S) NAMED: cfname01 cfname02
 CFNAME  STATUS/FAILURE REASON
 ------  ----------------------------------------------------
 LF01     PREFERRED CF 1
               INFO110: 00000001 AC007800 00010011
 TESTCF   PREFERRED CF 2
          PREFERRED CF HIGHER IN PREFLIST
               INFO110: 00000001 AC007800 00050011
 LF02     PREFERRED CF ALREADY SELECTED
          CFLEVEL REQUIREMENT MET BY PREFERRED CF
               INFO110: 00000001 9C007800 00020010
 A        INSUFFICIENT CONNECTIVITY
               INFO110: 00000000 00000000 00000000

 1 REALLOCATE STEP(S): REBUILD
```

Figure 4-9 Message IXC347I

Message IXC574I

Explanation: For the specified structure, either the REALLOCATE process was evaluating the structure or a structure rebuild process of the specified type was being started or in progress. This message provides further information about the evaluation, feasibility, or allocation decision based on the coupling facilities in the structure's preference list.

With z/OS V1R13, when sorting the CF eligibility queue for all system-managed rebuild processing and reallocate processing, SFM weight is given priority over other attributes. For a structure without any active connectors, the SFM weight of each coupling facility is the sum of the SFM weights of all systems connected to that coupling facility. For a structure with active connectors, only systems with active connectors are used to determine the SFM weight of the coupling facility. All systems are considered to have equal SFM weight if no SFM policy is active. Figure 4-10 shows message IXC5741.

```
IXC574I ALLOCATION INFORMATION FOR SYSTEM-MANAGED DUPLEXING
REBUILD OF STRUCTURE DUPALLOWED01
 AUTO VERSION: C606BE21 DC5BEA80
 CFNAME     STATUS/FAILURE REASON
 ------     --------------------
 LF02       RESTRICTED BY REBUILD OTHER
 SUPERSES   INSUFFICIENT CONNECTIVITY
                   INFO110: 00000000 00000000 00000000
 A          STRUCTURE ALLOCATED
                   INFO110: 00000001 CC005800 00000000
 LF01       PREFERRED CF ALREADY SELECTED
                   INFO110: 00000001 CC004800 00020011
            FAILURE ISOLATION REQUIREMENT MET BY PREFERRED CF
 TESTCF     PREFERRED CF ALREADY SELECTED
                   INFO110: 00000001 C4005800 00020011
            FAILURE ISOLATION FOR DUPLEXING MET BY PREFERRED CF
 A          INSUFFICIENT CONNECTIVITY
                   INFO110: 00000000 00000000 00000000

 1 REALLOCATE STEP(S): REBUILD
```

Figure 4-10 Message IXC574I

4.5 Controlling CF structure ALTER processing - OA34579

Structure ALTER is a non-disruptive process to connectors to the structure being altered. The structure ALTER function requires the following combination of hardware and software support:

- The structure alter function requires a coupling facility with CFLEVEL=1 or higher. You must add one or more coupling facilities with CFLEVEL=1 or higher to the structure's preference list in the CFRM policy. This enables XES to allocate the structure in a coupling facility that supports structure alter.

  Note: The event monitor controls are supported only for keyed list structures allocated in a coupling facility of CFLEVEL=3 or higher.

  For this type of structure:

  - When allocated in a coupling facility of CFLEVEL=3, only the size and the entry-to-element ratio can be altered.
  - When allocated in a coupling facility of CFLEVEL=4 or higher, the size, entry-to-element ratio, and the percentage of EMC storage can be altered.
- The structure alter function requires that MVS SP 5.2 or higher be running on all systems on which applications plan to use the function. All connectors to the structure must have specified ALLOWALTER=YES on the IXLCONN macro.

Severe performance problems can be seen when ALTER processing is internally initiated through IXLALTER at the same time that the logger structure is being accessed to perform reads and asynchronous deletes for logstream offload processing.

In particular, severe IBM IMS™ performance problems have been seen when internally initiated ALTER processing is performed for the IMS common queue server (CQS) logger structure during logstream offload processing.

There is currently no general way to disable CF structure ALTERs. Specifying ALLOWAUTALTER(NO) in the CFRM policy can disable system-initiated ALTER, but there is no way to disable ALTERs started with a program-initiated IXLALTER or an operator-initiated **SETXCF START,ALTER** command.

APAR OA34579

New function is provided by APAR OA34579 to allow internal ALTERs of logger structures to be disabled or enabled by a **SETXCF MODIFY** command.

```
SETXCF MODIFY,STRNAME=strname|ALL,ALTER=DISABLED|ENABLED
```

Important: The ALLOWAUTALTER specification in the CFRM policy determines whether system-initiated ALTER processing will be performed when the FULLTHRESHOLD is reached if the exploiter allows alter. This does not affect whether or not an IXLALTER can be performed.

Logger connects using ALLOWALTER(YES), which means that it can issue IXLALTER regardless of the ALLOWAUTALTER specification. The **SETXCF MODIFY,STRNM=strname,ALTER=DISABLED** command introduced through OA34579 will prevent an alter that is initiated by IXLALTER, even when ALLOWALTER(YES) is specified.

New and updated commands

You can initiate structure ALTER processing either by using the IXLALTER macro or by issuing the **SETXCF START,ALTER** command. The IXLALTER macro allows an authorized user to request a change to the structure's size, entry-to-element ratio, and percentage of storage allocated for event monitor controls (EMCs). The **SETXCF START,ALTER** command allows the operator to request a change only to the structure's size. Recall, however, that a request to contract the structure's size might also affect the entry-to-element ratio and the percentage of EMC storage.

The ALTER status of a CF structure can be changed by commands, as shown:

```
SETXCF MODIFY,STRNAME=strnmpat,ALTER=DISABLED
SETXCF MODIFY,STRNAME=strnmpat,ALTER=ENABLED
      STRNAME=strname
      STRNAME=strprfx*
      STRNAME=ALL | STRNAME=*
```

The ALTER status can be displayed by the following command:

```
DISPLAY XCF,STRUCTURE,ALTER={ENABLED|DISABLED}
```

The response to the following command, **D XCF,STRUCTURE**, is unchanged when ALTER is permitted. However, it has a new insert when ALTER is disabled, as shown here and in Figure 4-14 on page 88:

```
START ALTER NOT PERMITTED
```

The **DISPLAY XCF** and **SETXCF MODIFY** commands have been updated to add an ALTER keyword to display and modify the alter status, respectively.

Figure 4-11 on page 86 shows the syntax for a **DISPLAY XCF** command that includes the ALTER keyword.

```
D XCF[,{PATHIN|PI}]
...
        [,{STRUCTURE|STR}]
|             [,ALTER={DISABLED|ENABLED}]
              [,{STRNAME|STRNM}={(strname[,strname]...)|ALL}]
                 [,{CONNAME|CONNM}={(conname[,conname]...)|ALL}]
                 [,{STATUS|STAT}=...
...
   CONNAME= or CONNM=conname(s)
     Requests that the system display detailed information
     about one or more connectors to a structure. You may
     specify ALL to request information for all connectors to
     the structure.

|  ALTER=DISABLED
|    Display only CF structures for which CF structure alter
|    processing is disabled. Requests to start CF structure
|    alter processing for such structures is not permitted.
|
|  ALTER=ENABLED
|    Display only CF structures for which CF structure alter
|    processing is enabled.
```

Figure 4-11 DISPLAY XCF command syntax

Figure 4-12 on page 86 shows the syntax for **SETXCF MODIFY** command that includes the ALTER keyword.

```
SETXCF MODIFY,{PATHIN,{DEVICE=([/]indevnum[,[/]indevnum]...)}
|               {STRNAME=strname,
|                  ALTER={DISABLED|ENABLED}}
The parameters are:
|STRNAME or STRNM={strname|ALL}
|  Specifies a structure name pattern identifying the name
|  of one or more coupling facility structures that are to be
|  modified.
|
|  ALTER=DISABLED
|    CF structure alter processing is to be disabled for
|    matching structures. Starting CF structure alter
|    processing for such structures will not be permitted.
|    CF structure alter permission status will persist through
|    the re-initialization of the CFRM CDS that is performed
|    for a sysplex-wide IPL. Before disabling CF structure
|    alter, consider setting up systems to issue the
|    D XCF,STR,ALTER=DISABLED command when a system IPLs in
|    order to log the specific structures affected by
|    persistence of CF structure alter disablement.
|  ALTER=ENABLED
|    CF structure alter processing is to be enabled for
|    matching structures.
```

Figure 4-12 SETXCF MODIFY command syntax

New and updated messages

There are changes to a number of messages to support the new ALTER function.

Message ICX300I

Message IXC300I is updated with new text inserts for errors when processing a **SETXCF** system command to modify CF structure alter permission.

Messages IXC359I and IXC360I

Messages IXC359I and IXC360I are updated with a new text insert to provide the status of CF structure alter permission.

```
START ALTER NOT PERMITTED
  CF structure alter processing has been disabled -
  start alter is not permitted.
```

New message IXC556I

New message IXC556I provides the results of CF structure alter permission modifications.

```
IXC556I [SETXCF COMMAND COMPLETED:]
ALTER {ENABLED|DISABLED} FOR strcnt STRUCTURE(S).
```

Figure 4-13 shows an example message IXC556I that is the output of a **SETXCF MODIFY** command to disable the ALTER status.

```
SETXCF MODIFY,STRNAME=*,ALTER=DISABLED
IXC556I SETXCF COMMAND COMPLETED: ALTER DISABLED FOR 128 STRUCTURE(S).
D XCF,STR,ALTER=DISABLED
IXC359I  16.43.44  DISPLAY XCF
STRNAME            ALLOCATION TIME   STATUS                       TYPE
BIGONE                --       --    NOT ALLOCATED
                                       START ALTER NOT PERMITTED
CACHE01          05/12/2011 16:42:34 ALLOCATED                    CACHE
                                       START ALTER NOT PERMITTED
...
```

Figure 4-13 IXC556I for SETXCF MODIFY

Messages IXC531I and IXC591I

Messages IXC531I and IXC591I are updated with a new text insert indicating that alter cannot be started when it is not permitted.

```
IXC531I SETXCF {START|STOP} ALTER REQUEST FOR STRUCTURE strname
 REJECTED. REASON:
 START ALTER NOT PERMITTED

IXC591I AUTOMATIC ALTER REQUEST FOR STRUCTURE strname
 REJECTED. REASON: START ALTER NOT PERMITTED
```

Figure 4-14 on page 88 shows message IXC531I issued in response to an operator-initiated **ALTER** command for a CF structure that has an ALTER status of DISABLED.

```
SETXCF START,ALTER,STRNAME=CACHE01,SIZE=0
IXC531I SETXCF START ALTER REQUEST FOR STRUCTURE CACHE01
REJECTED. REASON:
START ALTER NOT PERMITTED
SETXCF MODIFY,STRNAME=CACHE01,ALTER=ENABLED
IXC556I SETXCF COMMAND COMPLETED: ALTER ENABLED FOR 1 STRUCTURE(S).

D XCF,STR,ALTER=ENABLED,STRNAME=*
IXC360I  16.46.22  DISPLAY XCF
STRNAME: CACHE01
...
```

Figure 4-14 IXC531I issued for SETXCF,START,ALTER

Changes to the IXLALTER macro

The IXLALTER macro now provides a new reason code when CF structure alter cannot be started because it is not permitted.

- New reason code xxxx0C6E for return code C:
  - Equate symbol: IXLRSNCODEALTERNOTPERMITTED
  - Meaning: CF structure alter is not permitted to start. This may be because of a **SETXCF MODIFY ALTER=DISABLED** command. A structure-specific ENF35 signal will be issued when alter is permitted.
  - Action: Retry the IXLALTER request when a structure-specific ENF35 is issued.

Migration and coexistence considerations

There are no migration or toleration APARs required. A system without OA34579 will ignore CF structure ALTER disablement.

Note: No exploiter problems are expected due to new “alter not permitted” responses from IXLALTER and the command **SETXCF START,ALTER**.

The alter permission status is stored in the CFRM CDS and is retained across a sysplex-wide IPL. Message IXC556I is issued when alter permission status is retained across a sysplex-wide IPL:

```
IXC556I ALTER DISABLED FOR 127 STRUCTURE(S).
```

Tip: Log the alter status for CF structures during IPL by using automation to issue the following command:

```
D XCF,STR,ALTER=DISABLED
```

5

Console service enhancements

When planning MVS operations for a system, you must take into account how operators use consoles to accomplish tasks, and how you want to manage messages and commands. Because messages are also the basis of automated operations, understanding message processing in an MVS system can help you plan MVS automation.

z/OS console support can be operated in one of two modes: shared mode and distributed mode.

This chapter describes the enhancements to the following console service functions that are introduced in z/OS V1R13:

- CMDS command extensions
- Make distributed mode be the default console services operating mode
- Message Flood Automation constraint
- SETMF command

5.1 CMDS command extensions

The **CMDS** command is used to display executing and waiting MVS commands, to delete commands that are waiting for execution, or to cancel commands that are executing.

In z/OS V1R13, command processing improvements have been implemented to this command. The **CMDS** command, which among other options can be used to terminate the processing of a particular command, has a new **FORCE** option. As with other **FORCE** commands and keywords, the **FORCE** option will allow you to specify that a command be terminated. Because the effects of forcing command termination are not always predictable, this option can be used only when there is no other option but to IPL.

z/OS V1R13 introduces the following enhancements:

- A new option for the attempt to abnormally end or cancel a command, which was introduced in z/OS V1R12
- A new **FORCE** option of the **CMDS** command, which is now introduced in z/OS V1R13
- An explanation about how to control the use of the **FORCE** option operand

5.1.1 Abnormally ending a command currently executing in z/OS V1R12

The possibility to abnormally end or cancel a command (by using the **CMDS ABEND** command) was implemented in z/OS V1R12. This new behavior allows a command dynamically to also indicate when it is non-abendable.

When you successfully cancel a command by using the **CMDS ABEND** command, the system terminates it with ABEND code 422, reason code 00010301.

However, if the **CMDS ABEND** command is rejected because the command is in a non-abendable state, the system issues message CNZ6002I as shown in Figure 5-1 on page 91.

```
CNZ6002I  COMMAND command WITH ID id NOT ABENDABLE
```

Explanation: A **CMDS ABEND** command was issued to terminate the command, but the command is not abendable.

In the message text:

command This is the command that was specified on the CMDS command.

id This is the command id that was specified on the CMDS command.

System action: The **CMDS ABEND** command is not processed.

Operator response: Try the **CMDS ABEND** command again. If the command is still rejected, contact your system programmer.

System programmer response: If the command you attempt to terminate does not complete, search the problem reporting databases for a fix for this problem. If no fix exists, contact the IBM Support Center about the command you are attempting to terminate.

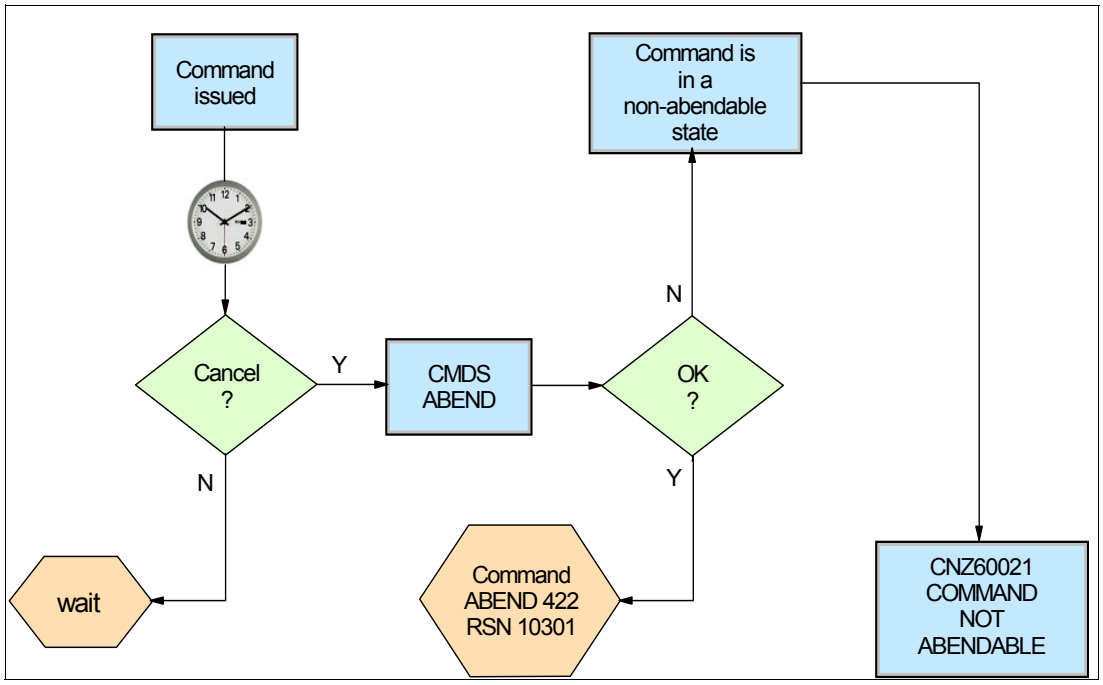


Figure 5-1 Flow of the CMDS ABEND command in R12

5.1.2 Using the new FORCE option with CMDS command in z/OS V1R13

It is best to avoid cancelling a running command, if possible. However, if the only way to prevent an IPL is to terminate the command, then you can use the **CMDS FORCE** command, as implemented in z/OS V1R13.

CMDS command

The **CMDS** command can be used to display executing and waiting MVS commands, to delete commands that are waiting for execution, and to cancel commands that are executing.

The complete syntax for the **CMDS** command is shown in Figure 5-2 on page 91.

```
CMDS {ABEND,CMD=cccccccc,ID=nnnn[,CLASS=classname][,JOB=jobname]}
     {DISPLAY|D[,CLASS=classname][,CMD=cccccccc][,ID=nnnn][,JOB=jobname]}
     {DUMP}
     {FORCE,CMD=cccccccc,ID=nnnn[,CLASS=classname][,JOB=jobname]}
     {REMOVE|R[,CLASS=classname][,CMD=cccccccc][,ID=nnnn][,JOB=jobname]}
     {SHOW|S[,CLASS=classname][,CMD=cccccccc][,ID=nnnn][,JOB=jobname]}
```

Figure 5-2 Syntax of the CMDS command

Where the new parameter is:

FORCE Use this parameter to abnormally end a command that is currently executing. z/OS V1R13 introduces the **FORCE** parameter, which requires subparameters CMD= and ID=. Using these options causes the system to terminate the current processing with an ABEND code 422, reason code 00010302, for the command that CMD=cccccccc and ID=nnnn identifies.

Use the **FORCE** option with extreme caution because you are terminating a command that may be updating critical system data. Use this parameter only as a last resort, such as when a re-IPL is needed if the command is not terminated.

Where existing parameters are:

CMD This is the name of the command.

ID This is the command's sequence number, which appears in the output from a CMDS DISPLAY.

Attention: If the command is considered non-abendable, **FORCE** will still terminate the command and message CNZ6002I will be issued indicating that **FORCE** overrode the abendable setting.

CMD parameter

The **CMD=** command verb specifies the name of the command, as displayed by the **SHOW** option; see Figure 5-5.

The system issues message IEE064I in response to this command. It does not send any response message to the console that issued the abended command.

Attention: Never use the **FORCE** parameter without understanding the following points:

- After issuing **CMDS FORCE** command, you might have to re-IPL the system or, depending on the command being terminated, a sysplex-wide IPL may be required.
- Make sure that the target command is hung, and not simply taking a long time to complete.

New command FORCE option

By using the **CMDS FORCE** command provided in z/OS V1R13, you may terminate a command even if the command indicates that it should not be canceled through a **CMDS ABEND** command. The flow of the **CMDS ABEND/FORCE** command in z/OS V1R13 is shown in Figure 5-3.

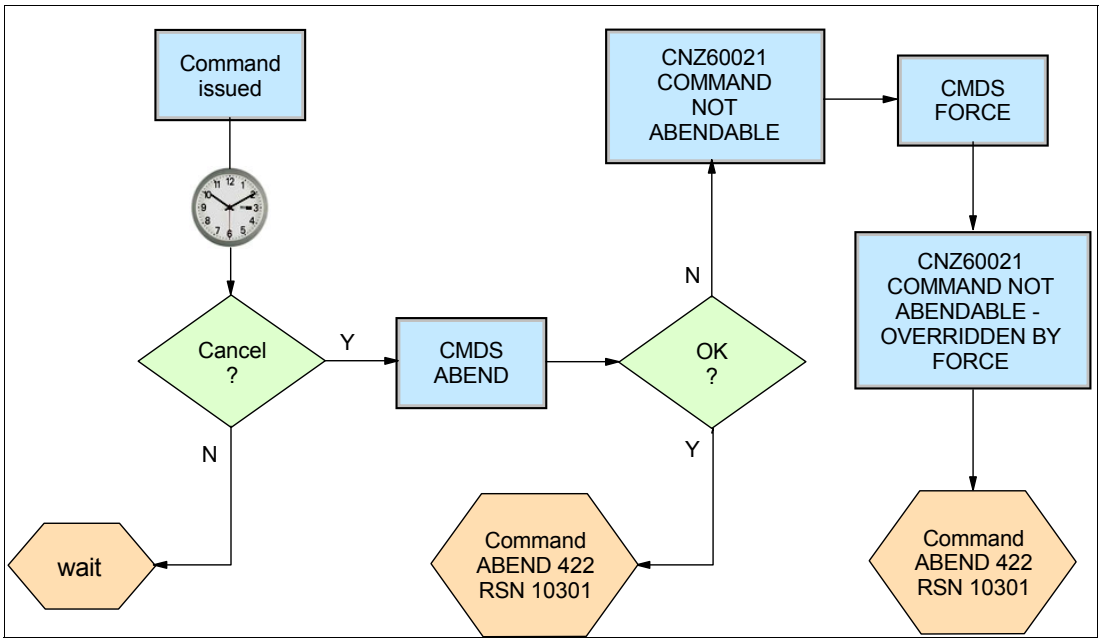


Figure 5-3 Flow of the CMDS ABEND/FORCE command in R13

Changed message CNZ6002I

The message CNZ6002I is changed in z/OS V1R13, indicating that **FORCE** overrode the abendable setting.

```
CNZ6002I COMMAND command WITH ID id NOT ABENDABLE [- OVERRIDDEN BY FORCE]
```

Explanation: A **CMDS ABEND** or a **CMDS FORCE** command was issued to terminate the command, and the command is not abendable.

In the message text:

command This is the command that was specified on the **CMDS** command.

id This is the command id that was specified on the **CMDS** command.

System action: If `OVERRIDDEN BY FORCE` is not displayed, the **CMDS ABEND** command is not processed. If `OVERRIDDEN BY FORCE` is displayed, the **CMDS FORCE** command is processed.

Operator response: Try the **CMDS ABEND** command again. If the command is still rejected, contact your system programmer.

System programmer response: If the command you attempt to terminate does not complete, search the problem reporting databases for a fix for this problem. If no fix exists, contact the IBM Support Center about the command you are attempting to terminate.

5.1.3 CMDS command example

Figure 5-4 on page 94 shows an example of using the **CMDS** command to determine whether there are any active long-running commands. It shows two active commands.

```
CMDS
IEE062I MATCHED: 2        COMMANDS 640
CMDS
MATCHING COMMANDS EXECUTING:  2
COMMAND  COMMAND ID CLASS DATE/TIME ISSUED
SET         1        M1    2011/146 23.47.03
SLIP        9        M1    2011/146 23.47.03
MATCHING COMMANDS WAITING FOR EXECUTION: 0
TOTAL ATTACHED COMMANDS: 2
TOTAL ATTACHED COMMANDS EXECUTING: 2
TOTAL ATTACHED COMMANDS WAITING: 0

CMDS SHOW
MATCHING COMMANDS EXECUTING:  2
 COMMAND NAME: SLIP     COMMAND ID:    8
 SLIP SET,C=222,ID=X222,A=NODUMP,END
 CLASS: M1 ISSUER: *MASTER*,A=0001 STARTED AT: 2011/217 10.27.36
 COMMAND NAME: SET      COMMAND ID:  120
 SET MPF=00
 CLASS: M1 ISSUER: CONSOLE ,A=000B STARTED AT: 2011/229 08.09.05
 MATCHING COMMANDS WAITING FOR EXECUTION: 0
```

Figure 5-4 CMDS command showing active commands

To force the **SET** command to be canceled, first use the **SHOW** operand to determine its characteristics; see Figure 5-5 on page 94.

```
CMDS SHOW,CMD=SET
IEE063I MATCHED: 1        COMMANDS 642
CMDS SHOW,CMD=SET
MATCHING COMMANDS EXECUTING:  1
COMMAND NAME: SET      COMMAND ID:    1
SET MPF=00
CLASS: M1 ISSUER: *MASTER*,A=0001 STARTED AT: 2011/146 23.47.03
MATCHING COMMANDS WAITING FOR EXECUTION: 0
IAT6395 00002 REQUEST(S) - WAITING FOR A MAIN, CLASS, OR GROUP
IEA989I SLIP TRAP ID=X422 MATCHED.  JOBNAME=BPXOINIT, ASID=001D.
```

Figure 5-5 CMDS command with the SHOW operand

Figure 5-6 on page 95 shows the new option, **FORCE**, being used to cancel the **SET** command. Notice that the following messages are issued:

```
IEF196I IEA848I NO DUMP WAS PRODUCED FOR THIS ABEND, DUE TO SYSTEM
IEF196I INSTALLATION REQUEST
```

```
CMDS FORCE,CMD=SET,ID=1
CNZ6002I COMMAND SET WITH ID 1 NOT ABENDABLE - OVERRIDDEN BY FORCE.
IEE064I MATCHED: 1        COMMANDS 645
CMDS FORCE,CMD=SET,ID=1
   1      SET
IEA848I NO DUMP WAS PRODUCED FOR THIS ABEND, DUE TO SYSTEM OR
INSTALLATION REQUEST
IEA989I SLIP TRAP ID=X422 MATCHED.  JOBNAME=*MASTER*, ASID=0001.
IEF196I IEA848I NO DUMP WAS PRODUCED FOR THIS ABEND, DUE TO SYSTEM
IEF196I INSTALLATION REQUEST
IEE480I SET COMMAND ABEND 422-00010302, TERMINATED
IEE480I SET COMMAND ABEND 422-00010302, TERMINATED
IEA989I SLIP TRAP ID=X422 MATCHED.  JOBNAME=BPXOINIT, ASID=001D.
```

Figure 5-6 CMDS command using the FORCE operand to cancel the command

5.1.4 Controlling the use of the FORCE operand

In addition to the **FORCE** option implementation, new support allows command processors to specify whether the **CMDS** command should terminate its commands without the use of **FORCE**.

You can control which groups of users (system programmers and operators) can issue commands. You can also RACF to authorize or restrict users from entering some or all commands, or specific variations of commands, or the consoles from which commands can be entered.

OPERCMDS class profile

A security profile in the OPERCMDS class can be used to limit the use of the **CMDS FORCE** command to authorized users. This new function helps improve system availability by making it clear which commands are intended to support termination at any time and which are not.

To control usage of the **FORCE** operand, the new security profile MVS.CMDS.FORCE will be checked. Table 5-1 on page 95 lists the information that is used to protect the **CMDS FORCE** command.

Table 5-1 Information to protect the use of the FORCE option

MVS commands, RACF authorities, and resource names		
Command/Keyword	Authority	Resource name
CMDS FORCE	CONTROL	MVS.CMDS.FORCE

5.2 Distributed mode as default console operating mode

Distributed mode console support was introduced during a console restructure in z/OS V1R10. However, continuing to support two modes of operation greatly increases console component size, complexity, and service load. Therefore, shared mode console support will be removed in a future release.

To encourage moving to distributed mode, the default has been changed to distributed mode in z/OS V1R13. If the mode of console operation is not explicitly specified, the system will now default to distributed mode console support rather than shared mode console support.

Note: The default is only applied when the first system in a sysplex (or the only system) is IPLed. Installations that currently explicitly specify their mode of console operation are unaffected.

Migration health check

A migration health check was implemented for z/OS V1R11 and z/OS V1R12 through APAR OA32930. This is intended to encourage explicitly specifying the console mode. This health check becomes a “best practices” health check on z/OS V1R13 to encourage clients to migrate to distributed mode if they are found to be operating in shared mode.

Figure 5-7 shows a display of the CNZ_CONSOLE_OPERATING_MODE health check on a z/OS V1R13 system where the console support mode was allowed to default.

```
CHECK(IBMCNZ,CNZ_CONSOLE_OPERATING_MODE)
START TIME: 04/29/2011 07:08:59.508239
CHECK DATE: 20100101  CHECK SEVERITY: LOW

CNZHS0014I Console Services is running in the preferred operating mode
of Distributed.

END TIME: 04/29/2011 07:08:59.508645  STATUS: SUCCESSFUL
```

Figure 5-7 Display of the CNZ_CONSOLE_OPERATING_MODE health check

Clients wishing to continue to use shared mode console support must explicitly request shared mode by using the IEASYSxx CON = modespec parameter and specifying SHARED.

5.3 Message Flood Automation constraint

The Message Flood Automation constraint enhancement in z/OS V1R13 relieves many of the constraints that affect the usability of the Message Flood Automation facility.

Message Flood Automation detects and handles message floods caused by device or software failures. It uses policy created by the installation to determine when a message flooding situation is occurring, and what actions to take.

Changing the message table

The message table is currently handled by a built-in table that requires a code change to alter. In z/OS V1R13, the MSGFLDxx parmlib member is read much earlier during system initialization, thereby removing the need for the built-in table. This allows the messages to be handled during early system initialization to be defined by standard MSGFLDxx parmlib member processing.

The CONSOLxx INIT statement allows a MSGFLDxx parmlib member to be automatically loaded and optionally enabled during system initialization. This eliminates the need for an operator or automation to enter a **SET MSGFLD=xx** command, followed by a **SETMF ON** command to initialize Message Flood Automation. The initial Message Flood Automation MSGFLDxx parmlib member may now be specified in the same way as the initial MPF parmlib member is specified, as follows:

```
MSGFLD(xx[,(wrd)])
```

Where:

xx xx is a two-character MSGFLDxx parmlib member suffix or the word NONE.

NONE NONE indicates that no MSGFLDxx parmlib is to be loaded.

wrd wrd is optional and is either the word ON or OFF (OFF is the default)

If ON is specified, Message Flood Automation is immediately enabled.

Figure 5-8 shows examples of messages issued when the parmlib member is processed.

```
MSGFLD(00)        -- MSGFLD00 loaded but not enabled
MSGFLD(00,(ON))   -- MSGFLD00 loaded and enabled
MSGFLD(NONE,ON)   -- built-in internal policy is used and enabled
```

Figure 5-8 Messages issued when parmlib member is processed

Specifying more than 50 SPECIFIC messages

There is currently a limit of 50 for the number of SPECIFIC messages that can be defined. The messages required by Basic IBM HyperSwap® and GDPS consume 35 of the 50 SPECIFIC messages, leaving only 15 for other uses.

There are many messages for which it is desirable to prevent Message Flood Automation from taking action because they are critical to system automation, but for which the use of the IGNORE option is not possible due to the SPECIFIC message limit of 50.

In z/OS V1R13, up to 1024 SPECIFIC messages can be specified. This improves the usability of Message Flood Automation in Basic HyperSwap, GDPS, and automation environments.

The Message Flood Automation MSGFLDxx parmlib member DEFAULTCMD statement now supports the optional symbolic substitution of an ASID into the command text, as shown here:

```
DEFAULTCMD 'jobchar[asidchar],command-text
```

Where:

jobchar jobchar is a single character that will be replaced by the jobname the first time that it is found within the command-text.

asidchar asidchar is optional and is a single character that will be replaced by a 4-digit hexadecimal ASID the first time that it is found within the command-text.

asidchar cannot be the same as jobchar.

command-text command-text is any system or subsystem command.

Figure 5-9 shows examples of using the DEFAULTCMD statement.

```
DEFAULTCMD '&,CANCEL &'       -- cancels a job solely by jobname
DEFAULTCMD '&%,CANCEL &,A=%'  -- cancels a job by jobname and ASID
```

Figure 5-9 Examples of using the DEFAULTCMD statement

Handling simultaneously flooding address spaces

There is currently a limit of 10 for the number of simultaneously flooding address spaces (for example, CICS regions). In addition, Message Flood Automation is unable to distinguish

between address spaces with the same jobname and can inadvertently take action against the wrong address space.

In z/OS V1R13, up to 128 address spaces to be tracked and acted on simultaneously. The address spaces can be tracked by both jobname and ASID. The ASID can be symbolically substituted into the CMD action command text, allowing it to be directed to a specific ASID. This improves the ability of Message Flood Automation to handle instances of the same program flooding in multiple address spaces.

A subset of the graph can be produced by the Message Flood Automation **D MSGFLD,MSGRATE** command:

```
DISPLAY MSGFLD,MSGRATE [,n][,m]
```

- `n` is the length of the message rate graph to be produced, in lines. Note that `n` defaults to 25, meaning a 25-line graph will be produced.
- `n` must be specified if `m` is specified.
- `n` is rounded to a value producing whole number values for each graph line.
  - For example, 25 yields 4% / line if `m`=100.
- `m` is the fraction of the graph to present, in %.
  - For example, if 5 is specified, then the top 5% of the graph will be presented in the number of lines specified by n.
- `m` defaults to 100, which mean 100% of the graph will be presented.

Figure 5-10 shows examples of using the **D MSGFLD,MSGRATE** command.

```
D MSGFLD,MSGRATE  -- default to 25 lines and all of graph
D MF,MSGRATE,20,5       -- display top 5% of graph in 20 lines
```

Figure 5-10 Examples of using the DISPLAY MSGFLD,MSGRATE command

Increasing the number of policies

Currently, the number of jobs that can have unique Message Flood Automation policies is limited to 10.

In z/OS V1R13, up to 64 REGULAR JOB statements and up to 64 ACTION JOB statements can be specified in a MSGFLDxx parmlib member. This allows clients to have unique Message Flood Automation policies for more jobs.

Displaying Message Flood Automation status

Currently, there is no reliable way to determine when a message flood event is occurring, what is causing the flood, and how long the flood has been happening. Message Flood Automation issues a message when it begins to take action against a flood, and another message when the flood is relieved, but there is no convenient way to tell that you are in the middle of a ongoing flood.

In z/OS V1R13, the output of the **D MSGFLD,STATUS** command has been enhanced to indicate:

- When a message flood is happening
- The type of flood
- The jobname and ASID of the program causing the flood
- How long the flood has been happening

This allows operators to more quickly identify whether they are experiencing a message flood, and if so, who is causing it, so that they can take appropriate action.

Figure 5-12 shows an example of the output from the **D MSGFLD,STATUS** command.

```
D MF,STATUS
SY1  CNZZ042I MSGFLD Status: ENABLED. 146
Policy INITIALIZED.    Using PARMLIB member: MSGFLDKK
Message rate monitoring DISABLED.          0 msgs          0 secs
A message flood is underway.
JOBNAME    ASID T MSGS-ACTED-ON  --DURATION   -------STARTED-----
WTOFLDR    0026 R        129311     2:52.33   2010148 10:07:38.47
WTOFLDR    0033 R Tracking count        2 not > JOBTHRESH       10
WTOFLDA    0025 A        129586     2:53.43   2010148 10:07:38.47
WTOFLDA    0034 R Tracking count        2 not > JOBTHRESH       10
MSG-ID     ASID T MSGS-ACTED-ON  --DURATION   -------STARTED-----
FLDS001I   001F S        129264     2:52.21   2010148 10:07:38.47
FLDS101I   0032 S Tracking count        2 not > MSGLIMIT        10
```

Figure 5-11 Example of the output from the DISPLAY MSGFLD,STATUS command

Setting MFA message rate thresholds

The **D MSGFLD,MSGRATE** command graph does not provide enough message rate detail and can be excessively long. It also does not provide recommended message rate thresholds at high enough percentages of the observed message rates.

In z/OS V1R13, the **D MSGFLD,MSGRATE** command has been enhanced as described here:

- The output can be restricted to the uppermost portion of the message rate graph.
- The **D MSGFLD,MSGRATE** command graph provides threshold recommendations at higher message rate percentages.

This allow clients to more quickly and accurately establish Message Flood Automation thresholds.

The recommended message rate thresholds, which follow the message rate graph produced by the **D MSGFLD,MSGRATE** command, now cover higher fractions of the observed message rates, as shown in Figure 5-12.

```
D MF,MSGRATE,25,2       -- display top 2% of graph in 25 lines
CNZZ043I MSGFLD Message Rates
                 Instantaneous Message Rates
           2007 messages in      1584 seconds     1.267 msg/sec
% of time at msg rate              467 messages w/most common rate
     100.000%|             |    ******
      99.920%|             |   *******
      99.840%|             |   *******
      99.760%|             |  ********
      99.680%|             |  ********
        :                          :
      98.080%|             | *********
            0+--+--+>-+--+|-+--+---+<-+-------------------
             0           1  8 64  1K 8K  messages/second

             Suggested threshold for  99.900 % is      26
             Suggested threshold for  99.750 % is      15
             Suggested threshold for  99.500 % is      12
             Suggested threshold for  99.250 % is       9
             Suggested threshold for  99.000 % is       8
             Suggested threshold for  98.500 % is       6
             Suggested threshold for  97.750 % is       5
             Suggested threshold for  97.000 % is       4
             Suggested threshold for  95.000 % is       3
```

Figure 5-12 Example of the DISPLAY MSGFLD,MSGRATE command

5.4 SETMF command

The **SETMF** command is used to alter message flood automation parameters or actions. With z/OS V1R13, there are changes to the **SETMF** command used to manage message flood automation processing, as follows:

- Enable or disable message flood checking
- Enable or disable message rate monitoring
- Change message flood automation parameters
- Change message flood automation actions

Note: Changes made by **SETMF** command persist only until the next SET MSGFLD= command or IPL.

To make the changes permanent, update the parameter in a MSGFLDxx parmlib member.

Command syntax

The complete syntax for the **SETMF** command is listed here:

```
SETMF [ON|OFF]
[MONITORON|MONITOROFF]
[MSGTYPE=msgtype,keyword=value[,keyword=value]]
[MSGTYPE=msgtype,DEFAULT=action[,action]]
[MSGTYPE={REGULAR|ACTION},JOB=jobname[,action][,action]]
[MSGTYPE=SPECIFIC,MSG=msgid[,action][,action]]
```

The parameters are as follows:

ON This enables message flood checking. The enable switch is set on and processing resumes. Messages received while message flood checking is disabled are not processed and not counted.

OFF This disables message flood checking. No values are changed and when subsequently enabled, processing resumes with the values as set at the time of the disablement command.

MONITORON This enables the collection of message rate information. Gather message rate information for at least an hour before displaying it. In general, the longer the sample, the more accurate the results.

Message rate monitoring incurs more overhead than standard (non-intensive mode) message flood processing and should probably not be run all the time. A 24-hour sample taken during a busy period every few months is probably sufficient.

The largest interval between successive messages that will be recorded by Message Rate Monitoring is 2048 seconds, or approximately 33 minutes. The smallest interval that can be recorded is 250 picoseconds.

Note: Message rate monitoring is turned off and the counters are reinitialized whenever a **SET MSGFLD=xx** command is issued.

MONITOROFF This disables the collection of message rate information. Disabling the collection of message rate information does not reinitialize the information already collected.

For more information, see *z/OS MVS System Commands*, SA22-7627.

Migration and coexistence

The compatibility APAR OA33652 for z/OS R11 and z/OS R12 allows the MSGFLD parameter on the CONSOLxx INIT statement to be tolerated (ignored) on z/OS V1R11 and z/OS V1R12 systems.

6

DFSMSdfp, DFSMSoam, and DFSMShsm

This chapter describes DFSMSdfp, DFSMSoam, and DFSMShsm areas that have been enhanced or improved in z/OS R1V13.

The following DFSMSdfp and DFSMSoam enhancements are covered:

- New PDSE diagnostic commands
- IEBPDSE utility to validate a PDSE data set
- IEBCOPY enhancements
- New catalog parmlib member support
- DASDM dynamic exits
- New option for the LISTCAT LVL command
- OPEN/CLOSE/EOV enhancements
- New OCE diagnostic data
- QSAM MULTDSN calculation with concatenation
- Free tape volumes at end of volume
- XTIOT support for subsystem DCBs and ACBs
- OAM file system support
- OAM usability and reliability enhancements

DFSMShsm is a functional component of the DFSMS family, which provides facilities for managing your storage devices. DFSMShsm is a DASD storage management and productivity tool for managing low-activity and inactive data. It relieves you from having to handle manual storage management tasks and improves DASD use by automatically managing both space and data availability in a storage hierarchy.

DFSMShsm cooperates with the other products in the DFSMSdfp family to provide efficient and effective storage management. DFSMSdfp provides a storage management subsystem (SMS) that allows storage administrators to control the use of storage. The storage

management subsystem provides storage groups, storage classes, management classes, and data classes that control the allocation parameters and management attributes of data sets.

DFSMShsm performs space management and availability management of each data set as directed by the management class attributes of that data set. In addition, the storage group controls the allocation of the data set when DFSMShsm returns the data set to level 0 (L0) storage.

In this chapter, the following topics are discussed:

- CDS Backup improvements
- Space Management Performance improvements
- Small Data Set Packing Performance improvements
- DFSMShsm Serviceability and Usability enhancements

6.1 New PDSE diagnostic commands

In z/OS V1R13, two new PDSE command operands are provided to aid PDSE diagnostics and refresh PDSE in-storage data. In addition, you can use a new IEBPDSE program to validate a PDSE data set and determine whether it is valid or corrupted.

Also in z/OS V1R13, a new IEBPDSE program is introduced that produces the following output:

- A message data set is produced that contains informational messages (for example, if the data set was found to be corrupted), the results of the validation check, and error messages.
- IEBPDSE is controlled by job control statements. Utility control statements are not used.

6.1.1 PDSE diagnostic aids

Although PDSE and PDSE1 errors are not common, when errors do occur in the code for PDSEs and PDSE1s, the results often extend beyond the job or user who encountered the error. This is a result of the complexity of the PDSE and PDSE1 implementation, its use of common control blocks in CSA, and the execution of the majority of its code under the user's tasks. This creates a situation where a failure during the recovery from cancel or force can break or prevent access to a data set.

Two operator commands simplify the determination of what may be broken. In some cases, the repair of the PDSE and PDSE1 structures or the selection of which element of the system should be terminated is also determined. These following commands are commonly used when the processing for one or more PDSEs or PDSE1s hang.The two commands are:

```
VARY SMS,PDSE | PDSE1, ANALYSIS command (referred to as the ANALYSIS command)
VARY SMS,PDSE | PDSE1, FREELATCH command (referred to as the FREELATCH command)
```

With z/OS V1R13, the following PDSE commands have new operands to aid PDSE diagnostics.

```
D SMS,PDSE<1>,CONNECTIONS,DSN(pdsename)<,VOL(volser)>
V SMS,PDSE<1>,REFRESH,DSN(pdsename)<,VOL(volser)>
```

PDSE connections operand and example

The CONNECTIONS operand is useful in determining which jobs are affected when an error occurs associated with a PDSE; see Figure 6-1. You can then determine if a reIPL or restart of the PDSE address space must be done immediately.

```
D SMS,PDSE<1>,CONNECTIONS,DSN(pdsename)<,VOL(volser)>
```

```
D SMS,PDSE1,CONNECTIONS,DSN(DB8Y8.SDSNLOAD)
IEF196I IEF237I 838F ALLOCATED TO SYS00032
IGW051I PDSE CONNECTIONS Start of Report(SMSPDSE1) 704
---------data set name---------------------- -----vsgt-------
DB8Y8.SDSNLOAD                               01-BH5DB1-000F0F
--asid-- --name-- --tcb--- -open-
  002D   DB8YDIST 007FF890 Input
  004E   DB8YDBM1 007FF890 Input
  0047   DB8YMSTR 007FF890 Input
PDSE CONNECTIONS End of Report(SMSPDSE1)
```

Figure 6-1 DISPLAY SMS example

PDSE refresh operand and example

The REFRESH operand is useful in discarding what may be bad data for a PDSE after an error; see Figure 6-2. Use this command to discard cached PDSE or PDSE1 directory pages. This ensures that the next access to the specified PDSE will use the data directly from the device.

DSN(dsname)[,VOL(volser)] This specifies the PDSE for which you want to discard cached directory pages. The VOL(volser) is required only when the PDSE is not cataloged.

```
V SMS,PDSE<1>,REFRESH,DSN(pdsename)<,VOL(volser)>
```

```
V SMS,PDSE1,REFRESH,DSN(DB8Y8.SDSNLOAD)
IEF196I IEF237I 838F ALLOCATED TO SYS00033
IGW052I The cached directory blocks for PDSE DB8Y8.SDSNLOAD have been
discarded
IEF196I IEF285I   SYS11161.T131838.RA000.MSTJCL00.R0100136     KEPT
IEF196I IEF285I   VOL SER NOS= BH5DB1.
```

Figure 6-2 VARY SMS command example

Important: These commands are the result of requests from PDSE level 2. Only issue these commands under the direction of PDSE level 2 when an error has occurred.

6.2 IEBPDSE utility to validate a PDSE data set

Over the years there have been a number of client requirements requesting the ability to verify the structural integrity of a PDSE. In some instances, clients back up broken data sets as part of their routine operations, only to become aware of the problem months after the operation occurred, or after multiple backups subsequent to the first one.

z/OS V1R13 enhancements

In z/OS V1R13, the new IEBPDSE utility can be used to validate a PDSE before or after copying the data set to a new location. The utility provides output in a message data set that contains informational messages, the results of the validation check, and error messages, if any.

The utility performs a number of checks on a specified PDSE, to verify that requirements for directory structures, members, and other rules are met. The utility provides output in the SYSPRINT data set, optionally including a message for each structure or a map that shows the allocation of pages. To invoke the tool, specify PGM=IEBPDSE on the EXEC statement in JCL, naming the data set to be validated.

Using the IEBPDSE utility

The PDSE validation utility can be invoked through JCL. Like most utilities, IEBPDSE can also be invoked from TSO if SYSLIB is allocated to a PDSE.

The following JCL statements are needed to execute the IEBPDSE utility:

EXEC statement This statement invokes the PDSE validation utility using PGM=IEBPDSE. The PARM keyword may be specified.

PARM=[DUMP|NODUMP] - If the DUMP option is specified, the PDSE validation utility will issue an ABEND in the PDSE address space when

an error has been found in the analysis of the PDSE and results in an SVC dump.

SYSPRINT This DD statement defines a sequential output message data set. If this statement is omitted, the output appears in the job log. The block size for the SYSPRINT data set must be a multiple of 121. If not, the job step will end with a return code of 8.

SYSLIB This statement defines the PDSE that will be accessed by the PDSE utility when performing the operations specified on the control statements. The DSNAME parameter is required.

Note: The PDSE validation utility does not validate the data in the members, and IEBPDSE does not require APF authorization.

Example JCL

The JCL in Figure 6-3 shows how to invoke IEBPDSE to validate one PDSE data set and send the results to SYSPRINT.

```
//STEPCHK  EXEC PGM=IEBPDSE
//SYSPRINT DD SYSOUT=A
//SYSLIB   DD DSN=IBMUSER.SIMPLE.V2.PDSE,DISP=OLD
```

Figure 6-3 Example JCL to validate one PDSE data set

The JCL in Figure 6-4 shows how to invoke IEBPDSE to validate one PDSE data set and send the results to the job log because there is no SYSPRINT DD.

```
//STEPCHK2 EXEC PGM=IEBPDSE
//SYSLIB   DD DSN=IBMUSER.SIMPLE.V2.PDSE,DISP=OLD
//         DD DSN=IBMUSER.SIMPLE.V3.PDSE,DISP=OLD
```

Figure 6-4 Example JCL to validate multiple PDSE data sets

The JCL in Figure 6-5 shows how to invoke IEBPDSE to validate one PDSE data set and specify that a dump will be taken if an error is found.

```
//STEPCHK3 EXEC PGM=IEBPDSE,PARM='DUMP'
//SYSPRINT DD SYSOUT=A
//SYSLIB   DD DSN=IBMUSER.SIMPLE.V2.PDSE,DISP=OLD
```

Figure 6-5 Example JCL to validate one PDSE data set with DUMP

Return codes

The following return codes can be given:

00 (X'00') Successful completion.

04 (X'04') Slightly damaged PDSE. The data set can be opened normally but has some form of corruption. Currently, the only instance of a “slightly” damaged PDSE is when the free space list marks free pages as used, thus wasting space that could normally be reclaimed. This does not prevent the user from opening the PDSE, but the user should copy the PDSE to a new data set.

08 (X'08') Corrupted PDSE.

12 (X'0C') PDSE could not be opened.

16 (X'10') Input data set not a PDSE.

6.2.1 Installation considerations

The new PDSE utility is provided to validate the directory structure of a PDSE. The utility can be used to do the following:

- You can validate critical data sets and data sets that are backed up frequently to guarantee the validity of the date set.
- You can invoke the PDSE validation utility in a separate JCL step before or after key operations like backing up a data set. Invoking the IEBPDSE should not add any significant impact to performance.

6.3 IEBCOPY enhancements

With the current version of IEBCOPY, a new requirement is to improve its performance by not requiring its caller (a program) to have authorized program facility (APF) authorization. This tends to endanger system integrity by requiring existing programs to obtain authorization, although they were not designed with system integrity in mind.

z/OS V1R13 adds the following DFSMSdfp utility enhancements:

- The IEBCOPY utility is enhanced with performance improvements, and the utility no longer requires APF authorization.
- Starting in z/OS V1R13, you can invoke the IEBCOPY utility without being APF authorized. Calling programs no longer need to be in an APF-authorized library. A pre-V1R13, APF-authorized copy of IEBCOPY is available as IEBCOPYO in SYS1.LINKLIB, if needed.

6.3.1 IEBCOPY performance improvement

Improvements have been made to several channel programs used by IEBCOPY that will decrease elapsed time in some cases.

Most IEBCOPY storage is below the line but now it uses buffers above the line. There are no JCL changes needed to obtain the improved performance, but in some cases a larger region size may be required.

To allow for the possibility that an error or incompatibility problem is found, the earlier level of IEBCOPY is available through module IEBCOPYO (for “OLD”).

Note: IEBCOPY has had an undocumented alias IEBDSCPY. That alias now applies to IEBCOPYO. The new IEBCOPY has no alias.

6.3.2 IEBCOPY APF removal considerations

Because the IEBCOPY load module is now APF authorized and requires its caller also to be APF authorized, this introduces a security exposure when IEBCOPY needs to be called from programs that were not designed with system integrity in mind.

In z/OS V1R13, the program calling IEBCOPY no longer needs to be APF authorized. IEBCOPY can adapt to the lack of APF authorization. If your program calls IEBCOPY, you probably can remove APF authorization from your program, making it safer.

Note: IEBCOPY is still delivered as APF authorized and tends to run slightly more efficiently when it retains APF authorization. However, it adapts to the lack of APF authorization, such as when called by an unauthorized program.

6.3.3 Installation considerations

There are no changes needed to obtain the improved performance, but there may be borderline cases where there is insufficient virtual storage. Most IEBCOPY storage is below the line but now it uses buffers above the line.

If your program calls IEBCOPY, you probably can remove APF authorization from your program, making it safer.

As delivered by IBM, IEBCOPY still has APF authorization and it should tend to run slightly more efficiently when it retains APF authorization. However, it adapts to the lack of APF authorization, such as when called by an unauthorized program.

6.4 New catalog parmlib member support

Currently, the only way to customize the catalog environment is through SYS1.NUCLEUS(SYSCATxx) and SYS1.PARMLIB(LOADxx). Only 1 line (80 characters) is available and it has been long filled, thereby preventing any new parameters from being added. Also, it prevented changing parameters after the system had been IPLed.

With z/OS V1R13, new support is added for catalog definitions to provide its own parmlib member. As well, any number of parameters can be added and changed, with and without a system IPL. This new supports allows system programmers to create catalog parmlib members, to customize the catalog environment.

6.4.1 IGGCATxx parmlib member

The IGGCATxx parmlib member allows you to define catalog system parameters and make permanent changes between IPLs. You can specify other catalog parameters in the LOADxx parmlib member and the SYSCATxx parmlib member. The **`DISPLAY IPLINFO,CATALOG`** command can be used to display the catalog system parameters currently in effect at a given time.

Note: The IGGCATxx parameters are processed both at IPL and when CAS is restarted.

You must specify the current IGGCATxx parmlib member or members in the CATALOG=xx parameter in the IEASYSxx parameter. The xx suffix can be any two alphanumeric characters or national characters (@,#,$). The default is 00 (zeros).

Consider the following rules:

- If the system finds no CATALOG parameter, it uses default IGGCAT00. If the system finds no IGGCAT00 default member, then default values will be used.

- If you specify multiple IGGCATxx members in the CATALOG=xx parameter, the system processes them in the order specified. When the same parameter appears in multiple members, then the value specified in the last member that is processed becomes the value in effect.
- If the system does not find a IGGCATxx member matching the CATALOG=xx specification, then the system issues a message for each missing IGGCATxx member.

Note: The IGGCATxx parameters are processed both at IPL and when CAS is restarted.

The IGGCATxx member(s) are optional but, if specified, the parameters specified within take precedence over the parameters specified in the LOADxx and SYSCATxx members.

The suffixes should be specified in the new IEASYSxx parameter, CATALOG=(list of suffixes separated by commas). When multiple members are specified, the members are processed in the order specified.

Examples:

```
CATALOG=AA (specifies one member. No parens needed if only one member)
CATALOG=(AA,BB,05)  (specifies multiple members. Values in IGGCAT05 will
override those in IGGCATBB and IGGCATAA.)
```

For the syntax rules for this member, see *z/OS MVS Initialization and Tuning Reference*, SA22-7592.

Statements and parameters for IGGCATxx parmlib member

The following statements are new in z/OS V1R13 for the IGGCATxx parmlib member:

VVDSSPACE VVDSSPACE(primary,secondary) - specifies the number of tracks for primary and secondary allocations that the Catalog Address Space (CAS) should use for an implicitly defined VVDS. The specified values are preserved across a CAS restart and are applied at IPL.

If a zero secondary value is specified on a VVDSPSACE(pri,sec) parameter and the primary value is valid, then on an IPL, a default of 10 tracks will be substituted for the secondary. On a restart of the Catalog Address Space, the value that existed prior to the CAS restart is retained.

Default: VVDSSPACE(10,10)

NOTIFYEXTENT NOTIFYEXTENT(percent) - specifies the desired percentage threshold of the extents allocated for a catalog before the system issues message IEC361I to warn that the catalog is becoming full. The specified value is preserved across a CAS restart and is applied at IPL. Note that you can also change this value with a CAS restart.

Default: NOTIFYEXTENT(80) - A percentage value of zero (0) indicates that the monitoring for extent usage is suppressed. If a catalog exceeds 90% utilization of the maximum extents, the system will issue message IEC361I even if the threshold has been set to zero.

Note: The parameters do not have to start in column 1, but they must fit entirely within columns 1 and 71. The parser ignores any text beyond column 71.

- You can specify any of the parameters multiple times, but the parser uses only occurrences of any parameters are allowed. The last valid value will be used for this parameter. For example, the following is allowed:
  - VVDSSPACE(10,10)
  - VVDSSPACE(14,14)

  In this case, the system uses VVDSSPACE(14,14) as the final value for VVDSSPACE.
- You can use blanks between or before values in a parameter. For example:
  - A specification of VVDSSPACE(10, 14) is valid, and the system interprets it as VVDSSPACE(10,14).
  - A specification of NOTIFYEXTENT(55) is valid, and the system interprets is as NOTIFYEXTENT(55).
- Although you can have blanks between or before values in a parameter, you cannot have blanks between the digits in a single parameter value. For example:
  - If you intend a value of VVDSSPACE(10,14) but specify VVDSSPACE(1 0, 14), then the parameter is not valid.
  - If you intend a value of NOTIFYEXTENT(66), but specify NOTIFYEXTENT(66), this parameter is not valid.

TASKMAX

TASKMAX(tasks) - specifies the Catalog address space user service task upper limit, which is the maximum number of Catalog Service tasks that can run at any given time.

After this limit has been reached, further requests for CAS services are delayed until a task control block becomes available. The value must be in the range 24 - 360.

Default: 180

Note: The value for TASKMAX cannot be higher than 90% of the maximum number of concurrent CAS service tasks allowed specified in SYS1.NUCLEUS(SYSCATxx) or SYS1.PARMLIB(LOADxx) members. The CAS maximum has a default value of 200, with a minimum of 200 and maximum of 400.

- The value for TASKMAX cannot be lower than the Catalog address space service task lower limit specified in SYS1.NUCLEUS(SYSCATxx) or SYS1.PARMLIB(LOADxx) members. Default for the lower limit is 60. You can display both the Catalog address space user service task upper limit and lower limit by issuing the F CATALOG,REPORT catalog command.
- You can use the **`MODIFY CATALOG`** command to decrease the TASKMAX value or restore it to 90% of the max number of concurrent catalog requests value specified in the SYSCATxx or LOADxx member. The new value specified on **`MODIFY CATALOG`** command is then in effect until the next IPL.

Operator commands

You can display both the Catalog address space user service task upper limit and lower limit by issuing the **F CATALOG,REPORT** catalog command.

You can use the **F CATALOG** command to decrease the TASKMAX value or to restore it to 90% of the max number of concurrent catalog requests value specified in the SYSCATxx or LOADxx member. The new value specified on **F CATALOG** command is then in effect until the next IPL.

New messages

The following list describes the new messages in z/OS V1R13. For a detailed explanation of all messages, see *z/OS MVS System Messages, Volume 7 (IEB - IEE)*, SA22-7637.

- `IEC385W IGGCATxx BYPASSED DUE TO xxxxxxxx ERROR. RETURN CODE IS rc`

  Explanation: An internal error was detected during a service that was called by catalog during parsing of the catalog PARMLIB member.

  In the message text:

  - `xxxxxxxx` is "`A LOAD`" or "`A STORAGE`" or "`AN IEEMB878.`"
  - `rc` is the return code returned by the internal service.

- `IEC386W INVALID KEYWORD DETECTED IN aaaaaaaa AT LINE: text`

  Explanation: A syntax error was detected on a particular line during the parsing of the IGGCATxx parmlib member.

  In the message text:

  - `aaaaaaaa` is the 8-character catalog PARMLIB member name starting with IGGCAT, followed by the 2-character suffix. `text` is the whole line of text with the invalid keyword, up to position 71.

- `IEC387W ERROR RELEASING IGGCATxx STORAGE. ERROR CODE IS rc`

  Explanation: An internal error was detected while releasing storage obtained during IGGCATxx processing.

  In the message text:

  - `rc` is the return code returned by the STORAGE service during storage release processing.

6.4.2 SYS1.PARMLIB(IEASYSxx) member changes

You must specify the current IGGCATxx parmlib member or members in the CATALOG=xx parameter in the IEASYSxx parmlib member. The xx suffix can be any two alphanumeric characters or national characters (@,#,$). The default is 00 (zeros).

Examples of the CATALOG= parameter

```
CATALOG={aa }
CATALOG={(aa,bb,...)}
```

This parameter identifies the IGGCATxx parmlib members to use during the current IPL. The two alphanumeric characters, represented by `aa` (or `bb`, and so forth), are appended to IGGCAT to form the names of the IGGCATxx members.

The IGGCATxx parmlib members specify catalog parameters for the initializing system.

Value Range: Any two alphanumeric characters.

Default Value: CATALOG=00

The associated Parmlib Member: IGGCATxx

The following rules apply:

- If the system finds no CATALOG parameter, it uses default IGGCAT00. If the system finds no IGGCAT00 default member, default values will be used.
- If you specify multiple IGGCATxx parmlib members in the CATALOG=xx parameter, the system processes them in the order specified. When the same parameter appears in multiple members, then the value specified in the last member processed becomes the value in effect.
- If the system does not find a IGGCATxx member matching the CATALOG=xx specification, the system issues a message for each missing IGGCATxx member.

Note: Down-level systems will not try to read the new IGGCATxx parmlib member or members.

6.4.3 Installation considerations

Although there is currently no overlap between the IGGCATxx parmlib member and the SYSCATxx/LOADxx parmlib parameters, the system gives the IGGCATxx parmlib member the highest priority.

The system applies the parameters in SYSCATxx, LOADxx, and IGGCATxx in the following order:

- Parmlib member IGGCATxx, if specified, takes the highest priority

  followed by:
- Parmlib member LOADxx

  followed by:
- SYSCATxx member of SYS1.NUCLEUS

  followed by:
- System defined defaults

Consider the need to create the new IGGCATxx parmlib member or members and populate them with the desired settings. If you do, the list of suffixes must be specified on the new IEASYSxx system parameter, CATALOG=. If the CATALOG= parameter is not found, the default member IGGCAT00 is processed. If IGGCAT00 is not found, default values will be applied.

The parmlib members are processed during IPL and a restart of the catalog address space.

If invalid values are detected in any of the parmlib member or members, then the parameter is ignored and a message displaying the invalid parameter is issued in the syslog and also to the console, if it is a CAS restart.

Important: The IGGCATxx parmlib members are processed during an IPL and a restart of the catalog address space by issuing a **F CATALOG,RESTART** command.

For additional information, see:

- *z/OS DFSMS Managing Catalogs*, SC26-7409
- *z/OS MVS Initialization and Tuning Reference*, SA22-7592

6.4.4 Changing the maximum number of catalogs and tasks in CAS

When you IPL a system, the maximum number of catalogs that can be open in the catalog address space is set at 9999. The maximum number of CAS service tasks available for user requests is set to either 180 (the default) or 90% of the value optionally specified in the SYSCATxx member of SYS1.NUCLEUS. You can specify the number of catalogs and tasks as follows:

- Specify the service task lower limit in SYSCATxx (SYS1.NUCLEUS), or LOADxx parmlib member.
- Specify the maximum number of concurrent user service tasks in the TASKMAX parameter of the IGGCATxx parmlib member. The value of TASKMAX in IGGCATxx parmlib member should be no more than 90% of the maximum number of concurrent Catalog requests specified in SYSCATxx or LOADxx. The TASKMAX parameter is new with z/OS V1R13.

6.4.5 MODIFY CATALOG command

This section explains the syntax and parameter changes to the **MODIFY CATALOG** command. This command communicates with the catalog address space, to display information or request services.

The following operands have been added to the command in z/OS V1R13 with the DISABLE keyword:

DELRECOVWNG This operand specifies that message IDC1999I should not be issued if a **DELETE UCAT RECOVERY** command is attempted.

DELRECOVWNG is disabled, by default.

EXTENDEDALIAS This operand disables the ability to create extension records for user catalog aliases on the current system.

EXTENDEDALIAS is disabled, by default.

The following operands have been added to the command in z/OS V1R13 with the ENABLE keyword:

DELRECOVWNG This operand specifies that message IDC1999I will be issued if a **DELETE UCAT RECOVERY** command is attempted.

EXTENDEDALIAS This operand enables the ability to create extension records for user catalog aliases on the current system.

Note: Only enable this feature when all systems in the sysplex are z/OS V1R13 or higher.

The following operands have been modified to the command in z/OS V1R13:

SYMREC This operand specifies that SYMREC records are not created. Use this option to temporarily disable the creation of SYMREC records. For example, if a problem is causing repeated creation of SYMREC records and this is disrupting how well you are able to manage of the SYMREC target data set, you can disable the SYMREC records.

UPDTFAIL This operand disables the message IEC390I when a VSAM update request against a catalog has been abnormally terminated. This message is intended to alert the installation that potential catalog damage may have resulted from the incomplete request. The default for this option is enabled.

VVRCHECK This operand disables enhanced VVR checking on VVDS I/O. VVRCHECK is disabled, by default.

Message IDC1999I

The following message in new in z/OS V1R13:

```
IDC1999I catname replaced by an import backup copy, its VVR and its VTOC to be
deleted, REPLY 'Y' TO DELETE 'N' TO CANCEL.
```

Explanation: A catalog is about to be deleted with the RECOVERY option. This means that the VSAM volume record (VVR) and VTOC for the catalog will be deleted, as follows:

- If you respond Y or YES, IDCAMS processes the command replacing the user catalog with an import backup copy and deletes the specified user catalog along with its VSAM volume record (VVR) and VTOC entries. The VVR and DSCB for each object in the catalog are not deleted.
- If you respond N or NO, the system ends the delete process with a return code of 8.

System programmer response: To enable or disable this WTOR message, use the **MODIFY CATALOG,DISABLE(DELFORCEWNG)** command. Use the command F CATALOG, REPORT to display the status of the DELRECOVWNG parameter.

6.5 DASDM dynamic exits

All DADSM functions (create, extend, scratch, partial release, and rename) call a common preprocessing dynamic exit routine (IGGPRE00_EXIT) and a common postprocessing exit routine (IGGPOST0_EXIT).

The system adds the IGGPRE00 and IGGPOST0 modules as exit routines to their associated dynamic exits. These two modules are equivalent to what was provided in releases prior to z/OS V1R13. DADSM functions will call each dynamic exit using dynamic exit services (CSVDYNEX) during preprocessing and postprocessing so that each added exit routine to IGGPRE00_EXIT and IGGPOST0_EXIT is called. This allows the exit routine to gain control before and after DADSM processing.

Adding exit routines

All existing operator commands and system services that apply to managing dynamic exits apply to these two dynamic exits. For example, to add an exit routine to a dynamic exit:

- A program can use CSVDYNEX REQUEST(ADD).
- You can use the **SETPROG EXIT** command.

- You can use the EXIT statement of a PROGxx PARMLIB member, followed by the **SET PROG=xx** command which will add the IGGPRE01 exit routine to the preprocessing dynamic exit of IGGPRE00_EXIT:

```
SETPROG EXIT,ADD,EXITNAME=IGGPRE00_EXIT,MODNAME=IGGPRE01
```

Multiple exit routines

You can now have multiple exit routines associated with each dynamic exit. Replacing an already active IGGPRE00 or IGGPOST0 dynamic exit routine without an IPL is also supported. For example, the **SETPROG EXIT,DELETE** command can be issued followed by the **SETPROG EXIT, ADD** command to replace a dynamic exit routine.

6.5.1 Installation considerations

You can now have multiple exit routines associated with each of the IGGPRE00_EXIT and IGGPOST0_EXIT dynamic exits for the DADSM preprocessing and postprocessing exits. The system calls them in an unpredictable order.

Words in the CVAF table continue to contain the addresses of IGGPRE00 and IGGPOST0. However, DADSM no longer uses these fields to call preprocessing and postprocessing exits.

If you use IGGPRE00 and IGGPOST0 exits, you do not need to change them in any way. You can install them as usual.

You do not need to change the load module names for IGGPRE00 nor IGGPOST0, but you can change their name or names.

If you do change the name or names, you must create or update a PROGxx parmlib member or issue the **SETPROG EXIT** command to activate the name changes.

To use multiple exits for a preprocessing and postprocessing function, issue the **SETPROG EXIT,ADD** command or use the PROGxx PARMLIB member to specify the load module names to be associated with each of the IGGPRE00_EXIT or IGGPOST0_EXIT dynamic exits.

New messages

The new messages for this new support are listed here:

- DADSM issues a new message if a dynamic exit ABENDs. Also note that an ABENDed dynamic exit is automatically disabled and will not be called again unless reactivated:

```
IEC615I ABEND=xxxxxx-yyyyyyyy OCCURRED IN THE yyyyyyyy EXIT MODULE FOR
DYNAMIC EXIT zzzzzzzzzzzzzzzz')
```

- DADSM issues a new message if it receives an unexpected return code from the CSVDYNEX service:

```
IEC616I NON ZERO RETURN CODE FROM CSVDYNEX xxxxxxxx RC = xxxx RSN = xxxx,
DYNAMIC EXIT = zzzzzzzzzzzzzzzz')
```

6.6 New option for the LISTCAT LVL command

In previous releases, ordinary LISTCAT with LVL commands sometimes do not list all dependent objects for a CLUSTER or an AIX. This happens when the LVL pattern for the DATA and the INDEX objects does not match the generic pattern expressed.

With z/OS V1R13, with a new CDILVL option to a LISTCAT LVL, you are able to see the other dependent objects if the pattern matches the main CLUSTER or AIX object.

This allows you to gain more information for a LISTCAT LVL across many catalogs, and have a better understanding of what dependent objects are associated with which primary objects.

New LISTCAT command option

The following new operands are added to the command:

```
NOCDILVL | CDILVL
```

Note: You can only specify NOCDILLVL | CDILVL parameters with the LEVEL parameter.

Where:

NOCDILVL NOCDILVL specifies that only the objects whose patterns match the LEVEL pattern be listed.

CDILVL CDILVL specifies that DATA and INDEX objects in CLUSTERs and AIXs be listed if at least one of the three objects matches the LEVEL pattern.

NOCDILVL is the default.

6.6.1 Installation considerations

When using this new parameter, this enables you to obtain CLUSTER, AIX, DATA, and INDEX elements whose names do not match the LVL pattern, if the pattern is matched by the parent object.

With the CDILVL option, for example, a CLUSTER, DATA, INDEX set of objects named USER1.CLUSTER, USER2.DATA, and USER3.INDEX with a LISTC LVL(USER1) ALL CDILVL would list the CLUSTER (because it matches the LVL pattern); the DATA (because of the CDILVL specification); and the INDEX objects (because of the CDILVL specification). Without it, only the CLUSTER object would display.

Note: With the default being NOCDILVL, you must specify the CTGCDI parameter to obtain the extra set of dependent objects.

LISTCAT LVL now has a new option, CDILVL, for its output.

If specified, a LISTCAT LVL will also list the dependent objects in a listing if its main object matches the LVL pattern. Without it, LVL listing is as before. The CDILVL specification has an effect only with the LVL option. The parameter allows you to see more in a LVL listing than simply what matches the LVL pattern, necessarily.

6.7 OPEN/CLOSE/EOV enhancements

Open/Close/End of Volume provides the following enhancements in z/OS V1R13:

- A new installation option, using the DEVSUPxx parmlib member, to append descriptive text for a subset of O/C/EOV abend messages associated with the more commonly experienced abend and return codes. This option minimizes the need to refer to the message manuals to interpret the abend and return codes for those messages.

- A change to calculations of the BUFNO value by QSAM, using the DCBE macro's MULTSDN value, to reduce out-of-storage conditions. For concatenated data sets, if MULTSDN is specified, QSAM dynamically recalculates the BUFNO value when switching from one data set to another.
- OCE now accepts subsystem DCBs with XTIOTs, in addition to BAM DCBs. This is an extension to the BAM XTIOT support that was provided in z/OS V1R12.
- Support for recovery for missing and out-of-order tape volumes, using the LABAN tape anomaly exit.
- A new FREEVOL=EOV JCL parameter to allow different systems in a sysplex to read multivolume tape files concurrently, in the sense that while one system is reading a multivolume tape data set, the volser of the last read volume is DEQd at EOV. This allows another system to obtain the volser ENQ and to begin processing the volume while the other system is still reading the same multivolume data set.
- New OCE diagnostic data added to SMF 14/15 Type records Extended Information Segment Type 8, to provide reasons when a DCBE is invalidated or a partial release is not performed for a DASD data set.

6.7.1 New DEVSUPxx parmlib member option

The OCE_ABEND_DESCRIP option in the DEVSUPxx parmlib member controls whether descriptive text is appended to certain O/C/EOV abend messages. With the default value (NO), the abend messages contain only numeric values for the ABEND code and associated reason code. If you code the OCE_ABEND_DESCRIP option with a value of YES, then descriptive text is appended to the messages, thereby minimizing the need to look up the ABEND and reason codes.

OCE_ABEND_DESCRIP= (YES | NO)

YES - Specifies that abend messages for selected OPEN, EOV and CLOSE determinant errors include descriptive text for the associated numeric abend code and numeric return code.

NO - Specifies that OPEN, EOV and CLOSE abend messages will not include a descriptive text for the associated numeric abend and numeric return code.

DEFAULT: NO

Note: Not all OPEN/CLOSE/EOV messages are affected in this release.

Example of the output

The following example output illustrates the use of this new option:

```
IEC146I 513-08,IFG0196T,CRTAAL1,CRTTSL1,SYSUT2,0920,,DATASET1  036
ERROR DESCRIPTION:
A LABEL VIOLATED THE PUBLISHED STANDARD FOR THAT LABEL, AND THE LABEL
VALIDATION EXIT ISSUED A RETURN CODE REQUESTING OPEN OR EOV TO REJECT
THE VOLUME.
END ERROR DESCRIPTION: IEC146I
```

6.8 New OCE diagnostic data

In current z/OS systems, OPEN allows the DCB to be in key 9 storage if the caller is running in key 8. However, OPEN does not allow the DCBE (DCB extension) to be in key 9 storage. If the user makes a mistake in supplying the DCBE, OPEN ignores it with no diagnostics. For example, the data set organization must be PS, DA or PO and the DCBE must not be in use by another DCB.

With z/OS V1R13, OPEN records a code in the SMF Types 14 or 15 records, extended information segment Type 8, to provide reasons when a DCBE is invalidated or a partial release is not performed for a DASD data set. This is intended to show what was wrong with the DCBE, to improve problem determination.

Extended information segment Type 8

This byte in the type 8 extended information segment in the SMF type 14 or 15 record explains why the DCBE is invalid, as shown in Figure 6-6 on page 119.

```
SMF14RAS       EQU  *        EXTENDED INFO SEGMENT TYPE 8
SMF14DCBEEXCP  EQU  X'80'    DCB IS EXCP AND NO FOUNDATION
                             EXTENSION IS PRESENT
SMF14DCBEDSORG EQU  X'40'    DSORG IS NOT PS, PO OR DA
SMF14DCBEFREE  EQU  X'20'    DCBE STORAGE IS NOT ADDRESSABLE
SMF14DCBEKEY   EQU  X'10'    DCBE STORAGE IS NOT IN KEY OF CALLER
SMF14DCBEID    EQU  X'08'    THE DCBEID IS NOT 'DCBE'
SMF14DCBEMIN   EQU  X'04'    DCBE IS NOT  AT LEAST THE MINIMUM
                             LENGTH REQUIRED (56 BYTES)
SMF14RAS       EQU  *        EXTENDED INFO SEGMENT TYPE 8
SMF14DCBEEXCP  EQU  X'80'    DCB IS EXCP AND NO FOUNDATION
                             EXTENSION IS PRESENT
SMF14DCBEDSORG EQU  X'40'    DSORG IS NOT PS, PO OR DA
SMF14DCBEFREE  EQU  X'20'    DCBE STORAGE IS NOT ADDRESSABLE
SMF14DCBEKEY   EQU  X'10'    DCBE STORAGE IS NOT IN KEY OF CALLER
SMF14DCBEID    EQU  X'08'    THE DCBEID IS NOT 'DCBE'
SMF14DCBEMIN   EQU  X'04'    DCBE IS NOT  AT LEAST THE MINIMUM
                             LENGTH REQUIRED (56 BYTES)
SMF14NODCBE    EQU  X'02'    DCBEHIARC FLAGS SET BUT DCBDCBE IS ZEROES
```

Figure 6-6 Type 8 extended information segment in the SMF type 14 or 15 record

RAS section (Type 8)

This section contains RAS (Reliability, Availability, Serviceability) information about the data set. If you coded DSECT=YES when calling the IFGSMF14 macro, it generates a DSECT statement at this point with the DSECT name with RAS appended to it. For example, if you did not code a label on the IFGSMF14 call, the name of this DSECT will be IFGSMFRAS.

6.9 QSAM MULTSDN calculation with concatenation

The QSAM user can code MULTSDN in the DCBE macro to request OPEN to calculate an optimum BUFNO value for tape and specific types of DASD data sets. OPEN calculates it based on the first data set in the concatenation.

When going to the next data set in the concatenation, an out-of-storage condition can occur when the next data set's block size is much larger than in the previous data set. This is because the system does not recalculate a smaller number of buffers for the larger blocks.

With z/OS V1R13, the system dynamically recalculates the BUFNO value when switching to the next concatenated data set when QSAM with MULTSDN is specified. With this implementation, the QSAM user that uses MULTSDN is less likely to run out of storage.

In z/OS V1R9, you could use the MULTSDN parameter of the DCBE macro with QSAM. In previous releases, QSAM ignored the MULTSDN parameter. The new support for MULTSDN allows the system to calculate a more efficient default value for the DCB BUFNO parameter, and reduces the situations where you need to specify a BUFNO value.

QSAM accepts a MULTSDN value for the following data sets:

- Tape data sets
- DASD data sets of the following types:
- Basic format data sets
- Large format data sets
- Extended format (non-compressed) data sets
- PDS data sets

With z/OS V1R13 for the supported types of data sets, the system uses MULTSDN to calculate a more efficient value for BUFNO when the following conditions are true:

- The MULTSDN value is not zero (0).
- DCBBUFNO has a value of zero after completion of the DCB OPEN exit routine.
- The data set block size is available.

When MULTSDN is specified, note that the default number of buffers may be less than what would have been derived without MULTSDN.

Installation considerations

No changes are needed for how you code MULTSDN on the DCBE macro. The system will calculate a number of buffers that more closely matches the logic that is used during OPEN. You are less likely to run out of storage.

Note: If the number of buffers for the first data set is small due to a large block size, then the current code gets the same number of buffers for the second data set even if it has a small block size that should deserve many buffers. Therefore this case works but is likely to give poor performance. With z/OS V1R13, this probable poor performance should be improved by this new implementation because of the BUFNO recalculation for each data set.

6.10 Free tape volumes at end of volume

Long-running programs that read or write multivolume tape data sets can prevent other jobs from accessing any of the volumes until the job unallocates all of the volumes. A DISP=OLD option allows authorized programs to avoid this.

With z/OS V1R13, a new DD statement keyword, FREEVOL=EOV, allows any program to release tape volumes early before closing the data set and therefore no source code change or special authorization is needed.

FREEVOL parameter

Use the FREEVOL parameter to specify whether to allow other jobs to read freed volumes of a multivolume tape file as the volume is dismounted by the job.

```
FREEVOL={EOV | END}
```

Where:

EOV Requests that when reading a multivolume data set, the system finish reading the current volume and then dequeue the volume serial number and demount the volume. This makes the volume immediately available to another job in another system. An attempt by the same task to reprocess the volume using the same JCL DD statement will result in an abnormal end.

END Requests that volumes be dequeued at the end of the job step.

Defaults: If no FREEVOL parameter is specified, the default is END. Also, if the FREEVOL parameter is incorrectly coded, the system substitutes END and issues a warning message.

Note: If you code FREEVOL=EOV when processing a second tape data set, you must assume that the same volume serial numbers have been coded in the DD statements for data set 1 and data set 2. As the volumes of data set 1 are demounted, they are dequeued even though those volumes still might be requested for data set 2. The dequeued volume can then be mounted for use by another job.

Do not code the FREEVOL parameters with PATH-related keywords, such as the following:

```
PATH PATHOPTS
PATHMODE PATHDISP
FILEDATA
```

6.10.1 Installation considerations

This function is honored only for input processing. EOV and CLOSE volume disposition processing will rewind and unload the volume. After a volume is released in this manner, OPEN or EOV will issue an abend if the task using the same JCL DD statement attempts to reprocess a previously dequeued volume serial number.

The new ABEND and return codes are:

- For OPEN: IEC145I with ABEND 413-60

  ```
  IEC145I 413-rc,mod,jjj,sss, ddname[-#],dev,ser,dsname(member)
  Explanation: The error occurred during processing of an OPEN macro
  instruction for a data set on magnetic tape or on a direct access device.
  ```

- For EOV: IEC026I with ABEND 637-C0

  ```
  IEC026I 637-rc,mod, jjj,sss,ddname[-#], dev,ser,dsname(member)
  Explanation: The error occurred during end-of-volume for a data set on
  magnetic tape or an end-of-volume during concatenation.
  ```

6.11 XTIOT support for subsystem DCBs and ACBs

OPEN/CLOSE/EOV XTIOT support for BAM DCBs was added in z/OS V1R12 but did not include subsystem DCBs. Using an XTIOT instead of a TIOT entry provides virtual storage constraint relief (VSCR).

With z/OS V1R13, OPEN will accept subsystem DCBs with associated XTIOTs if the subsystem supports it. As with XTIOT support for DASD and tape, the system programmer must set NON_VSAM_XTIOT=YES in the DEVSUPxx parmlib member and the user's DCBE macro must have LOC=ANY.

This enhancement provides 24-bit VSCR, meaning that several control blocks for that data set allocation will be above the 16 MB line.

DEVSUPxx parmlib member

The following new keyword, NON_VSAM_XTIOT, was introduced in z/OS V1R12 as follows:

NON_VSAM_XTIOT= **YES** enables support for XTIOT, uncaptured UCB, and DSAB control blocks that reside above the 16 megabyte line for data sets that use BSAM, QSAM, or BPAM.

NO disables support for XTIOT, uncaptured UCB, and DSAB control blocks that reside above the 16 megabyte line for data sets that use BSAM, QSAM, or BPAM.

The default value for NON_VSAM_XTIOT is NO.

Installation considerations

JES2 and JES3 set an indicator (SSALXTIO or SSAGXTIO) that results in the new DSABSSXT bit being on. This means that the subsystem supports these dynamic allocation options: XTIOT, uncaptured UCBS, and above-the-line DSABs.

Optionally you can set these options in the DEVSUPxx member of PARMLIB:

```
OCE_ABEND_DESCRIP={YES | NO}
NON_VSAM_XTIOT={YES | NO}
```

6.12 OAM file system support

This new function in z/OS V1R13 introduces OAM support for a new file system sublevel in the OAM storage hierarchy. This client requirement satisfies the following concerns:

- An additional “Disk” destination is provided in the OAM storage hierarchy. The existing hierarchy can consist of disk (implemented through DB2 tables on DASD), optical, and tape.
- Transition from fast DASD to slower DASD
- Additional storage hierarchy targets, such as slow DASD
- Non-DB2 disk storage in the OAM storage hierarchy

Z/OS V1R13 support

This new support provides additional flexibility in constructing the OAM storage hierarchy. For example, you can reuse older or slower DASD devices for zFS file system storage. Additionally, storage costs might be reduced with less expensive disk in an NFS file server

used for NFS file system storage. Another consideration is using the file system sublevel as a form of “cache” when implementing the OAM Recall to Disk functionality.

Attention: A PTF for z/OS V1R13 coexistence with APAR OA33022 must be installed on any pre-V1R13 level systems prior to starting OAM the first time on z/OS V1R13.

OAM on pre-V1R13 level systems will not process objects in the file system sublevel.

In z/OS V1R13, the Disk level of the OAM storage hierarchy is now comprised of disk sublevel 1 (existing DB2 sublevel using DB2 tables) and disk sublevel 2 (new file system sublevel). The new file system sublevel is a destination for primary objects stored as files in the z/OS UNIX file system hierarchy using either:

- zFS (on native attached DASD), or
- NFS (with a wide variety of storage options and technologies on network attached NFS file servers)

Objects are directed to the file system sublevel of the OAM storage hierarchy using the SMS storage class construct. Using ISMF, you create a new storage class or modify an existing storage class and specify an “Initial access response seconds” value of 0 and an “OAM sublevel” value of 2 to direct objects to the file system sublevel.

In an OAM environment, object storage groups allow the storage administrator to define an object storage hierarchy. The object storage hierarchy classifies storage areas according to location and, therefore, according to retrieval response time. Each object storage hierarchy must contain an object directory, containing control information about each object.

In z/OS V1R13, the OAM storage hierarchy can consist of:

- DB2 object storage tables on DASD
- A file system (zFS or NFS)
- Optical storage
- Tape storage

Note: Careful and complete planning is critical for a successful implementation of the file system capability within OAM. A number of considerations must be coordinated between the z/OS UNIX System Services environment and the OAM implementation. For more information, see *z/OS DFSMS OAM Planning, Installation, and Storage Administration Guide for Object Support*, SC35-0426.

6.12.1 Installation considerations

The following topics introduce several installation considerations when migrating to z/OS V1R13 and exploiting the new OAM file system support.

Attention: For migration, run the CBRSMR1D SAMPLIB job if you are migrating from any release earlier than z/OS V1R13.

- Whether or not you intend to take advantage of the new function, you must modify and run this job.
- This job adds the File System Delete Table to the OAM Configuration Database.
- Do not perform this step at initial installation. Perform this step for migration purposes only.

SMS storage class

Each object that OAM stores are directed to a storage class and a management class. OAM uses the storage class to determine the initial placement of an object in the OAM object storage hierarchy. OAM also uses the storage class during the OSMC storage management cycle to determine the correct placement of the object when the storage management cycle processes that object. OAM uses the Initial Access Response (IARS) parameter in the storage class to determine if a primary copy of an object is stored on disk (DB2 tables or a file system) or on removable media (optical or tape). If the IARS parameter in the storage class that is assigned to the object is zero, the primary copy of the object is stored on disk. If the IARS parameter is nonzero, the primary object is stored on removable media.

OAM object storage group implementation

For each OAM object storage group, implement the following:

- A new file system (zFS aggregate or NFS server definition)
- Create a new mount point directory in the z/OS UNIX file system hierarchy
- Mount the file system at a mount point directory and perform the following tasks for the directory:
  - Change owner/group to uid/gid for OAM address space
  - Change permissions to '700' (rwx - only OAM address space)
- Create an OAM “sentinel” file in the file system at mount point and perform the following tasks for the file:
  - Change owner/group to uid/gid for OAM address space
  - Change permissions to '600' (rw - only OAM address space)
- Add a SETDISK statement in the CBROAMxx parmlib member
- Create or update a SMS storage class and ACS routines, and activate the SCDS

CBROAMxx parmlib member

The following changes are made:

- A new SETDISK statement is available.

  The CBROAMxx parmlib member contains one or more SETDISK statements to configure the file system sublevel of the disk level in the OAM storage hierarchy. For each object storage group in which a file system sublevel will be defined, a SETDISK statement must be specified to provide the file system type and the file system directory to be used for the storage group. OAM uses these values to store objects in, and retrieve objects from, the file system. The file system type and file system directory for the object storage group must be carefully selected, because these are static values and cannot be changed.

  Note: The SETDISK statement is the only mechanism in OAM used to communicate the file system type and file system directory for the storage group. The file system type and file system directory specified is not recorded by OAM within DB2 database tables. The SETDISK statement must therefore continue to exist to provide access to objects in the file system sublevel.

  Therefore, backing up the CBROAMxx parmlib member to preserve these critical SETDISK statements must be included in your backup strategy. If a symbolic link is used, the value of the symbolic link should also be included in your backup strategy.

- A new configuration option for the existing SETOPT statement in the CBROAMxx parmlib member is available to specify “Automatic Access to Backup” for file system errors.

- A new configuration option for the existing SETOSMC statement in the CBROAMxx parmlib member is available to specify a disk sublevel for “Recall to Disk” as listed here:
  - 1 for existing DB2 sublevel
  - 2 for new file system sublevel

DB2 considerations

The OAM configuration database (CBROAM) contains configuration information related to the target destinations for objects including tape volumes, optical libraries, drives, slots, and volumes. CBROAM also identifies objects to be ultimately deleted by OAM from optical volumes and the file system. It is a DB2 database that has a new table:

- The File System Delete table contains one row for each object waiting to be deleted from the file system. Objects to be deleted from the file system can be deferred delete for delete requests from the file system or an undo of a write for uncommitted store requests. The DB2 name of this table is FSDELETE.

Also relative to DB2, note the following points:

- The existing ODINSTID field in the OAM Object Directory now may contain a value to identify unique instances of OAM files in file system sublevel.
- The existing ODLOCFL field in OAM Object Directory now may contain new values:
  - 'E' when the object is located in a new file system sublevel
  - '2' when the object is recalled to a new file system sublevel

File system security considerations

Security configuration for the file system is important to allow the OAM address space to access directories and files in the z/OS UNIX file system hierarchy. The Security Server (RACF) or equivalent security product must be configured to provide both a UNIX System Services group (with an associated group ID) and user (with an associated user ID) for the OAM started procedure.

When using the Security Server (RACF), a definition in the STARTED class is the preferred method for assigning identities to started procedures such as the procedure that you use to start the OAM address space. See *z/OS DFSMS OAM Planning, Installation, and Storage Administration Guide for Object Support*, SC35-0426, for a description of the steps required to configure the Security Server (RACF) for the OAM started procedure.

6.13 OAM usability and reliability enhancements

Many changes have been made in z/OS V1R13 regarding OAM usability, as listed here:

- Wildcard usage with the **F OAM,S,STORGRP** command
- Extend object expiration beyond 27 years
- Dynamic update of SGMAXTAPERETRIEVETASKS and SGMAXTAPESTORETASKS settings
- Improved media migration
- Enhanced OAM messages for specific DB2 errors
- SMF counter scalability
- CTICBR00 parmlib member
- CBR9875I recycle candidates display enhancement

6.13.1 Wildcard usage with the F OAM,S,STORGRP command

Operators have to enter a separate command for each object or object backup storage group they want to process. To help cut back on keystrokes, the **F OAM,START,STORGRP,group-name** command can now accept a wildcard asterisk (*) to replace zero (0) or more characters in the group-name.

This enables installations with multiple object or object backup storage groups to now start multiple storage group processes with a single command invocation.

Command examples

The **F OAM,S,STORGRP,groupname** command has been enhanced to support a single asterisk as the last or only character in the groupname.

The following command starts processing for all object or object backup storage groups defined in the ACDS that have groupnames that start with GROUP:

```
F OAM,S,STORGRP,GROUP*
```

The following command starts processing for all object or object backup storage groups defined in the ACDS:

```
F OAM,S,STORGRP,*
```

Note: The **F OAM,S,OSMC** command can be used to start processing of all object storage groups, but it ignores object backup storage groups.

6.13.2 Extend object expiration beyond 27 years

In previous releases, the maximum expiration criteria specified through SMS management class definitions (other than NOLIMIT) was 9999 days (roughly 27 years).

With z/OS V1R13, objects can still be retained FOREVER (or NOLIMIT). However, the 9999 day maximum associated with management class retention limit, Expire after Date/Days and Expire after Days Non-usage, has been expanded to 93000 days. Additionally, the maximum number of days specified through the RETPD and EVENTEXP keywords on the OSREQ API has also been expanded from 32767 to 93000.

SMS retention period

To exploit the expanded SMS retention period available in z/OS V1R13, the storage administrator can optionally set higher values in the following SMS Management Class attributes through ISMF.

- The retention limit
- Expire after Days Non-usage
- Expire after Days/Date

OSREQ keywords

Application programmers can optionally set higher values in the following OSREQ keywords:

- RETPD
- EVENTEXP

Table 6-1 on page 127 shows the maximum values that are supported by OAM running on z/OS V1R13 and pre-R13 level systems.

Table 6-1 New maximum values for retention periods

Attribute	z/OS V1R13	Pre-V1R13
Management Class Retention Limit	93000	9999
Management Class Expiration Attributes	93000	9999
Expire after Days Non-Usage	93000	9999
Expire after Days/Date	93000	9999
OSREQ RETPD	93000	32767
OSREQ EVENTEXP	93000	32767

6.13.3 SGMAXTAPESTORETASKS and SGMAXTAPERETRIEVETASKS

To change the distribution of tape drives allocated for OAM object and object backup storage groups, installations have to modify SGMAXTAPESTORETASKS and SGMAXTAPERETRIEVETASKS values in the CBROAMxx parmlib member and restart OAM.

With z/OS V1R13, there is a mechanism to alter these values dynamically. The values specified for the SETOAM keyword in SGMAXTAPERETRIEVETASKS and SGMAXTAPESTORETASKS can be dynamically changeable through the **`F OAM,UPDATE,SETOAM`** operator command. No restart of the OAM address space is required after issuing this command.

Note: This support applies to SGMAXTAPERETRIEVETASKS and SGMAXTAPESTORETASKS keywords specified at a storage group level.

The MAXTAPERETRIEVETASKS and MAXTAPESTORETASKS keywords specified at a global level are still not dynamically changeable.

6.13.4 Improved media migration

When processing tape or optical volumes with a large number of collections, a significant amount of time could elapse between the time the **`MOVEVOL`** command is issued for a volume and the time of the first write to a new volume.

The OAM media migration utility, MOVEVOL, was enhanced to provide better performance characteristics for certain scenarios.

- The MOVEVOL algorithm was changed to no longer process objects on a collection by collection basis, thus allowing for more efficient processing and data movement. The changes in MOVEVOL can result in reduced time to migrate objects off a volume that contains objects from multiple OAM collections. The performance improvement will be most significant when the source volume contains objects from a large number of OAM collections.
- Additionally, prior to this support, running MOVEVOL on one member of an OAMplex resulted in measurable CPU usage on “idle” members in the OAMplex in reaction to XCF messages broadcast by the “active” member. With this support, the frequency of the broadcast messages from the active member will be significantly reduced, resulting in much lower CPU usage on the idle systems.

To move objects from a source volume, issue the following **MOVEVOL** command, where volser is the volume serial of the source volume from which objects are to be moved.

```
F OAM,START,MOVEVOL,volser
```

6.13.5 Enhanced OAM messages for specific DB2 errors

OAM currently issues generic messages that display DB2 SQL codes when a DB2 error is encountered. The systems programmer must convert the hex return or reason code into a negative decimal SQL code and then look up the codes in DB2 manuals.

With z/OS V1R13, new messages are issued containing additional information for “common” SQL codes. This can save the operator and storage administrator the trouble of having to derive the SQL codes and look up the codes in the DB2 manuals.

6.13.6 SMF counter scalability

Some four-byte counter fields in SMF Type 85, subtypes 32-35 and 87 that contain kilobyte values potentially could overflow as workloads and tape capacity increase.

With z/OS V1R13, new eight-byte counter fields have been added to SMF Type 85, subtypes 32-35 and 87 to protect against potential overflow. The new eight-byte counters contain values in bytes. This enhancement avoids inaccuracies due to counter overflow (the four-byte counters will contain X'FFFFFFFF' if an overflow condition is detected).

The new eight-byte counters provide more granularity. They contain number of bytes (versus a number of kilobytes in the old four-byte fields).

The following OAM Subtype 32-35 counters are four-byte fields that could potentially overflow. A value of X'FFFFFFFF' in one of these fields indicates an overflow was detected. The new eight-byte fields introduced in R13 supersede these four-byte fields.

```
ST32PDWK  ST32PDRK  ST32PDDK  ST32POWK  ST32PORK   ST32PODK  ST32PTWK  ST32PTRK
ST32PTDK  ST32BOWK ST32BORK  ST32BODK  ST32BTWK  ST32BTRK  ST32BTDK ST32B2OWK
ST32B2ORK ST32B2ODK ST32B2TWK ST32B2TRK   ST32B2TDK ST32RCLK  ST32PUWK
ST32PURK  ST32PUDK
```

The following OAM Subtype 87 counters are four-byte fields that could potentially overflow. A value of X'FFFFFFFF' in one of these fields indicates an overflow was detected. New eight-byte fields are introduced in z/OS V1R13 that supersede these eight-byte fields.

```
ST87NKBW ST87NKBR
```

6.13.7 CTICBR00 parmlib member

Installations had to copy the CBRCTI00 parmlib member from SYS1.SAMPLIB into the parmlib with a rename to CTICBR00 to define OAM default trace options through the parmlib member.

With z/OS V1R13, OAM now ships the CTICBR00 parmlib member directly in parmlib. Therefore, the copy or rename step is no longer required. This can simplify OAM installation and migration.

6.13.8 RECYCLE candidates display enhancement

When an **F OAM,START,RECYCLE** command is issued, the Recycle candidates display message CBR9875I, followed by a list of up to 40 volumes that have met the criteria specified by the **RECYCLE** command is generated and sent to hard copy SYSLOG. The total number of volumes that meet the criteria for the **RECYCLE** command is not displayed.

With z/OS V1R13, the message line that is displayed at the end of the Recycle candidates display is updated to show a count of the total number of volumes that met the criteria specified in the **RECYCLE** command. This enables installation to better plan for tape recycle.

6.13.9 Changed operator commands

The following operator commands have been updated with this release:

- The **F OAM,UPDATE,SETOAM,scope,SGMAXTPS** command can be used to dynamically update the SGMAXTAPESTORETASKS associated with the specified storage group or groups.
- The **F OAM,UPDATE,SETOAM,scope,SGMAXTPR** command can be used to dynamically update the SGMAXTAPERETRIEVETASKS associated with the specified storage group or groups.
- The **F OAM,START,STORGRP,group-name** command can now accept a wild-card asterisk to replace zero or more characters in the group-name.

6.14 CDS backup improvements for DFSMShsm

DFSMShsm control data sets (CDS) are system-type data sets that DFSMShsm uses to keep track of all DFSMShsm-owned data. They consist of migration, backup, and offline control data sets. The control data sets are an inventory of low activity and inactive data that was stored by DFSMShsm and used to manage its environment. DFSMShsm logs its transactions and maintains multiple backup versions of its CDSs for recovery purposes as specified by the user.

DFSMShsm maintains three control data sets, as illustrated in Figure 6-7:

- Backup control data set (BCDS)

  The backup control data set is a VSAM key-sequenced data set that contains information about backup versions of data sets, backup volumes, dump volumes, and volumes under control of the backup and dump functions of DFSMShsm.

- Migration control data set (MCDS)

  The migration control data set contains information about migrated data sets and the volumes they migrate to and from. Additionally, internal processing statistics and processing-unit-environment data reside in the MCDS. DFSMShsm needs this information to manage its data sets and volumes. With this information you can verify, monitor, and tune your storage environment.

- Offline control data set (OCDS)

  The offline control data set contains DFSMShsm with information about each migration and backup tape and about each data set residing on these tapes.

To enable you to recover a lost or damaged control data set, DFSMShsm also maintains a journal of each transaction that occurs to the control data sets. To maintain availability with

using DFSMShsm control data sets, you need backup copies of them to allow you to recover data if the active control data sets become lost or damaged.

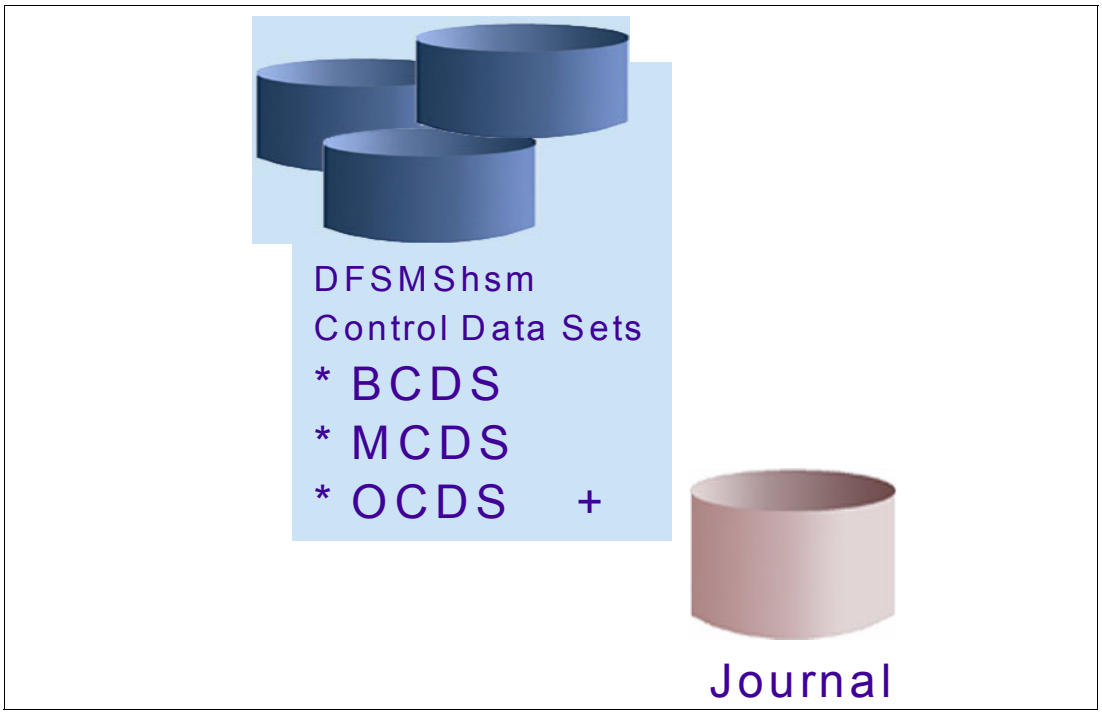


Figure 6-7 DFSMShsm control data sets and journal

6.14.1 Non-intrusive journal backup with z/OS V1R13

In past releases, CDS backup activity was largely dependant on the time it took to back up the CDS and journal. Journal backup was not performed at the beginning of the backup process due to the changing contents of the journal and the need to capture the entire journal contents.

During the backup of DFSMShsm control data sets (CDS), DFSMShsm activity is quiesced to ensure integrity of the backup. Although Concurrent Copy can reduce the backup of each control data set to a few seconds, the backup of the journal can take many minutes. This can impact production processing because jobs may be waiting for DFSMShsm recalls, recoveries, or backups.

6.14.2 Installation considerations for z/OS V1R13

A non-intrusive journal backup will minimize the time that DFSMShsm activity is quiesced during CDS backup. The time that DFSMShsm is quiesced during CDS backup would change from many minutes to only a few seconds. This could reduce delays to production processing.

The CDS backup window now allows DFSMShsm activity while the journal is being backed up when the CDSs are backed up using DSS Concurrent Copy and when the journal is configured to be written in RECOVERY mode (for example, with a **SETSYS JOURNAL(RECOVERY)** command specified). While simultaneously allowing DFSMShsm activity, the journal backup task backs up all journal records present at the time the backup is started.

DFSMShsm activity is quiesced briefly while all new updates (those occurring after the start of journal backup) are backed up and the journal is nulled. DFSMShsm activity continues to be quiesced while the backups of the CDSs occur.

For this to occur, the backup order is changed to Journal, MCDS, BCDS, and OCDS to allow the journal to start and complete before the backup of the CDSs begin; see Figure 6-8. This allows the journal backup and CDS backups to remain in sync.

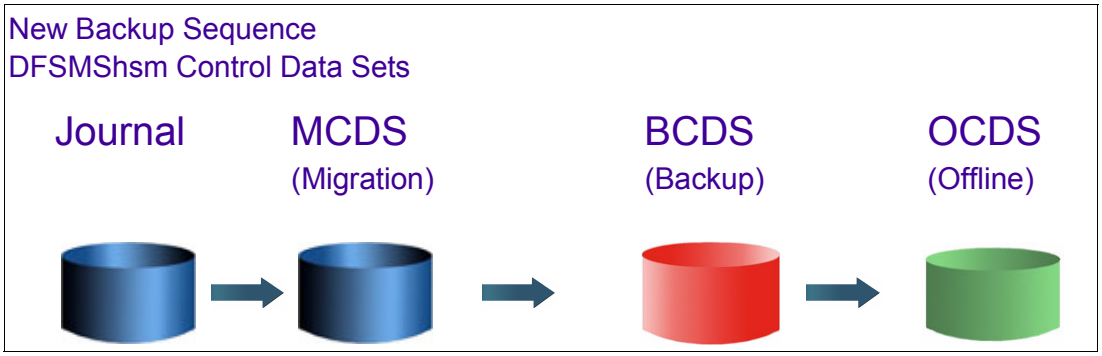


Figure 6-8 All CDSs are backed up concurrently when non-intrusive journal is eligible

Note: Non-intrusive journal backup will be automatically performed on DFSMShsm hosts with the following configuration:

- The **`SETSYS JOURNAL(RECOVERY)`** command is specified.
- The **`SETSYS CDSVERSIONBACKUP(DATAMOVER(DSS))`** command is specified.

  DFSMShsm writes the BCDS, MCDS, and OCDS data set records in the journal data set when updated. The parameter used is JOURNAL(RECOVER).
- MCDS, BCDS, and OCDS are SMS-managed.
- Each CDS's management class indicates a BACKUP COPY TECHNIQUE other than STANDARD.

Note: APAR OA32993 provides coexistence maintenance for lower-level systems that coexist (share resources) with V1R13 hosts.

Message ARC0750I

If the installation requirements are not met, the existing technique or “Quiesced Journal Backup” will be performed. If non-intrusive journal backup does not successfully run, then message ARC0750I is issued.

Message ARC0750I, an existing message that is always issued, indicates if the journal backup technique is non-intrusive or quiesced and why the backup was not performed. When quiesced is indicated, the message will also indicate the reason why non-intrusive backup was not performed; see Figure 6-9.

```
ARC0740I CDS BACKUP STARTING AT 10:01:17 ON 2011/05/05,  329
ARC0740I (CONT.)  SYSTEM SC74, TO DASD IN PARALLEL MODE, DATAMOVER=DSS
ARC0750I BACKUP FOR JRNL STARTING AT 10:01:17, ON  330
ARC0750I (CONT.) 2011/05/05, BACKUP TECHNIQUE IS NON-INTRUSIVE
ARC0743I JRNL SUCCESSFULLY BACKED UP TO  331
ARC0743I (CONT.) DINIZ.JRNL.BACKUP.V0000003, ON VOLUME(S) SBOX0Q,
ARC0743I (CONT.) TIME=10:01:17, DATE=2011/05/05
ARC0742I BACKUP FOR MCDS STARTING AT 10:01:17 ON  332
ARC0742I (CONT.) 2011/05/05, BACKUP COPY TECHNIQUE IS PREFERRED
ARC0742I BACKUP FOR BCDS STARTING AT 10:01:17 ON  333
ARC0742I (CONT.) 2011/05/05, BACKUP COPY TECHNIQUE IS PREFERRED
ARC0742I BACKUP FOR OCDS STARTING AT 10:01:17 ON  334
ARC0742I (CONT.) 2011/05/05, BACKUP COPY TECHNIQUE IS PREFERRED
ARC0743I BCDS SUCCESSFULLY BACKED UP TO  338
ARC0743I (CONT.) DINIZ.BCDS.BACKUP.V0000003, ON VOLUME(S) SBOX0V,
ARC0743I (CONT.) TIME=10:01:18, DATE=2011/05/05
ARC0743I MCDS SUCCESSFULLY BACKED UP TO  344
ARC0743I (CONT.) DINIZ.MCDS.BACKUP.V0000003, ON VOLUME(S) SBOX0W,
ARC0743I (CONT.) TIME=10:01:19, DATE=2011/05/05
ARC0743I OCDS SUCCESSFULLY BACKED UP TO  346
ARC0743I (CONT.) DINIZ.OCDS.BACKUP.V0000003, ON VOLUME(S) SBOX0X,
ARC0743I (CONT.) TIME=10:01:19, DATE=2011/05/05
ARC0748I LAST SUCCESSFUL CDS BACKUP-SET QUALIFIER IS  347
ARC0748I (CONT.) D0000003
ARC0741I CDS BACKUP ENDING AT 10:01:19 ON 2011/05/05,  348
ARC0741I (CONT.) STATUS=SUCCESSFUL
```

Figure 6-9 BACKVOL CDS with non-intrusive journal backup

6.15 ARCCAT release

CDS backup requires an exclusive enqueue on the resource with the major name of ARCCAT and the minor name of ARCGPA for non-RLS CDSs or ARCENQG for RLS CDSs. Long-running functions in DFSMShsm may hold this resource for indefinite amounts of time, causing CDS backup to wait.

Newly started functions then must wait for CDS backup to obtain and release the resource before they can start. This can lead to many minutes of delays to production processing that may rely on DFSMShsm functions to complete.

The ARCCAT release function will ensure CDS backup can start and complete quickly so that newly started functions are not delayed due to contention for the ARCCAT resource.

With z/OS V1R13, all functions release the ARCCAT resource when CDS backup starts so that CDS backup completes quickly. Pending CDS updates are allowed to complete before release of ARCCAT, and new updates are held until CDS backup completes, thus ensuring integrity of backup.

Newly started DFSMShsm functions are not required to wait until CDS backup can obtain the resource, complete backup of the CDSs, and then release the resource. Production processing that relies on these functions will not be delayed.

ARCCAT release will be performed on all z/OS V1R13 systems automatically.

Note: For configurations with multiple DFSMShsm hosts in an HSMplex, XCF services are required to communicate the start of CDS backup to other hosts. Only hosts on z/OS V1R13 systems will perform ARCCAT release.

Performance considerations

When concurrent copy is used for backing up the CDSs, and the **SETSYS JOURNAL(RECOVERY)** command is specified, the amount of time resources are held during CDS backup is dramatically shortened. In past releases, CDS backup activity was largely dependent on the time it took to back up the CDS and journal. Journal backup was not done at the beginning of the backup process due to the changing contents of the journal, and the need to capture the entire journal contents. Therefore, DFSMShsm activity was quiesced during the entire backup process. As a result, user and production jobs requiring recall or recovery waited until the CDS Backup function completed. This presented performance considerations that had to be factored in to backup planning. As a result, DFSMShsm activity is quiesced for much less time with non-intrusive journal backup.

MASH configuration considerations

When running in a multiple address space DFSMShsm (MASH) configuration where XCF services are not available, keep the following considerations in mind:

- Long-running functions on other DFSMShsm hosts (including those that start before automatic CDS backup) will not release the resources required to allow CDS backup to obtain an exclusive lock. This prevents starting CDS backup.
- The enqueue prevents all data set or volume-type functions from accessing the control data sets while they are being backed up. Therefore, carefully consider the best time to start backing up control data sets.

6.16 Space management performance improvements

Space management is the function of DFSMShsm that allows you to keep DASD space available for users to meet the service level objectives for your system. The purpose of space management is to manage your DASD storage efficiently. To do this, space management automatically and periodically performs functions that move low-activity data sets from user-accessible volumes to DFSMShsm volumes, and reduce the space occupied by data on both the user-accessible volumes and the DFSMShsm volumes.

6.16.1 Replace interval migration with on-demand migration

An interval migration (IM) is a scheduled process in which DFSMShsm performs a space check on each DFSMShsm volume that is being managed. DFSMShsm migrates eligible data sets to ML1 or ML2 volumes.

That process is well known to cause high processor usage and to consume significant “wall clock” time. Beginning in z/OS V1R13, installations can choose which type of space management processing they want DFSMShsm to perform:

- On the hour (called interval migration), or
- A new option called on-demand migration (ODM)

Note: This new option called on-demand migration (ODM) is a enhancement that provides a timely space management process that runs against volumes that have exceeded their threshold, rather than having the volumes wait for a top-of-the-hour space management scheduled task (such as interval migration).

When using ODM, SMS will issue a new ENF72 event when the volume threshold is exceeded during allocation of a data set on an SMS volume. DFSMShsm will process the volume after receiving the ENF72 from SMS when using on-demand migration space management.

Note: The ENF72 notification is SMS complex (SMSplex)-wide. The ENF72 signal is a newly created ENF signal. ENF72 event types and qualifiers are described in *z/OS MVS Authorized Assembler Services Reference EDT-IXG*, SA22-7610, and *z/OS MVS Authorized Assembler Services Guide*, SA22-7608.

SETSYS commands

The new **SETSYS** command parameters are called **ONDEMANDMIGRATION** and **ODMNOTIFICATIONLIMIT**, as explained here:

- The **SETSYS ONDEMANDMIGRATION** command

  ONDEMANDMIGRATION is an optional parameter specifying whether on-demand migration should be used instead of interval migration for SMS-managed volumes. This parameter can be abbreviated as ODM.

  Note: The **ONDEMANDMIGRATION** parameter applies only to SMS-managed volumes that have a storage group setting of AM=Y. The subparameters of the **ONDEMANDMIGRATION** parameter are:

  - **N** - This indicates that interval migration is performed for SMS-managed volumes at the top of every hour.
  - **Y** - This indicates that interval migration is ignored and not performed at the top of every hour. Instead, DFSMShsm listens for an SMS ENF72 signal (which is issued when a volume exceeds threshold due to a new data set allocation or extension) and performs on-demand migration space management on volumes that have a storage group setting of AM=Y.

    If you want to have TMM storage groups managed by on-demand migration, you must convert AM=I TMM storage groups to AM=Y.

- The **SETSYS ODMNOTIFICATIONLIMIT** command

  This command specifies a notification limit for the number of volumes being processed by on-demand migration. The subparameter limit is a decimal number from 1 to 99999.

  Default: 100 is the default setting for the **ODMNOTIFICATIONLIMIT** parameter.

  For example, if the **SETSYS ODMNOTIFICATIONLIMIT(250)** command is issued and the number of SMS-managed volumes with storage group settings of AUTOMIGRATE=Y whose high thresholds are exceeded reaches 250, a new highlighted message to the operator console will be issued. The new message is:

  ```
  ARC1901E NUMBER OF VOLUMES ELIGIBLE FOR ON-DEMAND MIGRATION HAS REACHED 250
  ```

  When the number of SMS-managed volumes with AM=Y whose high thresholds are exceeded becomes less than 250, this message will be removed from the operator console.

To display the value of the subparameter limit and the value for ODM, issue the **QUERY** SETSYS command.

The result of the **SETSYS ONDEMANDMIGRATION(Y)** and **SETSYS ODMNOTIFICATIONLIMIT(250)** commands will be displayed in the ARC0153I message on the operator console, as shown in Figure 6-10 on page 135.

```
ARC0153I SCRATCHFREQ=0007, SYSOUT(CLASS=A, COPIES=01,  288
ARC0153I (CONT.) SPECIAL FORMS=NONE), SWAP=NO, PERMISSION=NO,
ARC0153I (CONT.) EXITS=NONE, UNLOAD=NO, DATASETSERIALIZATION=DFHSM,
ARC0153I (CONT.) USECMS=NO, ONDEMANDMIGRATION=YES,
ARC0153I (CONT.) ODMNOTIFICATIONLIMIT=00250
```

Figure 6-10 ARC0153I - on-demand migration

On-demand migration provides a timely space management process that runs against volumes that have exceeded their threshold, rather than having the volumes wait for a top-of-the-hour space management scheduled task (such as interval migration).

6.16.2 Auto Migrate support

Support for configuring the use of DFSMShsm on-demand migration has been added in z/OS V1R13 to the pool storage group definition (Auto Migrate).

Values for defining a pool storage group are in the storage group and management class. The storage group Auto Migrate and Auto Backup parameters specify whether the volumes in this storage group are eligible to be processed automatically.

The management class, assigned to the data sets residing on the volumes, determines whether and how to process the data sets on the volume. In contrast, if you set Auto Migrate or Auto Backup to N (No) in the storage group, the volumes in the storage group are not processed and data sets residing in the storage group are not migrated or backed up.

Attention: Auto Migrate is not new, but with the new on-demand migration, there are changes to the Auto Migrate parameters.

You can specify the following attributes on the Pool Storage Group Define panel:

Auto Migrate This specifies whether the DASD volumes in this storage group are eligible for automatic space management processing. Auto Migrate is a required field, which ISMF primes with the value Y (yes).

Y (yes) - This specifies that data sets are eligible for primary space management and that DFSMShsm performs automatic space management if a processing window is defined to DFSMShsm. If the DFSMShsm system setting **SETSYS INTERVALMIGRATION** command is specified, the data sets are eligible for interval migration. If the DFSMShsm system setting **SETSYS ONDEMANDMIGRATION(Y)** command is specified, the data sets are eligible for on-demand migration and are not eligible for interval migration, even if a **SETSYS INTERVALMIGRATION** command is specified.

N (no) - This specifies that data sets are not eligible for automatic migration.

I (interval) - This specifies that data sets are eligible for primary space management and forces DFSMShsm to perform automatic interval

migration independent of the DFSMShsm system setting for **SETSYS INTERVALMIGRATION** command. Interval migration is performed on volumes that are at or above the midpoint between the high and low threshold. Interval migration is performed hourly. If a low threshold value of 0 is specified, DFSMShsm migrates all eligible data sets in the selected storage group. DFSMShsm on-demand migration is not performed. This value is most useful for storage groups used with tape mount management.

P (primary) - This specifies that data sets are eligible for primary space management. Interval migration and on-demand migration are not performed regardless of the DFSMShsm system setting for **SETSYS INTERVALMIGRATION** and **SETSYS ONDEMANDMIGRATION** commands.

Note: If for some reason (such as a system outage) system-managed temporary data sets are not deleted at the end of a job, DFSMShsm attempts to delete them during primary space management. For DFSMShsm to delete these data sets, you must specify Auto Migrate as Y, I, or P.

Migration and allocation thresholds

You can specify an upper and lower space limit for the DASD volumes in a pool storage group. SMS tries to stay within these thresholds by looking at the primary space allocation of each data set before assigning it to a given DASD volume.

For example, the SMS volume selection function attempts to prevent allocation of a data set to a given DASD volume if that allocation causes the volume's high threshold to be exceeded. When a volume reaches or exceeds the high threshold, SMS issues an ENF72 signal, regardless of the Auto Migrate setting.

6.16.3 Installation considerations

By default, a volume that remains over threshold is eligible to be processed again by on-demand migration (ODM) after 24 hours. However, sometimes a volume remains over threshold (after being processed by ODM) because there are no eligible data on the volume to be moved.

If you do not want DFSMShsm to reselect such a volume for ODM processing after 24 hours, you can adjust the frequency of running ODM on over-threshold volumes by issuing a patch similar to the one in the following example which will instruct DFSMShsm to wait 48 hours:

```
PATCH .MGCB.+138 X '0002A300' /* 48 hours in seconds */
```

6.17 SMS configuration change refresh

One of the unnecessary top-of-the-hour checks that DFSMShsm performs is to determine whether the SMS configuration was modified.

To avoid this hourly check DFSMShsm, z/OS V1R13 has implemented a new ENF 15 listening exit called ARCENF15 that will get control every time an SMS configuration change has occurred.

DFSMShsm checks to see if an ENF15 event has occurred every time primary space management, interval migration, or on-demand migration starts. If an ENF15 event has

occurred, then an internal SMS configuration information refresh is performed by DFSMShsm.

Because SMS configuration changes are infrequent, this will eliminate the additional overhead of work that DFSMShsm does by checking for the SMS configuration change each hour. No additional SMS configuration refresh will be done until the next ENF15 event occurs.

This enhancement now allows for DFSMShsm to listen for an SMS configuration change and refresh DFSMShsm internal SMS configuration information upon receiving the notification from SMS.

Note: The command **`PATCH .MCVT.+C8 BITS(1.......)`** can be removed from the SYS1.PARMLIB ARCCMDxx member. This patch command was used to prevent DFSMShsm from performing SMS configuration updates to internal records. It is not necessary to remove it but removal is recommended to avoid future misunderstandings of its usefulness.

The description of this command has been deleted from *z/OS DFSMShsm Implementation and Customization Guide*, SC35-0418.

Consider the following enhancements:

- The functional enhancement to replace interval migration with on-demand migration that is now supported by DFSMShsm using the new ENF72 over threshold event notification issued by SMS.
- The enhancement to listen for an SMS configuration change and refresh DFSMShsm internal SMS configuration information upon receiving the notification from SMS. DFSMShsm now listens and takes action when the ENF15 notification is issued by SMS when SMS configuration has changed.

6.18 Small data set packing performance (SDSP) improvements

Installations use DFSMShsm's SDSP to manage millions of small data sets and the ARCMMEXT migration exit to keep these small data sets on ML1 to reduce the migration overhead cost to ML2. Currently, DFSMShsm goes through its queue scanning and SDSP serialization for each data set, which includes those that the ARCMMEXT will later reject for migration. This processing is not very efficient if the ARCMMEXT exit rejects the request to migrate the data set.

A second situation occurs as currently DFSMShsm takes the ML1 volumes with SDSPs in the order in which they were added with an ADDVOL, using the same ones repeatedly until they are full. This results in RECALL contention problems and SDSP full conditions, which require a reorganization of the SDSP.

z/OS V1R13 enhancements

There are two main focus areas addressed in this new support in z/OS V1R13:

- The point of invocation for the ARCMMEXT exit is moved to reduce unnecessary overhead.

  Data sets will no longer go through the overhead of secondary space management (SSM) processing for data sets that the ARCMMEXT exit will later reject for migration.

Performance improvements can be seen in installations with high SDSP utilization that use the ARCMMEXT exit.

- The new, balanced SDSP selection algorithm is introduced.

  The new, balanced SDSP selection algorithm will fill the SDSPs evenly, resulting in fewer reorganizations and fewer contention problems.

6.18.1 Moving the point of invocation for the ARCMMEXT

In prior releases, DFSMShsm goes through its queue scanning and SDSP serialization for each data set, which includes those that the ARCMMEXT exit will later reject for migration. This processing is not very efficient if the ARCMMEXT exit rejects the request to migrate the data set.

With z/OS V1R13, the point of invocation for the ARCMMEXT exit is moved to prior to queue scanning and SDSP serialization. Now data sets no longer go through the overhead of secondary space management (SSM) processing for data sets that the ARCMMEXT exit will later reject for migration.

The ARCMMEXT exit will now receive control first during the MCD scanning to validate whether the data set is eligible to be migrated. Then, if the data set is going to be migrated, DFSMShsm places the data set onto the SMQE migration queue. Otherwise, if the ARCMMEXT rejects the data set for migration to ML2, then no additional SSM overhead is done. See *z/OS DFSMS Installation Exits*, SC26-7396, for further information about this topic.

New balanced SDSP selection algorithm

In z/OS V1R13, the new balanced SDSP selection algorithm will fill the SDSPs evenly, resulting in fewer reorganizations and fewer contention problems. This SDSP selection algorithm is based on free space rather than ADDVOL sequence.

Note: The new balanced SDSP selection algorithm is always used in z/OS V1R13 DFSMShsm. The main criterion for the SDSP data set selection is the free space in the SDSP data set. Free space will be calculated during two time periods of SDSP processing:

- During ADDVOL processing
- When an SDSP is closed at the completion of processing data

This enhancement takes into account the VSAM CA reclaim feature. SDSP free space is calculated using high allocated and high used RBA values for base DATA and INDEX components. If clients use the CA Reclaim feature for VSAM, this free space will be increased by the free space of reclaimed CAs.

6.19 DFSMShsm serviceability and usability enhancements

Many small enhancements are provided to improve the serviceability and usability of DFSMShsm:

- PDA trace during DFSMShsm startup
- Fast replication ARC1809I messages
- RELEASE RECALL(DASD)
- QUERY CRQ

- ONLYIF enhancements
- ARC0570I patch
- AUDIT COPYPOOL
- Change messages from I to E
- Change FR(DSR) default to NONE

6.20 PDA trace during DFSMShsm startup

The problem determination aid (PDA) facility gathers sufficient DFSMShsm processing information to pinpoint module flow and resource usage that is related to any DFSMShsm problem. The PDA facility is required for IBM service because it traces module and resource flow. DFSMShsm stores its trace information in the PDA log data sets.

If PDA=YES is not specified in the startup procedure, then PDA tracing will not occur during DFSMShsm startup, leading to difficulty diagnosing errors that can occur during startup.

z/OS V1R13 implementation

In z/OS V1R13, the default for PDA trace during DFSMShsm startup will be changed to PDA=YES. Clients set PDA=NO in the DFSMShsm startup procedure if they do not want PDA trace enabled during DFSMShsm startup.

After the ARCCMDxx parmlib member is processed during DFSMShsm initialization, DFSMShsm will begin tracing all other DFSMShsm activity unless the SETSYS PDA(NO) parameter is specified.

6.21 Fast replication ARC1809I messages

Fast replication backup processing issues excessive ARC1809I messages when reporting the reason why a target volume cannot be paired to a source volume. The ARC1809I RC2 message indicates that a source and target cannot be paired because the target volume has already been matched with a different source. For a single FRBACKUP command, the ARC1809I RC2 can be issued many times for each target volume.

In previous releases, a patch command can be used to turn off all ARC1809I messages. In z/OS V1R13, a new **SETSYS** command called **FASTREPLICATION** replaces the existing patch to enable or disable the redundant ARC1809I messages.

This enhancement will decrease the number of ARC1809I RC2 messages so that the message will be issued no more than once per target volume for each storage group in the copy pool.

If you currently have **PATCH .FRGCB.+9 BITS(.1.....)** in your ARCCMDxx parmlib member, replace the patch with the desired command as follows:

```
SETSYS FASTREPLICATION(VOLUMEPAIRMESSAGES(YES|NO))
```

YES This indicates that ARC1809I RC2 messages are to be issued when a target volume cannot be paired with a source volume.

NO This indicates that ARC1809I RC2 messages should not be issued.

Note: The existing patch will continue to be supported, but all external references to it will be removed from documentation.

6.22 RELEASE RECALL(DASD) command

RECALL is an optional parameter specifying whether DFSMShsm should release automatic recall, recall by command, and deletion of a migrated data set from all volumes, from all tape volumes, or recalls that need tapes and were submitted by a TSO user.

The RECALL parameter, without any subparameters, specifies that DFSMShsm release all recall and data set deletion tasks from all volumes.

When a **HOLD RECALL(TAPE)** command is issued in response to a tape-related problem, there is no method available to change the hold level to **HOLD RECALL(TAPE)** command without first releasing all recalls.

z/OS V1R13 enhancements

With z/OS R1V13, DFSMShsm has added support for a **RELEASE RECALL(DASD)** command. It enables recall requests from ML1 to continue when there is a tape-related issue.

DASD This specifies that DFSMShsm release all recalls from DASD volumes only, while maintaining the hold on recalls from tape volumes.

The **RELEASE RECALL(DASD)** command will convert a HOLD RECALL state to a HOLD RECALL(TAPE) state and QUERY ACTIVE will indicate a hold on tape recall tasks.

RECALL(DASD) This goes against the current restrictions of RELEASE RECALL levels. RELEASE RECALL(DASD) exists on the same level as RELEASE RECALL(TAPE), but is meant to be issued against the higher level HOLD RECALL.

Note: There is no accompanying **HOLD RECALL(DASD)** command. Instead, the **RELEASE RECALL(DASD)** command essentially converts a HOLD RECALL state to a HOLD RECALL(TAPE) state.

6.23 QUERY CRQ

The common recall queue (CRQ) is a single recall queue that is shared by multiple DFSMShsm hosts. This queue is implemented through the use of a coupling facility (CF) list structure. The CRQ balances the recall workload across multiple address spaces. It enables optimization for priority and single-tape mounts. In some Parallel Sysplex configurations, all hosts are not connected to all devices. The CRQ enables hosts that are not connected to a device, like a tape drive, to place recall requests on the CRQ. After it is on the CRQ, the request can be selected by hosts that are connected to the device.

A recall request in the common recall queue (CRQ) can only be canceled from the host that originated the request. To determine which host originated the request, a QUERY must be issued on each DFSMShsm host.

z/OS V1R13 enhancements

In z/OS V1R13, the output from the **QUERY COMMONQUEUE(RECALL)** command shows the originating host id, which can simplify data gathering.

As a result of the **QUERY COMMONQUEUE(RECALL)** command, message ARC1543I is issued, which now includes text showing the host id of the host that initiated the request:

```
{, REQUEST ORIGINATED ON HOST hostid}
```

Note: The ARC1543I message update is not displayed when it is the result of a **QUERY DATASETNAME**, **QUERY REQUEST**, or **QUERY USERID**.

HSMplex considerations

If there are hosts within an HSMplex that have data that cannot be shared between systems, then those hosts are not to share the same CRQ.

If there are sets of hosts within an HSMplex that cannot share data, then each of those sets of hosts can share a unique CRQ so that there are multiple CRQplexes within a single HSMplex (for example, test systems and production systems that are within the same HSMplex, but have data that they cannot share).

6.24 ONLYIF enhancements

Currently, the **ONLYIF** command allows a single command to be executed conditionally. Within a multi-host system, **ONLYIF** allows the definition of parameters for each host within a single ARCCMDxx parmlib member.

Storage administrators of multi-host systems want a single **ONLYIF** command to be able to conditionally execute a block of commands, so that the ARCCMDxx parmlib member is easier to understand and maintain.

Because the **ONLYIF** command processes the next command, then if the line after the **ONLYIF** command is blank or contains a comment, DFSMShsm considers that line; for example, the comment after the **ONLYIF** command is considered for processing, and not the actual **SETSYS** command:

```
ONLYIF HSMHOST(A)
/* The following is only done for Host A */
SETSYS ABSTART(0600 0700)
```

To avoid this error, use a continuation character to join the comment to the command, as shown:

```
ONLYIF HSMHOST(A)
/* The following is only done for Host A */ +
SETSYS ABSTART(0600 0700)
```

z/OS V1R13 enhancements

The **ONLYIF** command allows the single command, or group of commands, contained within a BEGIN...END block immediately following the **ONLYIF** command to be executed conditionally, depending on the host being started. An **ONLYIF** command may not follow another **ONLYIF** command, and it may not be nested within an ONLYIF BEGIN...END block.

Note: **ONLYIF** allows conditional execution of the single command, or group of commands, contained within a BEGIN...END block, immediately following the **ONLYIF** command in the ARCCMDxx parmlib member.

The following figures illustrate the methods of issuing the **SETSYS** commands to Host A and Host B.

Figure 6-11 illustrates the “old” method of using the **ONLYIF** command.

```
ONLYIF HSMHOST(A)
   SETSYS ABSTART(0600 0700)
ONLYIF HSMHOST(B)
   SETSYS ABSTART(0600 0700)
ONLYIF HSMHOST(A)
   SETSYS ABARSTAPES(NOSTACK)
ONLYIF HSMHOST(B)
   SETSYS ABARSTAPES(NOSTACK)
```

Figure 6-11 The “old” way to use the ONLYIF command

Figure 6-12 illustrates the new method of using the **ONLYIF** command.

```
ONLYIF HSMHOST(A,B)
BEGIN
SETSYS ABSTART(0600 0700)
SETSYS ABARSTAPES(NOSTACK)
END
```

Figure 6-12 The new way that the ONLYIF command can be used

Note: The old **ONLYIF** command usage still works and a coexistence PTF is available for releases V1R10, V1R11, and V1R12 to tolerate the **ONLYIF** command changes made on a V1R13 system to a shared parmlib member.

This coexistence support will properly parse and process BEGIN/END logic and multiple host ids within the HSMHOST keyword.

6.25 ARC0570I patch

ARC0570I messages are unconditionally issued when certain SMS constructs have not been defined. Until now, there is no way to suppress these messages when this is a known and acceptable condition. The message is as follows:

```
ARC0570I {PRIMARY SPACE MANAGEMENT | INTERVAL MIGRATION | ON-DEMAND MIGRATION |
COMMAND MIGRATION | AUTOMATIC BACKUP | COMMAND BACKUP | AUTOMATIC DUMP |
COMMAND DUMP | RESTORE | RECOVERY | FRBACKUP | FRRECOV} FOR {ALL SMS MANAGED |
volser | volser,SGROUP=sg | ALL COPY POOL | COPY POOL=cpname} VOLUME(S)
TERMINATED, RC=return-code, REASON=reason-code
```

z/OS V1R13 implementation

Two new patches are introduced in DFSMShsm for z/OS V1R13 to filter ARC0570I RC17 and RC36 messages, as follows:

- During autobackup
- During automigrate and autodump functions

Note: During AUTOBACKUP, AUTOMIGRATION and AUTODUMP, HSM will attempt to retrieve SMS storage group and copy pool information.

If no storage group or copypool information is found, ARC0570I RC 17 or RC36 is issued, respectively.

This item adds two patches to turn off these messages because, in non-SMS or non-copy pool environments, they can confuse users.

The messages will still be issued during a **LIST** command, regardless of whether or not the patches are ON.

This could eliminate periodic analysis of messages which are known to be acceptable by some, but not by others. The messages are as follows:

```
ARC0570I RC17 (No storage groups defined) messages will be turned off using the
following patch: PATCH .MCVT.+297 BITS(....1...)

ARC0570I RC36 (No copy pools defined) messages will be turned off using the
following patch: PATCH .MCVT.+297 BITS(.....1..)
```

6.26 AUDIT COPYPOOL

In previous releases, the AUDIT COPYPOOLCONTROLS function does not report fast replication target volume records that are invalidly paired with a source volume record.

z/OS V1R13 implementation

The AUDIT COPYPOOLCONTROLS function will now detect invalid target volume records.

When AUDIT detects an orphan FRTV record, a new AUDIT error message will be issued to the AUDIT output to identify the orphaned FRTV.

At the end of the COPYPOOLCONTROLS audit, *ERR 202 is issued for each FRTV record that is not indicated as associated with any of the copypool records that were visited during the audit.

```
*ERR 202 ORPHANED I (FRTV) RECORD FOUND FOR TARGET VOLUME <tgtvolser>, SOURCE
VOLUME <srcvolser>
```

Following the error message, AUDIT will issue the **FIXCDS DELETE** command required to resolve the error, if FIX is specified on the AUDIT command.

Note: This detection will not occur when a specific copy pool name has been specified.

6.27 Change messages from I to E

The following messages have the suffix changed from I to E:

```
ARC0036I to ARC0036E:
ARC0036E I/O {INHIBITED | DISABLED} FOR DFSMSHSM PROBLEM DETERMINATION OUTPUT
DATA SET, REAS=reason-code

ARC0503I to ARC0503E
ARC0503E ALLOCATION ERROR, {VOLUME=volser | DATA SET=dsname | DD DUMMY},
RC=return-code, REASON=reason-code, INFO CODE=infocode, EXTENDED REASON
CODE=extreas

ARC0704I to ARC0704E
ARC0704E {BACKUP | DUMP | RECOVER} OF VOLUME volser1 TERMINATED, ERROR
{ALLOCATING | OPENING | CLOSING | READING | WRITING} VTOC COPY DATA SET [ON
VOLUME volser2]
```

Note: If you have message automation that detects these message numbers, it needs to be validated.

6.28 Change FR(DSR) default to NONE

When fast replication PREFERRED is in effect, DFSMShsm will use fast replication as the copy method whenever possible for data set level recoveries. This may cause a subsequent copy pool backup to fail due to the background copy still being in progress.

It is difficult to analyze the copy pool backup failure and make adjustments given the messages only indicate the backup volume was already an IBM FlashCopy® source. When FASTREPLICATION(DSR(NONE)) is specified, the failure messages indicate serialization failure on the copy pool, which is more helpful in resolving the situation.

z/OS V1R13 implementation

To use traditional copy methods as the default, in z/OS V1R13 the default for data set level recovery has changed from PREFERRED to NONE and subsequent copy pool backups will not fail due to a background copy initiated by previous data set recoveries still being in progress.

```
SETSYS FASTREPLICATION(DATASETRECOVERY(PREFERRED | REQUIRED | NONE)
DEFAULT: NONE
```

FASTREPLICATION(PREFERRED | REQUIRED | NONE) is an optional keyword on the **FRRECOV** command, that overrides the **FASTREPLICATION(DSR(PREFERRED | REQUIRED | NONE))** value specified on the **SETSYS** command. When the FASTREPLICATION parameter is not specified on the **FRRECOV** command, the **SETSYS FASTREPLICATION(DSR)** specification is used.

If you do not specify FASTREPLICATION on a **FRRECOV DSNAME** command, or FASTREPLICATION(DSR) on a **SETSYS** command, the default is NONE.

Note: Clients currently relying on the previous default value of **SETSYS FASTREPLICATION(DSR(PREFERRED))** should determine whether to keep the **PREFERRED** option by specifying the **SETSYS FASTREPLICATION(DSR(PREFERRED))** command to override the new default on V1R13.

To specify **FASTREPLICATION(DSR(NONE))**, allow DFSMShsm to default to NONE.

7

ISPF enhancements

The Interactive System Productivity Facility (ISPF) is a dialog manager that provides tools to improve program, dialog, and development productivity and control.

ISPF is a multifaceted development tool set for the z/OS operating system. Since 1975, MVS programmers have used ISPF for host-based application development productivity. ISPF forms the basis of many TSO and CMS applications and also provides extensive programmer-oriented facilities.

ISPF helps programmers develop interactive applications called *dialogs*. Dialogs are interactive because ISPF uses them to communicate with terminal users through a series of panels while the users do application development tasks.

In this chapter, the following topics are described:

- z/OS UNIX Directory List enhancements
- New data set allocation command for ISPF Data Set List
- Display data set create jobname and stepname
- **`INFO`** command to display PDS member extended statistics

7.1 z/OS UNIX Directory List enhancements

Access control list (ACL) support, which began in z/OS, is based on a POSIX standard and other UNIX implementations. ACLs were introduced in z/OS as a way to provide a greater granularity for access to z/OS UNIX files and directories. This support provides for the use of ACLs to control access to files and directories by an individual user (UID) and a group (GID). ACLs are now created and checked by RACF. ACLs are created, modified, and deleted by using either of the following:

- **setfacl** shell command
- ISHELL interface

z/OS UNIX support for ACLs

ACLs have existed on various UNIX platforms for many years, but with variations in the interfaces. ACL support in z/OS is based on a POSIX standard that was never implemented when z/OS UNIX was first introduced as OpenEdition.

In the POSIX standard, two ACLs are referenced:

- Base ACL entries

  These entries are permission bits (owner, group, other). This refers to the FSP. You can change the permissions using **chmod** or **setfacl**commands. They are not physically part of the ACL, although you can use **setfacl** to change them and **getfacl** to display them.

- Extended ACL entries

  These entries are ACL entries for individual users or groups, such as the permission bits that are stored with the file, not in RACF profiles. Extended ACL entries, like the permission bits, are stored with the file, not in RACF profiles.

  Each ACL type (access, file default, directory default) can contain up to 1024 extended ACL entries. Each extended ACL entry specifies a qualifier to indicate whether the entry pertains to a user or a group; the actual UID or GID itself; and the permissions being granted or denied by this entry. The allowable permissions are read, write, and execute. As with other UNIX commands, **setfacl** allows the use of either names or numbers when referring to users and groups.

7.1.1 ISPF support for ACLs in z/O V1R13

Adding ACL support is part of the long-term plan for the z/OS UNIX Directory List Utility (ISPF Option 3.17) to provide most of the functions currently supported by the ISHELL utility. The objective is to reduce the need for users to switch between the z/OS UNIX Directory List Utility and the ISHELL. Also, adding and updating ACL entries using the UNIX command SETFACL from the OMVS command line can be difficult for z/OS systems programmers. With this enhancement, creating ACLs is performed using a more familiar technique.

The z/OS UNIX Directory List Utility now supports the new **MA** line command for the display and update of ACLs for a file or directory. When the **ML** line command is entered the z/OS UNIX ACL List panel is displayed, showing a list of ACL entries defined for the file or directory.

Adding a new ACL entry

To add a new ACL, first select the file or directory from IPSF Option 3.17 and then issue the **MA** line command for the file, as shown in Figure 7-1 on page 149. The figure shows the following access to the file:

- The owner of the file has read and write access (raw-).
- Any user ID in the same group as the owner has only read access (r--).
- All other z/OS UNIX users are in the other category, they also have read access (r--).

When you create an ACL, you are giving a specific user a specified access to the file.

```
ISRUUDL0                  z/OS UNIX Directory List            Row 1 to 5 of 5
Command ===>                                                 Scroll ===> PAGE

Pathname . : /u/soares

Command  Filename        Message          Type Permission Audit  Ext  Fmat
------------------------------------------------------------------------------
 _______ .                                 Dir  rwxr-xr-x  fff---      ----
 _______ ..                                Dir  r-xr-xr-x  ------      ----
 _______ .profile                          File rwx------  fff--- --s- ----
 _______ .sh_history                       File rw-------  fff--- --s- ----
 ma_____ acltest          Modified         File rw-r--r--  fff--- --s- ----
******************************* Bottom of data ********************************
```

Figure 7-1 Updating an ACL for a file using the MA line command

Figure 7-2 on page 150 then is displayed. To create the first ACL for this file, enter the **a** command on the command line. This panel (ISRUULMA) uses a table display to represent the Access Control Lists that is currently defined. An empty table can also be displayed, as shown in Figure 7-2 on page 150.

If there is a list, it is sorted in UID order; the following are valid on the command line:

A Add a new ACL.

SA Sort the ACL list alphabetically by user ID.

SN Sort the ACL list numerically on UID.

ST Sort the ACL list alphabetically by type.

If there are ACL entries to be displayed, the following line commands are available:

A Can be used to add further entries.

D Can be used to delete the entry.

X Can be used to list members of an OMVS group.

To enter a new ACL entry use the **a** command at the ACL List panel, as shown in Figure 7-2 on page 150.

```
ISRUULMA                    z/OS UNIX ACL List
Command ===> a                                          Scroll ===> PAGE

S   UID       Read  Write  eXecute  Name      Type
****************************** Bottom of data ********************************
```

Figure 7-2 Enter the a command on the command line to create an ACL

Figure 7-3 on page 150 is displayed, and this is where you create the ACL. As shown, enter an UID for the intended user and define the type of access to the file. In this example:

- The UID of the user was entered as 74.
- The UID is also given execute (W) to the file.
- The UID is not in the same group as the owner.

```
ISRUULMI                 z/OS UNIX ACL Attributes
Command ===>

Supply a numeric UID value or use the name field. If both are
are entered then the NAME field will be used.
Any non-blank character will set the r , w , x privileges.

UID  . . . . . 74_____
Read . . . . . -
Write  . . . . . w
eXecute  . . . . . _
Name . . . . . ________
Type . . . . . 1   Enter 1 for User or 2 for Group
```

Figure 7-3 Giving UID 74 read and write access to the file

UID 74 now has write access to the file. Note that UID 74 already had read access to the file; as shown in Figure 7-1 on page 149, all UIDs had read access to the file because the “other” category displayed read access.

```
ISRUULMA                    z/OS UNIX ACL List                    Row 1 from 1
Command ===> ___________________________________________ Scroll ===> PAGE

S   UID       Read  Write  eXecute  Name      Type
    74         R    W               DINIZ     USER
****************************** Bottom of data ********************************
```

Figure 7-4 Adding an ACL entry

At the ACL Attributes panel, you can specify the user or group privileges for the file. UID or user names can be used, and the GID or group name, as shown in Figure 7-5 on page 151.

```
ISRUULMI                 z/OS UNIX ACL Attributes
Command ===> ____________________________________________________________

Supply a numeric UID value or use the name field. If both are
are entered then the NAME field will be used.
Any non-blank character will set the r , w , x privileges.

UID  . . . . . . . . _______
Read . . . . . . . . X
Write  . . . . . . . _
eXecute  . . . . . . X
Name . . . . . . . . ZANON___
Type . . . . . . . . 1  Enter 1 for User or 2 for Group
```

Figure 7-5 ACL Attributes panel

To delete an ACL entry, enter `d` at the line command for the entry you want to delete; see Figure 7-6.

```
ISRUULMA                    z/OS UNIX ACL List                    Row 1 from 2
Command ===> _____________________________________ Scroll ===> PAGE
S   UID       Read  Write  eXecute  Name      Type
_   71          R              X     ZANON     USER
d   74          R    W         X     DINIZ     USER
****************************** Bottom of data ********************************
```

Figure 7-6 Deleting an ACL entry

7.2 New data set allocation command for ISPF Data Set List

Many ISPF users spend significant time working from the Data Set List display (ISPF Option 3.4) because it supports most ISPF functions to process data sets. However, to create a new data set they must navigate to ISPF Option 3.2.

With z/OS V1R13, ISPF provides a new Data Set List line command, the **AL** line command, for the allocation of a new data set. This means that users do not have to leave the Data Set List display to create a new data set. This line command may be issued against a data set with attributes that are similar to those needed for the new data set.

ISPF V1R13 support

The ISPF Data Set List panel (ISPF Option 3.4) now supports the new **AL** line command for the allocation of a new data set. This line command uses a new data set name as a parameter. If no parameter is supplied, then the displayed data set must have been previously deleted by another command. When a new data set name is provided, then the displayed data set can be used as a model for allocation attributes.

The name of the new data set must be entered with the **AL** line command, as shown in Figure 7-7 on page 152.

```
ISRUDSLO Data Sets Matching SOARES                                  Row 1 of 8
Command ===>                                                  Scroll ===> CSR

Command - Enter "/" to select action                Message          Volume
------------------------------------------------------------------------------
         SOARES                                                      *ALIAS
         SOARES.BRODCAST                                             BH5ST3
         SOARES.EXEC                                                 BH5ST1
         SOARES.HFS                                                  *VSAM*
         SOARES.HFS.DATA                                             BH5OE1
         SOARES.ISPO6555.SC74.SPFLOG1.LIST                           BH5ST6
al soares.newds JCL                                                  BH5ST2
         SOARES.SC74.ISPF42.ISPPROF                                  BH5ST5
***************************** End of Data Set list ***************************
```

Figure 7-7 Using the AL line command against an existing data set

When the **AL** line command is entered against an existing data set, a panel is displayed offering the following options; see Figure 7-8:

- Create the data set using the attributes of the existing data set
- Specify the attributes of the data set on the Allocate New Data Set panel

```
ISRMCALL             Allocate Target Data Set
Command ===>
                                                          More:     +
Specified data set SOARES.NEWDS
does not exist.
If you wish to allocate this data set, select one of the options
below.

Allocation Options:
 1_ 1. Allocate using the attributes of:
       SOARES.JCL
   2. Specify allocation attributes

   Use existing SMS attributes for option 1

Instructions:
  Press ENTER to allocate data set.
  Enter CANCEL or END to cancel allocation.
```

Figure 7-8 Allocate Target Data Set panel for AL command

New data set allocated

In this case, Figure 7-8 on page 152 shows that Option 1 was selected to allocate the new data set with the attributes of SOARES.JCL, which was the data set on the line where the **AL** line command was first entered. The new data set is automatically allocated using the attributes of the data set that the **AL** command was issued against.

Option 2 - Specify Allocation attributes

The Allocate New Data Set panel, shown in Figure 7-9 on page 153, is only displayed if you select Option 2 on the Allocate Target Data Set panel.

```
  Menu  RefList  Utilities  Help
------------------------------------------------------------------------------
Command ===>
                                                                More:     +
Data Set Name  . . . : SOARES.NEWDS

Management class . . .               (Blank for default management class)
Storage class  . . . .               (Blank for default storage class)
 Volume serial . . . . BH5ST4        (Blank for system default volume) **
 Device type . . . . .               (Generic unit or device address) **
Data class . . . . . .               (Blank for default data class)
 Space units . . . . . BLOCK         (BLKS, TRKS, CYLS, KB, MB, BYTES
                                      or RECORDS)
 Average record unit                 (M, K, or U)
 Primary quantity  . . 16            (In above units)
 Secondary quantity    15            (In above units)
 Directory blocks  . . 40            (Zero for sequential data set) *
 Record format . . . . FB
 Record length . . . . 80
 Block size  . . . . . 27920
 Data set name type    BASIC         (LIBRARY, HFS, PDS, LARGE, BASIC, *
                                      EXTREQ, EXTPREF or blank)
 Extended Attributes                 (NO, OPT or blank)
 Expiration date . . .               (YY/MM/DD, YYYY/MM/DD
Enter "/" to select option            YY.DDD, YYYY.DDD in Julian form
   Allocate Multiple Volumes          DDDD for retention period in days
                                      or blank)
```

Figure 7-9 Panel displayed when Option 2 is selected

Using the AL command for a deleted data set

You do not need to provide a data set name if the **AL** command is issued against a data set that has just been deleted. The name of the deleted data set will be used for the new data set.

```
  Menu  Options  View  Utilities  Compilers  Help
------------------------------------------------------------------------------
DSLIST - Data Sets Matching SOARES                        Data set deleted
Command ===>                                              Scroll ===> CSR

Command - Enter "/" to select action                 Message          Volume
------------------------------------------------------------------------------
         SOARES                                                       *ALIAS
         SOARES.BRODCAST                                              BH5ST3
         SOARES.EXEC                                                  BH5ST1
         SOARES.HFS                                                   *VSAM*
         SOARES.HFS.DATA                                              BH5OE1
         SOARES.ISPO6609.SC74.SPFLOG1.LIST                            BH5ST6
al       SOARES.JCL                                  Deleted
         SOARES.SC74.ISPF42.ISPPROF                                   BH5ST5
***************************** End of Data Set list ****************************
```

Figure 7-10 Using the AL line command against a deleted data set

After you press Enter, the panel shown in Figure 7-11 on page 154 is displayed.

```
  Menu  RefList  Utilities  Help
------------------------------------------------------------------------------
                           Allocate New Data Set
Command ===>
                                                                    More:     +
Data Set Name  . . . : SOARES.JCL

Management class . . .                (Blank for default management class)
Storage class  . . . .                (Blank for default storage class)
 Volume serial . . . . BH5ST4         (Blank for system default volume) **
 Device type . . . . .                (Generic unit or device address) **
Data class . . . . . .                (Blank for default data class)
 Space units . . . . . BLOCK          (BLKS, TRKS, CYLS, KB, MB, BYTES
                                       or RECORDS)
 Average record unit                  (M, K, or U)
 Primary quantity  . . 16             (In above units)
 Secondary quantity    15             (In above units)
 Directory blocks  . . 40             (Zero for sequential data set) *
 Record format . . . . FB
 Record length . . . . 80
 Block size  . . . . . 27920
 Data set name type    BASIC          (LIBRARY, HFS, PDS, LARGE, BASIC, *
                                       EXTREQ, EXTPREF or blank)
 Extended Attributes                  (NO, OPT or blank)
 Expiration date . . .                (YY/MM/DD, YYYY/MM/DD
Enter "/" to select option             YY.DDD, YYYY.DDD in Julian form
   Allocate Multiple Volumes           DDDD for retention period in days
                                       or blank)
```

Figure 7-11 A deleted data set is allocated again

7.3 Display data set create jobname and stepname

IBM introduced EAV volumes in z/OS V1R10. When you work with EAV volumes, DFSMS stores additional information in the new format-9 DSCB. Starting with z/OS V1R13, now the job name and step name that created the data set will be stored in the format-9 DSCB. The appropriate ISPF Data Set Information display panels have been enhanced to show the creation job and step name when they are available in the format-9 DSCB, as shown in Figure 7-12 on page 155.

```
ISRUAASE                    Allocate New Data Set
Command ===>
                                                                More:     +
Data Set Name  . . . : LUTZ.EAV.R13

Management class . . .                (Blank for default management class)
Storage class  . . . .                (Blank for default storage class)
 Volume serial . . . . LK907F         (Blank for system default volume) **
 Device type . . . . .                (Generic unit or device address) **
Data class . . . . . .                (Blank for default data class)
 Space units . . . . . CYLINDER       (BLKS, TRKS, CYLS, KB, MB, BYTES
                                       or RECORDS)
 Average record unit                  (M, K, or U)
 Primary quantity  . . 70000          (In above units)
 Secondary quantity    100            (In above units)
 Directory blocks  . . 0              (Zero for sequential data set) *
 Record format . . . . FB
 Record length . . . . 80
 Block size  . . . . . 8000
 Data set name type    LARGE          (LIBRARY, HFS, PDS, LARGE, BASIC, *
                                       EXTREQ, EXTPREF or blank)
 Extended Attributes                  (NO, OPT or blank)
 Expiration date . . .                (YY/MM/DD, YYYY/MM/DD
Enter "/" to select option             YY.DDD, YYYY.DDD in Julian form
   Allocate Multiple Volumes           DDDD for retention period in days
                                       or blank)
```

Figure 7-12 ISPF allocate dialog

After allocating the data set shown in Figure 7-12 on page 155, DFSMS creates a DSCB-9. In Figure 7-13 on page 156, from the DSCB-9, you can see who created the data set.

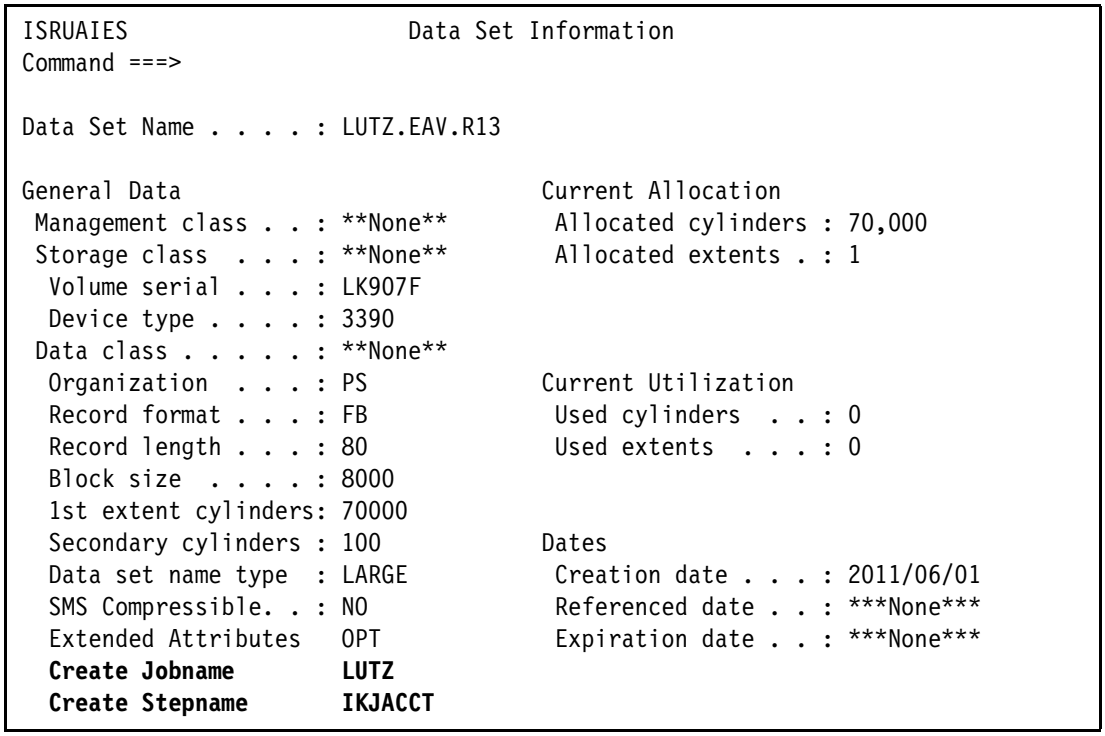


Figure 7-13 New information for large data sets

The ISPF services DSINFO and LMDLIST have also been enhanced to return these names in dialog variables.

DSINFO service

The DSINFO service returns the creation job and step names in the following variables:

ZDSCJOBN Creation job name

ZDSCSTPN Creation step name

The LMDLIST service returns the creation job and step names in the following variables:

ZDLCJOBN Creation job name

ZDLCSTPN Creation step name

These new services allow an ISPF dialog writer to have easy access to the creation job and step names for a data set, because the DSINFO and LMDLIST services are enhanced to return these names in ISPF dialog variables.

Figure 7-14 on page 157 shows a REXX example of how to obtain this new information.

```
/*REXX=============================================================*/
/*                                                                 */
/* title  : list data set EAV informations                         */
/*                                                                 */
/* author : ITSO                                                   */
/*                                                                 */
/* purpose: for use in Redbook SG247946 z/OS 1.13 Implementation */
/*                                                                 */
/*=================================================================*/
 dsn = '''LUTZ.EAV.R13'''
 address ISPEXEC "DSINFO DATASET("dsn")"

 if rc=0
  then do
   say 'EAV informations about data set :' dsn
   say 'dataset primary allocation   :' strip(ZDS1EX) ZDSSPC
   say 'dataset extended attributes  :' ZDSEATR
   say 'dataset created by jobname   :' ZDSCJOBN
   say 'dataset created by stepname  :' ZDSCSTPN
  end
  else say 'DSINFO failed !'
exit
```

Figure 7-14 Sample DSINFO call

After you run the REXX procedure against an EAV data set, you will see output similar to Figure 7-15 on page 157.

```
EAV informations about data set : 'LUTZ.EAV.R13'
dataset primary allocation   : 70000 CYLINDER
dataset extended attributes  : OPT
dataset created by jobname   : LUTZ
dataset created by stepname  : IKJACCT
```

Figure 7-15 Sample output from DSINFO

7.4 Editor support for line command macros

Changes have been made to the ISPF Table Utility to help users define EDIT line command tables. User-defined line commands and the edit macros they invoke are defined in an ISPF table.The ISPF editor reads the ISPF table to obtain information about the user-defined line commands and associated macros.

The ISPF Table Utility provides an option that helps in defining a user line command table. Users specify the name of their line command table on the edit and view entry panels. A dialog calling the EDIT or VIEW services can pass the name of a line command table. The ISPF Table Utility (ISPF Option 3.16) is modified to assist with defining the user line command table, as shown in Figure 7-16 on page 158.

To use this enhancement, select the option **Table is an EDIT line command table** on the ISPF Table Utility entry panel; see Figure 7-16 on page 158.

```
  Menu  RefList  Utilities  Options  Help
--------------------------------------------------------------------------------
ISRUTBP0                      ISPF Table Utility
Option ===> E_____________________________________________________________________

   blank Display table list                  E Edit table
       B Browse table                        I Import table data

Enter one of the parameters below:
   Table Data Set . . _____________________________________________
   or Table DD  . . . ISPTLIB   (Default is ISPTLIB)

   Table Name . . . . ROGERS    (Blank or pattern for table selection list)

   Import Data Set

   Enter "/" to select option
   __ Open table in SHARE mode
   /_ Table is an EDIT line command table
```

Figure 7-16 ISPF Table Utility

The ISPF EDIT Line Command Table, shown in Figure 7-17 on page 158, displays table columns where you define specific commands to a user edit line command table. Enter the user commands you want, then enter the MACRO name and N and Y for each command, as shown. The figure shows the rows for the user edit line command table ROGERS. The table contains rows for line commands CE, RV, LEF, and RIT, which were entered. Each of these line commands is processed by the edit macro POSLINE.

```
  Options  Help
--------------------------------------------------------------------------------
ISRUTBP3             ISPF EDIT Line Command Table  ROGERS      Row 1 to 4 of 4
Command ===> _____________________________________________ Scroll ===> PAGE
                                                            Shift ===> PAGE
    User     MACRO    Program  Block    Multi    Dest
    Command           Macro    format   line     Used
    ----+--- ----+--- ----+--- ----+--- ----+--- ----+---
 __ CE       POSLINE  N        Y        Y        Y
 __ RV       POSLINE  N        Y        Y        Y
 __ LEF      POSLINE  N        Y        Y        Y
 __ RIT      POSLINE  N        Y        Y        Y
****************************** Bottom of data ********************************
```

Figure 7-17 Enter the User Commands - (four commands shown)

Where:

CE	Center text on a line.
RV	Reverse text on a line.
LEF	Move text all the way left.
RIT	Move text all the way right.

When you press Enter, Figure 7-18 will display. This panel lists the data sets allocated to the table DD specified on the table utility entry panel. Enter **S** in the selection field for the data set

in which you want the updated table to be saved. If you press Enter without selecting a data set, the table update is canceled.

```
 Options  Help
--------------------------------------------------------------------------------
ISRUTBP9              Table Output Data Set Selection       Row 1 to 10 of 10
Command ===> _____________________________________________ Scroll ===> PAGE

Table ROGERS was opened in WRITE mode. No table data set was originally
specified, only a table DD. Since there was more than one table data set
allocated to this DD, please select which data set should receive the updated
table. Use END or CANCEL to return without saving the table.


S  Table Data Set
-  ---------------------------------------------
S  ROGERS.SC74.ISPF42.ISPPROF
   ISP.SISPTENU
   SYS1.SC75.ISPTLIB
   SYS1.SBPXTENU
   SYS1.DGTTLIB
   SYS1.SBLSTBLO
   CSF.SCSFTLIB
   ISF.SISFTLIB
   EOY.SEOYTENU
   GIM.SGIMTENU
****************************** Bottom of data ********************************
```

Figure 7-18 Select the data set to save your updated table

The Line Command Table field on the edit entry panel shown in Figure 7-19 on page 160 gives you the opportunity to define a set of user line commands that you can use during the edit session. The table you specify can be generated using the ISPF table editor.

This table contains the line commands that you want to have available, and also associates each line command with an edit macro that will be run if the line command is entered during the edit session.

```
Menu  RefList  RefMode  Utilities  Workstation  Help
--------------------------------------------------------------------------------
ISREDM01                        Edit Entry Panel
Command ===>  ____________________________________________________________

ISPF Library:
   Project . . . ROGERS
   Group . . . . JCL      . . . ________  . . . ________ . . . ________
   Type  . . . . VERS5
   Member  . . .                 (Blank or pattern for member selection list)

Other Partitioned, Sequential or VSAM Data Set, or z/OS UNIX file:
   Name . . . . .                                                          +
   Volume Serial . . ______    (If not cataloged)

Workstation File:
   File Name  . . ____________________________________________________________
                                       Options
Initial Macro  . . . . __________          Confirm Cancel/Move/Replace
Profile Name . . . . . __________          Mixed Mode
Format Name  . . . . . __________          Edit on Workstation
Data Set Password  . .                     Preserve VB record length
Record Length  . . . . _______             Edit ASCII data
Line Command Table . . ROGERS
```

Figure 7-19 ISPF Option 2 showing the new Line Command Table

LMAC tool: An unsupported LMAC tool that has been used by ISPF users for many years also provides the preceding capability. However, LMAC requires changes when the editor adds new line commands. User-written line command macros are now supported and this integration of support into the editor provides a more useful solution than working with LMAC.

A new line command table input field is added to the edit and view entry panels of ISPF. When the line command table is specified, the line commands specified in the table are able to be used during the edit session.

Line command examples

Figure 7-20 on page 160 shows a data set to use as examples of the new line commands.

```
 File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
--------------------------------------------------------------------------------
ISREDDE2   ROGERS.JCL.VERS5(DATA) - 01.00               Columns 00001 00072
Command ===>                                              Scroll ===> HALF
****** ***************************** Top of Data ******************************
000001 Move data to the center
000002 Reverse text on a line
000003             Move text all the way left
000004 Move text all the way right
****** **************************** Bottom of Data ****************************
```

Figure 7-20 Example data set

Figure 7-21 on page 161 displays examples of each of the four line commands.

```
CE0001 Move data to the center

Command ===>                                                Scroll ===> HALF
****** ***************************** Top of Data ******************************
000001                            Move data to the center
000002 Reverse text on a line
000003             Move text all the way left
000004 Move text all the way right
****** **************************** Bottom of Data ***************************

RV0002 Reverse text on a line

Command ===>                                                Scroll ===> HALF
 ****** ***************************** Top of Data ******************************
000001                            Move data to the center
000002                                                    enil a no txet esre
000003             Move text all the way left
000004 Move text all the way right
****** **************************** Bottom of Data ***************************

LEF0003             Move text all the way left

****** ***************************** Top of Data ******************************
000001                            Move data to the center
000002                                                    enil a no txet esre
000003 Move text all the way left
000004 Move text all the way right
****** **************************** Bottom of Data ***************************

RIT004 Move text all the way right

Command ===>                                                Scroll ===> HALF
****** ***************************** Top of Data ******************************
000001                            Move data to the center
000002                                                    enil a no txet esre
000003 Move text all the way left
000004                                                    Move text all the w
****** **************************** Bottom of Data ***************************
```

Figure 7-21 Examples of the new line commands

7.4.1 Installation considerations

Editor support for user line commands coexists with the LMAC tool. Edit macros written to process line commands with the LMAC tool will also work with the new user line command support. The editor will give LMAC precedence to process a user-written line command. This should simplify the process of existing LMAC users transitioning to the new user edit line macro support.

Note: *z/OS ISPF Services Guide*, SC34-4819, describes the new LINECMDS parameter for the EDIT and VIEW services.

z/OS ISPF Edit and Edit Macros, SC34-4820, describes how to define and invoke user edit line commands.

7.5 INFO command to display PDS member extended statistics

In z/OS V1R11, the ISPF statistics maintained for PDS members were extended to support line count values greater than 65535. However, no function was provided to display these values. The ISPF enhanced member list now supports the new INFO (**i**) line command to display the panel ISRUDSI, which shows the ISPF statistics for a member.

The **i** line command can be used against any member in a partitioned data set. It is not restricted to members that have extended line count statistics defined.

View extended line count values

To view extended line count values, select ISPF Option 3.4 and then enter b for a PDS data set. This enhanced member list is displayed using either **M**, **B**, **E**, or **V** line commands in the Data Set List display shown in Figure 7-22 on page 162.

```
ISRUDSL0 Data Sets Matching ROGERS                          0 Members processed
Command ===> _____________________________________________    Scroll ===> PAGE

Command - Enter "/" to select action                        Tracks %Used   XT
------------------------------------------------------------------------------
         ROGERS.OPERLOG.TXT                                     15   80     1
b        ROGERS.PDS1                                            90   37     1
         ROGERS.PRIVATE.JCL                                     75    1     1
         ROGERS.REPORT.JCL                                      10   20     1
```

Figure 7-22 ISPF Option 3.4 display

Member BIG1 is displayed; see Figure 7-23 on page 162.

```
ISRUDSM BROWSE    ROGERS.PDS1                                  Row 00001 of 00001
Command ===>                                                    Scroll ===> PAGE
           Name     Prompt       Size   Created          Changed          ID
I________  BIG1                 65535  2011/06/21  2011/06/21 13:15:58  RC74
          **End**
```

Figure 7-23 Member BIG1

When you enter the **i** line command, Figure 7-24 on page 163 is displayed. The ISPF enhanced member list now supports using the new **i** line command to display the panel ISRUDSI, which shows the ISPF statistics for a member.

```
ISRUDSI BROWSE    ROGERS.PDS1
Command ===> ____________________________________________________________


                                          Directory flag byte
   Member name  . . . . . . . . : BIG1     Bit 0  : 0  SCLM
   Concatenation number . . . . : 1        Bit 1  : 0
   Version . Modification . . . : 01.00    Bit 2  : 0  Ext Stats
   Create Date  . . . . . . . . : 2011/06/21 Bit 3  : 0
   Modification Date  . . . . . : 2011/06/21 Bit 4  : 0
   Modification Time  . . . . . : 13:15:58  Bit 5  : 0
   Userid that Created/Modified : RC74      Bit 6  : 0
                                          Bit 7  : 0

   Line counts : max values are 65536     Extended line counts
   Current  : 65535                       Current  :
   Initial  : 65535                       Initial  :
   Modified : 0                           Modified :
```

Figure 7-24 New panel display with z/OS V1R13

An alternate option

If you enter a forward slash (/) on the command line in Figure 7-22 on page 162, then Figure 7-25 is displayed. If you select Option 15 from this panel, Figure 7-24 is displayed.

```
  Menu  Functions  Confirm  Utilities  Help
s
I  ISRCMLEP    Action for Member BIG1                  Row 00001 of 00001
C                                                        Scroll ===> PAGE
   Member Action                                      Changed          ID
/  15  1.  Edit              8.  Copy                 06/21 13:15:58  RC74
       2.  View              9.  Reset
       3.  Browse            10. Open   **None**
       4.  Delete            11. Submit
       5.  Rename            12. TSO Cmd
       6.  Print             13. WS Cmd
       7.  Move              14. Select BROWSE
                             15. Info

   Prompt Action . .          (For prompt field)

   Select a choice and press ENTER to continue
```

Figure 7-25 Action for Member BIG1 display

As mentioned, the **Info** command displays the same information as the member list. When the Extended PDS statistics function has been enabled, the extended line counts fields contain data.

8

DFSMSrmm enhancements

DFSMSrmm is a z/OS feature. In your enterprise, you probably store and manage removable media in several types of media libraries. For example, in addition to your traditional tape library, which is a room with tapes, shelves, and drives, you might have several automated, virtual, and manual tape libraries. You probably also have both onsite libraries and offsite storage locations, also known as “vaults” or “stores.”

With DFSMSrmm, you can manage your removable media as one enterprise-wide library across systems and sysplexes. DFSMSrmm manages your installation's tape volumes and the data sets on those volumes. DFSMSrmm also manages the shelves where volumes reside in all locations except in automated tape libraries.

DFSMSrmm manages all tape media, such as cartridge system tapes and 3420 reels, and other removable media that you define to it. For example, DFSMSrmm can record the shelf location for optical disks and track their vital record status. However, it does not manage the objects on optical disks.

In this chapter, the new DFSMSrmm features with z/OS V1R13 are described:

- Retention date in the volume and data set search results
- SEARCHDATASET parameter extensions
- TVEXTPURGE extra days
- Expiry data set by information
- Exclude data set from VRSEL
- Retention method
- Data set attribute copy function
- Support RETPD(93000)
- Selective volume movement
- Last change detail
- VRS last reference date
- Display navigation enhancements

8.1 Retention methods for tapes

Among the important decisions to be made when using DFSMSrmm is how to retain tape data sets and for how long. You might want to retain a given data set for a specific period of time after it is created; retain it based on an event (for example, while the data set is cataloged); or retain it permanently.

With z/OS V1R13, DFSMSrmm offers two retention methods, EXPDT and VRSEL, for retention of your volumes and data sets on your volumes. For each volume set, you can specify whether the volumes and data sets in that volume set are to be managed by the expiration date or by VRSEL retention vital record specification (VRS) policies.

8.1.1 Specifying a retention method

The EDGRMMxx parmlib member has a new operand to set the system-wide retention method for new tape volume sets created during Open/Close/End-of-Volume (O/C/EOV) processing, and for tape volumes added to the DFSMSrmm CDS.

```
RETENTIONMETHOD(EXPDT | VRSEL)
```

Note: RETENTIONMETHOD can be abbreviated as RM.

Every volume has a retention method, either EXPDT or VRSEL. The default retention method specified in parmlib is used, unless one is assigned by the EDG_EXIT100 installation exit or by the ADDVOLUME command.

When data is created the storage administrator can select an appropriate retention method for a volume set. The basic choice is either VRSEL or EXPDT, as explained here:

VRSEL The VRSEL retention method can retain a volume beyond its expiration date by using a vital record specification to define a retention policy. The VRSEL retention method can also be used for setting a movement policy. Volumes and data sets managed by this retention method are subject to frequent VRSEL processing. In every run of VRSEL processing, the matching of your data sets and volumes to the vital record policies is done again, so the retention and movement management can change from one inventory management run to another.

EXPDT The EXPDT retention method is based on the expiration date and avoids VRSEL processing. As a result, the retention information can be known when a tape data set is created. You can also manually override an expiration date to release a volume before the original expiration date is reached. The movement of the volumes must be managed manually.

Expiration date

An important parameter for both retention methods is the expiration date of the data set and the volume. The data set expiration date or retention period is determined when a new tape data set is created. The expiration date or retention period can be specified at multiple levels:

1. The default retention period RETPD specified in the DFSMSrmm parmlib member
2. The EXPDT or RETPD in the DFSMS data class if the data set is associated with a DFSMS data class
3. The JCL DD statement, using the EXPDT or RETPD keywords
4. The DFSMSrmm installation exit EDG_EXIT100

Note: When you change the expiration date for a data set record representing one part of a multivolume data set on volumes managed by the EXPDT retention method, DFSMSrmm updates the expiration date and time for all the data set records for the data set.

Retention date

z/OS V1R13 DFSMSrmm provides enhancements to make tape management easier and to improve administrator productivity, as described here:

- A retention date in the volume

  With this enhancement, the retention date is displayed instead of the expiration date in the search results list, if the volume or data set is VRS-retained.

- A data set search result

  When a resource is retained by VRS, the search results list for volumes or data sets might show retained resources with an expiration date that has already passed. The storage administrator can more easily determine from the search results list why the volume is retained, without viewing the volume and data set details.

Note: If you not specify a default retention method, the system uses the VRSEL retention method as the default. This new function displays the retention date instead of the expiration date in the search results list, if the volume or data set is VRS-retained. Again, the storage administrator can more easily determine from the search results list why the volume is retained, without viewing the volume and data set details.

Displaying system options

To display system options, from ISPF enter the RMMISPF EXEC to enter the dialog. If you then select Option 4.8.2 to display the system options, you can see the retention method that is currently active (Figure 8-1).

```
EDGPC200              DFSMSrmm System Options Display
Command ===>

                                                                      More:     +
Parmlib suffix . : 13

Operating mode . : WARNING

Data sets:
  Control  . . . : RMM.CONTROL.DSET
  Journal  . . . : RMM.JOURNAL.DSET
  CDS id . . . . : WTSCPLX1                Journal threshold . : 75 %
  Catalog SYSID  : NOTSET                  Journal transaction : NO

                                           SMF:
Retention method : EXPDT                     System id . . . . : SC74
                                             Audit . . . . . . : 0
Retention period:                            Security  . . . . : 0
  Default  . . . : 0
  Maximum  . . . : NOLIMIT                 Report options:
  Catalog  . . . : 0      hours              Lines per page  . : 54
                                             Date format . . . : JULIAN
PDA: ON
  Block count  . : 255                     Auto-start procedures:
  Block size . . : 12                        Scratch procedure : EDGXPROC
  Log  . . . . . : ON                        Backup procedure  :
```

Figure 8-1 System Options display - Option 4.8.2

8.1.2 Retaining source data sets managed by the EXPDT retention method

The tape copy application must decide how to manage the retention of the source data set and volume. To manage the retention of the source data set and volume, the tape copy application must issue volume and data set subcommands to set the required retention policy. The application can decide whether it is the specific data set, the volume, or the entire volume set that is no longer required. Retention of data sets is subject to retention of the volume. Only when the volume expiration date is reached will the volume be set pending release.

Your choices include the following:

- Keep the current retention

  In this case, the data sets and volumes continue to be retained as they were prior to the copy. You can later decide when you want to release the volumes. Because the EXPDT retention method does not support cycle retention, no other data set can affect the retention of the source data.

- Release the volume or volume set

  In this case, when copying all the files on a volume, you can decide to manage the retention and expiration at the volume level. If you no longer require the data, you can release the volume manually, as follows:

  ```
  RMM DV volser RELEASE
  ```

To release the volume chain, you can use the DFSMSrmm SEARCH with CLIST capability:

```
RMM SV VOLUME(volser) CHAIN LIMIT(*) CLIST('RMM DV ',' RELEASE')
EXEC RMM.EXEC
```

- Retain the volume for a defined period

  In this case, if you want to retain the volume for a defined period, you can update the existing volume expiration date using a retention period:

  ```
  RMM CV volser RETPD(days)
  ```

  You can change the retention for an entire volume set with this command:

  ```
  RMM SV VOLUME(volser) CHAIN LIMIT(*) CLIST('RMM CV ',' RETPD(days)')
  EXEC RMM.EXEC
  ```

- Set a new retention policy for the data set

  Data set on volumes managed by the EXPDT method can be expired after a specified period of time by including the RETPD of the COPYFROM suboperand. The source data set records are updated to enable expiration to occur. The data set EXPDT is set to the specified date. When you specify the COPYFROM RETPD operand to set the source data set expiration date, it will also update the volume expiration date if it is a higher value than the current volume expiration date.

8.1.3 Subcommands for RETENTIONMETHOD parameters

The subcommands shown in Figure 8-2 can be used to set the retention method by specifying one of the following operands:

```
RETENTIONMETHOD(EXPDT | VRSEL)
```

```
>--AV---+---------------------------------------+-->
        |                     .-VRSEL--------.  |
        '---RETENTIONMETHOD---(-+--------------+-)-'
          '-RM--------------'   '-EXPDT--------'
ADDOVOLUME subcommand - Use this operand to set the retention method for a tape
volume set. Specify this operand for the first volume in a multi-volume
sequence. All other volumes added o the set assume the same retention method.

>--CV---+---------------------------------------+-->
        |                     .-VRSEL--------.  |
        '---RETENTIONMETHOD---(-+--------------+-)-'
          '-RM--------------'   '-EXPDT--------'
CHANGEVOLUME subcommand - Use this operand fo any volume to set the retention
method for a tape volume set. Specify this operand for the first volume in a
multi-volume sequence. All other volumes added o the set assume the same
retention metho

>--SV---+---------------------------------------+-->
        |                     .-VRSEL--------.  |
        '---RETENTIONMETHOD---(-+--------------+-)-'
          '-RM--------------'   '-EXPDT--------'
SEARCHVOLUME subcommand - Use this operand to restrict the returned volumes
based on the retention method.
```

Figure 8-2 Subcommands for RETENTIONMETHOD parameters

Where:

AV The ADDVOLUME subcommand can be used to set the retention method for a tape volume set. Specify this operand for the first volume in a multivolume sequence. All other volumes added to the set assume the same retention method. After a retention method is defined for a non-scratch volume, it is not overridden to the system-wide default during OPEN output processing, but it can be changed by installation exit EDG_EXIT100. Volumes in a set always assume the retention method of the first volume in the set.

Specify EXPDT to set the retention method for a tape volume set to be based on EXPDT. Data sets and volumes managed by this retention method are never processed by VRSEL inventory management.

Specify VRSEL to set the retention method for a tape volume set to be VRSEL. This option enables DFSMSrmm inventory management to attempt to match data sets and volumes to vital record specifications, and if a match is found, to determine if the data set or volumes are to retained by VRS.

CV The CHANGEVOLUME subcommand changes existing DFSMShsm-managed volumes. You can use the CHANGEVOLUME subcommand with RETENTIONMETHOD operand on single volumes and on the first volumes in a multivolume set. DFSMSrmm will apply the specified retention method to all volumes in the multivolume set. All EXPDT retention method-managed volumes must have a valid expiration date. Multivolume sets that use the EXPDT retention method are managed at the volume level, which means that the desired EXPDT must be specified for each volume in a multivolume set.

SV RETENTIONMETHOD(EXPDT | VRSEL) specifies a list limited to volumes that have a specified retention method.

Specify EXPDT to select volumes with the EXPDT retention method.

Specify VRSEL to select volumes with the VRSEL retention method.

Note: When you change the retention method from VRSEL to EXPDT and the volume was retained as a vital record, the vital record attribute is reset and the retention date set to the current date. If the new retention method is the EXPDT retention method, the data set records will be excluded from VRSEL and the vital record attribute and retention date similarly changed.

In addition, DFSMSrmm performs additional processing for all multivolume data sets in the volume set. The maximum expiration date and time for each multivolume data set is determined from the data set records, and used to equalize the expiration for the multivolume data set.

8.1.4 Benefiting from the EXPDT retention method

DFSMSrmm provides the programming interface EDGTVEXT that is used from DFSMShsm, OAM, or any other APF-authorized program that needs to obtain the same services as the DFSMShsm ARCTVEXT exit.

If you have a product with similar requirements for releasing tapes as DFSMShsm, you can use the EDGTVEXT program interface. You can also use the EDGDFHSM interface. The difference between EDGTVEXT and EDGDFHSM is that EDGTVEXT accepts the ARCTVEXT parameter list, whereas EDGDFHSM accepts only a single volume at a time in the parameter list.

Note: You can use DFSMSrmm parmlib option TVEXTPURGE to control the processing that EDGTVEXT performs. You can release volumes or set the volume expiration date either to the current date or based on a number of days from the current date. The effect of setting the expiration date depends on the retention method (EXPDT or VRSEL) already specified for the volume.

Changing the retention method for existing VRS retained data

For single volumes and first volumes in each multivolume set, issue the following TSO RMM subcommand:

```
RMM CV volser RM(EXPDT) EXPDT(retdate)
```

Because the EXPDT retention method-managed multivolume sets are managed on a volume level, you must also set the desired expiration date for all other volumes in each multivolume set by using the following subcommand:

```
RMM CV volser EXPDT(retdate)
```

Note: Do not specify Readapting) because DFSMSrmm subcommand processing applies this to the current date and not the creation date. Instead use the volume retention date, which is the date calculated by VRSEL.

For new data sets added to a volume changed from the VRSEL to the EXPDT retention method, consider the following points:

- A RETPD or EXPDT must be specified at the time of allocation, or the EDG_EXIT100 installation exit must assign a date.
- If these are not done, the default retention period will be applied.

Using retention methods

With z/OS V1R13, data sets on volumes managed by the VRSEL retention method are unchanged.

With z/OS V1R13, all files of a multi volume data set on a volume set managed by RM(EXPDT) have the same expiration date and time, as shown in Figure 8-3.

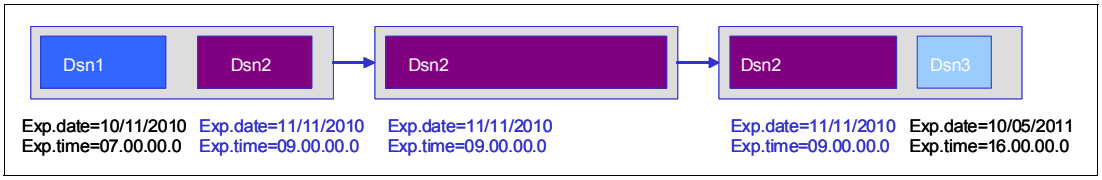


Figure 8-3 EXPDT retention method

RMM maintains a consistent data set expiration date and time for data sets on volumes managed by the EXPDT retention method for the following processing phases:

- During O/C/EOV processing, in which the expiration time is rounded up to a whole hour to help avoid I/O to the RMM CDS

  This is done for the first data set record for a data set, and only repeated if data set creation continues onto a new volume and the then-current time exceeds the rounded-up value.
- When you specify EXPDT/RETPD on an ADDDATASET or CHANGEDATASET subcommand

- During CHANGEVOLUME PREVVOL processing
- When the retention method of the volume set is changed from VRSEL to EXPDT

During conversion, regardless of retention method, the expiration date of the first data set record of a multivolume data set is propagated to all other data set records of the data set.

8.1.5 Installation considerations and coexistence

Use the DFSMSrmm EDGRMMxx parmlib member to specify the installation options for DFSMSrmm. If you are using expiration dates to manage tapes, consider the values you specify for the parmlib OPTION command MAXRETPD, RETENTIONMETHOD, the TVEXTPURGE operands, and the VLPOOL command EXPDTCHECK operand.

For z/OS V1R13 DFSMSrmm, consider the following points:

- The RETENTIONMETHOD operand specifies the retention method (EXPDT or VRSEL) that DFSMSrmm is to use for DFSMShsm volumes.
- You can use the EXPDT retention method to avoid processing DFSMShsm volumes on each inventory management VRSEL run. This retention method requires that you use expiration date protection for DFSMShsm tape volumes, namely EXPDT=99365.
- You do not need to run VRSEL processing unless volumes are defined with the VRSEL retention method. Only EXPROC processing is required to handle expiration of all volumes managed by the EXPDT retention method.
- EXPROC processing provides a summary of volumes by retention method.
- The expiration date of volumes is set during OPEN processing, so for volumes managed by the EXPDT retention method, there are no special considerations for open data sets; they are managed based on the volume EXPDT.
- Volumes containing data sets closed by ABEND processing or which are deleted are handled as though no special ABEND/DELETED VRS had been defined. That is, all retention is based only on the volume EXPDT.
- Volumes managed by the EXPDT retention method are included only in the EXPDTDROP limit. VRSRETAIN and VRSDROP limits apply only to volumes managed by the VRSEL retention method.

Setting DFSMSrmm options when using DFSMShsm

When DFSMShsm sets 99365 as the expiration date to manage its tapes, 99365 means permanent retention or to never expire. If you choose to use DFSMShsm expiration date protected tape volumes, DFSMShsm sets the date 99365 to prevent DFSMSrmm from considering the volumes for release at any time. You must specify the MAXRETPD(NOLIMIT) operand to ensure that DFSMSrmm honors the 99365 date.

You may also use the DFSMSrmm parmlib MAXRETPD operand value to reduce the expiration date for all volumes including DFSMShsm volumes. If you want to reduce the 99365 permanent retention expiration date, specify the MAXRETPD with a value between 0 and 9999 days.

Attention: To avoid the overhead of VRSEL processing for tape volumes managed by DFSMShsm (and similar applications), use the EXPDT retention method.

However, if you require DFSMSrmm to manage the movement of DFSMShsm volumes, use DFSMSrmm vital record specifications instead of using the 99365 permanent retention date to retain DFSMShsm volumes.

Volume and data set displays

The output of the ISPF panel VOLUME SEARCH results is shown in Figure 8-4

```
EDGPT020           DFSMSrmm Volumes (Page 1 of 2)            Row 1 to 20 of 45
Command ===> _____________________________________________  Scroll ===> PAGE

Enter HELP or PF1 for the list of available line commands
Use the RIGHT command to view other data columns
   Volume          Assigned   Expir./    S                  Dest-    Tr- Data
S  serial Owner    date       Retn. date R Status  Location ination  ans sets
-- ------ -------- ---------- ---------- - ------- -------- -------- --- -----
__ THM000 STC      2010/123   CYCL/99999   VRS     ITSOTL1           N   60
__ THM001 STC      2010/124   PERMANENT    VRS     ITSOTL1           N   60
__ THM002 STC      2010/125   CYCL/99999   VRS     ITSOTL1           N   60
__ THM003 STC      2010/126   CYCL/99999   VRS     ITSOTL1           N   60
__ THM004 STC      2010/126   CYCL/99999   VRS     ITSOTL1           N   60
__ THM006 STC      2010/127   CYCL/99999   VRS     ITSOTL1           N   15
__ THM010 STC      2010/127   CYCL/99999   VRS     ITSOTL1           N   60
__ THM013 STC      2010/127   CYCL/99999   VRS     ITSOTL1           N   27
__ THM016 STC      2010/128   CYCL/99999   VRS     ITSOTL1           N   60
__ THM017 STC      2009/069   CYCL/99999   VRS     ITSOTL1           N   1
__ THM020 STC      2007/293   PERMANENT    VRS     ITSOTL1           N   39
__ THM024 STC      2010/128   CYCL/99999   VRS     ITSOTL1           N   60
```

Figure 8-4 Volume search results

The output of ISPF panel DATA SET SEARCH results is shown in Figure 8-5.

```
EDGPD030        DFSMSrmm Data Sets (Page 2 of 2)         Row 37 to 54 of 1,473
Command ===>                                               Scroll ===> PAGE


Enter HELP or PF1 for the list of available line commands
Use the LEFT command to view other data columns
                                                Create     Expir./    V
S  Data set name                                date       Retn. date R
-- ---------------------------------------- ---------- ---------- -
__ TOTHSM.DMP.SYSPLEX.VTOTMIA.D10125.T111800    2010/125   CYCL/99999 Y
__ TOTHSM.DMP.SYSPLEX.VTOTMIB.D10118.T231800    2010/118   CYCL/99999 Y
__ TOTHSM.DMP.SYSPLEX.VTOTMIB.D10125.T291800    2010/125   PERMANENT  Y
__ TOTHSM.DMP.SYSPLEX.VTOTMIC.D10118.T511800    2010/118   CYCL/99999 Y
__ TOTHSM.DMP.SYSPLEX.VTOTMIC.D10125.T561800    2010/125   CYCL/99999 Y
__ TOTHSM.DMP.SYSPLEX.VTOTMID.D10118.T261900    2010/118   CYCL/99999 Y
__ TOTHSM.DMP.SYSPLEX.VTOTMID.D10125.T311900    2010/125   CYCL/99999 Y
__ TOTHSM.DMP.SYSPLEX.VTOTMIE.D10118.T581900    2010/118   PERMANENT  Y
__ TOTHSM.DMP.SYSPLEX.VTOTMIE.D10125.TO42000    2010/125   CYCL/99999 Y
__ TOTHSM.DMP.SYSPLEX.VTOTMIG.D10118.T591900    2010/118   CYCL/99999 Y
__ TOTHSM.DMP.SYSPLEX.VTOTMIG.D10125.TO52000    2010/125   CYCL/99999 Y
__ TOTHSM.DMP.SYSPLEX.VTOTMIH.D10118.T422000    2010/118   CYCL/99999 Y
```

Figure 8-5 Data set search results

Note: The SD or SV TSO subcommands will return the REXX variables EDG@RTDT, calendar date, and EDG@RTDJ, julian date, in any case.

Coexistence considerations

Sharing z/OS releases lower than z/OS V1R13 requires a PTF for coexistence APAR OA32984 to be installed before exploitation of new functions is attempted on z/OS V1R13.

Important considerations: For a volume level RETENTIONMETHOD, DFSMSrmm EXPROC processing only processes volumes with the VRSEL retention method.

All files of a multivolume data set on a volume set managed by RM(EXPDT) have the same expiration date and time. DFSMSrmm coexistence maintains the data set records in sync both while tape data sets are processed and when an RMM ADD or CHANGE subcommand is issued.

8.1.6 EDG_EXIT100 retention method support

You can use the EDG_EXIT100 installation exit to set the retention method to be used for a new tape data. When you create a new tape volume set, or rewrite an existing set from the first file, you can override the system default retention method.

With z/OS V1R13, every volume has a retention method, either EXPDT or VRSEL. The default retention method specified in parmlib is used, unless one is assigned by the EDG_EXIT100 installation exit or by the **ADDVOLUME** command.

The tape copy application can use the EDGPL100 macro to invoke the installation exit EDG_EXIT100 to copy the data set attributes from the source data set to the copy data set during OPEN processing. EDG_EXIT100 can notify DFSMSrmm that the data set being created is being copied from another.

During OPEN processing, the exit can identify the source data set from which DFSMSrmm will obtain all existing data set attributes, which will be used for the target data set. DFSMSrmm EOV processing ensures that the attributes are copied to all target data set records when the output data set becomes a multivolume data set.

Note: During the creation of the target data set, tape copy application programs can use the EDG_EXIT100 installation exit to copy all applicable attributes from the source data set to the target data set. Retention of the target volumes and data sets can be selected at the volume set level, and even switched between VRSEL and EXPDT retention methods.

8.2 SEARCHDATASET extensions

The SEARCHDATASET subcommand is used to create a list of data sets that match criteria you specify. You can specify a generic name using the full set of filter masks. Specify a fully-qualified name to restrict the search to data sets that match the name exactly. Specify the FILESEQ operand to restrict the search to data sets at a relative position on the volume. You can also use the JOBNAME operand to restrict the search to data sets that are created by a particular jobname.

With z/OS V1R13 DFSMSrmm, additional operands are added enabling more extensive searches including many specific date ranges (Figure 8-6). With this implementation, the SEARCHDATASET search is more efficient with a large number of data sets.

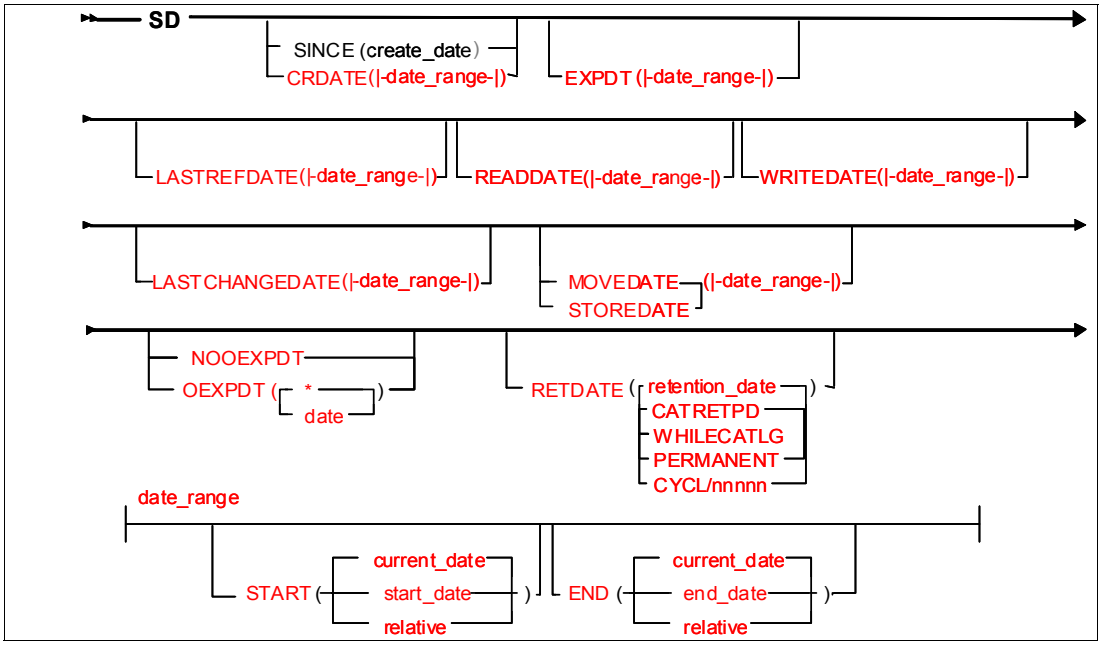


Figure 8-6 SEARCHDATASET subcommand parameters that are new with z/OS V1R13

The new operands and syntax changes with the SEARCHDATASET subcommand are listed here:

CATALOG This specifies to limit the search to data sets based on catalog status. Specify CATLG(UNKNOWN) to search for data sets that have not yet been cataloged. Specify CATLG(YES) to search for data sets that are currently cataloged. Specify CATLG(NO) to list only data sets that have been uncataloged. CATALOG can be abbreviated as CATLG.

CRDATE This lists the data sets with a creation date that matches the specified date criteria.

CRDATE is mutually exclusive with the SINCE operand. CRDATE may be specified as any of the following:

CRDATE - Only data sets whose creation date is the current date are listed.

CRDATE(START(start_date)) - Only data sets whose creation date is on or after the specified start date are listed, where start_date is either an absolute date or relative date.

CRDATE(END(end_date)) - Only data sets whose creation date is on or before the specified end date are listed, where end_date is either an absolute date or relative date. Note that because START defaults to the current date, the specified end date equal to or greater than the current date when START is omitted.

CRDATE(START(start_date)END(end_date)) - Only data sets with a creation date that is within the range delimited by the specified start and end dates are listed, where both start_date

and end_date are either an absolute date or relative date. The specified end date equal to or greater than the specified start date.

Note: Each of the start_date and end_date values can be absolute or relative dates.

DATACLASS (dataclass_name) - This specifies to limit the search to data sets with the specified data class name. A data class name is one-to-eight alphanumeric, national, or special characters. DATACLASS is mutually exclusive with NODATACLASS.

EXPDT (date_range) - This lists the data sets with a current expiration date that matches the specified date criteria. The date criteria can be specified using the START and END suboperands. See the description of the SEARCHDATASET CRDATE operand for an explanation of how to use the START and END suboperands and for examples of their use.

LASTCHANGEDATE (date_range) - This lists the data sets with a last changed date that matches the specified date criteria. The date criteria can be specified using the START and END suboperands. See the description of the SEARCHDATASET CRDATE operand for an explanation of how to use the START and END suboperands and for examples of their use.

LASTREFDATE (date_range) - This lists the data sets based on both the last read date and last write date, using the most recent of both dates. The most recent of the two values within the date range for a data set to be selected. The date criteria can be specified using the START and END suboperands. See the description of the SEARCHDATASET CRDATE operand for an explanation of how to use the START and END suboperands and for examples of their use.

MANAGEMENTCLASS (managementclass_name) - This specifies to limit the search to data sets with the specified management class name. A management class name is one-to-eight alphanumeric, national, or special characters. MANAGEMENTCLASS is mutually exclusive with NOMANAGEMENTCLASS.

NODATACLASS This specifies to limit the search to data sets that do not have a data class. NODATACLASS is mutually exclusive with DATACLASS.

NOMANAGEMENTCLASS This specifies to limit the search to data sets that do not have a management class. NOMANAGEMENTCLASS is mutually exclusive with MANAGEMENTCLASS.

NOOEXPDT This specifies to limit the search to data sets that do not have an original expiration date. NOOEXPDT is mutually exclusive with OEXPDT.

NOSTORAGECLASS This specifies to limit the search to data sets that do not have a storage class. NOSTORAGECLASS is mutually exclusive with STORAGECLASS.

NOSTORAGEGROUP This specifies to limit the search to data sets that do not have a storage group. NOSTORAGEGROUP is mutually exclusive with STORAGEGROUP.

OEXPDT (date) - This lists the data sets with an original expiration date that matches the specified date. If you use an asterisk (*), DFSMSrmm returns data set information for all data sets that have any original expiration date. OEXPDT is mutually exclusive with NOOEXPDT.

READDATE (date_range) - This lists the data sets with a last read date that matches the specified date criteria. The date criteria can be specified using the START and END suboperands. See the description of the SEARCHDATASET CRDATE operand for an explanation of how to use the START and END suboperands and for examples of their use.

RETDATE (retention_date) - This specifies that DFSMSrmm lists only data sets that will expire up to and including the specified date. Data sets on scratch volumes are excluded. You can specify a specific date as RETDATE(retention_date). You can also specify the DFSMSrmm special date formats such as CATRETPD, PERMANENT, WHILECATLG, or CYCL/nnnnn, where nnnnn is five numeric digits.

When you specify one of the special dates, DFSMSrmm lists only data sets that are VRS retained with that special retention date. When you specify the special cycles format date, CYCL/nnnnn, DFSMSrmm lists data sets that are VRS retained and have a cycles retention date and the same number or fewer cycles. For example, RETDATE(CYCL/00255) searches for all data sets with a retention date set to CYCL/00255 or lower, such as CYCL/00001.

For data sets retained by a VRS, DFSMSrmm uses the retention date. For data sets not retained by a vital record specification, DFSMSrmm uses the expiration date for the search. To obtain a list of data sets that have a permanent expiration date and that are not retained by vital record specifications, specify the expiration dates 1999/365 or 1999/366.

STORAGECLASS (storageclass_name) - This specifies to limit the search to data sets with the specified storage class name. A storage class name is one-to-eight alphanumeric, national, or special characters. STORAGECLASS is mutually exclusive with NOSTORAGECLASS.

STORAGEGROUP (storagegroup_name) - This specifies to limit the search to data sets with the specified storage group name. A storage group name is one-to-eight alphanumeric, national, or special characters. STORAGEGROUP is mutually exclusive with NOSTORAGEGROUP.

VRSELEXCLUDE (YES | NO) - This specifies to limit the search to data sets excluded (or not excluded) from VRSEL processing. Specify NO to search for data sets that are not excluded from VRSEL processing. Specify YES to search for data sets that are excluded from VRSEL processing. VRSELEXCLUDE can be abbreviated as VX.

WRITEDATE (date_range) - This lists the data sets with a last write date that matches the specified date criteria. The date criteria can be specified using the START and END suboperands. See the

description of the SEARCHDATASET CRDATE operand for an explanation of how to use the START and END suboperands and for examples of their use.

Note: Relative dates are n days(-n), n (months(-nm) or n years(-nY) back in time from the current date.

The new CRDATE operand is an alternate to the existing SINCE operand.

From the ISPF panels for DFSMSrmm, if you select Option 5.3.5, the panel shown in Figure 8-7 on page 178 displays the new options to the subcommand.

```
EDGPD010                  DFSMSrmm Data Set Search
Command ===>

Enter fully qualified or partial data set name and job name:

Data set name  . . . .
Job name . . . . . .            Specific or generic name

Enter optional parameters to qualify search
                                                                More:     +
Owner  . . . . . . . . *          Owner of volumes (Default is your userid)
Volume serial  . . . .          OR Volume serial
List entire set  . . .            YES, For all data sets in the
                                  multi-volume set, otherwise NO
Status . . . . . . . .            PRIVATE, SCRATCH, or blank for all
Retained by VRS  . . .            YES, NO, or blank for all
Excluded from VRSEL               YES, NO, or blank for all
Dates           Start          End       Date, date range or relative value
  Create  . . .            . .
  Expiration               . .
  Reference . .            . .
  Read  . . . .            . .
  Write . . . . -99999     . . -9M
  Changed . . .            . .
Physical file seq  . .            Relative position on the volume
Limit  . . . . . . . . *          Limit search to first nnnn data sets
Clist  . . . . . . . .            YES to create a data set, or NO, or blank
Program name . . . . .            Specific or generic name
Closed by Abend  . . .            YES, NO, or blank for all
Deleted  . . . . . .              YES, NO, or blank for all
BES key index  . . . .             CA BTE tape encryption key index
Original EXPDT . . . .            YES, NO, or a specific date YYYY/DDD
Retention  . . . . . .            Data set retained up to YYYY/DDD
Cataloged  . . . . . .            YES, NO or UNKNOWN
Force  . . . . . . . .            Data sets used with FORCE ( YES or NO )
Data class . . . . . .            Data class name or NO
Storage class  . . . .            Storage class name or NO
Management class . . .            Management class name or NO
Storage group  . . . .            Storage group name or NO
```

Figure 8-7 Search data set subcommand

SEARCHDATASET command examples

This section shows you examples of how to use the new options available with the SEARCHDATASET (SD) command.

- List all data sets that were last read or written a month ago or newer:

```
SD LASTREFDATE(START(-1M)) OWNER(*) LIMIT(*)
```

- List all data sets with a last CDS change of 1 year ago or older and that have no original expiration date set:

```
SD LASTCHANGEDATE(START(1900/001) END(-1Y)) NOOEXPDT OWNER(*) LIMIT(*)
```

- List all data sets that are retained forever and cataloged:

```
SD RETDATE(PERMANENT) CATLG(YES) OWNER(*) LIMIT(*)
```

- List all data sets, defined with data class “DC000001”, but no storage class:

```
SD DATACLASS(DC000001) NOSTORAGECLASS OWNER(*) LIMIT(*)
```

8.3 TVEXTPURGE extra days

With z/OS V1R12 DFSMSrmm, the parameter TVEXTPURGE had the options RELEASE, EXPIRE, and NONE. With z/OS V1R13, DFSMSrmm has a new option for EXPIRE(days). If tapes are expired using the EDGTVEXT HSM exit, extra days for retention can only be defined with no additional consideration.

You can use DFSMSrmm to release DFSMShsm tapes that are requested to be purged by DFSMShsm. You can also specify that DFSMSrmm retain a tape for a few days after its expiration date has been reached. By default, the expiration date protection for DFSMShsm tapes is done by DFSMShsm.

DFSMShsm uses 1999/365 as the expiration date for permanent retention. To enable extra days retention for purged DFSMShsm tape volumes, and depending on the retention method used, you need to either use the TVEXTPURGE(EXPIRE(days)) option or set up retention options in the vital record specifications that are used to retain the tape volumes.

EDGRMMxx parmlib member

This member specifies how DFSMSrmm processes volumes to be purged by callers of EDGTVEXT or EDGDFHSM. The TVEXTPURGE operand specifies how you want to handle the release of DFSMShsm tape volumes, using the following new specification:

```
TVEXTPURGE(EXPIRE(days) | NONE | RELEASE)
```

Where:

EXPIRE(days) Use the EXPIRE(days) option to set the volume expiration date to the current date plus days for volumes to be purged. Use the EXPIRE(days) parameter when you use the EXPDT retention method, using days as a way to delay expiration of the volume.

When using the VRSEL retention method, you can optionally use this operand in combination with vital record specifications that use the UNTILEXPIRED retention type. First, the EXPIRE(days) is applied to set a new volume EXPDT. Next, by running VRSEL you can then optionally extend retention using the extra days retention type. For example, specify EXPIRE(0) when using a VRS with extra days retention, or use a non-zero EXPIRE(days) value and avoid using an extra days retention VRS.

days - This is the number of days for which DFSMSrmm retains the volume before considering it for release. The value is a one-to-four digit decimal number which is added to today's date to compute the new expiration date. If the value exceeds the maximum retention period (MAXRETPD), it is reduced to the MAXRETPD value. The default value for days is 0. That is, TVEXTPURGE(EXPIRE) is the same as TVEXTPURGE(EXPIRE(0)).

NONE DFSMSrmm takes no action for volumes to be purged.

RELEASE DFSMSrmm releases the volume to be purged according to the release actions set for the volume. You must run expiration processing to return a volume to scratch status.

Default: TVEXTPURGE(RELEASE)

You can specify that volumes purged from DFSMShsm are to be retained a few extra days to be sure that purged volumes do not contain any data that might still be needed. DFSMShsm migration and backup volumes can be retained by EXPDT=99365. You can optionally set EXPDT to the current date or a future date when the volume is purged from DFSMShsm with EDGTVEXT.

You can release volumes or set the volume expiration date either to the current date or based on a number of days from the current date. The effect of setting the expiration date depends on the retention method (EXPDT or VRSEL) already specified for the volume.

Note: EXPIRE (days) - Use the EXPIRE(days) option to sent the volume expiration date to the current date plus days for volumes to be purged.

EXPIRE(days) option display

Figure 8-8 shows the output from RMMISPF panels, Option 4.8.2, where the option Days is set to 3.

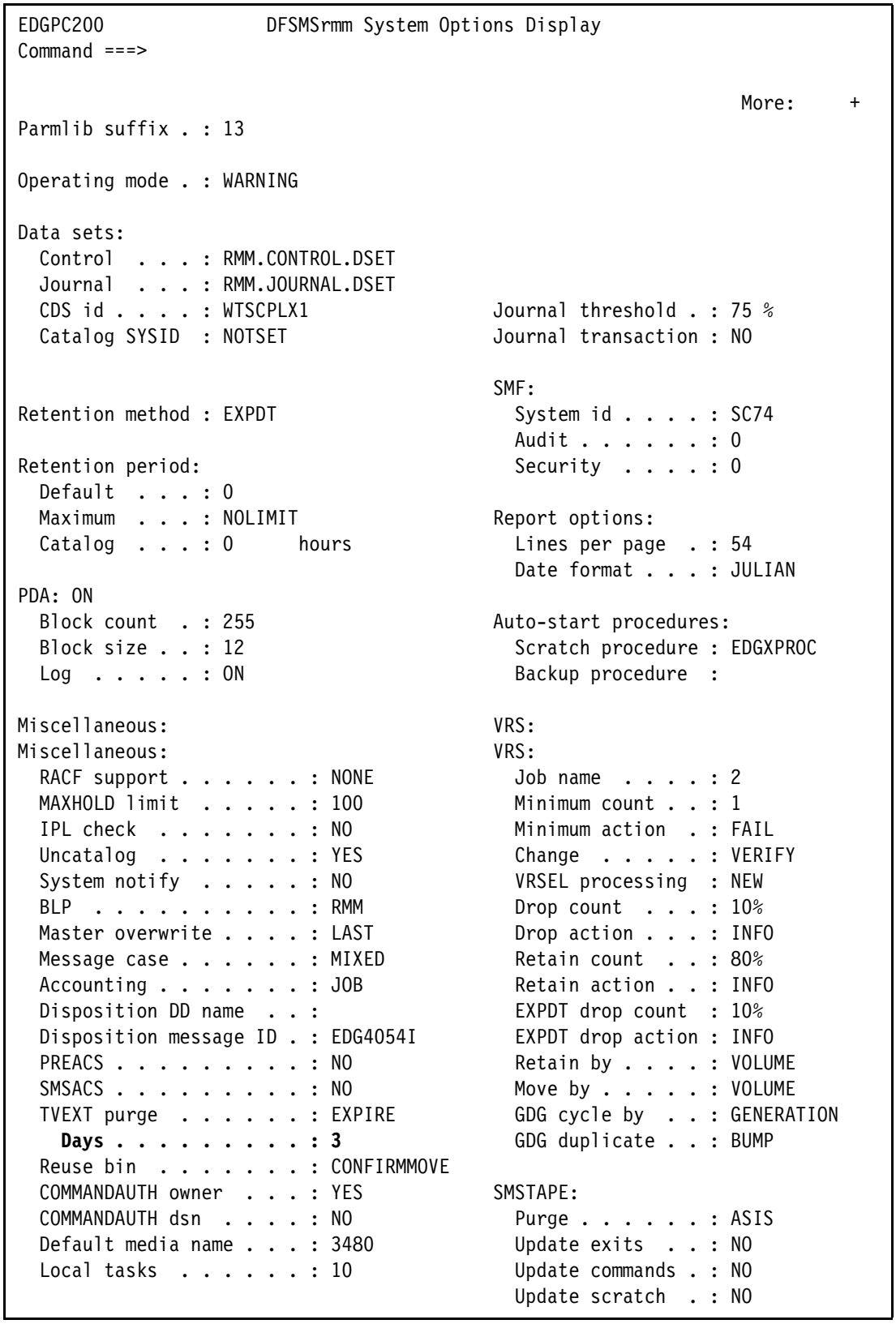


Figure 8-8 RMMISPF panel Option 4.8.2

LISTCONTROL command

From the ISPF Option 6 command line, if you enter the **`listcontrol option`** command, the output is displayed as shown in Figure 8-9.

```
System options:
PARMLIB Suffix  = 13
Operating mode  = W    Retention period: Default = 0       Maximum = NOLIMIT
                                         Catalog = 0     hours
                       Retention method: Method  = EXPDT
Control data set name      = RMM.CONTROL.DSET
Journal file data set name = RMM.JOURNAL.DSET
Journal threshold          = 75%               Journal transaction = NO
Catalog SYSID              = Notset
Scratch procedure name     = EDGXPROC
Backup procedure name      =
IPL date check  = N        Date format    = J       RACF support   = N
SMF audit       = 0        SMF security   = 0       CDS id         = WTSCPLX1
MAXHOLD value   = 100      Lines per page = 54      System ID      = SC74
BLP             = RMM      TVEXT purge    = EXPIRE  Notify         = N
                                days     = 3
Uncatalog       = Y        VRS job name   = 2       Message case   = M
MASTER overwrite= LAST     Accounting     = J       VRS selection  = NEW
VRS change      = VERIFY   GDG duplicate  = BUMP    GDG cycle by = GENERATION
VRSMIN action   = FAIL     VRSMIN count   = 1
VRSDROP action  = INFO     VRSDROP count  = 0              percent = 10
VRSRETAIN action= INFO     VRSRETAIN count= 0              percent = 80
EXPDTDROP action= INFO     EXPDTDROP count= 0              percent = 10
Disp DD name    =          Disp msg ID    = EDG4054I
Retain by       = VOLUME   Move by        = VOLUME  CMDAUTH Owner  = YES
PREACS          = NO       SMSACS         = NO      CMDAUTH Dsn    = NO
Reuse bin       = CONFIRMMOVE                       Media name     = 3480
                                                    Local tasks    = 10
PDA: ON
 Block count    = 255      Block size     = 12      Log            = ON
SMSTAPE:
 Update scratch = NO       Update command = NO      Update exits   = NO
 Purge          = ASIS
Client/Server:
 Subsystem type = STANDARD Port           = 0
 Server                    Server tasks   = 0
  host name  =
  IP address =
```

Figure 8-9 LISTCONTROL OPTION command

8.4 Expiry by data set information

A structured field introducer (SFI) is a structure that identifies one line or field of output data from another. The DFSMSrmm API returns these types of SFIs in your output buffer. With z/OS V1R13, there are new `Set by` fields (Figure 8-10) in a new REXX variable.

```
blank       - Not set
CMD         - Set by TSO subcommand
CMD_DEF     - Default RETPD applied during subcommand processing
CMD_VOLCAT  - EXPDT obtained from VOLCAT during subcommand processing
OCE_JFCB    - EXPDT obtained from EXPDT/RETPD keywords or from Data
              Class applied during tape recording
OCE_EXIT    - EDG_EXIT100 updated the JFCB EXPDT during tape recording
OCE_DEF     - Default RETPD applied during tape recording
OCE_MAX     - MAXRETPD was used to reduce the requested EXPDT during
              tape recording
OCE_VOLCAT  - EXPDT obtained from VOLCAT during tape recording
LCS         - EXPDT obtained from VOLCAT for system managed tapes when
              called from OAM installation exits
LCS_DEF     - Default RETPD applied for system managed tapes when
              called from OAM installation exits
TVEXTPURGE  - Set as a result of TVEXTPURGE parmlib option
CNVT        - Set during conversion by EDGCNVT
EXPORT      - Set during export processing
```

Figure 8-10 Set by fields in a new REXX variable

By simply looking at the expiration date of the volume or data set, it is hard to understand how it was set. Now DFSMSrmm records details of what event caused the EXPDT to be set or changed, and with this is easy to determine the event. The `Set by` field shows the events that caused the expiration data to be set or changed (Figure 8-11).

The SFI fields from the REXX variables are used to display information in DFSMSrmm volume and data sets details displays when used from the ISPF RMM panels.

```
EDGPT110              DFSMSrmm Volume Details - T00002              Multi-File
Command ===>

Volume . . . . . : T00002     VOL1 volser :         Rack number : T00002
Media name . . . . : MEDIA3                         Status  . . : USER
                                                                  More:     +
Volume type  . . . : PHYSICAL       Stacked count  . . . . . . . : 0
WorldWide ID . . . :                                Worm . . : NO
Retention date . . :                Expiration date  . . . . . . : 2011/136
Set retained . . . : NO               Set by . . . . . . . . . . : OCE_DEF
Hold . . . . . . . : NO             Original expiration date . :
Retention method . : EXPDT
  Set by . . . . . : CMD_DEF

Description  . . . :

Data set name  . . : 'BACKUP.BH5ST4'
Media information  : IBM
Media type . . . . : ETC
Vendor . . . . . :                  Release actions:
Label  . . . . . . : SL               Return to SCRATCH pool . : YES
  Current version  :                  Replace volume . . . . . : NO
  Required version :                  Return to owner  . . . . : NO
Density  . . . . . : IDRC            Initialize volume  . . . : NO
Recording format . : EFMT1           Erase volume . . . . . . : NO
Compaction . . . . : YES             Notify owner . . . . . . : NO
Attributes . . . . : NONE            Expiry date ignore . . . : NO
.................
```

Figure 8-11 Volume details (partial display)

8.5 Exclude data sets from VRSEL processing

You can use the EDG_EXIT100 installation exit to exclude specific data sets from DFSMSrmm VRSEL processing as they are created or rewritten. You can specify this for any data set, but DFSMSrmm ignores the request unless the data set is on a volume that is managed by the VRSEL retention method. The data set VRSELEXCLUDE attribute is set for all data sets on volumes managed by the EXPDT retention method, and is not affected by this support.

When DFSMSrmm excludes a data set from VRSEL processing, it ensures that the data set vital record attribute is reset and the retention date is set to current date. The matching VRS information is left unchanged.

Inventory management considerations

DFSMSrmm records the complete inventory of the removable media library and storage locations in the DFSMSrmm control data set, which is a VSAM key-sequenced data set. DFSMSrmm records all changes made to the inventory, such as adding or deleting volumes, and also keeps track of all movement between libraries and storage locations.

With z/OS V1R13, consider the following new processing features:

- You do not need to run VRSEL processing unless volumes are defined with the VRSEL retention method. Only EXPROC processing is required to handle expiration of all volumes managed by the EXPDT retention method.
- EXPROC processing provides a summary of volumes by retention method.
- The expiration date of volumes is set during OPEN processing, so for volumes managed by the EXPDT retention method, no special considerations exist for open data sets; they are managed based on the volume EXPDT.
- For volumes managed by the EXPDT retention method, no special considerations exist for data sets closed by ABEND processing or that are DELETED; they are managed based on the volume EXPDT.
- Volumes managed by the EXPDT retention method are included only in the EXPDTDROP limit. VRSRETAIN and VRSDROP limits apply only to volumes managed by the VRSEL retention method.

New command operands

All data in the DFSMSrmm inventory is managed by dynamic VRS policies. With z/OS V1R13 DFSMSrmm there are new operands for the commands CHANGEDATASET, SEARCHDATASET, and VRSELEXCLUDE.

CHANGEDATASET Use the CHANGEDATASET COPYFROM subcommand to copy the data set attributes from one data set dsname to another data set.

The CHANGEDATSET operand overwrites DFSMSrmm VRSEL processing. You can specify this for any data set already retained as a vital record. Its vital record attribute is reset and the retention date is set to the current date. The data set VRSELEXCLUDE attribute is set to YES for all data sets on volumes managed by the EXPDT retention method (Figure 8-12).

SEARCHDATASET The SEARCHDATASET operand specifies to limit the search to data sets excluded (or not excluded) from VRSEL processing. Specify NO to search for data sets that are not excluded from VRSEL processing. Specify YES to search for data sets that are excluded from VRSEL processing.

VRSELEXCLUDE Data sets that have the VRSELEXCLUDE attribute set to “on” are excluded from vital record processing. The overhead will be reduced by eliminating certain types of data from VRSEL processing.

When the data set is created on a tape volume managed by the EXPDT retention method, the data set VRSELEXCLUDE attribute is automatically set. If the data set is created on a tape volume managed by the VRSEL retention method, the VRSELEXCLUDE attribute can be set by the EDG_EXIT100 installation exit, and can be set or reset later by command.

Figure 8-12 shows the output of ISPF panel Data Set Details with the new field `VRSEL exclude`.

```
 EDGPD110                 DFSMSrmm Data Set Details                 Multi-File
 Command ===>

 Data set name . . : 'BACKUP.BH5ST4'
 Volume serial . . : T00002      Physical file sequence number . . : 1
 Owner . . . . . . : CESAR       Data set sequence number  . . . . : 1
                                                                More:     +
Job name  . . . . : DFDSSDMP
Step name . . . . : BH5ST4                 Record format . . . . : U
Program name  . . : ADRDSSU                Block size  . . . . . : 262144
DD name . . . . . : TAPE                   Logical record length : 0
Create date . . . : 2011/136    YYYY/DDD   Block count . . . . . : 10470
Create time . . . : 14:19:05               Total block count . . : 10470
System id . . . . : SC74                   Data set size (KB)  . : 2680320
                                           Percent of volume . . : 1
Expiration date . : 2011/136    YYYY/DDD   Device number . . . . : 0B00
 Set by . . . . . : OCE_DEF
Original  . . . . :             YYYY/DDD

Last job name . . : DFDSSDMP               Last DD name  . . . . : TAPE
Last step name  . : BH5ST4                 Last device number  . : 0B00
Last program name : ADRDSSU
Date last read  . : 2011/136               VRS management value  :
Date last written : 2011/136               Management class  . . :
                                           Data class  . . . . . :
VRSEL exclude . . : YES                    Storage class . . . . :
Retention date  . :                        Storage group . . . . :
VRS retained  . . : NO
..........
```

Figure 8-12 Data set details (partial display)

8.5.1 Retaining source data sets managed by the VRSEL retention method

With z/OS V1R13, the tape copy application can optionally decide how to manage the retention of the source data set and volume. To manage the retention of the source data set and volume, the tape copy application must issue volume and data set subcommands to set the required retention policy. The application can decide whether the specific data set, the volume, or the entire volume set is no longer required. Retention of data sets is subject to retention of the volume. Only when all data sets are no longer VRS-retained and the volume expiration date is reached or ignored will the volume be set pending release.

Note: Review the information in *z/OS DFSMSrmm Managing and Using Removable Media*, SC26-7404, if you have decided to use the VRSEL retention method for any of your volumes and data sets. Vital record specifications are used only for retention and movement of volumes that are managed by the VRSEL retention method and data sets on these volumes that are not excluded from VRSEL processing.

You can control the way DFSMSrmm retains and moves volumes that are managed by the VRSEL retention method and data set on these volumes that are not excluded from VRSEL processing by defining these DFSMSrmm EDGRMMxx parmlib member OPTION operands.

You can use the following options for VRSEL retention:

- Keep the current retention.

  Assuming the target data set matches the same VRS as the source data set, this means that the target data set and the source data set are being possibly retained by the same VRS.

  When the copy and the source data sets use the same data set name, consider that there are two generations or cycles of the data set. And, if managed by cycles retention (the VRS specifies CYCLES or BYDAYSCYCLE), the retention can be unpredictable.

  If the data sets are GDSs, then the setting of the DFSMSrmm parmlib option OPTION GDG(DUPLICATE()) determines how the two data sets are managed. For other data sets, the data sets might count as two cycles, potentially causing roll-off of the oldest cycle, which may be neither the source nor the target.

- Use a new retention method for the data set.

  If you want to retain a data set currently managed by the VRSEL retention method for a defined period, you can change to use the EXPDT retention method for the data set's volume and set a retention period:

  ```
  RMM CV volser RETENTIONMETHOD(EXPDT) RETPD(days)
  ```

  If the volume is part of a multivolume set, you must change the retention method for the entire multivolume set by:

  1. Changing the retention method for the first volume in the multivolume set. DFSMSrmm changes the retention method for all volumes in the multivolume set.

     ```
     RMM CV volser RETENTIONMETHOD(EXPDT)
     ```

  2. Set the required expiration date for each volume in the multivolume set:

     ```
     RMM CV volser RETPD(days)/EXPDT(date)
     ```

  This can be performed for the multivolume set as follows:

  ```
  RMM SV VOLUME(volser) CHAIN LIMIT(*) CLIST('RMM CV ','
  RETPD(days)')
  EXEC RMM.EXEC
  ```

- Release the volume or volume set.

  When copying all the files on a volume, you can decide to manage the retention or expiration at the volume level. If you no longer require the data, you can release the volume manually:

  ```
  RMM DV volser RELEASE
  ```

  To release the volume chain, you can use the DFSMSrmm SEARCH with CLIST capability:

  ```
  RMM SV VOLUME(volser) CHAIN LIMIT(*) CLIST('RMM DV ',' RELEASE')
  EXEC RMM.EXEC
  ```

- Prevent expiration of the volume.

  Set the hold attribute for the source volume to prevent it being released until the data is no longer needed:

  ```
  RMM CV volser HOLD
  ```

- Delete the source data set

  Mark the source data set as deleted and ensure that a DELETED VRS exists to manage deleted data sets. The specified retpd value is used to set the data set EXPDT, and if it is greater than the volume EXPDT, DFSMSrmm will increase the volume EXPDT to match.

Whether this affects how long the source data is retained depends on the matching VRS. The COUNT(retpd) value and the retention type on the DELETED VRS will ultimately determine how long the deleted data set is retained. For example:

```
RMM CD dsname SEQ(seq) VOLUME(volume) DELETED(YES)
RMM AS DSN('DELETED') LOCATION(CURRENT) LASTREF COUNT(retpd)
RELEASE(EXPIRYDATEIGNORE)
```

- Exclude the source data set from VRSEL retention.

  You can exclude the source data set from VRSEL by:

  ```
  RMM CD dsname VOLUME(volser) SEQ(seq) VRSELEXCLUDE(YES)
  ```

  You can optionally specify the RETPD operand, which sets the data set expiration date and also updates the volume expiration date, if it is a higher value than the current volume expiration date. You can combine this with the copying of attributes in a single command:

  ```
  RMM CD datasetname VOLUME(tgtvol) COPYFROM(VOLUME(volser) RETPD(3)
  VRSELEXCLUDE)
  ```

Figure 8-13 shows the output of ISPF panel Change Data Set Details, where the field `VRSEL exclude` can be changed.

```
EDGPD110                DFSMSrmm Change Data Set Details          Multi-File
Command ===>

Data set name . . : 'BACKUP.BH5ST6'
Volume serial . . : T00002      Physical file sequence number . . : 3
Owner . . . . . . : CESAR       Data set sequence number  . . . . . 3
                                                                More:     +
Job name  . . . . . DFDSSDMP
Step name . . . . : BH5ST6                Record format . . . . . U
Program name  . . . ADRDSSU               Block size  . . . . . . 262144
DD name . . . . . : TAPE                  Logical record length   0
Create date . . . . 2011/111    YYYY/DDD  Block count . . . . . . 647
Create time . . . . 17:14:28              Total block count . . : 647
System id . . . . . SC74                  Percent of volume . . : 0
                                          Device number . . . . . 0B00

Last job name . . : DFDSSDMP              Last DD name  . . . . : TAPE
Last step name  . : BH5ST6                Last device number  . : 0B00
Last program name : ADRDSSU
Date last read  . . 2011/111              VRS management value
Date last written   2011/111              Management class  . . .
                                          Data class  . . . . . .
VRSEL exclude . . . NO                    Storage class . . . . .
Retention date  . :                       Storage group . . . . :
VRS retained  . . : NO
```

Figure 8-13 Change data set details

DFSMSrmm Data Set Search panel

Figure 8-14 shows the highlighted fields are new with z/OS V1R13.

```
EDGPD010                    DFSMSrmm Data Set Search
 Command ===>

 Enter fully qualified or partial data set name and job name:

 Data set name  . . . . 'BACKUP.*'
 Job name . . . . . .              Specific or generic name

 Enter optional parameters to qualify search
                                                                More:     +
 Owner  . . . . . .                Owner of volumes (Default is your userid)
 Volume serial  . . . .           OR Volume serial
 List entire set  . . .            YES, For all data sets in the
                                   multi-volume set, otherwise NO
 Status . . . . . .                PRIVATE, SCRATCH, or blank for all
 Retained by VRS  . . .            YES, NO, or blank for all
 Excluded from VRSEL               YES, NO, or blank for all
 Dates           Start          End       Date, date range or relative value
   Create  . . .             . .
   Expiration                . .
   Reference . .             . .
   Read  . . . .             . .
   Write . . . .             . .
   Changed . . .             . .
```

Figure 8-14 Data set search

8.5.2 DFSMSrmm dynamic exit services

DFSMSrmm uses the services of the z/OS Dynamic Exit Facility for all of its installation exits. This permits the use of multiple installation exit routines for each DFSMSrmm exits (Figure 8-15). It also enables you to control these exits and their exit routines with the EXIT statement of the PROGxx parmlib member, the SET PROG=xx operator command, the SETPROG EXIT operator command, or the CSVDYNEX macro.

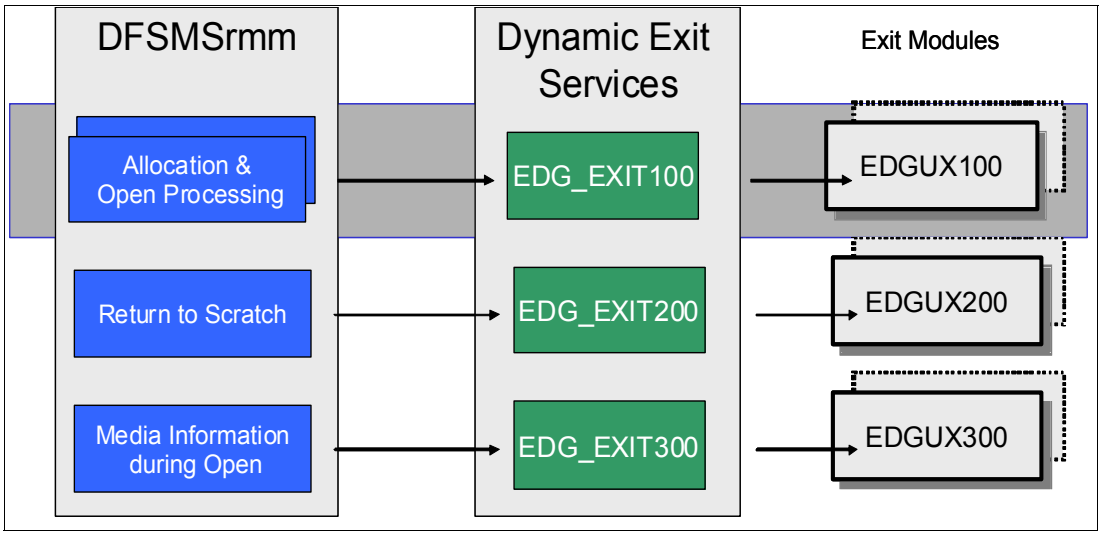


Figure 8-15 DFSMSrmm dynamic installation exits

z/OS V1R13 enhancements

For z/OS V1R13, when you assemble DFSMSrmm installation exits, use the IBM-supplied default assembler options unless noted otherwise. Ensure that the ALIGN option is in effect.

Note: By default, the system disables the exit routines for EDG_EXIT300 after one abend. By default, the system does not disable the exit routines for EDG_EXIT100 and EDG_EXIT200.

The following exit changes can be made:

EDG_EXIT100 You can use the EDG_EXIT100 installation exit to set the retention method to be used for new tape data sets. When you create a new tape volume set, or rewrite an existing set from the first file, you can override the system default retention method.

You can use the EDG_EXIT100 installation exit to exclude specific data sets from DFSMSrmm VRSEL processing as they are created or rewritten. You can specify this for any data set, but DFSMSrmm ignores the request unless the data set is on a volume that is managed by the VRSEL retention method. The data set VRSELEXCLUDE attribute is set for all data sets on volumes managed by the EXPDT retention method and is not affected by this support.

When DFSMSrmm excludes a data set from VRSEL processing, it ensures that the data set vital record attribute is reset and the retention date is set to current date. The matching VRS information is left unchanged.

Note: For details about these changes, see *z/OS DFSMSrmm Implementation and Customization Guide*, SC26-7405.

EGD_EXIT100 considerations

To tailor your user exit, make the following changes:

- Copy the sample EDGUX100 exit module and use the copy as a base for your exit module.
  - Only perform your processing when the PL100_CAN_VRSELEXCLUDE bit is on.
  - Set PL100_SET_VRSELEXCLUDE bit to B'1' for data sets. If you do not request VRSELEXCLUDE, then the default for the retention method will be used. If the installation exit sets PL100_SET_VRSELEXCLUDE, then any VRS management value set in PL100_VRS is ignored.
  - You do not need to set the PL100_SET_VRSELEXCLUDE bit when you also request to set the retention method to EXPDT. DFSMSrmm always sets the VRSELEXCLUDE attribute for data sets managed by the EXPDT retention method.
- Make any other changes required such as setting the retention method when creating the first file on the tape, or clearing the EXPDT.

Note: The sample EDGUX100 exit module includes an example of setting the VRSELEXCLUDE attribute. The order in which the table entries are listed is important because the exit scans the table until it finds the first entry where the job name, data set name and program name masks match the current request. You can change the priority of matching by changing the order of the table entries.

Migration considerations

z/OS releases lower than z/OS V1R13 require the PTF for coexistence APAR OA32984 to be installed before exploitation of the new functions is attempted on z/OS V1R13.

Standard coexistence recognizes and supports the data set level VRSELEXCLUDEVRSEL processing on releases below z/OS V1R13 and thus skips these data sets.

There is a new report in DFSMSrmm, EDGRT18, which includes data sets and volumes other than those that are scratch. REPORT18 is split by retention method and sorted by data set name, create date, and create time. Figure 8-16 shows a sample of this report.

```
DFSMSrmm INTERNAL USE ONLY Inventory of Data Set Names by Volume Retention Method EXPDT PAGE-1
EDGRPT18                                                                    DATE -   27/04/2011
                                                                            TIME -     10:07:23
                 Volume Vol- DSN-  Creating Create     Crate  Volume     DSN       V EXPDT
  Data Set Name  Serial Seq. Seq.  Jobname  Date       Time   Exp. Date  Exp. Date X Set by
  ---------------------- ------ ---- ----- -------- ---------- ------ ---------- ----------
  BACKUP.BH5ST4  T00002    1     1 DFDSSDMP 21/04/2011 171303 21/04/2011 21/04/2011 Y OCE_DEF
  BACKUP.BH5ST5  T00002    1     2 DFDSSDMP 21/04/2011 171400 21/04/2011 21/04/2011 Y OCE_DEF
  BACKUP.BH5ST6  T00002    1     3 DFDSSDMP 21/04/2011 171428 21/04/2011 21/04/2011 Y OCE_DEF
                          End of Report.   3 Entries listed
DFSMSrmm INTERNAL USE ONLY Inventory of Data Set Names by Volume Retention Method VRSE  PAGE-2
EDGRPT18                                                                     DATE -  27/04/2011
                                                                             TIME -    10:07:23
                      Volume Vol- DSN-  Creating Create     Create Volume     DSN        V V
  Data Set Name       Serial Seq. Seq.  Jobname  Date       Time   Ret. Date  Ret. Date  X R
  ---------------------------------- ---- ----- -------- ---------- ------ ---------- ----- -
  BACKUP.BH5ST3       T00001    1     1 DFDSSDMP 26/04/2011 150351                       N N
  ES976.DUMP.#@$#A1   THS305    1     1 ALLDUMP1 17/12/2004 140015 WHILECATLG WHILECATLG N Y
  ES976.DUMP.#@$#B1   THS305    1     2 ALLDUMP1 17/12/2004 140143 WHILECATLG WHILECATLG N Y
  ES976.DUMP.#@$#B2   THS305    1     3 ALLDUMP1 17/12/2004 140857 WHILECATLG WHILECATLG N Y
```

Figure 8-16 EDGRPT18 report

In the VRSRETN report shown in Figure 8-17, a new column is added to include the data set VRSEXCLUDE attribute.

```
Newly assigned volumes subject to VRSRETAIN          01/20/09        05:55:21        - 1 -

Status: RETAINED

D A T A   S E T                                D A T A   S E T   V R S                                   V O L U M E
                                             V          DROP   REASON                             VRS           RETAIN    E
VOLSER  FSEQ  DSNAME                JOBNAME  X RETAINED PRIM   2nd.     PRIMARY VRS       JOB MASK TYPE  VRS     REASON    C
A22251     1  RMMUSER.DSN1                   N Y                        RMMUSER.*                  D            DATASET
A22252     1  RMMUSER.DSN20                  N Y                        RMMUSER.*                  D            DATASET
A22252     2  RMMUSER.DSN21                  N Y                        RMMUSER.*                  D            DATASET
A22253     1  D046059.DSN01                  Y N        W               D046059.*                  D            IMPLICIT
A22253     2  DSN02                          Y                                                                  IMPLICIT
A22253     3  D046059.DSN03                  N Y                        D046059.*                  D            DATASET
VOL001     1  First.data.set        F1J      N N        D               ABEND             *        D     VOL001 VOLUME
VOL001     2  Second.data.set       F1J      Y                                                           VOL001 VOLUME
VOL002     1  Second.data.set       F1J      Y                                                                  SET
VOL002     2  third.data.set        F1J      Y                                                                  SET

data sets in this status:            10

Newly assigned volumes subject to VRSRETAIN          01/20/09        05:55:21        - 2 -

Status: NOTRETAINED

D A T A   S E T                                D A T A   S E T   V R S                                   V O L U M E
                                             V          DROP   REASON                             VRS           RETAIN    E
VOLSER  FSEQ  DSNAME                JOBNAME  X RETAINED PRIM   2nd.     PRIMARY VRS       JOB MASK TYPE  VRS     REASON    C
A22250     1  D046059.WCATALOG      SSTEINHA N N        W               D046059.*                  D
A22256     1  DSN6                           N
NO0001     1  ANOTHER.DSET          WOODY1   Y
NO0001     2  YET.ANOTHER           WOODY2   Y

data sets in this status:             4
```

Figure 8-17 VRSRETN report

In the EXPDROP report, a new column is added to include the retention method, as shown in Figure 8-18.

```
EXPDT retained volumes subject to EXPDTDROP          01/20/09        05:55:21        - 1

Status: RELEASED

VOLSER  VSEQ  DSNAME                 JOBNAME   EXPRSN  ASSIGNED    EXPDT       RM  SR  RETDATE
A22255     1  NOMATCH.DSN5           SSTEINHA  X       12/12/1999  01/01/2009  V   N   19/02/200
A22256     1                                   X       12/12/1999  12/31/2008  E   N

Volumes in this status:          2

EXPDT retained volumes subject to EXPDTDROP          01/20/09        05:55:21        - 2

Status: NOCHANGE

VOLSER  VSEQ  DSNAME                 JOBNAME   EXPRSN  ASSIGNED    EXPDT       RM  SR  RETDATE
A22257     1  DSN7                   SSTEINHA          02/06/1993  PERM        V   N   12/12/200
A22258     1  DSN7                   SSTEINHA          02/06/1993  01/05/1990  V   Y   12/12/200

Volumes in this status:          2
```

Figure 8-18 EXPDROP report

8.6 Data set attribute COPYFROM function

At some point, you might need to copy or move tape data from one tape to another. You can do this by using a tape copy utility that uses DFSMSrmm services to correctly copy and restack tape data sets. This will preserve the data set attribute settings and, optionally, set retention periods for the source data set, copy data set, or both. If you use your own JCL and the IEBGENER utility, you can use the RMM CHANGEDATASET subcommand with the COPYFROM operand to complete the copying of the source data set attributes. However, a tape copy application can select the method used to communicate with DFSMSrmm.

After the target data set has been created, the tape copy application can use the CHANGEDATASET COPYFROM subcommand to copy all applicable attributes from the source data set to the target data set. DFSMSrmm determines which attributes are to be copied.

During the creation of the target data set, tape copy application programs can use the EDG_EXIT100 installation exit to copy all applicable attributes from the source data set to the target data set.

By using DFSMSrmm COPYFROM support you can ensure that all applicable data set attributes are copied, and avoid using multiple RMM subcommands to modify only the attributes supported by the subcommands.

8.6.1 Using the CHANGEDATASET COPYFROM subcommand

For a target data set that has already been created, a tape copy application can use the CHANGEDATASET COPYFROM subcommand to copy all applicable attributes from the source data set to the target data set. DFSMSrmm determines which attributes are to be copied. Retention of the source data set can be specifically set. Retention of the target volumes and data sets can be selected at the volume set level, and even switched between VRSEL and EXPDT retention methods.

By using the CHANGEDATASET COPYFROM subcommand you can ensure that all applicable data set attributes are copied, and avoid using multiple RMM subcommands to modify only various attributes.

The CHANGEDATASET COPYFROM subcommand identifies a single volume data set or any part of a multivolume data set. Validation is done to ensure that the source and target data sets have the same recording format and record length. You use the CHANGEDATASET subcommand one time for each target data set record. For multivolume data sets, this means that you must issue the subcommand one time for each volume that the target data set is written on.

Figure 8-19 shows the syntax of the CHANGEDATASET subcommand with this new option.

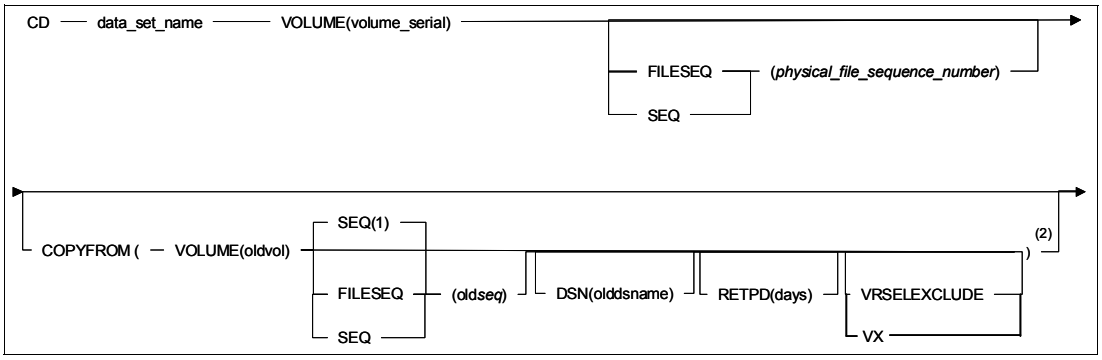


Figure 8-19 CHANGEDATASET subcommand

CHANGEDATASET subcommand

When you specify other CHANGEDATASET subcommand operands, the processing sequence in DFSMSrmm is that the COPYFROM operand processes first, and then the additional operands process. This means that additional operands can specify data that overrides the attributes copied.

Note: Define a new RACF profile to control the use of the CHANGEDATASET COPYFROM subcommand to copy the data set attributes.

```
STGADMIN.EDG.CD.COPYFROM.dsname
```

Various data set attributes are not copied. Table 8-1 lists these attributes.

Table 8-1 Data set attributes not copied

Command operand No cmd operand	Extract file field	REXX Variable/SFI
dsname	RDDSNAME	EDG@DSN
VOLUME	RDVOLSER	EDG@VOL
SEQ|FILESEQ	RDDSNSEQ	EDG@FILE
LABELNUMBER	RDLABNO	EDG@DSEQ
TOTALBLKCOUNT	RDTOTAL_BLKCNT	EDG@BLKT
PERCENT	RDPERCENT	EDG@DPCT
DEVNUM	RDUNITAD	EDG@DEV
LRECL	RDLRECL	EDG@LRCL
RECFM	RDRECFM	EDG@RCFM
BLKSIZE	RDBLKSZ	EDG@BLKS
BLKCOUNT	RDBLKCNT	EDG@BLKC
owner	RDOWNDSN	EDG@OWN
data set size	RDDSSIZE, RDSIZE	EDG@DSS6
catalog status	RDCAT	EDG@CTLG
STORAGECLASS	RDSCNAME	EDG@SC
storage group	RDSGNAME	EDG@SG
DATACLASS	RDDCNAME	EDG@DC
start block ID	n/a	n/a
end block ID	n/a	n/a
last device number	RDLDEVN	EDG@LDEV
BESKEY	RDBESKEY	EDG@BESK
VRSELEXCLUDE	RDVEX	EDG@VEX Note: This attribute is not copied unless both the source and target volumes are managed by RM(VRSEL)
ABEND	RDABEND	EDG@ABND Note: This attribute is not copied unless it is set. The source setting is merged with the target setting.

Using the EDG_EXIT100 installation exit

The tape copy application can use the EDGPL100 macro to invoke the installation exit EDG_EXIT100 to copy the data set attributes from the source data set to the copy data set

during OPEN processing. EDG_EXIT100 can notify DFSMSrmm that the data set being created is being copied from another.

During OPEN processing, the exit can identify the source data set from which DFSMSrmm will obtain all existing data set attributes, which will be used for the target data set. DFSMSrmm EOV processing ensures that the attributes are copied to all target data set records when the output data set becomes a multivolume data set.

Note: The tape copy application that activates an EDG_EXIT100 exit module must satisfy the requirements enforced by the CSVDYNEX macro, such as system state and key, or APF authorized, or authorized by SAF. Alternatively, the installation has control by an update to PROGxx or by issuing the SETPROG system command.

The tape copy application must use the installation exit to ensure that the source and target data set are one and the same.

EDG_EXIT100 parameter list

All communication is done using the parameter list. The parameter list is mapped by the EDGPL100 macro.

The parameter list input values that are new are listed here:

PL100_VALID This field defines functions you can request during this call of the installation exit.

- **PL100_CAN_COPYFROM** If set to B'1', you can request to notify DFSMSrmm that the data set being created is being copied from another.
- **PL100_CAN_RETMET** If set to B'1', you can request to set the volume retention method when the first file on the volume is written.

PL100_VALID2 This field defines functions you can request during this call of the installation exit.

- **PL100_CAN_VRSELEXCLUDE** If set to B'1', you can request to set the VRSELEXCLUDE attribute for a data set.

PL100_FUNCTION2 You can use this field to change several of the processing decisions that were made during disposition control processing. DFSMSrmm uses the PL100_LOCATION value to set the destination location or the current location. When DFSMSrmm sets the destination location, you must confirm the volume move later using the RMM CHANGEVOLUME subcommand.

- **PL100_SET_RETMET** Set this bit to B'1' to indicate that the volume's retention method has been set to the value defined in PL100_RETENTIONMETHOD.
- **PL100_SET_COPYFROM** Set this bit to B'1' to indicate that the data set being created is being copied from another.
- **PL100_SET_VRSELEXCLUDE** Set this bit to B'1' to indicate that the data set being created has the attribute VRSELEXCLUDE set.

PL100_RETENTIONMETHOD Set this field to 0 or 1 for:

PL100_RM_VRSEL EQU 0 RETENTION METHOD VRSEL

PL100_RM_EXPDT EQU 1 RETENTION METHOD EXPDT

PL100_COPYFROM_DSN	Set this field to the name of the source data set.
PL100_COPYFROM_VOLSER	Set this field to the source volume number.
PL100_COPYFROM_DSEQ	Set this field to the source data set sequence number.
PL100_COPYFROM_OWNER	Set this field if you want to specify the owner of the volume being written to. It can be applied only when the first file on the first volume of a multivolume set is written, and will be propagated by DFSMSrmm during EOV to any additional volumes in the set. This field is optional for the COPYFROM function.

Copying data set attributes in a tape copy application

A tape copy application can use the EDG_EXIT100 installation exit to copy the data set attributes from the original to the copy during OPEN processing. In the EDG_EXIT100 installation exit, you can specify the data set details for the source data set from which attributes is to be copied, as explained here.

To modify your tape copy application to exploit use of EDG_EXIT100, follow these steps:

1. Verify that EDG_EXIT100 is active.
2. Activate your exit module.
3. Copy the tape data sets and communicate your processing to your exit module.
4. When all copies are completed or as processing progresses successfully, you can update the source data sets and volumes to set specific retention methods, expiration dates, or release volumes that are no longer required.
5. Deactivate your exit module.

See *z/OS DFSMSrmm Implementation and Customization Guide*, SC26-7405, for detailed information about EDG_EXIT100.

Note: It is your responsibility to provide the correct source data set details. If you pass the wrong source COPYFROM data set details, one of the following applies:

- DFSMSrmm ignores your request if you do not provide valid values for the following:
  - PL100_COPYFROM_DSN
  - PL100_COPYFROM_VOLSER, and
  - PL100_COPYFROM_DSEQ
- If the data set record is not already defined in the DFSMSrmm CDS, processing continues, but WTO EDG4063I is issued to the job log and the system log.
- If the data set record is found, all relevant attributes are copied. In all cases, you can always complete processing or correct processing using the CHANGEDATASET subcommand with COPYFROM specifying the correct source data set record.

8.7 Support RETPD (93000)

With z/OS V1R13, when specifying the number of days that DFSMSrmm retains the data set before considering it for release, the retention_period can be a decimal number from 0 to 93000. This value is used in RETPD and MAXRETPD in all specifications in DFSMSrmm.

This value is set on the EDGRMMxx parmlib member, as shown in Figure 8-20.

```
|-+--------------------+------->
  |        .---5---.   |
  '-RETPD(-+-------+-)-'
           '-nnnnn-'

|-+-------------------------+-->
  |           .-NOLIMIT-.   |
  '-MAXRETPD(-+---------+-)-'
              '--nnnnn--'

The new maximum value is 93000.
```

Figure 8-20 EDGRMMxx parmlib member OPTION statement parameters

DFSMSrmm subcommands

Figure 8-21 shows the DFSMSrmm subcommands that allow the following specification:

```
RETPD(retention_period) - Specifies the number of days that DFSMSrmm retains
the volume before considering it for release. retention_period is a decimal
number from 0 to 93000.
```

```
ADDDATASET
ADDVOLUME
CHANGEDATASET
CHANGEVOLUME
GETVOLUME
```

Figure 8-21 DFSMSrmm subcommands that allow RETPD to be specified

8.8 Selective volume movement

Many clients have non-IBM virtual tape solutions and need another way to prevent volume movement driven by VRS. With the new LOCDEF operand, AUTOMOVE(YES/NO), you can define locations that are not applicable for automated movement.

This new capability in z/OS V1R13 is designed for libraries that contain virtual volumes or other volumes which either cannot be moved or for which you do not want DFSMSrmm to initiate the movement.

LOCDEF operand

With the new LOCDEF operand, AUTOMOVE(YES/NO), you can define locations that are not applicable for automated movement. When the volumes current location is defined in the EDGRMMxx parmlib member with LOCDEF location AUTOMOVE(NO), DSTORE processing does not set the destination from the required location.

During inventory management, DSTORE processing, DFSMSrmm validates the current location name for a volume and determines if automated movement is required. If validation fails, no movement is initiated. If a location is not defined through LOCDEF on the inventory management system, then automated movement is started.

All volumes can be manually moved by RMM subcommands. Moves requested manually from an AUTOMOVE(NO) location to bin-managed storage locations will not be initiated by DSTORE processing.

The new operand is shown in Figure 8-22.

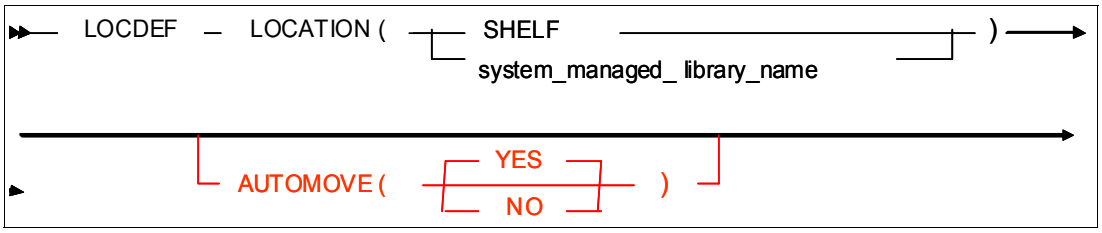


Figure 8-22 The new AUTOMOVE option

AUTOMOVE The AUTOMOVE operand is used to identify libraries or storage locations that hold tape volumes that are not eligible for automated movement initiated by inventory management. During inventory management, DSTORE processing the volume's current location is validated against the matching LOCDEF. Consider this operand for libraries that contain virtual volumes or other volumes that either cannot be moved or for which you do not want DFSMSrmm to initiate the movement.

YES - This specifies that DSTORE will initiate automated movement when the required location does not match the current location of the volume. This is the default.

NO - This specifies that DSTORE will not initiate automated movement.

Note: All volumes can be manually moved by DFSMSrmm subcommands.

RMM LISTCONTROL command

In the output of the TSO RMM LISTCONTROL command for the location definition, there is a new field AM(AUTOMOVE), as shown in Figure 8-23.

```
EDGPC800                DFSMSrmm Location Definitions          Row 1 to 6 of 6
Command ===>                                                  Scroll ===> CSR

Location Locdef Management Location Prio- AM Media Names
                Type       Type     rity
-------- ------ ---------- -------- ----- -- --------------------------------
         NO                AUTO     4800  Y
         NO                MANUAL   4900  Y
DISTANT  NO                STORE    200   Y
LOCAL    NO                STORE    300   Y
REMOTE   NO                STORE    100   Y
SHELF    NO                SHELF    5000  Y
```

Figure 8-23 Location Definitions

8.9 Last change details

There is a need for an easy way to audit changed media whenever possible without running the EDGAUD audit reports. With z/OS V1R13, the last change information is added to all list command output and to the dialog for all resources stored in the RMM CDS. This can reduce the need to run EDGAUD audit reports.

The EDGAUD and EDGRPTD utilities and the EDGRRPTE exec produce information about your removable media library and storage locations. You can also get security trail information about volumes and data sets defined to DFSMSrmm and audit trail information about volumes, shelf assignments, and user activity.

New information is added to all **list** command outputs and **list**, **change**, and **delete** to the panel for all resources stored in the RMM CDS. This reduces the need to run EDGAUD audit reports.

The following figures show panels that now display information indicating when information was changed on that display. Figure 8-24 shows the Last Change information for the Data Set Details panel.

```
EDGPD110                 DFSMSrmm Data Set Details                End of da
Command ===>

Data set name . . : 'BACKUP.BH5ST3'
Volume serial . . : T00001      Physical file sequence number . . : 1
Owner . . . . . . : CESAR       Data set sequence number  . . . . : 1
                                                                  More:   -
Retention date  . :                          Storage group . . . . :
VRS retained  . . : NO

Security name . . :                          BES key index . . . . : 0
Classification  . :

Primary VRS details:  (Use MATCHVRS primary command to display matching VRSe
 VRS name . . . :
 Job name . . . :                              VRS type  . . . . . :
 Subchain name  :                              Subchain start date :

Secondary VRS details:
 Value or class :
 Job name . . . :
 Subchain name  :                              Subchain start date :

Catalog status  . : UNKNOWN
Closed by Abend . : NO                       Deleted . . . . . . . : NO

Last Change information:
 Date . . . . . . : 2011/116       Time  . : 15:04:52     System  : SC74
 User change date :                Time  . :              User ID : *OCE
```

Figure 8-24 Data Set Details

Figure 8-25 shows the Last Change information for the Volume Details panel.

```
EDGPT110              DFSMSrmm Volume Details - T00001              Multi-File
Command ===>

Volume . . . . . : T00001     VOL1 volser :         Rack number : T00001
Media name . . . . : MEDIA3                         Status  . . : USER
                                                                  More:   - +
  User . . . . . :                 User . . . . . . . . . . :

Last Change information:
 Date . . . . . : 2011/116       Time  . : 15:07:01     System  : SC74
 User change date : 2011/116     Time  . : 15:07:01     User ID : CESAR

Volume use count . : 2             Volume usage (KB)  . . . . : 2630144
Capacity (MB)  . . : 286102        Percent full . . . . . . . : 1
Create date  . . . : 2011/116      Create time  . . . . . . . : 14:23:36
                                   System ID  . . . . . . . . : SC74
Date last read . . : 2011/116      Date last written  . . . . : 2011/116
Drive last used  . : 0B02          Write Mount count  . . . . : 2

Volume sequence  . : 1             Number of data sets  . . . : 1
                                   Data set recording . . . . : ON
Errors:
  Temporary read . : 0             Temporary write  . . . . . : 1
  Permanent read . : 0             Permanent write  . . . . . : 0
```

Figure 8-25 Volume Details

8.10 VRS last reference date

Over time the volume of vital record specifications can grow to a vast number, especially for VRSes that are no longer used. Now with DFSMSrmm V1R13, users can list a VRS based on the last reference date.

The SEARCHVRS subcommand is used to create a list of vital record specifications. Figure 8-26 shows an overview of the SEARCHVRS subcommand with the two new operands, LASTREFDATE and LASTCHANGAEDATE. Use the start_date operand to specify a date range for your search.

A date_range consists of a start date and an end date. Each date can be an absolute date in either yyyy/ddd or yyddd format, or it can be a relative value from which DFSMSrmm calculates the date.

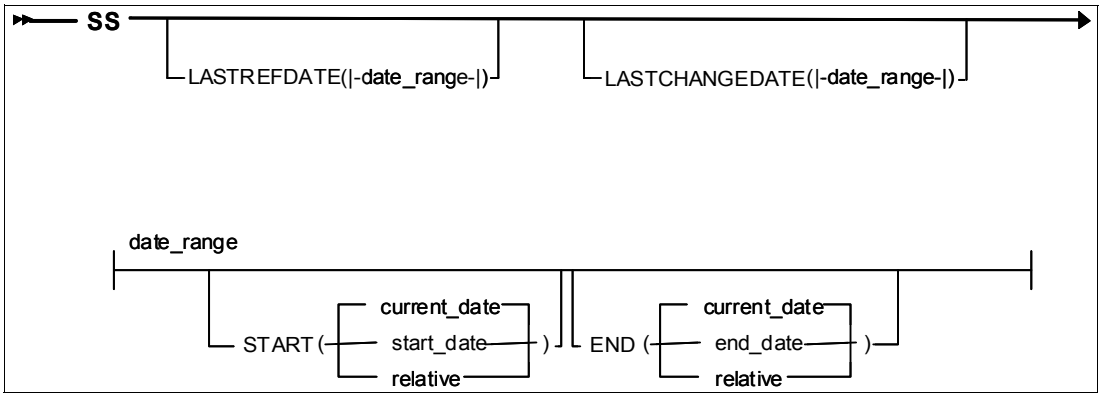


Figure 8-26 SEARCHVRS subcommand

Where:

LASTREFDATE This is the last reference date of the vital record specification record means the date of the last VRSEL run, when this VRS was used to retain a data set or volume.

LASTCHANGEDATE This is the last change date of the vital record specification record means the date, when this VRS was created or updated the last time.

START(start_date) You can optionally specify a starting date. If you omit the date, the current date is used. The search is performed inclusive of the start date.

END(end_date) You can optionally specify an ending date. If you omit the date, the current date is used. The search is performed inclusive of the end date.

Note: The dates are considered as local dates and not subject to time zone conversion. A relative value is specified as a negative number of days, months,

or years.

- 0 - means the current day, current month, current year.
- n - means the date is n days before the current date
- nM - means the date is n months before the current month and the current day in the month is as the current date. <internal level detail: For start date, the day in the month is adjusted down until a valid date is obtained. This handles, for example, 31->30, 30->28 etc. For end date, the day is adjusted down or up where the current day in month is the last day. This handles 28-> 31, 31->30 etc.>
- nY - means the date is n years before the current year and the current day in the year is as the current date. <internal level detail: For start date, the day in the year is adjusted down until a valid date is obtained. This handles, for example, 366->365. For end date, where current date 365/366 is last day in current year, we adjust up or down to allow for leap year 366/365.>

The value range for relative values is 0 to 99999, with a required leading dash (-) and an optional suffix of M|Y.

RMM ISPF panel updates

Figure 8-27 shows the new fields for the Display Data Set VRS panel. The fields will show details of the most recent change to the record.

```
  Date             - Displays the last change date
  Time             - Displays the last change time
  System           - Displays the ID of the system on which the last change
                     occurred
  User change date - Displays the date of the most recent change by a user,
                     other than by a DFSMSrmm internal function
  User change time - Displays the time of the most recent change by a user,
                     other than by a DFSMSrmm internal function
  User ID          - Displays the ID of the user who caused the most recent
                     change
```

Figure 8-27 New fields for the Display Data Set VRS panel

If the most recent change was made by DFSMSrmm processing, the ID starts with an asterisk (*). Internal IDs include these values:

- *OAM - DFSMSrmm system managed tape support
- *HKP - Inventory management
- *OCE - DFSMSrmm OPEN/CLOSE EOV support

```
EDGPV110               DFSMSrmm Display Data Set VRS              End of data
Command ===>


Data set mask . : 'BACKUP.BH5ST3'                                 GDG . : NO
     Job name mask . :

Count . . . : 99999                Retention type  . . . . . : CYCLES
                                   While cataloged . . . . . : NO
Delay . . . : 0   Days             Until expired . . . . . . : NO

Location  . . . . . : HOME
Number in location  : 99999
Priority  . . . . . : 0
                                   Release options:
Next VRS in chain . :              Expiry date ignore  . . . : NO
    Chain using . . :              Scratch immediate . . . . : NO

Owner . . . . . : CESAR
Description . . : TEST VRS
Last reference  : 2011/118  11:20:00         ( YYYY/DDD  HH:MM:SS )
Delete date . . : 1999/365    ( YYYY/DDD )

Last Change information:
 Date . . . . . : 2011/118       Time  . : 11:23:10     System  : SC74
 User change date : 2011/118     Time  . : 11:23:10     User ID : CESAR
```

Figure 8-28 Display Data Set VRS

Figure 8-29 shows the last change information field details of the most recent change to the record in the first part of the Search VRSs panel.

```
EDGPV010                    DFSMSrmm Search VRSs
Command ===>

Optionally specify one of:
Data set mask   *                                          GDG  . .
     Job name mask                                         ( Yes or No )
Volume serial                  Retention type
VRS name  . . .                While cataloged       ( Yes or blank for all)
                               Until expired         ( Yes or blank for all)
Location  . . . .              Release options:
Next VRS in chain              Expiry date ignore        ( Yes or No )
  Chain using . .              Scratch immediate         ( Yes or No )
Owner . . . . . *
Limit . . . . . *              Limit search to first n VRSs. Default is *
Dates           Start          End       Date, date range or relative value
  Reference . .             . .
  Changed . . .             . .
Clist . . . . . .              YES to create a data set, or NO, or blank

Enter HELP or PF1 for the list of available line commands
```

Figure 8-29 DFSMSrmm Search VRSs panel

Figure 8-30 shows the last change information field details of the most recent change to the record in the first part of the Search VRSs panel.

```
                DFSMSrmm Search VRSs (Page 4 of 4)          Row 1 to 7 of 7
Command ===>                                               Scroll ===> CSR

Enter HELP or PF1 for the list of available line commands.
Use the LEFT and RIGHT commands to view other data columns.
                                                     Store  Last
S  Volume/Data set/Name specification          Count number reference
-- ----------------------------------------- ----- ------ ----------
   *.**                                       99999 99999
   *.**                                       99999 99999
   **                                         99999 99999
   **                                         99999 99999
   BACKUP.BH5ST3                              99999 99999  2011/04/27
   TOTHSM.**                                  99999 99999  2009/05/19
   TOTHSM.**                                  99999 99999
```

Figure 8-30 DFSMSrmm Search VRSs panel

8.11 Display navigation enhancements

Previously there were no path command exits to display multivolume and multifile list. Now, in DFSMSrmm V1R13, there are two new primary commands, CHAINV and CHAIND, to display multivolume and multifile information.

Also implemented are 16 point-and-shoot fields on the volume panel and five point-and-shoot fields on the data set panel. To easily see the fields that are enabled for point-and-shoot, you must customize the color, intensity, and highlighting of the point-and-shoot fields.

Issue the ISPF system command **PSCOLOR** from any ISPF command line and adjust the Point-and-Shoot Panel Element as desired, as shown in Figure 8-31.

```
                          ISPF Settings
ISPOPT11              CUA Attribute Change Utility
Command ===>                                                  Defaults

Change colors, intensities, or highlights for panel attribute elements.
Enter the EXIT command to save changes or enter the CANCEL command to exit
without saving. To restore the defaults for a type, clear the field and
press Enter or select the Defaults point-and-shoot field to restore all
default settings for all types.

Panel Element                  Color         Intensity  Highlight
                                                            More:   - +
Point-and-Shoot  . . . . . . . RED           HIGH       NONE
PD Available Choices . . . . . WHITE         LOW        NONE
PD Unavailable Choices . . . . BLUE          LOW        NONE
Reference Phrase . . . . . . . WHITE         HIGH       NONE
Scroll Information . . . . . . WHITE         HIGH       NONE
Sel. Available Choices . . . . WHITE         LOW        NONE
Sel. Unavailable Choices . . . BLUE          LOW        NONE
Variable Output Info.  . . . . TURQ          LOW        NONE
Warning Message Text . . . . . YELLOW        HIGH       NONE
 F1=Help        F2=Split       F3=Exit        F7=Backward    F8=Forward
 F9=Swap       F12=Cancel
```

Figure 8-31 ISPF settings for PSCOLOR command

Using the CHAINVOLUME command

To use the **CHAINVOLUME** command, first display the DFSMSrmm Volume Details panel as shown in Figure 8-32. Next, issue the **CHAINVOLUME** command on the command line. (**CHAINVOLUME** can be abbreviated **CHAINV**.) The command can be provided on the Display Data Set Details or on the Display Volume Details panel, and it shows the search result list for all volumes of the multivolume set.

```
EDGPT110             DFSMSrmm Volume Details - THM000             Top of data
Command ===> chainv

Volume . . . . . : THM000     VOL1 volser :         Rack number : THM000
Media name . . . . : MEDIA5                         Status  . . : MASTER
                                                                  More:     +
Volume type  . . . : PHYSICAL       Stacked count  . . . . . . : 0
WorldWide ID . . . :                                Worm . . : NO
Retention date . . : CYCL/99999     Expiration date  . . . . . : 2010/123
Set retained . . . : NO               Set by . . . . . . . . . :
Hold . . . . . . . : NO             Original expiration date . :
Retention method . : VRSEL
-------- rest of screen lines not shown here----------------
```

Figure 8-32 DFSMSrmm Volume Details panel

Figure 8-33 shows the output of the **CHAINV** command.

```
EDGPT020          DFSMSrmm Volumes (Page 1 of 2)              Row 1 to 1 of 1
Command ===>                                                 Scroll ===> CSR

Enter HELP or PF1 for the list of available line commands
Use the RIGHT command to view other data columns
   Volume          Assigned   Expir./    S                  Dest-    Tr- Data
S  serial Owner    date       Retn. date R Status  Location ination  ans sets
-- ------ -------- ---------- ---------- - ------- -------- -------- --- -----
   THM000 STC      2010/123   CYCL/99999   VRS     ITSOTL1           N   60
-------- rest of screen lines not shown here----------------
```

Figure 8-33 Panel displayed by issuing the CHAINV command

Using the CHAIND command

To use the **CHAIND** command, first display the DFSMSrmm Data Set Details panel as shown in Figure 8-34. Next, issue the **CHAIND** command on the command line. The command can be issued on the Display Data Set Details or on the Display Volume Details panel.

```
EDGPD110                  DFSMSrmm Data Set Details                 Multi-File
Command ===> chaind

Data set name . . : 'TOTHSM.DMP.SYSPLEX1.V$DSNTA.D10123.T160023'
Volume serial . . : THM000      Physical file sequence number . . : 1
Owner . . . . . . : STC         Data set sequence number  . . . . : 1
                                                                More:     +
Job name  . . . . : DFSMSHSM
Step name . . . . : HSMSC47                Record format . . . . : U
Program name  . . : ARCCTL                 Block size  . . . . . : 65520
DD name . . . . . : SYS24437               Logical record length : 0
Create date . . . : 2010/123    YYYY/DDD   Block count . . . . . : 126482
Create time . . . : 23:00:45               Total block count . . : 126482
System id . . . . : SC47                   Data set size (KB)  . : 8092872
-------- rest of screen lines not shown here----------------
```

Figure 8-34 Issue CHAIND command

Figure 8-35 shows the output of the **CHAINDATASET** command. The command can be abbreviated to **CHAIND** or any other matching abbreviation and it returns the search result list for all data sets of the multivolume or multifile set.

```
EDGPD020        DFSMSrmm Data Sets (Page 1 of 2)            Row 1 to 20 of 60
 Command ===>                                                 Scroll ===> CSR


 Enter HELP or PF1 for the list of available line commands
 Use the RIGHT command to view other data columns
                                                    Volume          File
 S  Data set name                                   serial Owner    seq
 -- ------------------------------------------- ------ -------- -----
    TOTHSM.DMP.SYSPLEX1.V$DSNTA.D10123.T160023   THM000 STC      1
    TOTHSM.DMP.SYSPLEX1.V$DSNTB.D10123.T440423   THM000 STC      2
    TOTHSM.DMP.SYSPLEX1.V$DSNTC.D10123.T480523   THM000 STC      3
    TOTHSM.DMP.SYSPLEX1.V$DSNTD.D10123.T310723   THM000 STC      4
    TOTHSM.DMP.SYSPLEX1.V$DSN7C.D10123.T430823   THM000 STC      5
    TOTHSM.DMP.SYSPLEX1.V$DSN7D.D10123.T151023   THM000 STC      6
    TOTHSM.DMP.SYSPLEX1.V$DSN7E.D10123.T381023   THM000 STC      7
    TOTHSM.DMP.SYSPLEX1.V$DSN8A.D10123.T531023   THM000 STC      8
    TOTHSM.DMP.SYSPLEX1.V$DSN8B.D10123.T331123   THM000 STC      9
```

Figure 8-35 CHAIND command display

9

Establishing RACF security for RRSF TCP/IP connections

The RACF remote sharing facility (RRSF) allows you to easily manage RACF databases across the enterprise by propagating RACF TSO commands, application updates, and user password changes. It allows you to control what updates will be propagated and to where.

An RRSF node is an MVS system image or a group of MVS system images sharing a RACF database that has been defined as an RRSF node to RACF by a **TARGET** command. An MVS system image must meet the following requirements to be defined as an RRSF node:

- The RACF component of the z/OS Security Server is enabled.
- The RACF subsystem address space is active.

To direct commands or application updates from one MVS system image to another, or to synchronize passwords between two MVS system images, both system images must first be defined to RACF as RRSF nodes that can communicate with each other.

In this chapter the following topics are described:

- RRSF concepts
- Setting up RRSF with TCP/IP

9.1 RACF security for RRSF TCP/IP connections

An RRSF network is a collection of RRSF nodes in a network using APPC/MVS as the network transport mechanism. The RACF remote sharing facility (RRSF) currently exploits the networking services provided by APPC/MVS. Now, with z/OS V1R13, TCP/IP can be used to extend RACF functionality beyond the single host and shared DASD environments to a network of RRSF nodes capable of communicating with each other.

Using the function provided by RRSF, it is possible to administer RACF databases distributed throughout an enterprise from any location in the enterprise. Several implementation steps are required to enable your RRSF network to use TCP/IP node connections.

Note: Using TCP/IP connections for RRSF nodes provides benefits over APPC such as improved overall security, including the availability of stronger encryption levels. RRSF will support TCP/IP (IPV4 only) as an alternate transport protocol.

9.2 RRSF concepts

Several concepts are central to understanding the RACF remote sharing facility. When your programmer implements TCP/IP for RRSF node connections, you must issue RACF commands to enable RRSF to use TCP/IP node connections.

The RRSF network, the foundation of the RRSF environment, is the structure through which RRSF functions operate. An RRSF network is made up of RRSF nodes. An RRSF node is made up of one or more z/OS systems running with active RACF subsystems, as shown in Figure 9-1 on page 208.

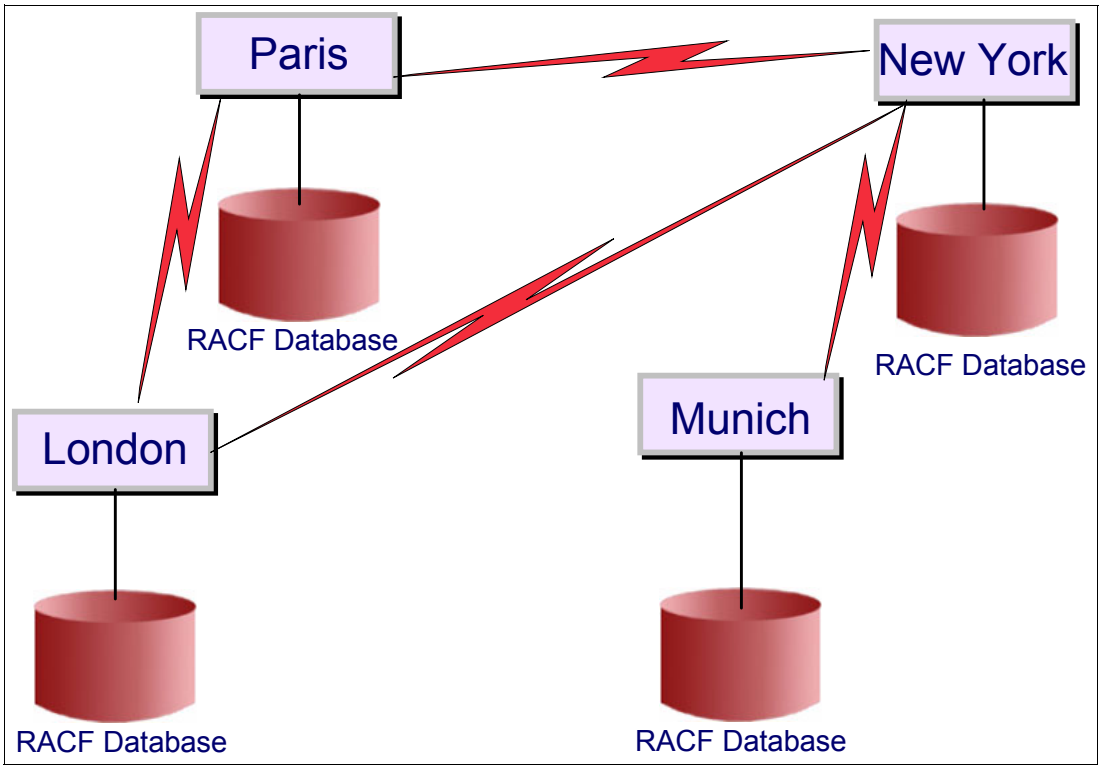


Figure 9-1 An RRSF network

In an RRSF network, the local node is the node whose viewpoint you are speaking from. Its remote nodes are the other nodes in the network that the local node communicates with.

RRSF nodes and the RRSF network

The RACF remote sharing facility is built on the concept of a network of RRSF nodes. As mentioned, an RRSF node is an MVS system image or several MVS system images sharing a RACF database that has been defined as an RRSF node to RACF. Before you can use the functions provided by RRSF, you must configure your MVS system images into a network of RRSF nodes.

RRSFLIST user data set

When you direct a command, the results are returned to you and are appended to the bottom of your RRSFLIST user data set. If you do not have a RRSFLIST user data set, RRSF allocates one and adds the results. The RRSFLIST user data set name is made up of the user's prefix as specified by the user through the TSO PROFILE command, the user ID, and RRSFLIST. When the prefix and the user ID are the same, the duplicate qualifier is dropped. Thus, the data set name would be either prefix.userid.RRSFLIST or userid.RRSFLIST.

RRSFDATA class

Profiles in the RRSFDATA class determine which RRSF functions are available on a node, and which users have access to them. The RRSFDATA class must be active for the functions it protects to be available.

User ID associations

Some RRSF functions require a previously established user ID association. A user ID association is an association between two different user IDs, on the same or different RRSF nodes. Typically user ID associations are established between user IDs used by the same person.

After the RRSF environment has been configured, users can establish associations (called user ID associations) between two RACF user IDs on nodes in the network. The user ID associations are used to:

- Link (associate) two user profiles on the same or different nodes
- Enable password synchronization to occur
  - Between a user's user IDs on the same node
  - Between a user's user IDs on different nodes
- Allow a user to manually direct most RACF commands to execute on other user IDs that the user's user ID has an appropriate association with

A RACF user ID can have multiple user ID associations. User ID associations can be set up by each user or by the security administrator using the **RACLINK** command. To enable the use of RACLINK, you must create a profile in the RRSFDATA class. After you create the profile, you can restrict it to a subset of users.

9.2.1 RRSF functions

Some users might need to use RRSF functions. RRSF provides the functions listed here.

Command direction

A user can issue a RACF command and direct that command to run under the authority of the same or another user ID on the same or another RRSF node. A user directs a command by

using the **AT** keyword on the command to specify the RRSF node and user ID that the command is to be directed to. The command runs asynchronously in the RACF subsystem address space, and the output is returned to the issuing user's RRSFLIST data set.

Password synchronization

If password synchronization is enabled between two different user IDs, then when the password or password phrase is changed for one of the user IDs, RACF automatically changes the password or password phrase for the other user ID.

Password synchronization is enabled between two user IDs by creating a user ID association between the two IDs that specifies password synchronization. Profiles in the RRSFDATA class control who may define user ID associations with password synchronization enabled, and whether password synchronization occurs on an RRSF node and for which users.

Automatic direction

Automatic direction allows you to have RACF automatically direct updates made to the RACF database on an RRSF node to one or more other RRSF nodes. If profiles on two or more RRSF nodes are already synchronized, you can use automatic direction to have RACF automatically keep the profiles synchronized.

RACF provides the following types of automatic direction:

- Automatic command direction, which broadcasts RACF operator commands issued at the local node to remote nodes.
- Automatic direction of application updates, which broadcasts updates at RACF made by an application.
- Automatic password direction, which does not require user ID associations. Instead, automatic password direction assumes that if the same user ID exists on two different nodes, those user IDs belong to the same person.

9.3 Setting up RRSF with TCP/IP

When implementing TCP/IP for RRSF node connections, you must issue RACF commands to allow TCP/IP communication to take place between RRSF nodes.

The steps to setting up a RRSF network using TCP/IP are listed here:

1. Changing the protocol for a connection from APPC/MVS to TCP/IP.
2. Defining the RRSFDATA class to RACF.
3. Creating an IRROPTxx parmlib member.
4. Setting up security for TCP/IP connections.

9.3.1 Changing the protocol for a connection

You can change the protocol that a connection uses. For example, if two nodes are communicating using APPC/MVS, you can change the protocol that the nodes use to TCP/IP. The procedure shown here for changing the protocol for a connection has the following characteristics:

- The original protocol continues to work until the new protocol successfully establishes a communications channel.

- No updates are lost during the conversion. Any requests that are queued in the workspace data sets for the original protocol are processed.
- The order of the requests is maintained.
- The conversion process can be restarted after an abend or the stop and restart of the subsystem address space without losing updates.
- The conversion process is bidirectional. That is, it can convert an APPC/MVS connection to TCP/IP, or a TCP/IP connection to APPC/MVS.
- During the conversion, there will be two sets of workspace data sets for the connection, one for the old protocol and one for the new protocol. Both sets are shown if you issue a **TARGET LIST** command during the conversion.
- The original node definition and its workspace data sets are deleted when the conversion process completes.

Setting up your RRSF network to use TCP/IP

To set up your RRSF network to use TCP/IPF, complete these tasks:

1. Protect the RRSF listener port.
2. Set up AT/TLS.
3. Prevent RACF from trying to establish connections before the AT-TLS policy is available.
4. Give the RACF subsystem address space access to the TCP/IP stack.
5. Allow the RACF subsystem address space to use z/OS UNIX socket APIs.

9.3.2 Defining the RRSFDATA class with RACF

With RRSFDATA, you define which functions are available to RRSF. Initially, the RRSFDATA class is not active, and no profiles are defined in the class. Therefore the RRSF functions controlled by the RRSFDATA class are not available to any users. You must define profiles for the functions you want to use, and activate the RRSFDATA class to make the functions available. See *z/OS Security Server RACF System Programmer's Guide*, SA22-7681, for a full description of the profiles in this class.

To support communications through TCP/IP, a new profile has been added in z/OS V1R13:

```
IRR.RRSF.CONNECT
```

This profile permits a connection to the local node when the AT-TLS rule covering the connection specifies a client authentication level by using an SAF check.

9.3.3 Creating an IRROPTxx parmlib member

The RRSF configuration is a combination of the RRSFDATA class and the IRROPTxx parmlib member. After you enable the RRSF functions in the RRSDATA class, you must define the nodes and activate the functions using RACF **TARGET** and **SET** commands. You put these commands in an IRROPTxx member of a PDS, allowing RACF to automatically configure the nodes during RACF address space initialization. For TCP/IP support there are new PROTOCOL keywords for the **TARGET** command:

```
PROTOCOL(
   TCP(
   ADDRESS(address)
   PORTNUM(number)
   )
```

For the ADDRESS keyword you can specify a hostname or an IP number and a port number for the PORTNUM keyword.

Here is an example of a **TARGET** command for a multisystem node:

```
TARGET NODE(PLEXBSB) SYSNAME(BSBA) PREFIX(BRS.RRSFB) -
      WORKSPACE(VOLUME(BD3X01) FILESIZE(5000)) -
      PROTOCOL(TCP(ADDRESS(rrsfbr.racf.test) PORTNUM(18136)) -
      DESCRIPTION('PLEXBSB BSBA LOCAL') -
      LOCAL OPERATIVE MAIN
```

Updating the RACF started task to include DD RACFPARM

The RACF started task reads the IRROPTxx member at initialization, finds the JCL for your RACF started task, and includes the DD RACFPARM:

```
//RACFPARM DD DSN=SYS1.PARMLIB,REGION=0M,PARM='OPT=$$'
```

In this example, RACF will look for a member named IRROPT$$ at the SYS1.PARMLIB data set.

9.3.4 Setting up security for TCP/IP communication

When your programmer implements TCP/IP for RRSF node connections, you must issue RACF commands to enable RRSF to use TCP/IP node connections.

z/OS Communications Server provides the TCP/IP networking protocol on z/OS. It also provides Application Transparent Transport Layer Security (AT-TLS), which allows client and server applications to communicate safely using TCP/IP.

RACF uses AT-TLS to provide authentication between RRSF nodes and to provide encryption of RRSF traffic. RACF does not allow RRSF nodes to connect unless the connection is protected by an AT-TLS rule enforcing client authentication.

z/OS Communications Server provides a sample AT-TLS policy for RRSF in its IBM Configuration Assistant for z/OS Communications Server, which you can download from the z/OS Communications Server web page at http://www.ibm.com/support/entry/portal/Overview/Software/Other_Software/z~OS_Communications_Server. For information about configuring TCP/IP, see *z/OS Communications Server: IP Configuration Guide*, SC31-8775, and *z/OS Communications Server: IP Configuration Reference*, SC31-8776.

9.3.5 Setting up the TCPIP environment

To set up your RRSF network to use TCP/IP, complete these tasks:

1. Protect the RRSF listener port.
2. Set up AT/TLS.
3. Prevent RACF from trying to establish connections before the AT-TLS policy is available.
4. Give the RACF subsystem address space access to the TCP/IP stack.
5. Allow the RACF subsystem address space to use z/OS UNIX socket APIs.

Protecting the RRSF listener port

Protect the RRSF listener port so that only the RACF subsystem address space has access to it.

A default port number of 18136 has been reserved with the Internet Assigned Numbers Authority (IANA) for the RRSF listener port. Define the RRSF port in the TCP/IP profile, and then assign it a name using the SAF keyword. For information about protecting ports, see the section on controlling access to particular ports in *z/OS Communications Server: IP Configuration Guide*, SC31-8775. For information about the **PORT** command, see *z/OS Communications Server: IP Configuration Reference*, SC31-8776.

A default port number of 18136 has been reserved with the Internet Assigned Numbers Authority (IANA) for the TCP/IP listener socket.

1. To assign the name RRSF to the listener port, edit the port definitions of the TCPIP started task:

   ```
   PORT 18136 TCP * SAF racfstc; Listener port for RACF Remote Sharing
   ```

   Here is an example:

   ```
   PORT 18136 TCP * SAF RACF; Listener port for RACF Remote Sharing
   ```

   To protect access to the port, the security administrator must create a profile in the SERVAUTH class:

   ```
   RDEFINE SERVAUTH EZB.PORTACCESS.nodename.tcpstc.racfstc
   ```

   Add the user ID of the RACF subsystem to the access list for the RRSF listener port:

   ```
   PERMIT EZB.PORTACCESS.nodename.tcpstc.racfstc CLASS(SERVAUTH) ID(racfuid)
   ACCESS(READ)
   ```

2. Determine whether access to the TCP/IP stack is protected. If it is, the following resource is protected in the SERVAUTH class:

   ```
   EZB.STACKACCESS.nodename.tcpstc
   ```

3. If the TCP/IP stack is not protected, skip this step. If it is protected, add the user ID of the RACF subsystem to the access list for the TCP/IP stack:

   ```
   PERMIT EZB.STACKACCESS.nodename.tcpstc CLASS(SERVAUTH) ID(racfuid)
   ACCESS(READ)
   ```

4. Activate your SERVAUTH profile changes:

   ```
   SETROPTS RACLIST(SERVAUTH) REFRESH
   ```

Implementing an RRSF trust policy

You must implement a trust policy based on digital certificates to allow TCP/IP communication to take place between RRSF nodes. RRSF node connections that use TCP/IP are protected using Application Transparent Transport Layer Security (AT-TLS). This trust policy is based on the requirements of AT-TLS. It requires that you create a RACF key ring for each node and one or more signed server certificates.

In z/OS V1R13, you can implement a trust policy using an internal CA (Certificate Authority), external CA or the same self-signed certificate for all RRSF nodes. See *z/OS Security Server RACF Security Administrator's Guide*, SA22-7683, to determine the best method for your enterprise.

This section explains the steps to follow to implement a trust policy using the same self-signed certificate for all RRSF nodes. To learn about other ways to implement a trust policy, see *z/OS Security Server RACF Security Administrator's Guide*, SA22-7683.

The examples in these steps use the same user ID (RACFSUB) for the RACF subsystem on each RRSF node. In general, security administration is simplified when the subsystem user IDs are kept consistent across a network.

Using the same self-signed certificate for all RRSF nodes

In this approach, you implement an RRSF trust policy for TCP/IP node connections by creating one self-signed certificate that you export and send to each TCP/IP node in your RRSF network.

For each node, you also create a key ring to hold only the node's server certificate. For a multisystem node, a single server certificate and key ring are shared among the member systems.

Creating the same self-signed certificate for all RRSF nodes

To create one self-signed certificate and send an exported copy of it to each RRSF node, complete these steps:

1. Choose a node in your RRSF network and create a self-signed certificate.

   ```
   RACDCERT GENCERT
   WITHLABEL('RRSF Server')
   SUBJECTSDN(CN('RACF Address Space') O('YOURORG') C('US'))
   KEYUSAGE(HANDSHAKE)
   NOTAFTER(DATE(2016-09-01))
   ```

   Do *not* specify the PKDS, PCICC, or ICSF option. The private key in this step must be stored in the RACF database so that it can be exported with the certificate in Step 2.

2. Export the certificate as a PKCS #12 package.

   ```
   RACDCERT EXPORT(LABEL('RRSF Server'))
   DSN(RACFSUB.PK12DER)
   FORMAT(PKCS12DER)
   PASSWORD('The circus is coming 2 town')
   ```

3. Using FTP in binary mode, transfer the export package from the local node to a data set on each remote TCP/IP node in your RRSF network. For a multisystem node, transfer the package to only one of the member systems.

4. (Optional) On the local node, move the private key from the RACF database to the ICSF PKA key data set (PKDS), if available, where it will have hardware protection. If your installation controls resources in the CSFSERV and CSFKEYS classes, ensure that the user ID of the RACF subsystem has sufficient authority even if the user ID has the TRUSTED attribute.

   a. Delete the self-signed certificate you created in Step 1.

      ```
      RACDCERT DELETE(LABEL('RRSF Server'))
      ```

      Re-add the self-signed certificate, using the export package you created in Step 2, and then store the private key in the ICSF PKDS.

      ```
      RACDCERT ADD(RACFSUB.PK12DER) ID(RACFSUB)
      TRUST
      WITHLABEL('RRSF Server')
      PASSWORD('The circus is coming 2 town.')
      PKDS(RRSFserverkey)
      ```

5. (Optional) Delete the data set containing the export package because it is no longer needed. If you opted to leave the private key in the RACF database, you can delete the export package. If you want to add a new TCP/IP node in the future, you can reuse the same RRSF server certificate by exporting it, as you did in Step 2, and transferring it to the new node.

   If you opt in Step 4 to move the private key to the ICSF PKDS, do not delete the export package when you want to use the same RRSF server certificate with any new TCP/IP

node that you might add in the future. If you delete the export package, you will need to create and distribute a new self-signed server certificate (with a different subject's distinguished name) for a new TCP/IP node.

6. On the local node, create a RACF key ring for RRSF and add the server certificate to the ring.

   a. Create the RRSF key ring.

      ```
      RACDCERT ID(RACFSUB) ADDRING(IRR.RRSF.KEYRING)
      ```

      Specify the key ring name provided by the programmer in “Before you begin” on page 457.

   b. Connect the server certificate to the key ring.

      ```
      RACDCERT ID(RACFSUB) CONNECT(LABEL('RRSF Server')
      RING(IRR.RRSF.KEYRING)
      DEFAULT
      USAGE(PERSONAL))
      ```

   c. Permit the user ID of RACF subsystem to access the key ring.

      ```
      RDEFINE FACILITY IRR.DIGTCERT.LISTRING UACC(NONE)
      PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY) ID(RACFSUB) ACCESS(READ)
      ```

      Note: Ensure that you complete this step even when the user ID of RACF subsystem has the TRUSTED or PRIVILEGED attribute on your system.

   d. Activate the profile change in the FACILITY class.

      ```
      SETROPTS RACLIST(FACILITY) REFRESH
      ```

7. On each remote RRSF node, add the self-signed certificate using the export package you transferred in Step 3.

   Note: These are the same steps that you perform for the local node in Steps 4b and 5b.

   a. Add the certificate and store the private key in the ICSF PKDS, if available.

      If your installation controls resources in the CSFSERV and CSFKEYS classes on the remote node, ensure that the user ID of the RACF subsystem has sufficient authority even if the user ID has the TRUSTED attribute.

      ```
      RACDCERT ADD(RACFSUB.PK12DER) ID(RACFSUB)
      TRUST
      WITHLABEL('RRSF Server')
      PASSWORD('The circus is coming 2 town')
      PKDS(RRSFserverkey)
      ```

   b. (Optional) Delete the data set containing the export package because it is no longer needed.

8. On each remote RRSF node, create a RACF key ring for RRSF and add the server certificate to the ring.

   Note: These are the same steps that you perform for the local node in Step 6.

   a. Create the RRSF key ring.

      ```
      RACDCERT ID(RACFSUB) ADDRING(IRR.RRSF.KEYRING)
      ```

      Specify the key ring name provided by the programmer in “Before you begin” on page 457.

b. Connect the server certificate to the key ring.

```
RACDCERT ID(RACFSUB) CONNECT(LABEL('RRSF Server')
RING(IRR.RRSF.KEYRING)
DEFAULT
USAGE(PERSONAL))
```

c. Permit the user ID of RACF subsystem to access the key ring.

```
RDEFINE FACILITY IRR.DIGTCERT.LISTRING UACC(NONE)
PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY) ID(RACFSUB) ACCESS(READ)
```

Note: Ensure that you perform this step even when the user ID of RACF subsystem has the TRUSTED or PRIVILEGED attribute on your system.

d. Activate the profile change in the FACILITY class.

If the FACILITY class is not already active, activate and RACLIST it.

```
SETROPTS CLASSACT(FACILITY) RACLIST(FACILITY)
```

If the FACILITY class is already active and RACLISTed, refresh it.

```
SETROPTS RACLIST(FACILITY) REFRESH
```

When you complete these steps, you will have created a key ring for each TCP/IP node and added its signed server certificate to the ring. You have now implemented an RRSF trust policy for TCP/IP node connections.

10

GRS enhancements

In a multitasking, multiprocessing environment, resource serialization is the technique used to coordinate access to resources that are used by more than one program. When multiple users share data, a way to control access to that data is needed. Users who update data, for example, need exclusive access to that. If several users try to update the same data at the same time, the result is a data integrity exposure with the possibility of incorrect or damaged data. In contrast, users who only read data can safely access the same data at the same time.

z/OS provides multiple ways of providing serialized access to data on a single or multiple systems, but global resource serialization is a fundamental way for programs to get the control they need and ensure the integrity of resources in a multisystem environment.

Because global resource serialization (GRS) is automatically part of your system and is present during z/OS initialization, it provides the application programming interfaces that are used by the applications on your system.

This chapter discusses the following GRS enhancements provided with z/OS V1R13:

- Latch Identity for the **D GRS,C** command
- New ISGQUERY request types
- New authorized QNAMEs
- Diagnostics for GQSCAN C,C
- IPCS enhancements

10.1 Latch Identity for D GRS,C

Identifying latches can be difficult because they are known by a hexadecimal string. Latches can now be given an identity that gives the latch a meaningful name. Prior to z/OS V1R13, the **D GRS,ANALYZE** command supported latch identity but **D GRS,C** and **D GRS,JOBNAME** commands did not.

New function has been adding for GRS in z/OS V1R13 to support latch identity in the **D GRS,C** and **D GRS,JOBNAME** commands. This provides improved diagnostics for latch contention problems.

10.1.1 Display GRS command syntax

A new keyword, LATCHID or LID has been added for **D GRS,C** and **D GRS,JOBNMAME** commands, as follows:

```
D GRS,C,LID
D GRS,JOBNAME=jobname,LID
```

Note: To identify latch ID information (displayed with the latch number in the output message ISG343I), specify LATCHID or LID. The latch ID can help with problem diagnosis when the latch creator has provided each latch number with a name.

When LATCHID or LID is specified, the output in message ISG343I contains the requested latch number and identity.

When you use **D GRS**.... you can also specify:

```
D GRS {,{CONTENTION|C}[,ENQ|,E][,{LATCH|L}[,{JOBNAME|JOB}=jobname]][,LATCHID|LID][,HEX]}
D GRS {,{LATCH|L}[,{JOBNAME|JOB}=jobname][,CONTENTION|,C]}[,LATCHID|LID][,HEX]
```

If LATCHID is specified but no latch identity is provided by the latch creator, the message displays the latch number followed by the information (`ID NOT SPECIFIED`). If LATCHID is specified and an available latch identity with a printable string exists, the message displays the identity string in one or more lines truncated after 255 characters. If truncation occurs, the character `T` follows the string at the end of the line.

If the latch identity information is not available, the message ISG375I is issued:

```
ISG375I THE LATCH IDENTITY TABLE IS NOT USABLE FOR CREATOR ASID: asid JOBNAME:
jobname LATCH SET NAME: lsetname.
```

10.1.2 Examples of D GRS output

The following examples shows the **D GRS** command displays when contention exists for a latch with a latch identity defined. The **D GRS,C** and **D GRS,C,LID** commands displayed in Figure 10-1 on page 219 show the output with and without the latch identify displayed.

```
D GRS,C * TC=GRJLLY06
ISG343I 13.32.20 GRS STATUS 666
NO ENQ RESOURCE CONTENTION EXISTS
LATCH SET NAME:  VAR040#SET1
CREATOR JOBNAME: GRALLY06  CREATOR ASID: 0027
  LATCH NUMBER:  0
    REQUESTOR  ASID  EXC/SHR    OWN/WAIT  WORKUNIT  TCB  ELAPSED TIME
    GRALLY06   0027  EXCLUSIVE  OWN       005FF110   Y   00:00:00.008
    GRALLY06   0027  EXCLUSIVE  WAIT      005E4AF8   Y   00:00:00.005
  LATCH NUMBER:  1
    REQUESTOR  ASID  EXC/SHR    OWN/WAIT  WORKUNIT  TCB  ELAPSED TIME
    GRALLY06   0027  EXCLUSIVE  OWN       005FF110   Y   00:00:00.008
    GRALLY06   0027  EXCLUSIVE  WAIT      005E4AF8   Y   00:00:00.005

D GRS,C,LID * TC=GRJLLY06
ISG343I 13.20.43 GRS STATUS 612
NO ENQ RESOURCE CONTENTION EXISTS
LATCH SET NAME:  VAR040#SET1
CREATOR JOBNAME: GRALLY06  CREATOR ASID: 0027
  LATCH ID:  0:GRJLLY06_SHORT_LID_VAR040
    REQUESTOR  ASID  EXC/SHR    OWN/WAIT  WORKUNIT  TCB  ELAPSED TIME
    GRALLY06   0027  EXCLUSIVE  OWN       005FF110   Y   00:00:00.026
    GRALLY06   0027  EXCLUSIVE  WAIT      005E4AF8   Y   00:00:00.022
  LATCH ID:  1:GRJLLY06_SHORT_LID_VAR040
    REQUESTOR  ASID  EXC/SHR    OWN/WAIT  WORKUNIT  TCB  ELAPSED TIME
    GRALLY06   0027  EXCLUSIVE  OWN       005FF110   Y   00:00:00.026
    GRALLY06   0027  EXCLUSIVE  WAIT      005E4AF8   Y   00:00:00.022
```

Figure 10-1 D GRS,C output

The **D GRS,ANALYZE** command output in Figure 10-2 on page 220 includes the latch identity and shows the WAITER and BLOCKER displays.

```
D GRS,ANALYZE,LATCH,WAITER,CJOBNAME=GRALLY06 * TC=GRJLLY06
ISG374I 13.20.44 GRS ANALYSIS 614
LONG WAITER ANALYSIS: CJOBNAME=GRALLY06
----- LONG WAITER #1
WAITTIME  JOBNAME  E/S  CASID LSETNAME/LATCHID
00:00:00  GRALLY06 *E*  0027  VAR040#SET1
                             0:GRJLLY06_SHORT_LID_VAR040
BLOCKER   GRALLY06  E
----- LONG WAITER #2
WAITTIME  JOBNAME  E/S  CASID LSETNAME/LATCHID
00:00:00  GRALLY06 *E*  0027  VAR040#SET1
                             1:GRJLLY06_SHORT_LID_VAR040
BLOCKER   GRALLY06  E

D GRS,ANALYZE,LATCH,BLOCKER,CJOBNAME=GRALLY06 * TC=GRJLLY06
ISG374I 13.20.44 GRS ANALYSIS 616
LONG BLOCKER ANALYSIS: CJOBNAME=GRALLY06
----- LONG BLOCKER #1
OWNTIME   JOBNAME  E/S  CASID LSETNAME/LATCHID
00:00:00  GRALLY06 *E*  0027  VAR040#SET1
                             0:GRJLLY06_SHORT_LID_VAR040
                             OTHER BLOCKERS: 0  WAITERS: 1
----- LONG BLOCKER #2
OWNTIME   JOBNAME  E/S  CASID LSETNAME/LATCHID
00:00:00  GRALLY06 *E*  0027  VAR040#SET1
                             1:GRJLLY06_SHORT_LID_VAR040
                             OTHER BLOCKERS: 0  WAITERS: 1
```

Figure 10-2 D GRS,ANALYZE output

The examples in Figure 10-3 show the **D GRS,C** and **D GRS,ANALYZE** command displays for a latch with a latch identity of greater than 255 characters defined. The latch identity is truncated at 255 characters and the truncation is denoted by the T at the end of the line.

Note: Resource contention information for the current global resource serialization complex is to be displayed. If a **DISPLAY GRS,CONTENTION** command is entered without the LATCH or ENQ operands, the system displays both ENQ and LATCH contention information.

As mentioned, specify LATCHID or LID when you want to identify latch ID information (displayed with the latch number in the output message ISG343I), because the latch ID can help you diagnose problems if the latch creator has provided each latch number with a name.

Specify HEX if you want resource names displayed in EBCDIC and in hexadecimal.

```
D GRS,C,LID * TC=GRJLLY06
ISG343I 13.36.12 GRS STATUS 724
NO ENQ RESOURCE CONTENTION EXISTS
LATCH SET NAME:  VAR087#SET1
CREATOR JOBNAME: GRALLY29  CREATOR ASID: 0027
  LATCH ID:  0:GRJLLY29-MYLIDSTRING256-123456789-123456789-1
     23456789-123456789-123456789-123456789-123456789-123456789-123
     456789-123456789-123456789-123456789-123456789-123456789-12345
     6789-123456789-123456789-123456789-123456789-123456789-1234567
     89-123456789-123456789-1                                      T
    REQUESTOR  ASID  EXC/SHR    OWN/WAIT  WORKUNIT  TCB  ELAPSED TIME
    GRALLY29   0027  EXCLUSIVE  OWN       005FF110   Y   00:00:00.027
    GRALLY29   0027  EXCLUSIVE  WAIT      005E4AF8   Y   00:00:00.025

D GRS,AN,LATCH,WAITER,CJOBNAME=GRALLY29,DETAIL * TC=GRJLLY29
ISG374I 13.36.13 GRS ANALYSIS 726
LONG WAITER ANALYSIS: CJOBNAME=GRALLY29
----- LONG WAITER #1
WAITER   JOBNAME: GRALLY29  (ASID=0027, TCB=005E4AF8)
         REQUEST: EXCLUSIVE                      LT:7F51407800000000
WAITING  00:00:00 FOR RESOURCE (CREATOR ASID=0027)
VAR087#SET1                                    LST:7F51660000000085
0:GRJLLY29-MYLIDSTRING256-123456789-123456789-123456789-123
  456789-123456789-123456789-123456789-123456789-123456789-123456789
  -123456789-123456789-123456789-123456789-123456789-123456789-12345
  6789-123456789-123456789-123456789-123456789-123456789-123456789-1 T
BLOCKER  JOBNAME: GRALLY29  (ASID=0027, TCB=005FF110)
         REQUEST: EXCLUSIVE                      LT:7F51401000000000
```

Figure 10-3 D GRS output for a latch with a latch identity of greater than 255 characters

10.2 New ISGQUERY request types

The ISGQUERY and GQSCAN macros enable you to obtain information about the status of each resource that is identified to global resource serialization, including information about the tasks that have requested the resource.

Using ISGQUERY and GQSCAN, you can inquire about a particular scope of resources (such as STEP, SYSTEM, or SYSTEMS), a specific resource by name, a specific system's resources, a specific address space's resources, or resources requested through the RESERVE macro.

The system collects the information that you request from the resource queues and consolidates that information before returning it. The authorized calls return a “snapshot” of the outstanding global resource serialization requests. The system serializes the resource queue so it does not change while the system gathers the information. (Authorized calls are those that specify either LOCAL or GLOBAL on the SCOPE parameter.)

The information returned by unauthorized calls, which do not perform serialization, might be inconsistent because of changes in the resource queue that can occur while the system collects the information. (Unauthorized calls are those that specify either STEP, SYSTEM, SYSTEMS, or ALL on the SCOPE parameter.)

The GRS query service routine is given control from the ISGQUERY macro to:

- Search a resource name list (RNL) for a given QNAME/RNAME pair
- Obtain information about resources and requesters of outstanding ENQ requests

In previous releases, there was no programmatic interface to enhanced contention analysis for latches or to gather overall GRS statistics.

z/OS V1R13 enhancement

New function in z/OS V1R13 for GRS provides two new request types in its ISGQUERY service. With exploitation in other products, this simplifies latch contention diagnosis as well as overall GRS usage monitoring.

Important: The following list describes the z/OS V1R13 enhancements to the ISGQUERY macro services:

- REQINFO=QSCAN

  Searches global resource serialization queues for resource and requester information.

- REQINFO=LATCHECA

  Searches global resource serialization queues for query latch enhanced contention analysis (ECA) data for waiters that might indicate contention issues.

  LATCHECA search does not return data for blockers or dependency data.

- REQINFO=USAGESTATS

  Search global resource serialization queues for address space-level contention information related to ENQs (all scopes) and latches (all latch sets). Global resource serialization gathers latch statistics in requester and latch set owner address space categories. Statistics are provided for all address spaces as described here:

  – ENQ by scope: This includes contention counts, total delay times, and the sum of the squared delay (SUMSQ) times. The SUMSQ times can be used to compute the standard deviation.

  – Latch: For both requesters and latch set owners, this includes contention counts, total delay times, and the sum of the squared delay (SUMSQ) times

  – ENQ usage counts. Note that latch counts are kept in “fast counts” in latch sets and not on an address space basis.

10.2.1 ISGQUERY REQINFO=LATCHECA

A new ISGQUERY, option REQINFO=LATCHECA, provides comparable functionality to the **D GRS,ANALYZE,WAITER** command.

The new syntax for ISGQUERY is:

```
ISGQUERY REQINFO=LATCHECA
,ANALYZE=WAITER
   ,ANSAREA=xansarea
   ,ANSLEN=xanslen
```

Attention: Callers who specify REQINFO=LATCHECA must not hold any FRRs.

- ,ANALYZE=WAITER

  When REQINFO=LATCHECA is specified, then ,ANALYZE=WAITER is a required output parameter that queries LATCHECA waiter data to determine if any long-term latch contention exists that might be cause for concern. ISGQUERY only returns LATCHECA data for waiters.

- ,ANSAREA=ansarea

  When REQINFO=LATCHECA is specified, then ,ANSAREA=ansarea is a required output parameter that is to contain the returned information. The area is mapped by the macro ISGYQUAA.

  A header area, mapped by DSECT ISGYQUAAHdr, is returned followed by additional data mapped by ISGYQUAALd and ISGYQUAALrd.

  To code: Specify the RS-type address, or address in register (2)-(12), of a character field.

- ,ANSLEN=anslen

  When REQINFO=LATCHECA is specified, then ,ANSLEN=anslen is a required input parameter that is the length of the answer area provided. The minimum size is the amount needed to describe a single resource with a single requester. Use an answer area length of at least 4 K.

  To code: Specify the RS-type address, or address in register (2)-(12), of a fullword field, or specify a literal decimal value.

This new function is being exploited by RunTime Diagnostics (RTD).

Note: The BLOCKER and DEPENDENCY options are not available at this time.

ISGYQUAC macro

The ISGYQUAC macro replaces ISGYQUAA due to an upward compatibility issue. ISGYQUAALd and ISGYQUAALrd data structures have been added. All future ISGQUERY enhancements will use the ISGYQUAC macro. Existing ISGQUERY invocations are unaffected.

The caller must include the ISGYQUAC macro to get a mapping for the answer area.

Note: The ISGYQUAA macro is stabilized as of z/OS V1R12.

10.2.2 ISGQUERY REQINFO=USAGESTATS

A new ISGQUERY option REQINFO=USAGESTATS provides overall statistics on GRS activity.

Note: For REQINFO=RNLSEARCH, the caller may be unlocked or hold both a local lock (LOCAL or CML) and the CMSEQDQ lock.

For all other REQINFO requests, the caller must not hold any locks.

The new syntax is:

```
ISGQUERY REQINFO=LATCHECA
   ,ANSAREA=xansarea
   ,ANSLEN=xanslen
```

This new function provides usage statistics for ENQ, Qscan, and latches for each address space. Aggregated statistics are also provided for terminated address spaces. It is being exploited by RMF.

ISGYQUAC is also used for this functionality. However, ISGYQUAAHdrUs is slightly different than other Qscan headers. ISGYQUAAUs used for each address space's data.

Important: The following list provides information about REQINFO=LATCHAREA and REQINFO=USAGESTATS:

- ,ANSAREA=ansarea

  When REQINFO=LATCHAREA is specified, then ,ANSAREA=ansareaa is a required output parameter that is to contain the returned information. The area is mapped by macro ISGYQUAA.

  A header area, mapped by DSECT ISGYQUAAHdr, is returned followed by additional data mapped by ISGYQUAARs, ISGYQUAARsx, ISGYQUAARq, and ISGYQUAARqx. Note that the ANSDETAIL specified determines which data is returned.

  To code: Specify the RS-type address, or address in register (2)-(12), of a character field.

- ,ANSAREA=ansarea

  When **REQINFO=USAGESTATS** is specified, then **,**ANSAREA=ansarea is a required output parameter that is to contain the returned information. The area is mapped by macro ISGYQUAA.

  A header area, mapped by DSECT ISGYQUAAHdrUs, is returned followed by additional data mapped by ISGYQUAAUs.

  To code: Specify the RS-type address, or address in register (2)-(12), of a character field.

10.3 New authorized QNAMEs

Authorized programs using unauthorized QNAMEs are vulnerable to denial-of-service, because any program can own the resource. DFSMShsm has been using several unauthorized QNAMEs.

With z/OS V1R13, a solution is to have GRS to provide a migration path to a larger authorized Qname list. This can improve the consistency of DFSMShsm.

GRS provides the following list of QNAMEs that only authorized callers may specify:

```
ADRDFRAG, ADRDSN, ARCENQG, BWODSN, SYSCTLG, SYSDSN, SYSIEA01, SYSIEECT,
SYSIEFSD, SYSIGGV1, SYSIGGV2, SYSPSWRD, SYSVSAM, SYSVTOC, SYSZ* (Where '*' is a
wildcard. For example, SYSZABC is an authorized  QNAME.)
```

Enqueue contention on critical DFSMShsm resources can cause system outages.

Beginning with z/OS V1R13, global resource serialization provides an additional list of qnames that are conditionally authorized:

```
ARCDSN, ARCBTAPE, ARCGPA, ARCBACV, ARCMIGV
```

Note: You can set the AUTHQLVL parameter in the GRSCNFxx parmlib member to indicate whether the system is to recognize the second list of authorized qnames in addition to the original list.

10.3.1 Managing authorized qname lists

To support these additional authorized QNAMEs you can set the AUTHQLVL parameter in the GRSCNFxx parmlib member to indicate whether the system is to recognize the DFSMShsm QNAMEs as authorized in addition to the original list of authorized QNAMEs.

The default is AUTHQLVL(1), which includes the original list of authorized QNAMEs. By setting AUTHQLVL(2), the following additional list of DFSMShsm QNAMEs will also be authorized:

```
ARCDSN, ARCBTAPE, ARCGPA, ARCBACV, ARCMIGV
```

The **D GRS,SYSTEM** command can be used to display the GRS AUTHQLVL parameter. The **D GRS,SYSTEM** command output displayed in Figure 10-4 shows the default AUTHQLVL of 1.

```
D GRS,SYSTEM
ISG343I 10.22.46 GRS STATUS 873
   SYSTEM    STATE               SYSTEM    STATE
SC74      CONNECTED           SC75      CONNECTED
GRS STAR MODE INFORMATION
  LOCK STRUCTURE (ISGLOCK) CONTAINS 1048576 LOCKS.
  THE CONTENTION NOTIFYING SYSTEM IS SC75
  SYNCHRES:      YES
  ENQMAXU:     16384
  ENQMAXA:    250000
  GRSQ:   CONTENTION
  AUTHQLVL:        1
```

Figure 10-4 D GRS,SYSTEM command output for default AUTHQLVL

Using AUTHQLVL

Because the default AUTHQLVL is 1, setting the AUTHQLVL to 2 will cause GRS to authorize the additional DFSMShsm QNAME list. This change can be made through a rolling IPL for the systems in the sysplex.

If you IPL your sysplex with AUTHQLVL=2, the AUTHQLVL can be backed out dynamically by using the following command:

```
SETGRS AUTHQLVL=1
```

The command displayed in Figure 10-5 on page 226 shows the output of the **D GRS,SYSTEM** command for a system IPLed with the GRSCNFxx parmlib member specifying AUTHQLVL(2). The AUTHQLVL is the changed back to 1 through a **SETGRS AUTHQLVL=1** command.

Attention: The **SETGRS AUTHQLVL=2** command cannot be used to dynamically change to using the DFSMShsm authorized QNAMEs. The **SETGRS AUTHQLVL** command is intended to be used for backout in case of ISV program issues; see Figure 10-5 on page 226.

Potential impact with ISV products

A number of ISV products have programs that issue ENQs for DFSMShsm QNAMEs. Programs that issue ENQs for the AUTHQLVL(2) list of DFSMShsm QNAMEs must be APF authorized. An unauthorized program that issues ISGENQ for an authorized qname will be abended as follows:

- For COND=YES, an abend ABEND040D is issued.
- For COND=NO,
  - ABEND338 is issued for an OBTAIN or CHANGE request.
  - ABEND330 is issued for a RELEASE request.

```
D GRS,SYSTEM
ISG343I 15.15.32 GRS STATUS
SYSTEM    STATE               SYSTEM    STATE
SC74      CONNECTED           SC75      CONNECTED
GRS STAR MODE INFORMATION
  LOCK STRUCTURE (ISGLOCK) CONTAINS 1048576 LOCKS.
  THE CONTENTION NOTIFYING SYSTEM IS SC75
  SYNCHRES:      YES
  ENQMAXU:     16384
  ENQMAXA:    250000
  GRSQ:   CONTENTION
  AUTHQLVL:        1

SETGRS AUTHQLVL=1
IEE712I SETGRS   PROCESSING COMPLETE

D GRS,SYSTEM
ISG343I 15.17.27 GRS STATUS
SYSTEM    STATE               SYSTEM    STATE
SC74      CONNECTED           SC75      CONNECTED
GRS STAR MODE INFORMATION
  LOCK STRUCTURE (ISGLOCK) CONTAINS 1048576 LOCKS.
  THE CONTENTION NOTIFYING SYSTEM IS SC75
  SYNCHRES:      YES
  ENQMAXU:     16384
  ENQMAXA:    250000
  GRSQ:   CONTENTION
  AUTHQLVL:        1

SETGRS AUTHQLVL=2
IEE708I AUTHQLVL KEYWORD, VALUE INVALID
```

Figure 10-5 D GRS,SYSTEM output displaying AUTHQLVL for a system IPLed with AUTHQLVL(1)

EQDQ monitor enhanced to aid migration

The EQDQ monitor has been enhanced to aid migration to AUTHQLVL(2). A new filter detects unauthorized callers of the authorized DFSMShsm QNAMEs.

```
REQTYPE=AUTHQ2
```

Important: Before you make the change to remove the unauthorized ENQ resource and leave the new authorized ENQ resource in place, you must require that all systems are using both ENQ resources. The EQDQ Monitor can filter on qname to help diagnose usage. Performing a rolling IPL should be sufficient for global requests.

New health check

A new heath check called GRS_AUTHQLVL_SETTING has been added to warn when AUTHQLVL(2) is not being used.

The display in Figure 10-6 on page 227 is the GRS_AUTHQLVL_SETTING health check display from a V1R13 system that is running with AUTHQLVL(1).

```
CHECK(IBMGRS,GRS_AUTHQLVL_SETTING)
START TIME: 04/18/2011 10:37:43.444578
CHECK DATE: 20100816  CHECK SEVERITY: LOW
CHECK PARM: 2

* Low Severity Exception *

ISGH0322E Global Resource Serialization AUTHQLVL is not set to the
maximum.

  Explanation:  The AUTHQLVL option allows an installation to specify
    what level of authorized QNAME protection to provide. When the value
    is not at the maximum, there is opportunity for authorized programs
    to be blocked by unauthorized programs when attempting to obtain
    resources that are protected at higher levels.

  System Action:  N/A
  Operator Response:  Contact your system programmer.
  System Programmer Response:  The AUTHQLVL option can be updated
    through the GRSCNFxx parmlib member.
  Problem Determination:  N/A
  Source:  IBM Global Resource Serialization
  Reference Documentation:
      z/OS MVS Planning: Global Resource Serialization
  Automation:  N/A
  Check Reason:  If the AUTHQLVL parameter is not set to the maximum
    level, certain requests may be susceptible to denial of service
    attacks.
  END TIME: 04/18/2011 10:37:43.445280  STATUS: EXCEPTION-LOW
```

Figure 10-6 GRS_AUTHQLVL_SETTING heath check output

10.4 Diagnostics for GQSCAN

The ISGQUERY and GQSCAN macros enable you to obtain information about the status of each resource that is identified to global resource serialization, including information about the tasks that have requested the resource. Use ISGQUERY or GQSCAN to obtain the status of resources and requesters of resources.

A GQSCAN failure that results in return code C, reason code C, is ambiguous because it can be the result of communication problems between systems, and also be the result of several internal errors.

An ISGQUERY QSCAN failure that results in return code 10, reason code 1005 has the same ambiguity issue.

New function in z/OS V1R13 now provides two new messages to clarify the problem when GQSCAN results in return code C, reason code C or ISGQUERY QSCAN results in return code 10, reason code 1005:

```
ISG377I GRS QSCAN INTERNAL ERROR, DIAG=diag
ISG378I GRS QSCAN ERROR COMMUNICATING WITH SYSTEM sysname, DIAG=diag
```

10.5 IPCS enhancements

The RESERVE macro serializes access to a resource (a data set on a shared DASD volume) by obtaining control of the volume on which the resource resides to prevent jobs on other systems from using any data set on the entire volume.

There is an ENQ associated with the volume RESERVE. This is required because a volume is reserved by the z/OS image and not the requester's unit of work. The ENQ is used to serialize across the requesters from the same system.

When diagnosing RESERVE contention problems, it is difficult to distinguish between a problem with the RESERVE and the underlying ENQ. The IPCS command VERBX GRSTRACE can be used to display the status of ENQ and RESERVE requests. It displays the time the ENQ was requested and the time the ENQ was granted. For RESERVE requests, there is no indication of whether or not the SYNCHRES option was used.

New function in z/OS V1R13 adds additional information in the GRSTRACE report to improve diagnostics for RESERVE problems:

- A time stamp is added for the Reserve's start of I/O.
- An indication is added for the SynchRes option.

10.5.1 Examples of GRSTRACE reports

This section provides examples from GRSTRACE reports from a z/OS V1R13 system. Figure 10-7 on page 228 shows the output for a RESERVE using the SYNCHRES option that has completed.

```
MAJOR NAME: SYSZJES2

    MINOR NAME: SPOOL1SYS1.CASE#1
      SCOPE: SYSTEM    SYSNAME: S1        STATUS: *EXCLUSIVE* /OWN
        ASID: 00000029   TCB: 004E6D90    JOBNAME: GRSTOOL
        Reserve Device: 027E Volser: TMPPAK - SYNCHRES COMPLETE
        Critical ENQ Time(s):
          Request:     07/21/2010 12:56:50.099689
          Started I/O: 07/21/2010 12:56:50.099716
          Grant:       07/21/2010 12:56:50.100263
```

Figure 10-7 GRSTRACE report output for a completed RESERVE with SYSNCHRES

Figure 10-8 on page 229 shows the output for a RESERVE without using the SYNCHRES option.

```
MAJOR NAME: SYSZJES2

    MINOR NAME: SPOOL1SYS1.CASE#2
      SCOPE: SYSTEM    SYSNAME: S1        STATUS: *EXCLUSIVE* /OWN
        ASID: 0000002A   TCB: 004E6D90    JOBNAME: GRSTOOL
        Reserve Device: 027D Volser: TMPPK1 - NOT SYNCHRES
        Critical ENQ Time(s):
          Request:     07/21/2010 13:02:03.623645
          Grant:       07/21/2010 13:02:03.623672
```

Figure 10-8 GRSTRACE report output for a completed RESERVE without SYSNCHRES

Figure 10-9 on page 229 shows the output for a RESERVE using SYNCHRES where the I/O has not completed.

```
MAJOR NAME: SYSZJES2

    MINOR NAME: SPOOL1SYS1.CASE#3
      SCOPE: SYSTEM    SYSNAME: S1        STATUS: *EXCLUSIVE* /OWN
        ASID: 00000029   TCB: 004E6D90    JOBNAME: GRSTOOL
        Reserve Device: 0182 Volser: LOWDSD - WAITING FOR SYNCHRES TO COMPLETE
        Critical ENQ Time(s):
          Request:     07/21/2010 12:58:26.940649
          Started I/O: 07/21/2010 12:58:26.940671
          Contention:  0, BUT IS WAITING
```

Figure 10-9 GRSTRACE report output for RESERVE with SYSNCHRES where the I/O has not completed

Figure 10-10 on page 229 shows the output for a reserve using SYNCHRES where there is ENQ contention.

```
MINOR NAME: SPOOL1SYS1.CASE#4
      SCOPE: SYSTEM    SYSNAME: S1        STATUS: *EXCLUSIVE* /OWN
        ASID: 0000001E   TCB: 004E6D90    JOBNAME: GRSTOOL
        Reserve Device: 027E Volser: TMPPAK - SYNCHRES COMPLETE
        Critical ENQ Time(s):
          Request:     07/21/2010 13:58:21.688497
          Started I/O: 07/21/2010 13:58:21.688733
          Grant:       07/21/2010 13:58:21.688734
      SCOPE: SYSTEM    SYSNAME: S1        STATUS: *EXCLUSIVE* /WAIT
        ASID: 0000002B   TCB: 004E6D90    JOBNAME: GRSTOOL
        Reserve Device: 027E Volser: TMPPAK
        Critical ENQ Time(s):
```

Figure 10-10 GRSTRACE report output for a ESERVE with SYSNCHRES where there is ENQ contention

11

BCP supervisor, contents supervisor, and RSM updates

This chapter discusses the BCP supervisor, contents supervisor enhancements, and RSM for z/OS V1R13. There are a number of minor changes to enhance usability and RAS:

- SSRB above 2 GB
- SRB enclave join and leave services
- Join/Leave for subtasks
- RMODE 64 Phase 1
- HiperDispatch = Yes for z196
- WARNUND system parameter
- RMTR control with local lock option
- Hardware Instrumentation Services miscellaneous update
- LLA use old suffix on restart
- CSVDLPAU message for RC=8
- Additional EXAA information
- 64-bit subspace support

11.1 SSRB above 2 GB

The use of common storage below 2 GB is increasing, partly due to the increasing use of SRBs that require SSRBs for suspended SRBs and preemptable SRBs. To reduce the use of common storage below 2 GB, most of the fields in the SSRB are moved to a new control block called the SSRX that resides in storage above 2 GB.

Although no interfaces are changed, there can be programs outside the BCP that reference the SSRB and thus need to change due to the movement of fields to the SSRX.

11.2 SRB enclave join and leave services

An *enclave* is an anchor for a transaction that can be spread across multiple dispatchable units in multiple address spaces. These multiple address spaces can even span across multiple systems in a Parallel Sysplex.

The value of using an enclave to represent a transaction is that the resources used to process the transaction can be accounted to the transaction itself, rather than to the address space or spaces that the transaction runs in. In addition, you can assign a performance goal to the enclave, which means that as a transaction consumes system resources, it can switch periods to run with a new goal.

An SRB is a control block that represents a routine that performs a particular function or service in a specified address space. An SRB is similar to a TCB in that it identifies a unit of work to the system.

Any number of tasks and SRBs can be grouped together in an enclave.

- Enclave SRBs offer the benefit that they are preemptable and will not tie up the system. SRBs in enclaves work well for higher volume, small requests, because SRBs have little overhead compared to tasks. The subsystem can create an enclave using the IWM4ECRE macro, and then schedule SRBs to run in the enclave using the IEAMSCHD macro.
- Tasks in enclaves automatically associate the enclave with the address spaces where they are dispatched, so workload management can manage the storage of those address spaces to meet the goal of the enclave. The enclave can perform functions that require a task environment, such as supervisor calls. Tasks can dynamically leave and join an enclave as they finish one piece of work and begin another. The subsystem creates an enclave using the IWM4ECRE macro, and then the task joins the enclave using the IWMEJOIN macro.

Note: A subsystem can create an enclave using the IWM4ECRE macro, join the task to the enclave using the IWMEJOIN macro, process the work request, and leave the task from the enclave using the IWMELEAV macro to remove the task. If a task joins an enclave and subsequently attaches subtasks, the subtasks are automatically joined to the enclave.

SRBs can join an enclave only upon a schedule. SRBs can leave an enclave only upon SRB termination or enclave termination.

However, there was no service that allowed an SRB that was already running to join or leave an enclave.

11.2.1 z/OS V1R13 enhancements - SRB enclave join and leave services

To implement various performance-sensitive code, DB2 has a requirement to be able to have a preemptabe SRB switch its enclave allegiance while the SRB is running. New function has been added for z/OS V1R13 to provide SRB enclave join/leave functionality for a preemptable SRB.

Invoking the SRB join and leave services

The new functionality can only be invoked by a direct call. No macro is provided for this function. The following fields in the extended communications vector table (ECVT) support this new function:

- The ECVT field ECVTSRBJ (offset x'90') contains the pointer-defined address for a “join”.
- The ECVT field ECVTSRBL (offset x'94') contains the pointer-defined address for a “leave”.

Note: If the field is 0 (zero), then the support is not present. It will never be the case that one of the fields is 0 and the other non-0 other than due to an overlay.

Performing SRB join through the ECVTSRBJ field

The environment requirements for the caller are listed here

- AMODE 31 or 64
- KEY 0, supervisor state
- Enabled for I/O and external interrupts
- Not holding any locks
- SRB mode (preemptable non-Client SRB only)
- Primary ASC mode
- Any P, Any S, Any H

Calling the SRB join service

To use this service, the following conditions are required:

- Set GR 1 to the below-2 G address of the 8-byte enclave token. Bits 0-31 of 64-bit GR 1 are ignored.
- Load the address in ECVTSRBJ into GR 15. Do not use the LLGT instruction. You do not need to set bits 0-31 of 64-bit GR 15.
- If AMODE 64, issue BASSM 14,15
- If AMODE 31, issue BASSM 14,15 or BASR 14,15

Returning from the join service

When returning to the caller of the service, the settings are:

- 31-bit GRs 2-13, high halves 2-14, and ARs 2-14 will be preserved.
- GR 15 contains the return code:
  - 0 = Join successfully completed.
  - 8 = Enclave token is not or is no longer valid
  - 12 = Work unit is already in an enclave
  - 16 = Non-preemptable SRB
  - 20 = Client SRB

Performing SRB leave through the ECVTSRBL field

The environment requirements for the caller are listed here:

- AMODE 31 or 64
- KEY 0, supervisor state
- Enabled for I/O and external interrupts
- Not holding any locks
- SRB mode (preemptable non-Client SRB only)
- Primary ASC mode
- Any P, Any S, Any H

Calling the SRB leave service

When using this service, set up the following conditions:

- Set GR 1 to the below-2 G address of the 8-byte enclave token. Bits 0-31 of 64-bit GR 1 are ignored.
- Load the address in ECVTSRBL into GR 15. Do not use the LLGT instruction. You do not need to set bits 0-31 of 64-bit GR 15.
- If AMODE 64, issue BASSM 14,15
- If AMODE 31, issue BASSM 14,15 or BASR 14,15

Returning from the SRB leave service

When returning from the service, the following conditions exist:

- 31-bit GRs 2-13, high halves 2-14, and ARs 2-14 will be preserved.
- GR 15 contains the return code:
  - 0 = Leave successfully completed.
  - 8 = Enclave token is not or is no longer valid
  - 12 = Work unit is not in an enclave
  - 16 = Work unit is not in the enclave identified by the input

11.2.2 Installation considerations and software requirements

The SRB join and leave services are included in z/OS V1R13. You need to apply APAR OA35146 for z/OS V1R10, z/OS V1R11, and z/OS V1R12.

Note: A program must test the ECVT field containing the entry point address of the service, and call only if that field is not 0 (zero).

DB2 will exploit the SRB join and leave services through APAR PM28626 for DB2 V8, V9, and V10.

A program must test the ECVT field containing the entry point address of the service, and call only if that field is not 0 (zero).

11.3 Join and leave services for subtasks

The enclave join and leave services are not flexible regarding how subtasks are treated, which can lead to performance problems for applications such as DB2. In z/OS V1R13, enclave join and leave processing now has a new option available to request different

behavior with respect to subtasks. This new function allows subtasks to implicitly join or leave an enclave.

The new functionality is invoked by WLM services IWMEJOIN and IWM4STBG through the new SUBTASKS keyword:

- IWMEJOIN SUBTASKS={NO | YES} and IWM4STBG SUBTASKS={NO | YES}
  - SUBTASKS=NO (the default) requests the “old behavior”
  - SUBTASKS=YES requests the “new behavior”

You can determine whether the support is available through SYSEVENT REQSRMST. A new bit SRMSTSTS within the IRASRMST macro indicates that the support is available.

Note: If SUBTASKS=YES is requested but the support is not available, the option is ignored and treated as though SUBTASKS=NO is in effect.

Behavior with SUBTASKS=NO

When a task joins an enclave, a subtask subsequently attached by that task is implicitly joined to that enclave. The “enclave root” is the task that did the join. Whenever a subtask is attached, it is given the same enclave root as the attaching task.

If that subtask had been attached before the join, the subtask does not join the enclave when the join occurs.

A task may not leave an enclave if the task has any subtasks that have the leaving task as its enclave root. This includes leaves done by IWM4STEN. It gets a return code or reason code of `8/xxxx0859`.

A task may leave an enclave only if the task's enclave root is itself. A root of “Not itself” means that the task was implicitly joined to that enclave. This includes leaves done by IWM4STEN.

Behavior with SUBTASKS=YES

When enclave join is done through IWMEJOIN or the join that is done as part of IWM4STBG, any existing subtask (not simply daughter tasks) that is not already joined to an enclave is implicitly joined to this enclave. It has its enclave root set to the joining task.

- SUBTASKS=YES is restricted to invocations of IWMEJOIN that have HASN=PASN.
- New return code 4 from IEAVJOIN if SUBTASKS=YES is seen on a system that supports it with HASN <> PASN, which is surfaced to the WLM caller as the return code or reason code of `4/xxxx044D`.
- If HASN<>PASN, the join works, but no subtasks are processed.

Note: If SUBTASKS=YES is specified on a system without the support, it is treated as SUBTASKS=NO with no new return information.

When an enclave leave is done through IWMELEAV or the leave that is done as part of IWM4STEN, then the subtasks will be removed from the enclave if the following conditions apply:

- The subtasks have the same enclave root as the leaving task, whether by join or attach processing.
- HASN=PASN
- The join for this task had specified SUBTASKS=YES.

Note: If no subtasks exist that have the same enclave root and HASN <> PASN and the join had specified SUBTASKS=YES, then you get return code 4 from IEAVLEAV, which is surfaced to the WLM caller as return code or reason code `4/xxxx044D`.

In all other cases, when there are subtasks with the same enclave root, you get the “cannot leave because subtasks exist” return code or reason code.

Software dependencies

The function is included in z/OS V1R13 and is rolled back to z/OS V1R11 and z/OS V1R12 through:

- WLM APAR OA33344 - PTFs UA58270 and UA58271
- Supervisor APAR OA33406 - PTFs UA58254 and UA58255

DB2 exploits this new function through APAR PM22154 for DB2 V9 (PTF UK63959) and DB2 V10 (PTF UK63958).

11.4 RMODE 64 Phase 1

DB2 imbeds code in its data and, for performance reasons, does not want to move that code below 2 GB before executing it. To meet this DB2 requirement, new function in z/OS V1R13 now supports executing code that resides above 2 GB.

Prior to z/OS V1R13, directed load of non-executable code above 2 GB is supported. With z/OS V1R13, this is extended to allow executable code to reside above 2 GB if that code does not invoke system services. It might get external or I/O interrupts or retryable machine checks or resolvable program interrupts and will have its status properly saved and later restored.

Restriction: Invoking system services is not supported for executable code residing above 2 GB. The service will most likely be invoked but there is no support to return to the caller.

11.4.1 Infrastructure changes to support 16 bytes PSW

Many services and formatting functions already support displaying data above 2 GB; others have been updated to support a 16-byte PSW.

- IPCS can already handle displaying data above 2 GB.
- SLIP can already handle ranges above 2 GB.
- GTF is extended to capture 16-byte PSW information.
- System trace is extended to capture 16-byte PSW information.

Key control structures

Prior to z/OS V1R13, the RBOPSW contains the 8-byte PSW for dispatch. In z/OS V1R13, the RBOPSW still contains that value, but it is no longer relied upon.

- XSBOPSW16 contains the official (16-byte) PSW.
- XSB_Orig_RBOPSW contains the original value of RBOPSW.

It is common for both IBM and ISV programs to modify RBOPSW. When the system detects that RBOPSW now differs from XSB_Orig_RBOPSW, the corresponding update is made to XSBOPSW16.

- An update to bytes 0-3 is copied directly.
- A “delta” update to the address is applied as a delta to the 8-byte address (this is defined as a change of plus-or-minus 6).
- A “replacement” of the address results in replacing the 8-byte address with the 4-byte replacement.

In macro CVT, a new field CVTBSM0F is defined. A recovery routine that wants to retry to an address above 2 G can set up 64-bit retry GPR 15 (such as by setting field SDWAG6415 and indicated RETREGS=64 on SETRP) with the pointer-defined address of the “real” retry point. At CVTBSM0F, BSM 0,15 is issued, so an address above 2 G must have bit 63 on so that the target will get control in AMODE 64.

In macro IHAEPIE, a new field EPIEPS16 is added. This is the 16-byte error PSW.

In the IEARBUP service, an extension is made to the PSWADDR parameter:

- The high 33 bits must be zero unless the result is to be AMODE 64.

Changes to message IEA995I

When the error address is below 2 G, there is no change. The PSW address line looks as shown here:

```
PSW AT TIME OF ERROR xxxxxxxx xxxxxxxx ILC x INTC xx
```

When the error address is above 2 G, that line is not presented. Instead two lines are used, as shown here:

```
PSW AT TIME OF ERROR xxxxxxxx xxxxxxxx xxxxxxxx xxxxxxxx
ILC x INTC xx
```

The full message now is as shown here:

```
IEA995I SYMPTOM DUMP OUTPUT{SYSTEM|USER} COMPLETION CODE=cde [REASON
CODE=reason-code]TIME=hh.mm.ss SEQ=sssss CPU=cccc ASID=asid
PSW AT TIME OF ERROR xxxxxxxx xxxxxxxx xxxxxxxx xxxxxxxx
ILC x INTC xx
....
```

Changes to message IEA844I

Message IEE844I, which is issued when an ACTION=WAIT SLIP matches, is changed to display an 8-byte address where the PER event occurred and a 16-byte PSW; see Figure 11-1.

```
*IEE844W SLIP TRAP 0002 MATCHED.  ACTION=WAIT  TYPE=PER
 PER INFO:     4070 00000000_01A30900
 PSW:       44040000 80000000 00000000 01A30906
 CR 3-4:    80400001 00010001
 AR/GR 0: 00000000/00000000_00009000   1: 00000000/00000000_7F3E9000
       2: FFFFFFFF/FFFFFFFF_00000000   3: FFFFFFFF/FFFFFFFF_005EA33C
       4: FFFFFFFF/FFFFFFFF_005EA33C   5: FFFFFFFF/FFFFFFFF_00000000
       6: FFFFFFFF/FFFFFFFF_051E3570   7: FFFFFFFF/FFFFFFFF_7F3F13A5
       8: FFFFFFFF/00000000_7F3F03A6   9: FFFFFFFF/FFFFFFFF_7F3EF3A7
       A: FFFFFFFF/FFFFFFFF_7F3EE3A8   B: FFFFFFFF/FFFFFFFF_051E6467
       C: FFFFFFFF/FFFFFFFF_051E64F8   D: FFFFFFFF/00000000_005EA39C
       E: 00000000/00000000_851E6190   F: 00000000/00000000_0000E603
 RESTART THE SYSTEM TO CONTINUE
```

Figure 11-1 Message IEE844I

Changes to messages BLW004A and IEA500A

Messages BLW004A and IEA500A have been changed to display a 16-byte PSW; see Figure 11-2

```
BLW004A   RESTART INTERRUPT DURING {jobname stepname | UNKNOWN JOBNAME}
ASID=asid MODE=mode PSW=pppppppp pppppppp pppppppp pppppppp
SYSTEM NON-DISPATCHABILITY INDICATOR IS {ON|OFF}
[text]
REPLY ABEND TO ABEND INTERRUPTED PROGRAM,
      RESUME TO RESUME INTERRUPTED PROGRAM,
            REPAIR TO PERFORM REPAIR ACTIONS.

IEA500A   RESTART INTERRUPT DURING {jobname stepname|UNKNOWN JOBNAME}
ASID=asid MODE=mode PSW=pppppppp pppppppp pppppppp pppppppp
REPLY RESUME TO RESUME INTERRUPTED
PROGRAM REPLY ABEND TO ABEND INTERRUPTED PROGRAM
[PREVIOUS REPLY WAS INVALID, ENTER A VALID REPLY]
```

Figure 11-2 Messages BLW004A and IEA500A

Changes to messages IEE854I

The current format is split across two lines to display the 16-byte PSW. The following two lines are changed:

```
PSW AT TIME OF ERROR=070C0000 80000002 ILC=2 INT=01
TRANSLATION EXCEPTION ADDR=7F1EF000
```

The changes to the first line are that it is split into two and extended for a 16-byte PSW. The second line is extended for an 8-byte address:

```
PSW AT TIME OF ERROR=07040000 8000000 00000000 0000002
ILC=2 INT=01
TRANSLATION EXCEPTION ADDR=00000000_7F1EF000
TRANSLATION EXCEPTION ADDR=7F1EF000
```

11.4.2 Changes to system trace records

Various system Trace records are changed to contain 16-byte PSWs. The formatted data will be updated. The physical records will be changed (incompatibly) as well. This applies to the following trace entries: Dispatcher, I/O, External interrupt, SVC, SSRV, Program check, Machine Check, Restart, and SPER/SPR2.

Dispatcher entries cover DSP, SRB, and SSRB. External interrupt entries cover CALL, CLKC, EMS, EXT, and SS. SVC entries cover SVC, SVCE, and SVCR.

For the changed records, wherever the 8-byte current PSW is present (field header such as PSW----- ADDRESS-), the output will contain the 8-byte address on the first line, with the two 4-byte parts separated by an underscore. Bytes 0-7 of the PSW will be underneath, on the second line. This format is used at the suggestion of Level 2 support.

Figure 11-3 shows examples of system trace records that have changed to use a 16-byte PSW.

```
PR ASID WU-ADDR- IDENT  CD/D PSW----- ADDRESS-  UNIQUE-1 ...
                                                UNIQUE-4 ...
00 0022 006E1BF8  SVC      1 00000000_082C21B0  00000000 ...
                               07141000 80000000
00 0022 006E1BF8  SVCR     1 00000000_082C21B0  806FF0C0 ...
                               07141000 80000000
00 000A 024E8F00  SRB        00000000_011ADA8E  0000000A ...
                               07040000 80000000  006F9758 ...
00 000A 006F9758  DSP        00000000_08119448  00000000 ...
                               07041000 80000000
00 000A 006FA0B0  EXT   1005 00000000_061EA096  00001005 ...
                               07040000 80000000
```

Figure 11-3 System trace records that have a 16-byte PSW

Changes to GTF trace records

GTF records and formatting are changed incompatibly. The 8-byte PSW fields are extended to 16 to contain the 16-byte PSW.

Figure 11-4 shows examples of GTF trace records that have changed to use a 16-byte PSW.

```
SVC..... 002      ASCB.... 00F55000 CPU..... 0000     JOBNAME. JES2
                  OLD-PSW. 07140000 80000002 00000000 08175258
                  TCB..... 006FF368 MODN.... HASJES20 R15..... 0005A9F0
                  R0...... 00000000 R1...... 0005AAE0
              GMT-04/22/2011 13:04:35.263540  LOC-04/22/2011 13:04:35.263540

SVCR.... 002      ASCB.... 00F55000 CPU..... 0000     JOBN.... JES2
                  DSP-PSW. 07140000 80000002 00000000 08175258
                  TCB..... 006FF368 MODN.... HASJES20 R15..... 00FE00FF
                  R0...... 00000002 R1...... 00FE00FF
              GMT-04/22/2011 13:04:35.263543  LOC-04/22/2011 13:04:35.263543

DSP               ASCB.... 00F55000 CPU..... 0000     JOBN.... JES2
                  DSP-PSW. 07141000 80000000 00000000 0818127C
                  TCB..... 006D99C0 MODN.... HASPCKVR R15..... 00FE00FF
                  R0...... 00000002 R1...... 00FE00FF
              GMT-04/22/2011 13:04:35.263565  LOC-04/22/2011 13:04:35.263565

SRB               ASCB.... 00FCC800 CPU..... 0003     JOBN.... JES2MON
                  SRB-PSW. 07040000 80000000 00000000 011397FE
                  SRB..... 0273AA5C PARM.... 8273AA30
                  TYPE.... INITIAL DISPATCH OF SRB
              GMT-04/22/2011 13:04:35.271753  LOC-04/22/2011 13:04:35.271753

EXT..... 1202     ASCB.... 00FD5B00 CPU..... 0000     JOBN.... *MASTER*
                  OLD-PSW. 07060000 00001202 00000000 00000000
                  TCB..... 00000000 PARM.... 00000000 SIG-CPU. 0003
              GMT-04/22/2011 13:04:35.271777  LOC-04/22/2011 13:04:35.271777
```

Figure 11-4 GTF trace records that have a 16-byte PSW

Changes in IPCS

The data formatted by the following command has changed to show 8-byte addresses when needed:

```
SYSTRACE PERFDATA(SHOWTRC SIGCPU(time))
```

If the address is below 2 G, the existing line format will be used where the PSW area shows the following:

```
ffffffff aaaaaaaa
```

Where `ffffffff` is bytes 0-3 of the PSW, and `aaaaaaaa` is the address.

If the address is above 2 G, that line will show the following:

```
aaaaaaaa_aaaaaaaa
ffffffff gggggggg
```

Where the 8-byte address and a second line will be added that contains bytes 0-3 then 4-7 of the PSW aligned with the 8-byte address.

IPCS LIST PSW will return or display a 16-byte PSW:

```
PSW - Program status word
LIST 0154. HEADER LENGTH(X'10') STRUCTURE(Ihapswe)
+00000 00000154. 07041000 80000000 00000000 12492C52
```

SDUMP 4 K buffer

The SDUMP 4 K buffer will have the 16-byte PSW added at the end, which can be displayed in IPCS with the following command:

```
LIST 0 DOMAIN(SDUMPBUFFER) LENGTH(4096)
```

- Bytes x'FEC'-x'FEF' will have the EBCDIC character “P16” to identify the 16-byte PSW.
- Bytes X'FF0'-'xFFF' will have that 16-byte PSW.

For PER cases, the variable portion will add the 8-byte PER address to the (current) PER interrupt code.

Figure 11-5 displays an example of the SDUMP 4 K buffer.

```
LIST 00. DOMAIN(SDUMPBUFFER) LENGTH(X'1000') AREA
00000000. E3E8D7C5 00000004 C3D7E440 00000080 |TYPE....CPU ....|
00000010. D9C5C7E2 7F543808 007E9944 00000040 |REGS"....=r.... |
00000020. 00A5A8D8 00FDE798 7F5436A8 07A67A44 |.vyQ..Xq"..y.w:.|
00000030. 7F5BFA18 07A68968 00A5A90C 7E9EB6D0 |"$...wi..vz.=..}|
00000040. 7F5BFA18 007E9940 007E99AC 80FDE7E8 |"$...=r .=r...XY|
00000050. 007E996C D7E2E640 440C0000 8184B606 |.=r%PSW ....ad..|
00000060. D7C1E2C4 0001E2C1 E2C40001 C1D9E240 |PASD..SASD..ARS |
00000070 LENGTH(X'40')==>All bytes contain X'00'
000000B0. C7F6F4C8 00000000 00000000 00000000 |G64H............|
000000C0 LENGTH(X'30')==>All bytes contain X'00'
000000F0. 00000000 40700000 00000184 B6000000 |.... ......d....|
00000100 LENGTH(X'0EE0')==>All bytes contain X'00'
00000FE0. 00000000 00000000 00000000 D7F1F640 |............P16 |
00000FF0. 44040000 80000000 00000000 0184B606 |............d..|
```

Figure 11-5 SDUMP 4 K buffer

11.5 HiperDispatch=yes for the z196

In z/OS V1R13, HiperDispatch=yes is the default in the IEAOPTxx parmlib member when the system detects that it is running on a z196 server.

IBM expects all LPARs on a z196 with at least one vertical high (VH), that is LPARs with more than 1.5 CPU’s worth of share, to be running with HiperDispatch=yes. This is because the z196 gets more value from HiperDispatch because cache misses cost more CPU cycles.

Note: HiperDispatch = No in IEAOPTxx will still be honored when running on a z196.

The SUP_HiperDispatch health check has been enhanced to remind clients to migrate images running with HiperDispatch=No to HiperDispatch=Yes when a z196 is installed.

To prevent the health check from issuing a warning message when running on a z196 with HiperDispatch=No, the z196 machine type must be specified where HiperDispatch=No is expected. This is done by specifying a SUP_HiperDispatch parameter of:

```
HiperDispatch(NO),MachTypes(2817)
```

11.6 WARNUND system parameter

When new system parameters are introduced for a z/OS release, they cause an IPL of a prior release to fail. This causes problems for installations that maintain multiple systems at various release levels that share a common parmlib data set.

A new system parameter, WARNUND, is introduced with z/OS V1R13 so that the system will warn on finding an undefined system parameter rather than prompting for new system parameters.

After the WARNUND parameter has been found, it applies to all subsequent system parameter processing.

- If you want WARNUND processing for all IEASYSxx parmlib members, put WARNUND at the beginning of the IEASYS00 parmlib member. It will then cover all subsequent system parameters within the IEASYS00 parmlib member and any other IEASYSxx parmlib members processed.
- If you need WARNUND processing active for a specific parameter in an IEASYSxx parmlib member, you can place the WARNUND [parameter just before that parameter.
- If you want WARNUND processing for your reply to the IEA101A message, you must specify it within the reply text (earlier within that reply than any undefined parameter), for example:

```
"R 0,WARNUND,SYSP=01,NEWPARM=VALUE"
```

Note: The value of WARNUND is not provided in the IPA data area (mapped by IHAIPA), and the **DISPLAY IPLINFO,WARNUND** command is not supported.

The WARNUND keyword is available on z/OS V1R11 and z/OS V1R12 through APAR OA35929.

New message IEA660I

If an undefined keyword is encountered and its syntax is valid, the system warns by issuing new message IEA660I to inform, and then moves onward rather than prompting to respecify system parameters, as shown here:

```
IEA660I PROCESSING CONTINUES AFTER UNDEFINED KEYWORD DUE TO WARNUND
```

11.7 RMTR control with local lock option

The PURGEDQ macro allows a task to purge a particular SRB activity. When purging SRBs scheduled in an address space other than primary, using a resource manager termination routine (RMTR) is recommended if you need to know whether a particular SRB has been purged.

However, the RMTR might not run if a “not my fault” abend occurs during PURGEDQ processing. The specific problem that highlighted this problem area is that an asynchronous cross-memory (XMEM) post might not complete due to an abend in PURGEDQ processing that results in the RMTR not being invoked even though PURGEDQ tried to call it.

The window occurs between the time PURGEDQ decides to call the RMTR and the time the RMTR gets control. If a “not my fault” abend occurs in this window then the RMTR will not be invoked.

SRB resource manager termination routine local lock option

This window is addressed by a new option in the SRB schedule request, SRBRMTLL, which causes PURGEDQ to obtain the local lock when calling the RMTR. XMEM post SRBs are not scheduled specifying SRBRMTLL to ensure the RMTR is invoked during PURGEDQ processing.

11.8 Hardware Instrumentation Services miscellaneous update

This section discusses an update to Hardware Instrumentation Services (HIS). This change is part of an attempt to make the HIS output more stand-alone.

The machine type and model in now included in HIS output. This includes the SMF Type 113 Subtype 2 record and the .CNT z/OS UNIX System Services (USS) output file. The machine type and model is in the same format as returned from CSRSI service. This will reduce the need to collect data from multiple sources when collecting HIS data for analysis.

The following updates have been made in z/OS V1R13:

- Machine type set from SI11V1CPCType returned by CSRSI Service
- Machine model set from SI11V1CPCModel or SI11V1CPCModel1 returned by CSRSI Service
- Machine type and model will be EBCDIC zeroes when CSRI Service does not provide the appropriate information
- SMF Type 113 Subtype 2 contains:
  - SMF113_2_MachType Char(4)
  - SMF113_2_MachModel Char(16)
- The .CNT USS output file changes:
  - First line containing output version number is increased to indicate new line in output

    ```
    HIS019I  EVENT COUNTERS INFORMATION VERSION 3
    ```

  - New line after “STATECHANGE” line

    ```
    MODEL: tttt-mmmmmmmmmmmmmmmm”
    ```

11.9 LLA use old suffix on restart

Stopping and restarting without including the LLA=xx in the start command suffix results in LLA managing only the LNKLST. This can result in performance problems because critical program libraries are no longer being managed by LLA.

New function in z/OS V1R13 changes the default to apply the old suffix unless LLA=NONE is specified. To get the pre z/OS V1R13 behavior of “only LNKLST”, the start of LLA must specify LLA=NONE.

Note: In an abend situation where LLA is internally restarted, the previous LLA option is already used prior to z/OS V1R13.

11.10 CSVDLPAU message for RC=8

CSVDLPAU is the “Dynamic LPA Utility” and has the primary function of displaying modules added using dynamic LPA. It also secondarily can display modules added through active LPA (for example, MLPA and FLPA and device support), and pageable LPA (that is, the LPALST). It had been in the product but not documented until DOC APAR OA34525.

When the CSVDLPAU utility encounters a problem in the CSVINFO service, it does not identify what problem CSVINFO reported.

CSVDLPAU has been enhanced in z/OS V1R13 to issue an additional diagnostic message.

If CSVDLPAU encounters an unexpected bad return code from CSVINFO, it will write a message to the joblog:

```
CSVD002W CSVINFO WAS NOT SUCCESSFUL. RC=xx, RSN=yyyyyyyy
```

Note: This is a utility message; it is not described in System Messages.

11.11 Additional EXAA information

CSVDYNEX REQUEST=QUERY returns data mapped by macro CSVEXAA. However, dynamic exits exploiters cannot access various parameter settings.

In z/OS V1R13, the following additional information is now presented to indicate whether:

- The exit is of type Installation (ExaaeExitTypeInstallation) or type Program (ExaaeExitTypeProgram)
- LOADAPF=Yes was requested for this exit (ExaaeLoadAPFYes)
- PERSIST=JobstepTask was requested (ExaaePersistJobstepTask)
- PERSIST=AddressSpace was requested (ExaaePersistAddressSpace)
- PERSIST=IPL was requested (ExaaePersistIPL)
- The exit routine was added and specified the POS parameter, that value (Exaam2RequestedPos)

11.12 64-bit subspace support

CICS transactions run in subspace mode to achieve storage isolation. In subspace mode, all the storage below 2 GB that does not belong to another subspace is accessible but storage above 2 GB is not. This means that system services exploiting 64-bit private or shared storage cannot be called in subspace mode.

To access 64-bit private or shared storage, a CICS transaction program must switch from subspace mode to base space mode through the branch in subspace group (BSG) instruction. The use of the BSG command introduces performance degradation.

In z/OS V1R13, new function is introduced in RSM to allow a program running in subspace mode direct access to 64-bit private or shared storage. Storage isolation for 31-bit private storage ranges remains unchanged but there is no subspace storage isolation for high virtual storage and therefore, 64-bit virtual storage ranges cannot be assigned to a subspace.

This means that a program running in subspace mode can directly access high virtual storage without issuing a BSG instruction, which results in a significant performance benefit.

Note: This new function is rolled back to z/OS V1R12 through APAR OA34311 and PTF UA58696.

Installation considerations

Programs running in subspace mode can directly access any valid high virtual storage. A new RCE flag, RceSubspaceV64, is set when access to 64-bit storage is supported. The application program can test to see if 64-bit subspaces are supported through RceSubspaceV64.

When RceSubspaceV64 is set, IARV64 system services are available in subspace mode. The following IARV64 request types can be issued in subspace mode:

- GETSTOR
- DETACH
- PAGEOUT/PAGEIN
- DISCARDDATA
- CHANGEGUARD
- GETSHARED
- PAGEFIX/PAGEUNFIX
- PROTECT/UNPROTECT
- SHAREMEMOBJ
- CHANGEACCESS
- LIST
- GETCOMMON

Note: Unauthorized called can only issue request types GETSTOR, DETACH, PAGEOUT/PAGEIN, DISCARDDATA, and CHANGEGUARD.

12

Improved channel recovery

This chapter describes the I/O supervisor (IOS) improvements that have been made to I/O error recovery in z/OS V1R13. The new design tracks path-related errors automatically to remove failing paths from all affected devices for the affected control unit.

In this chapter, the following topics are described:

- Improved channel recovery
- Enabling the IOS channel recovery function
- Displaying the current IOS recovery options
- New messages about path-related errors
- Displaying the reason why the path is offline
- Restoring after the path-related error has been corrected

12.1 Improved channel recovery

In all recent z/OS releases, there have been indications that installations prefer z/OS to be more pro-active when various types of path-related errors occur, instead of removing the path from only the device that incurred it.

With z/OS V1R13, z/OS will track path-related errors at the control unit level and will, at a threshold point (number of failures in specific time interval), respond by removing the failing path from all devices in the control unit. z/OS will respond to flapping links conditions and dynamic pathing errors by removing the failing path from all devices in the control unit. This can have the following effects:

- Reduce the time for z/OS to recover from path-related errors, and automatically remove failing paths from all affected devices for the affected control unit, as shown in Figure 12-1.
- Recovery is performed for the entire scope of devices impacted by the failing resource, all at one time.

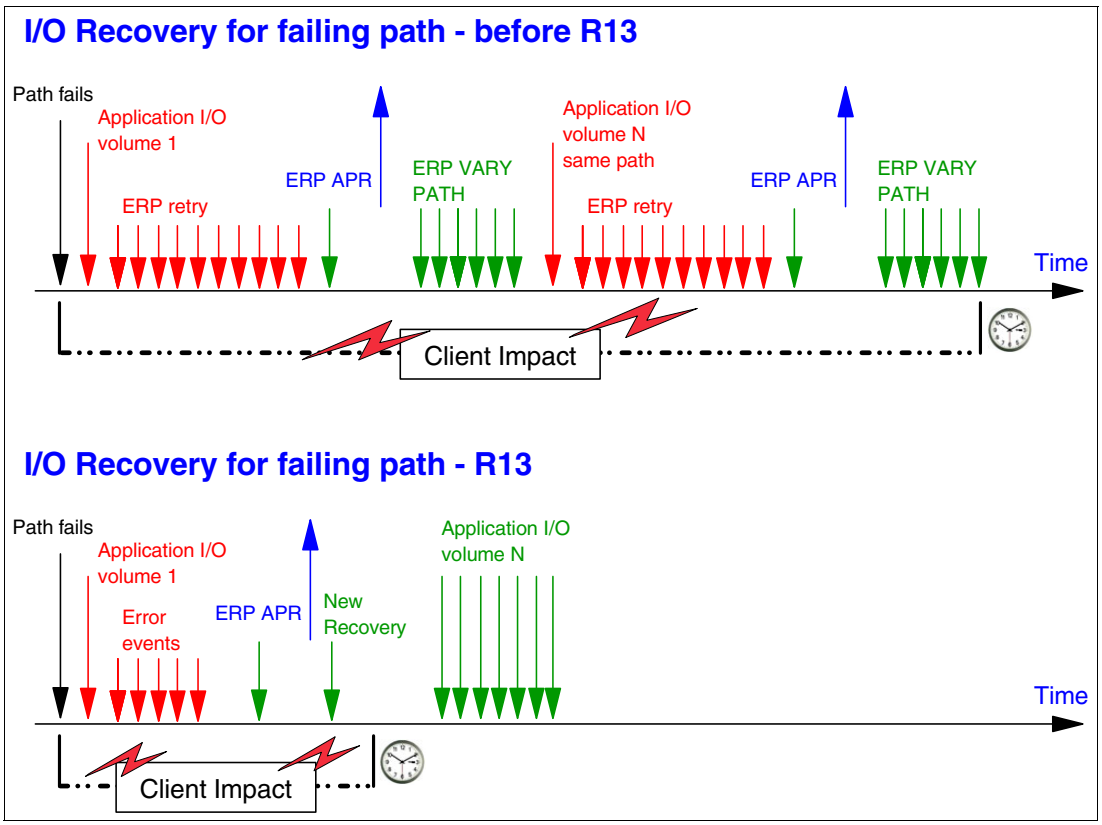


Figure 12-1 Path recovery before and after R13

The objective is to provide improved system resilience following various types of hardware failures, including fabric and control unit ports. This is done rather than the current processing, where each device detecting a path-related error independently begins a recovery process for the individual device, with each device running independently. IOS will track path-related errors at the control unit level and will, at a threshold point, respond by removing the failing path from all devices. Path-related error messages will identify the hardware component that detected the error.

12.2 Enabling the IOS channel recovery function

With z/OS V1R13, you can use the RECOVERY statement to enable or disable certain IOS RECOVERY options on a system. The following recovery options are available:

- Limited recovery time
- Path recovery

It is possible to enable the function using parmlib settings (IECIOSxx parmlib member) or operator commands (**SET IOS=xx** or **SETIOS RECOVERY**).

With the IECIOSxx parmlib member, the new options are for path recovery and you need to specify a PATH_SCOPE of either CU or DEVICE to enable path recovery.

Limited recovery time

A LIMITED_RECTIME between 2 and 14 seconds can be specified for either DASD or only devices that have the I/O timing facility enabled. When the limited recovery time function is enabled, IOS will use this value to time an IOS recovery I/O request to the qualified devices. If this I/O request suffers a start pending, a missing channel end or device end, or an interface timeout condition, the IOS recovery I/O request will not be retried on that device-path. To disable the limited recovery time function, specify:

```
LIMITED_RECTIME=0
```

With z/OS V1R13, when specifying a limited recovery time interval, consider the following situations.

- If the limited recovery time interval is longer than the I/O timing time interval for the device, an I/O time out condition is detected first and taking advantage of the limited recovery time function provides no value.
- If the limited recovery time interval is longer than the MIH time interval for the device, the MIH time out condition is detected first and taking advantage of the limited recovery time function provides no value.

Attention: With the specification of LIMITED_RECTIME, a Sense Path Group ID (SNID) CCW timeout on a device-path will be considered a permanent error, which will result in the path being taken offline. If the last path is taken offline, the device will be boxed.

Therefore, the LIMITED_RECTIME option is to be used with caution by installations that want to reduce the allowable time for a recovery I/O request, but are also able to handle device availability issues. Installations that consider exploiting the limited recovery time function are making a conscious trade-off between minimizing IOS recovery elapsed time and 100% availability of I/O resources.

Path recovery

With z/OS V1R13, when you define your I/O configuration, many devices share common hardware components (such as channels, channel cards, switches, control unit ports, control unit adapter cards, and fiber-optic links). So, for example, all devices for a specific control unit definition share the same hardware components because they share the same channels and control unit ports. Therefore, when a hardware-related error occurs on a channel path, multiple devices are affected.

As a result of an error, z/OS performs path recovery consisting of issuing one or more recovery-related I/Os to test the channel path to see if it is still usable. If path recovery

determines that the channel path is no longer usable, the path is removed (varied offline) from the affected device. Otherwise, the channel path remains online to the device.

Path recovery is typically performed one device at a time. This means that when an error occurs on one device, only that device is processed. Errors on other devices are processed independently, even if they share common hardware components. This can affect application performance because the application is delayed while z/OS performs path recovery and then retries the original I/O request. If the application uses multiple devices that share a failing hardware component, additional errors are encountered and further delays occur.

Additionally, certain types of path errors can be intermittent. That is, an error occurs, but path recovery is successful, so the path is not removed from the device. This also affects performance because applications can encounter errors multiple times. In many cases, when this occurs, you might need to manually remove the bad path or paths from the affected devices to stop the errors from occurring.

12.2.1 IECIOSxx parmlib member specifications

The new IECIOSxx parmlib member specifications allow you to reduce the elapsed time it takes for the system to recover from channel path-related errors, and help prevent system performance problems that can occur when a significant amount of time is spent in repetitive channel path error recovery. These specifications are:

- The PATH_SCOPE option
- The PATH_THRESHOLD option
- The PATH_INTERVAL option

The RECOVERY statement parameters are listed here:

PATH_SCOPE={CU | DEVICE} This specifies whether the scope of the path recovery is for all devices attached to the control unit (CU) or on a device-by-device basis (DEVICE).

Specify CU to enable the path recovery function for an LSS. When a path-related error occurs for a device that results in the path being taken offline, the path will also be taken offline to all devices in the LSS, unless that will remove the last path. In addition, IOS will monitor devices for path-related errors and take the path offline when the number of errors exceeds a threshold. Specify DEVICE to disable the path recovery function for an LSS. This keyword is independent of the LIMITED_RECTIME and DEV keywords. Changing this value will not affect actions previously taken.

Default: DEVICE

PATH_INTERVAL=nn This specifies the length of monitoring interval in minutes. Valid values are 1 through 10, where 10 is the default. This keyword can only be used when PATH_SCOPE has been set to CU. This keyword is independent of the LIMITED_RECTIME and DEV keywords. Changing this value will not affect actions previously taken.

PATH_THRESHOLD=nnn This specifies the number of errors that must be seen for each minute in the specified interval before IOS takes action. Valid values are 1 through 100, where 10 is the default. This keyword is independent of the

LIMITED_RECTIME and DEV keywords. Changing this value will not affect actions previously taken.

Default: DEVICE

Note: When using PATH_SCOPE=DEVICE, path recovery is on a device-by-device basis. If the error is associated with a hardware component shared by multiple devices (for example, channel, channel card, switch, control unit port, control unit adapter card, and so on), each device encounters the error and the path from the failing hardware component is removed, unless that will remove the last path.

This can affect application performance because the application is delayed while z/OS performs recovery and retries the request. If the application uses multiple devices that share the failing hardware component, additional errors are encountered and further delays occur.

PATH_INTERVAL=nn This specifies the length of time to monitor for errors in minutes. This keyword can only be used when PATH_SCOPE has been set to CU.

Value Range: 1-10 (minutes)
Default: 10

PATH_THRESHOLD=nnn This specifies the minimum number of path-related errors that must occur every minute for the specified period of time (PATH_INTERVAL) before IOS takes action. This keyword can only be used when PATH_SCOPE has been set to CU.

Value Range: 1-100 (errors)
Default: 10

Attention: PATH_INTERVAL and PATH_THRESHOLD are not valid keywords with PATH_SCOPE=DEVICE.

Do *not* set both the interval and threshold to low values (for example, 1) because this may cause z/OS to remove paths unnecessarily.

IECIOSxx parmlib member example

In this example, specifying a PATH_INTERVAL of 2 (minutes) and a PATH_THRESHOLD of 50 (errors), as shown in Figure 12-2, means that at least 50 errors must occur every minute for 2 consecutive minutes before the path from the failing hardware component is removed from all the affected devices.

```
 BROWSE    SYS1.PARMLIB(IECIOS51)
 Command ===>
******************************** Top of Data ************
RECOVERY PATH_SCOPE=CU,PATH_INTERVAL=2,PATH_THRESHOLD=50
******************************** Bottom of Data **********
```

Figure 12-2 Syntax example to enable path recovery through IECIOSxx

Important:

- As mentioned, setting both the interval and the threshold to low values (for example, 1) can cause IOS to remove paths unnecessarily.
- Changing the value of the interval and the threshold keywords will not take effect until the next error is encountered and will not affect actions previously taken.

12.2.2 IEASYSxx parmlib member

After having created the IECIOSxx parmlib member, you can update the IEASYSxx parmlib member and wait until the next IPL, or use the **SET IOS=xx** command to enable the function.

IOS=xx The two alphanumeric characters indicate the IECIOSxx member of the logical parmlib that contains the parameters that the system is to use to control MIH processing, I/O timing processing, and other IOS functions.

For example, if you have created a IECIOS51 parmlib member with your set of the RECOVERY statement parameters, now you can use the command **SET IOS=51** to activate the function as shown in Figure 12-3.

```
11:48:38.74 NARO     00000290  SET IOS=51
11:48:38.76          00000290  IEE252I MEMBER IECIOS51 FOUND IN SYS1.PARMLIB
11:48:38.76          00000090  IOS090I SETIOS. RECOVERY UPDATE(S) COMPLETE
11:48:38.76 NARO     00000090  IEE536I IOS      VALUE 51 NOW IN EFFECT
```

Figure 12-3 Using the SET IOS=xx command to enable path recovery

12.2.3 SETIOS RECOVERY command

In contrast to the **SET** command, which allows an installation to specify a different IECIOSxx parmlib member, the **SETIOS** command can dynamically enable or disable the IOS recovery function. The **SETIOS** command can dynamically change the missing interruption handler (MIH) or I/O timing (IOT) parameter. The parameters can appear in any order in the command, but there can only be one DEV and TIME parameter pair or DEV and IOTIMING pair in a command. You can create user classes for particular situations such as test environments and special job processing.

SETIOS RECOVERY command parameters

With z/OS V1R13, the new parameters are added to the **SETIOS RECOVERY** command to enable the function; see Figure 12-4.

```
SETIOS RECOVERY,PATH_SCOPE={CU|DEVICE}
               ,PATH_INTERVAL=nn
               ,PATH_THRESHOLD=nnn
```

Figure 12-4 Syntax for the SETIOS RECOVERY command

Where:

- RECOVERY,PATH_SCOPE={CU or DEVICE}

  This specifies whether the scope of the path recovery is for all devices attached to the control unit (CU) or on a device-by-device basis (DEVICE).

Note: Specify CU to enable the path recovery function for an LSS. When a path-related error occurs for a device that results in the path being taken offline, the path will also be taken offline to all devices in the LSS, unless that will remove the last path. In addition, IOS will monitor devices for path-related errors and take the path offline when the number of errors exceeds a threshold.

Specify DEVICE to disable the path recovery function for an LSS. This keyword is independent of the LIMITED_RECTIME and DEV keywords. Changing this value will not affect actions previously taken.

Default: DEVICE

- RECOVERY,PATH_INTERVAL=nn

  This specifies the length of time to monitor for channel path errors in minutes. This keyword can only be used when PATH_SCOPE=CU is specified.

  Note: This keyword is independent of the LIMITED_RECTIME and DEV keywords. Changing this value will not affect actions previously taken.

  Value Range: 1-10 (minutes)

  Default: 10

- RECOVERY,PATH_THRESHOLD=nnn

  This specifies the minimum number of channel path-related errors that must occur every minute for the specified period of time (PATH_INTERVAL) before IOS takes action. This keyword can only be used when PATH_SCOPE=CU is specified.

  Note: This keyword is independent of the LIMITED_RECTIME and DEV keywords. Changing this value will not affect actions previously taken.

  Value Range: 1-100 (errors)

  Default: 10

Attention: Remember, do not set both the interval and threshold to low values (for example, 1) because this can cause z/OS to remove paths unnecessarily.

To achieve the same results as the example shown in Figure 12-2 to enable the path recovery through the IECIOSxx parmlib member, issue a command as shown in Figure 12-5.

```
14:44:31.62  SETIOS RECOVERY,PATH_SCOPE=CU,PATH_INTERVAL=2,PATH_THRESHOLD=50
14:44:31.62  IOS090I SETIOS. RECOVERY UPDATE(S) COMPLETE
```

Figure 12-5 Using the SETIOS RECOVERY command to enable path recovery

Attention:

- Setting both the interval and the threshold to low values (for example: 1) can cause IOS to remove paths unnecessarily.
- Changing the value of the interval and the threshold keywords will not take effect until the next error is encountered and will not affect actions previously taken.

12.3 Displaying the current IOS recovery options

The path recovery options as shown in 12.2.1, “IECIOSxx parmlib member specifications” on page 250 can be enabled and changed dynamically. Therefore, it is necessary to have a way to know how the options are currently set.

Use the **D IOS,RECOVERY** command to display the current IOS recovery options, as shown in Figure 12-6.

```
D IOS,RECOVERY[,L={a|name|name-a}]
```

Figure 12-6 D IOS,RECOVERY command

IOS,RECOVERY The system displays status information about the IOS recovery function.

L=a, name, or name-a This specifies the display area (a), console name (name), or both (name-a) where the display is to appear.

If you omit this operand, the display is presented in the first available display area or the message area of the console through which you enter the command.

For example, if you issued a command as shown in Figure 12-5 and now you issue the **D IOS,RECOVERY** command, the result is shown in Figure 12-7.

```
09:56:33.38 NARO     00000290  D IOS,RECOVERY
09:56:33.39 NARO     00000090  IOS103I 09.56.33 RECOVERY OPTIONS 147
                  147 00000090  LIMITED RECOVERY FUNCTION IS DISABLED
                  147 00000090  PATH RECOVERY SCOPE IS BY CU
                  147 00000090  PATH RECOVERY INTERVAL IS 2 MINUTES
                  147 00000090  PATH RECOVERY THRESHOLD IS 50 ERRORS
```

Figure 12-7 Display iOS recovery options

12.4 New messages about path-related errors

When a path-related error occurs, clients have a difficult time determining where the problem resides (it can be in the switch card attached to the channel, the link between the channel and switch, and so on).

The following messages are now provided to identify the component that detected the error and show when the path recovery has been initiated.

IOS054I message

This new message is displayed to identify the hardware components, as follows:

```
IOS054I sdddd,chp ERRORS DETECTED BY component[, component]...
```

Explanation: This message is preceded by message IOS050I or IOS051I and identifies the hardware component or components that detected the error. This information can be used to isolate the faulty hardware component. The errors on device [s]dev and channel chp were detected by the components indicated.

In the message text:

sdddd This is the subchannel set identifier and device number.

chp The channel path identifier (CHPID), if known; otherwise, this field is set to asterisks.

component This is the detecting component, which can be one or more of the following:

- CHANNEL
- CHAN SWITCH PORT
- CU SWITCH PORT
- CONTROL UNIT

System action: The system writes a logrec data set error record.

Operator response: See the operator response for message IOS050I or IOS051I.

IOS210I message

This new message is displayed when a path-related error occurs with the following message:

```
IOS210I PATH RECOVERY INITIATED FOR PATH chp ON CU cccc, REASON=rsntext
```

Explanation: The system displays this message when a path recovery-related error occurs and the installation has specified PATH_SCOPE=CU option in the IECIOSxx parmlib member or through the **SETIOS** command. The system will attempt to vary the channel path offline for all devices on the control unit.

In the message text:

chp This is the channel path identifier (CHPID) that encountered the path recovery error.

cccc This is the control unit for the device that encountered the path recovery error.

rsntext This explains why the channel path is being varied offline, and is one of the following reasons:

`LINK RECOVERY THRESHOLD REACHED`

The hardware IBM FICON® link recovery threshold has been reached and the path is no longer available to all devices on the control unit. This message can be preceded by message IOS2001I or IOS2002I.

`PATH ERROR THRESHOLD REACHED`

The system has determined that the number of errors on a path over a period of time has reached an installation-specified threshold and the path needs to be taken offline to all devices on the control unit. This message can be preceded by message IOS050I or IOS051I.

`DYNAMIC PATHING ERROR`

The system encountered an error on a path while validating a dynamic pathing device. As a result of the error, the path was taken offline from the device. This message can be preceded by message IOS450E.

`REQUESTED BY DEVICE ERP ROUTINE`

The device support error recovery procedure routine requested that control unit path error recovery be performed.

System action: The system attempts to vary the path offline to all devices on the control unit. If the reason text does not indicate `LINK RECOVERY THRESHOLD REACHED`, then the path will not be taken offline to a particular device if it is the last path to the device and the device is online,

reserved, assigned, or in use by a system component. Otherwise, the path is not operational to all devices for the control unit, so the path will be taken offline.

Operator response: After the problem that caused the path-related errors to occur has been corrected, you must bring the path online manually; the path is not automatically varied back online by the system. To bring the path back online, issue one of the following commands: **VARY CU**, **VARY PATH**, **VARY device**, or **CONFIG CHP**. First vary the path online to a single device to ensure that the problem has been corrected before varying the path online to the remaining devices.

System programmer response: Correct the errors that caused the path to be taken offline and vary the path back online. See the associated error messages for guidance about how to identify and correct the error.

IOS167I message

The following message is issued when the channel subsystem reported that the subchannel for device dev has undergone recovery and has been restored to an operational state.

```
IOS167I DEVICE dev RESTORED BY SUBCHANNEL RECOVERY
```

In the message text:

dev This is the device number. The device number is prefixed by the subchannel set identifier when appropriate.

System action: None, as the system has processed the request from the channel subsystem and no further action is required.

12.5 Displaying the reason why the path is offline

Use the **DISPLAY M=DEV(dddd,(chp))** command to display the reasons why the path is offline:

- **PATH OFFLINE DUE TO THE FOLLOWING REASON(S):**
  - PATH RECOVERY ERROR
  - BY OPERATOR
  - CONTROL UNIT INITIATED RECOVERY
  - CONFIGURATION MANAGER

One or more reasons can be displayed. If none of these reasons exist, then the heading nor the reasons will be displayed.

An example of the path offline reason returned by the **DISPLAY M=DEV(dddd,(chp))** command is shown in Figure 12-8.

```
V PATH(D428,5C),OFFLINE
IEE303I PATH(D428,5C) OFFLINE
D M=DEV(D428,(5C))
IEE174I 11.45.03 DISPLAY M 076
DEVICE D428   STATUS=ONLINE
CHP                  5C
ENTRY LINK ADDRESS   2C
DEST LINK ADDRESS    0A
PATH ONLINE          N
CHP PHYSICALLY ONLINE Y
PATH OPERATIONAL     Y
MANAGED              N
CU NUMBER            D400
DESTINATION CU LOGICAL ADDRESS = 04
SCP CU ND         = 002107.922.IBM.75.0000000BALB1.0230
ACTUAL CU ND      = 002107.900.IBM.75.0000000BALB1.0301
SCP TOKEN NED     = 002107.900.IBM.75.0000000BALB1.0400
ACTUAL TOKEN NED  = 002107.900.IBM.75.0000000BALB1.0400
SCP DEVICE NED    = 002107.900.IBM.75.0000000BALB1.0428
ACTUAL DEVICE NED = 002107.900.IBM.75.0000000BALB1.0428
PATH IS AVAILABLE, INITIALIZED AND OPERATIONAL (00)
HYPERPAV ALIASES CONFIGURED = 159
FUNCTIONS ENABLED = MIDAW, ZHPF
PATH OFFLINE DUE TO THE FOLLOWING REASON(S):
    BY OPERATOR
```

Figure 12-8 The reason why the path is offline

12.6 Restoring after the path-related error has been corrected

When the path-related error has been corrected, the path taken offline to the devices on the control unit can be restored by the following commands:

- VARY CU
- VARY PATH
- VARY DEVICE
- CONFIG CHP

Note: We suggest you first issue a **VARY DEVICE** or **VARY PATH** for only one device or path to check whether success is achieved before issuing **VARY CU** for all devices or paths.

13

Service aids enhancements

MVS service aids are a group of tools used to help to detect problems, gather documentation needed to resolve the causes of those problems, and communicate with support locations remote from where materials might have originally been collected.

This chapter describes the following service aids enhancements:

- z/OS Problem Documentation Upload Utility (PDUU)
- IPCS SYSTRACE enhancement
- IPCS subcommand DOCPU

13.1 z/OS Problem Documentation Upload Utility

z/OS Problem Documentation Upload Utility (PDDU) is a parallel file transfer protocol (FTP) utility designed to send documentation in a more efficient manner to IBM FTP sites. This utility sections the input file into smaller files that are sent in parallel, resulting in shortened transmission time for large data sets such as stand-alone dumps. You can optionally encrypt the data sets.

Prior to z/OS V1R13, the utility is an independent tool known as MTFTPS that is available for download from an IBM web site. In z/OS V1R13, the utility is called Problem Documentation Upload Utility and is now a z/OS service aids program that simplifies and speeds up the process of sending problem-related materials from client sites to IBM.

PDDU provides a readily available industry standard level of encryption that uses triple DES encryption (CPACF) between two z/OS end points. It allows client-defined criteria to drive multiple simultaneous FTP sessions, thereby allowing higher utilization of client networking infrastructure and reducing the transmission time for large data sets.

13.1.1 Work data sets and work data sets guidelines

As explained here, PDUU can be used as the primary utility for sending large volumes of documentation to the IBM FTP site.

FTP work data sets

The work data sets are dynamically allocated and can range in size from 1 MB to 9999 MB. You can experiment to see what works best in your environment, but note the following guidelines:

- Use medium-size work data sets.
- The data sets are compressed and sent in parallel using multiple, simultaneous FTP sessions.
- Up to 20 FTP sessions can run simultaneously using two work files for each FTP session. Start with three or four parallel FTP sessions. Too many parallel FTP sessions can saturate the network link.

FTP work files

The two work files used for each parallel FTP session are known as the A file and the B file. Each A work file is filled by copying records from the input file. When the A file is full, the FTP sessions are started in parallel.

At the same time, each B work file is filled by copying records from the input file. When the B file is full and the transfer of the A file is complete, transfer of the next B file begins.

This process continues between the A and the B files until everything in the input file is sent. The work data sets are dynamically allocated and can range in size 1 MB to 9999 MB.

Important: If the work data sets are significantly smaller in relation to the input data set, you can end up with too many files on the IBM FTP sites. For example, if you are sending a 100 GB z/OS stand-alone dump and make the work data set size 1 MB, you create 100,000 files on the IBM FTP site, which exceeds the IBM limit of 999 files. This also causes significant delay by starting and stopping the FTP sessions for each file.

If the work data sets are significantly larger in relation to the input file size, the amount of overlap time is decreased. When the program first starts, it must fill the A work files before it starts transmitting any data. This means the copy time is not overlapping with data that needs to be sent through FTP. For example, if you send a 1 GB dump and you set the work data set size to 1 GB (1,000 MB), there is no overlap between copying the records and sending the work files.

13.1.2 PDDU compression and encryption

PDUU always compresses the input data before it is written to the work data sets. Therefore, it is not necessary to use a tool such as AMATERSE or TRSMAIN to compress the input data set. In addition, 192-bit triple Data Encryption Standard (DES) can be requested by using the CIPHER_KEY keyword. Without the keyword, the data is simply compressed. By using the keyword, the data is compressed and then encrypted.

Encryption is provided by the CP Assist for Cryptographic Functions (CPACF), DES/TDES Enablement (feature 3863). Encryption is available on all processors starting with the z990 (2084) and z890 (2086). CPACF feature 3863 enables clear key DES and TDES instructions on all supported CPs. The encryption algorithm used for 3DES is documented in *z/Architecture Principles of Operation*, SA22-7832, which is available at:

`http://www.ibm.com/systems/z/os/zos/bkserv/r10pdf/#zarchpops`

Changes made to package the PDUU as part of z/OS

The PDUU utility program name is AMAPDUPL with an alias entry point of MTFTPS for compatibility. AMAPDUPL resides in SYS1.MIGLIB, which is a data set in the LNKLST concatenation). Thus, a STEPLIB DDNAME is no longer needed to invoke AMAPDUPL. For complete descriptions of related AMA messages, see *z/OS MVS System Messages Volume 1 (ABA - AOM)*, SA22-7631.

Note: z/OS Communications Server must be configured to use the FTP client program. For more information, see the topic about transferring files using FTP in *z/OS Communications Server IP Configuration Guide*, SC31-8775.

Supported data set types

The following data set types are supported by PDUU:

- Members of partitioned data sets (PDS) and partitioned data sets extended (PDSE)
- Large format (DSNTYPE=LARGE) and traditional sequential data sets
- Extended format sequential data sets
- Fixed and variable, blocked and unblocked, unspanned record formats (RECFM = F,FB,FBS,V,VB)
- Data sets with records containing ISO/ANSI or machine code control characters
- Data sets in cylinder-managed space

Unsupported data set types

The following data set types are not supported by PDUU:

- Large block interface (LBI) (no BLKSIZE value)
- VSAM and direct access (DSORG=DA) data sets
- Data sets with keys (KEYLEN)
- Spanned record formats (VBS)
- Partitioned data sets (PDS) and partitioned data sets extended (PDSE)
- z/OS UNIX files
- Any data set with an undefined-length record format (RECFM=U)

13.1.3 JCL statements for PDUU

The AMAPDUPL utility is invoked using JCL, (PGM=AMAPDUPL), consisting of the following DD statements:

SYSPRINT This specifies the job output data set. The data set can be either SYSOUT or a sequential data, and it must be RECFM=FB,LRECL=134.

SYSUT1 This specifies the sequential input data set to transfer to IBM.

SYSIN This specifies the sequential data set that uses following control statements.

The data set must be RECFM=FB,LRECL=80).

- **TARGET_SYS** The name of the TCP/IP system to transfer the files to using FTP.
- **USERID** The user ID on the target system that is used to send the files.
- **PASSWORD** The password for the USERID on the target system.
- **ACCOUNT** The account data that is sent when an FTP session is started.
- **WORK_DSN** The prefix for the data set names of work files on the sending system.
- **WORK_DSN_SIZE** The size of the work files in megabytes.
- **KEEP_WORK** The parameter to save the work data sets that are dynamically allocated for each FTP session.
- **DATACLAS** The data class to use when allocating the work files on the sending system.
- **STORCLAS** The storage class to use when allocating the work files on the sending system.
- **CC_FTP** The number of parallel FTP sessions to use when transmitting the files.
- **DIRECTORY** The directory on the target system where the files will be sent with FTP.
- **PMR** The PMR number that this file is to be associated with.
- **CIPHER_KEY** The encryption key to use for 192-bit triple DES encryption.

FTPCMDS This is an optional DD statement that provides additional flexibility for traversing firewall or proxy servers.

13.1.4 JCL examples

The following examples demonstrate starting points for using a simple FTP connection, and for traversing a firewall or proxy server. There are few common characteristics for firewall or proxy servers, and local customization. If you are able to traverse the firewall or proxy server with a plain FTP statement, then modifications to the parameters USERID, PASSWORD, ACCOUNT, and TARGET_SYS, in conjunction with commands in the FTPCMDS data set, the ftp_data file, or both, can permit the z/OS Problem Documentation Upload Utility to traverse your firewall or proxy server.

Tip: You might want to place your user ID and password into a separate catenated data set. This provides added security because the user ID and password are not directly in the JCL. It is also easier to change the user ID and password across multiple jobs.

Simple FTP connection

Figure 13-1 shows the JCL to use for a simple FTP connection.

```
//FTP EXEC PGM=AMAPDUPL
//SYSUDUMP DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DISP=SHR,DSN=IPCS.PROBLEM.DUMP
//SYSIN DD *
USERID=anonymous
PASSWORD=anonymous
TARGET_SYS=testcase.boulder.ibm.com
TARGET_DSN=fredsamp.bigfile
WORK_DSN=fred.ftpout
CC_FTP=03
WORK_DSN_SIZE=500
DIRECTORY=/toibm/mvs/
PMR=12345.123.000
//
```

Figure 13-1 Simple FTP connection

FTP connection using a proxy server

Figure 13-2 shows an example of JCL for an FTP connection using a proxy server.

```
//FTP EXEC PGM=AMAPDUPL
//SYSUDUMP DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DISP=SHR,DSN=IPCS.PROBLEM.DUMP
//SYSIN DD *
USERID=anonymous@testcase.boulder.ibm.com
PASSWORD=proxyid@proxypw
TARGET_SYS=your.proxy.server.name
TARGET_DSN=fredsamp.bigfile
WORK_DSN=wes.ftpout
CC_FTP=03
WORK_DSN_SIZE=500
DIRECTORY=/toibm/mvs/
PMR=12345.123.000
//
```

Figure 13-2 FTP connection using a proxy server

FTP connection using a proxy server and user ID

Figure 13-3 shows an example of JCL for an FTP connection using a proxy server with proxy user ID.

```
//FTP EXEC PGM=AMAPDUPL
//SYSUDUMP DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DISP=SHR,DSN=IPCS.PROBLEM.DUMP
//SYSIN DD *
USERID=anonymous@proxyid@testcase.boulder.ibm.com
PASSWORD=proxyid@proxypwTARGET_SYS=your.proxy.server.name
TARGET_DSN=fredsamp.bigfile
WORK_DSN=wes.ftpout
CC_FTP=03
WORK_DSN_SIZE=500
DIRECTORY=/toibm/mvs/
PMR=12345.123.000
//
```

Figure 13-3 FTP connection using a proxy server with proxy user ID

FTP connection and proxy server with a FTPCMDS DD statement

Figure 13-4 shows an example of JCL for an FTP connection using a proxy server with the FTPCMDS DD statement.

```
//FTP EXEC PGM=AMAPDUPL
//SYSUDUMP DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DISP=SHR,DSN=IPCS.PROBLEM.DUMP
//FTPCMDS DD DISP=SHR,DSN=FRED.FTPCMDS.DATA
//SYSIN DD *
USERID=proxyid@testcase.boulder.ibm.com
PASSWORD=proxypw
TARGET_SYS=proxy.server.name 2121
TARGET_DSN=SVCD
WORK_DSN=HLQ.FTPOUT
CC_FTP=03
WORK_DSN_SIZE=500
DIRECTORY=/toibm/mvs/
PMR=12345.123.000
//
*****************************************
* The FRED.FTPCMDS.DATA data set contains *
* user anonymous pw userid@company.com *
*****************************************
```

Figure 13-4 FTP connection using a proxy server with the FTPCMDS DD statement

13.1.5 Using an NRTRC data set

Figure 13-5 shows an example of JCL that is storing the proxy login and password in the userid.NETRC data set. This can be submitted as a surrogate job where the userid.NETRC is not visible to the job originator. Use of the userid.NETRC data set requires NETRCLEVEL=2, which is set in the FTP.DATA data set.

```
//FTP EXEC PGM=AMAPDUPL
//SYSUDUMP DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DISP=SHR,DSN=IPCS.PROBLEM.DUMP
//SYSIN DD *
TARGET_SYS=-f"//'FRED.MYFTP.DATA'" testcase.boulder.ibm.com
PMR=12345.123.000
DIRECTORY=/toibm/mvs/
TARGET_DSN=fred.gb01tst
work_dsn=wes.ftpout
cc_ftp=03 WORK_DSN_SIZE=500
//
The userid.NETRC dataset consists of one line:
   machine testcase.boulder.ibm.com login anonymous password ibmusr@ibm.com

The FTP.DATA dataset contains:
;*********************************************************************
;*
NETRCLEVEL 2
;*
```

Figure 13-5 Using a userid.NETRC data set

Return codes for PDUU

Table 13-1 lists the return codes for PDUU and their explanations. The return code is placed into general purpose register 15.

Table 13-1 PDDU return codes

Return code	Explanation
0	Successful completion
8	Invalid parameters in control statement
10	Unsupported data set format
12	Storage obtain failure
16	Required input parameters are missing
20	Invalid input data set specified
24	Severe error occurred during dictionary building
28	Severe error occurred during file open
32	Severe error occurred during compression process
36	Error in FTP operation
64	Severe error in file operation
99	System or user abend occurred

13.2 IPCS SYSTRACE subcommand enhancement

z/OS support of LPARS that have more CPUs has increased the size of dumps and traces. This means that greater complexity has been added to the problem determination process as much larger volumes of diagnostic data need to be analyzed.

In z/OS V1R13, new function has been added to SYSTRACE processing to allow filtering by CPU number and CPU type.

- CPUMASK and CPUTYPE keywords have been added to the IPCS SYSTRACE subcommand.
- There is the same filtering implication as the existing CPU keyword.

This enhancement improves the filtering, sorting, and searching of large amounts of diagnostic data for faster and more efficient problem determination, and helps to minimize the impact of these tasks on a client's system.

New keywords for the SYSTRACE command

The SYSTRACE command now supports the following new keywords.

- CPUMASK('cpu-hexadecimal-mask')

  This keyword limits formatting to only the trace entries produced on the processors specified in the CPU (cpu-hexadecimal-mask). Specify the processors using a string of hexadecimal characters. Each hexadecimal character identifies four processors, and the leftmost bit designates a lower processor address, starting from zero (0).

The processor maximum in z/OS defines the length of this hexadecimal string. The current processor maximum is 128. Therefore, the maximum length of the hexadecimal mask string is 32.

```
Example: CPUMASK(34F) means CPUs 2,3,5,8,9,10,11
```

- CPUTYPE((ZAAP|ZA) | (ZIIP|ZI) | (STANDARD | CP | S))

  The CPUTYPE keyword limits the entries to specific processor types. CPUTYPE(ZAAP) selects all IBM System z Application Assist Processors (zAAP) in the configuration. CPUTYPE(ZIIP) selects all IBM System z Integrated Information Processors (zIIP) in the configuration. CPUTYPE(STANDARD) selects all standard processors. You can use the following abbreviations:

  – ZA for ZAAP

  – ZI for ZIIP

  – CP or S for STANDARD

  To view a combination of CPU types, you can combine, in any order, the ZAAP, ZIIP, and STANDARD keywords. Use a space or comma (,) as the delimiter.

  You can also combine CPUTYPE, CPU, and CPUMASK as a union of sets. If all of the parameters are omitted, all processors are included as the default.

New options on the system trace analysis panel

The system trace analysis panel accessed through IPCS Option 2.7.4 has been enhanced to include panel options for CPU list, CPUMASK, and CPUTYPE. Figure 13-6 shows the new panel options.

```
---------------------- IPCS - SYSTEM TRACE ----------------------------------
Option ===>
   ALL ASIDs     ===>
   CURRENT ASID  ===>  *    (default)
   ERROR ASID    ===>  *    (default)
   TCBERROR ASID ===>
   ASID list     ===>
   Jobnames      ===>
   EXCLUDE(BR)   ===>
   EXCLUDE(MODE) ===>
   TIME format   ===>  HEX   (HEX, GMT or LOCAL)
   SORT by CPU   ===>      (SHOW        ENTRIES BEFORE AND AFTER TIME
                                                - MM/DD/YY,HH:MM:SS:DDDDDD)
   CPU list      ===>
   CPUMASK       ===>
   CPUTYPE       ===>      ZAAP      ZIIP      STANDARD
 SYSTRACE specification:
 SYSTRACE  CURRENT ERROR TIME(HEX)

 S = start SYSTRACE, U = get SYSTRACE performance data report,
 R = reset panel fields, END/PF3 = terminate SYSTRACE
```

Figure 13-6 System trace analysis panel - new options

13.3 IPCS subcommand DOCPU

z/OS support for up to 100 CPUs increases the volume of data that needs to be analyzed for CPU-related problem diagnosis. To minimize the work required to get the same data for many CPUs a new IPCS subcommand, **DOCPU**, has been added to produce output for multiple CPUs.

The **DOCPU** subcommand is used to gather stand-alone dump data for tasks that need to be repeated for each of the specified processors, for example, to display the contents of a processor-related control block for a group of processors. Using this command, you can obtain processor-related diagnostic data from a stand-alone dump with one command rather than repeating the command for each processor.

Restriction: The **DOCPU** subcommand can only be used for stand-alone dumps.

Command syntax for DOCPU

The **DOCPU** subcommand syntax is shown in Figure 13-7.

```
{ DOCPU }
-------- Data Selection Parameters -------------------------
[ ( CPU ( cpu-address-range-list ) )|
CPUTYPE ( (ZAAP|ZA) | (ZIIP|ZI) | (STANDARD | CP | S) ) |
CPUMASK ( cpumask ) ]

-------- SETDEF-Defined Parameters -------------------------
Note: You can override the following SETDEF parameters.
[ FLAG(severity) ]
[ PRINT | NOPRINT ]
[ TERMINAL | NOTERMINAL ]
[ TEST | NOTEST ]
```

Figure 13-7 DOCPU command syntax

13.3.1 Data selection parameters for DOCPU

The data selection parameters limit the scope of the DOCPU output. If no selection is made, then output is produced for all CPUs. Use the following parameters to limit the scope of the data in the report. If you omit these parameters (CPU, CPUTYPE, and CPUMASK), all processors are included as the default.

CPU (cpu-address-range-list)

This parameter specifies the CPU or CPUs that are selected for the execution of the specified IPCS subcommand.

Specify the processors (CPU) that are selected to run the specified IPCS subcommand, for example:

- CPU-address-range-list is a CPU number or a range of numbers or a combination of both.
- The CPU number can be in decimal or in hex (`X'..'`).
- A colon (:) may be used to indicate a range of CPUs.

- A space or a comma (,) may be used as a delimiter.

  Examples:

  ```
  CPU(0)
  CPU(5:10)
  CPU(0 5:10)
  CPU(0,3,5:10)
  CPU(X'A')
  ```

CPUMASK(CPU hexadecimal mask)

This parameter specifies processors in a string of hexadecimal characters. Each hexadecimal character identifies four processors. The maximum number of processors supported by z/OS defines the maximum length of this hexadecimal string.

Currently, the maximum number of processors supported by z/OS is 128, so the maximum length of the hexadecimal mask is 32. The leftmost bit designates the lower processor address starting from zero (0), for example:

- CPUMASK(FFF)
- CPUMASK(F0F0)
- CPUMASK(80) CPU

You can combine CPUTYPE and CPUMASK as a union of sets. If all of the processors are omitted, the default is to include all processors, for example:

```
CPUMASK(34F) means CPUs 2,3,5,8,9,10,11
```

CPUTYPE((ZAAP|ZA) | (ZIIP|ZI) | (STANDARD|CP|S))

You can combine the ZAAP, ZIIP, and STANDARD options in any order to select a combination of CPU types, for example, CPUTYPE(ZAAP STANDARD). You can use spaces or commas as delimiters.

Here are other examples showing the use of this parameter:

- ZAAP or ZA will select all ZAAP processors in the configuration.
- ZIIP or ZI will select all ZIIP processors in the configuration.
- STANDARD or CP or S will select all standard processors in the configuration.
- The ZAAP, ZIIP, and STANDARD options may be combined in any order to select a combination of CPU types, for example:

  ```
  CPUTYPE(ZAAP STANDARD) to select all zAAPs and standard CPUs.
  ```

Note: If all CPU, CPUTYPE, and CPUMASK keywords are omitted, all processors will be included as the default.

EXEC((ipcs subcommand))

This executes the specified IPCS subcommand for each CPU specified by appending CPU(X'..') to the IPCS subcommand.

Examples of DOCPU command

To display 4 bytes of storage at 414 for CPU 0 and CPU 1:

```
DOCPU CPU(0,1) EXEC((L 414 LEN(4)))
```

To format the PSA of processor 0,1,2,3,8,9,10,11:

```
DOCPU CPUMASK(F0F0) EXEC((CBF 0 STR(PSA)))
```

To format the LCCA for all standard CPUs:

```
DOCPU CPUTYPE(S) EXEC((CBF 210? STR(LCCA)))
```

14

System logger - SMF

System logger is a set of services that allows an application to write, browse, and delete log data. You can use system logger services to merge data from multiple instances of an application, including merging data from multiple systems across a sysplex.

When you are concurrently running multiple instances of an application in a sysplex, and each application instance can update a common database, then it is important for your installation to maintain a common log of all updates to the database from across the sysplex so that if the database is damaged, it can be restored from the backup copy. You can merge the log data from applications across the sysplex into a log stream. A *log stream* is simply a collection of data in log blocks residing in the coupling facility and or on DASD.

This chapter discusses SMF log stream changes that have been implemented in z/OS V1R13, namely:

- System logger monitoring for log stream offload data set allocations

14.1 Monitoring for log stream offload data set allocations

System logger with z/OS V1R13 provides monitoring for log stream offload data set allocations and recalls. It also produces warning and action message after predefined intervals.

The IXGCNFxx parmlib member contains statements that can be used to control the following functions when z/OS system logger starts or is restarted on the initializing system within the sysplex.

1. Identify the tracing options when system logger initialized.
2. Specify the logger monitoring intervals for warning and action messages.

IEASYSxx parmlib member

The IEASYSxx parmlib member contains the following system logger options for its IXGCNFxx parmlib member:

```
IXGCNF={aa }
       {(aa,bb...)}
```

This parameter identifies the IXGCNFxx parmlib member or members to be used when system logger starts or is restarted on the initializing system within the sysplex.

- Syntax IXGCNF=aa specifies a single member.
- Syntax IXGCNF=(aa,bb,....) identifies groups of system logger initialization statements across several IXGCNFxx members.

You need to create the member or members and place them in SYS1.PARMLIB. When multiple IXGCNFxx parmlib members are concatenated, the individual system logger options are merged, with the last parmlib member option taking precedence. The following command displays the merged information:

```
D LOGGER,IXGCNF[,options]
```

Value range: Any two alphanumeric characters

Default value: None

14.1.1 IXGCNFxx parmlib member statement for logger monitoring

To identify system logger monitoring you can specify the new parameters, shown in Figure 14-1, in the IXGCNFxx parmlib member.

```
[ CTRACE(parmlib_member_name) ]
[ ]
      MONITOR ]
     [OFFLOAD ]
        [ WARNALLOC(initial-delay-interval) ]
        [ ACTIONALLOC(secondary-delay-interval) ]
        [ WARNRECALL(initial-delay-interval) ]
        [ ACTIONRECALL(secondary-delay-interval) ]
[ ]
```

Figure 14-1 New parameters for logger monitoring

Where:

MONITOR This provides system logger monitoring specifications.

OFFLOAD Use this to specify the initial and secondary delay intervals for monitoring system logger log stream offload activity. Refer to the topic about offload and service task monitoring in *z/OS MVS Setting Up a Sysplex*, SA22-7625, for more information about logger offload monitoring.

WARNALLOC (initial-delay-interval) - Use this to specify the amount of time, in seconds, allowed before the log stream offload monitor will start issuing warning messages indicating an offload is delayed waiting for a data set allocation or deletion request to complete.

If the data set allocation or deletion request does not complete within this interval, system logger issues message IXG310I to warn the operator of the log stream offload being delayed.

If this value is not less than the ACTIONALLOC secondary-delay-interval, then the logger will not issue the IXG310I warning message as part of its log stream offload monitoring.

`D LOGGER,IXGCNF[,MONITOR]` system logger monitoring settings.

Value range: It must be a decimal number from 5 to 86400 seconds (meaning 5 seconds to 24 hours, respectively).

Default: 30 (30 seconds, 1/2 minute).

ACTIONALLOC (secondary-delay-interval) - Use this to specify the amount of time, in seconds, allowed before the log stream offload monitor will issue messages to allow possible action be taken by the installation because of an offload being delayed waiting for a data set allocation or deletion request to complete.

If the data set allocation or deletion request does not complete within this interval, system logger issues message IXG311I identifying the log stream and data set resources, and WTOR message IXG312E to allow action to be taken if necessary.

`D LOGGER,IXGCNF[,MONITOR]` system logger monitoring settings.

Value range: Must be a decimal number from 5 to 86400 seconds (meaning 5 seconds to 24 hours, respectively).

Default: 60 (60 seconds, 1 minute).

WARNRECALL (initial-delay-interval) - Use this to specify the amount of time, in seconds, allowed before the log stream offload monitor will start issuing warning messages indicating an offload is delayed waiting for a migrated data set recall request to complete.

If the migrated data set recall request does not complete within this interval, system logger issues message IXG310I to warn the operator of the log stream offload being delayed.

If this value is not less than the ACTIONRECALL secondary-delay-interval, then logger will not issue the IXG310I warning message as part of its log stream offload monitoring.

D LOGGER,IXGCNF[,MONITOR] system logger monitoring settings.

Value range: Must be a decimal number from 5 to 86400 seconds (meaning 5 seconds to 24 hours, respectively).

Default: 60 (60 seconds, 1 minute)

ACTIONRECALL (secondary-delay-interval) - Use this to specify the amount of time, in seconds, allowed before the log stream offload monitor will issue messages to allow possible action be taken by the installation because of an offload being delayed waiting for a migrated data set recall request to complete.

If the migrated data set recall request does not complete within this interval, system logger issues messages IXG311I identifying the log stream and data set resources, and logger also issues WTOR message IXG312E to allow action to be taken if necessary.

D LOGGER,IXGCNF[,MONITOR]system logger monitoringsettings.

Value range: Must be a decimal number from 5 to 86400 seconds (meaning 5 seconds to 24 hours, respectively).

Default: 120 (120 seconds, 2 minutes).

Logger command examples

The **D LOGGER,IXGCNF[,options]** command displays the merged information.

```
D LOGGER,IXGCNF,MONITOR
IXG607I   16.51.08  LOGGER DISPLAY 777
 LOGGER PARAMETER OPTIONS
 KEYWORD           SOURCE   VALUE
 ----------------- -------- -----------
 MONITOR OFFLOAD
  WARNALLOC        DEFAULT  00030
  ACTIONALLOC      DEFAULT  00060
  WARNRECALL       DEFAULT  00060
  ACTIONRECALL     DEFAULT  00120
```

Warning message scheduled time delays

The default values for issuing the warning message IXG310I are 30 seconds for allocating a data set and 60 seconds for recalling a data set.

```
IXG310I SYSTEM LOGGER CURRENT OFFLOAD IS NOT PROGRESSING FOR LOGSTREAM
logstream STRUCTURE: strname request DSN=dsnhlq.dsnlsn.dsnllq
```

The default values for issuing action messages IXG311I and IXG312E are 60 seconds for allocating a data set and 120 seconds for recalling a data set.

```
IXG311I SYSTEM LOGGER CURRENT OFFLOAD HAS NOT PROGRESSED DURING THE PAST
seconds SECONDS  FOR LOGSTREAM logstream STRUCTURE: strname request
DSN=dsnhlq.dsnlsn.dsnllq
```

```
IXG312E OFFLOAD DELAYED FOR logstream, REPLY "MONITOR", "IGNORE",
"FAIL","AUTOFAIL", OR "EXIT".
```

14.2 RELATIVEDATE parameter update

Prior to z/OS V1R13, there was a risk of a failure when trying to ARCHIVE or DELETE an entire log stream because there was a “race” between IFASMFDL and any SMF record writers, as SMF does not stop record writing to the log stream during the dump process. This limited the configuration options for clients when setting up SMF log stream recording.

z/OS V1R13 update

z/OS V1R13 provides a method so that the ARCHIVE and DELETE options can now operate on the entire log stream. You can set a production dump procedure without changes on JCL each time you run it. Being able to ARCHIVE or DELETE the entire log stream provides better migration options for clients who want to use SMF log stream recording.

Note: There are no new keywords or new options for this support. The RELATIVEDATE parameter was introduced in z/OS V1R10 and it has been updated with z/OS V1R13.

14.2.1 RELATIVEDATE parameter with z/OS V1R13

The RELATIVEDATE parameter sets the range of time the IFASMFDL program is going to process. IFASMFDL will now write a marker into the log stream to allow for the deletion of all of the data selected. If the log stream cannot be written to this output data set, the job will still fail.

Without this support, if an ARCHIVE or DELETE of an entire log stream is attempted, then IFASMFDL can fail and an IFA832I message will be issued to the job log, as shown in Figure 14-2 on page 275.

IFA832I INVALID PARAMETER COMBINATION FOR xxxxxx

Where: xxxxxx can be ARCHIVE / DELETE / RELATIVEDATE.

Explanation: The system encountered an error when running with the ARCHIVE, DELETE option or the RELATIVEDATE parameter:

- If the system was running with the ARCHIVE option, the encountered record was not written,and the record did not match any of the criteria of any OUTDD statement.
- If the system was running with the DELETE option, the error is that an OUTDD statement was specified.
- If the system was running with the RELATIVEDATE parameter but DATE was also specified.

Note: If multiple log streams are specified and then an error occurs with any one log stream, the job will fail with this message.

Figure 14-2 IFA832I message

Updated message with z/OS V1R13

This message was updated with z/OS V1R13 to include information about RELATIVEDATE, so if you try to define both RELATIVEDATE and DATE, they are not compatible parameters.

Currently the Summary Statistics Report written by the IFASMFDL program provides the date range of data read in all logstreams. But there was no way to quickly tell what data was processed (written and/or deleted) from any logstream.

The new IFA846I message will now be displayed for all operations (DUMP, ARCHIVE, DELETE), and it will detail the range of data processed for each logstream, as shown in Figure 14-3.

```
IFA846I PROCESSED DATA RANGES FOR LOGSTREAMS
 LSNAME                          START DATE/TIME       END DATE/TIME
 IFASMF.DEFAULT             04/21/2011 16:30:00 04/21/2011 16:50:15
 IFASMF.PERF                04/21/2011 16:30:00 04/21/2011 16:50:00
```

Figure 14-3 Message IFA846I

14.2.2 Keeping the log stream archive process similar to SMF MANx

With SMF data set recording, it is common to see JCL that dumps and clears a data set. The same JCL is reused without requiring changes to the JCL. Now with this support, a similar concept is available. You can use ARCHIVE and RELATIVEDATE to dump and archive SMF log stream information, as shown in the example in Figure 14-4.

The job will copy log stream data to OUTDD1 and OUTDD2 and will delete information from the log streams after copying. OUTDD1 will keep only records 70 to 79. UTDD2 will keep all records from 10 to 255, as shown in the figure. After copying information to OUTDD1 and OUTDD2, the records that were copied are deleted from the log streams.

```
//DIANAG1   JOB (999,POK),'LOGR POLICY',CLASS=A,REGION=0M,
//             MSGCLASS=T,TIME=10,MSGLEVEL=(1,1),NOTIFY=&SYSUID
//STEP EXEC PGM=IFASMFDL
//OUTDD1 DD DSN=DIANAG.SMFARCH.PERFORM2,DISP=(NEW,CATLG,DELETE),
//    SPACE=(CYL,(50,20),RLSE),DCB=(LRECL=32760,RECFM=VBS,BLKSIZE=0)
//OUTDD2 DD DSN=DIANAG.SMFARCH.OTHER2,DISP=(NEW,CATLG,DELETE),
//    SPACE=(CYL,(50,20),RLSE),DCB=(LRECL=32760,RECFM=VBS,BLKSIZE=0)
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
 LSNAME (IFASMF.PERF,OPTIONS(ARCHIVE))
 LSNAME (IFASMF.DEFAULT,OPTIONS(ARCHIVE))
 RELATIVEDATE (BYDAY,0,1)
 OUTDD (OUTDD1,TYPE(70:79))
 OUTDD (OUTDD2,TYPE(10:255))
 SID(SC74)
/*
```

Figure 14-4 Archive job example

Attention: The SMF log stream dump program requires APF authorization. Running the SMF log stream dump program, as shown in the Figure 14-4 on page 276 examples, preserves the APF authorization assigned to the SMF log stream dump program.

Invoking the SMF log stream dump program in a way other than as shown in the figure (for example, if you invoke the SMF log stream dump program from another program or invoke it as a TSO/E command) can cause it to lose its authorization.

Figure 14-5 shows the SYS1.PARMLIB.SMFPRM01 member specification that OUTDD2 will use from the information for the IFASMF.PERF and IFASMF.DEFAULT log streams.

```
RECORDING(LOGSTREAM)              /* LOGSTREAM RECORDING  */
LSNAME(IFASMF.PERF,TYPE(70:79))   /* RMF GOES IFASMF.PERF */
DEFAULTLSNAME(IFASMF.DEFAULT)     /*DEFAULT LS NAME   */
```

Figure 14-5 SMF log stream parmlib definitions

You can set a production job to run every day using the following specification:

```
RELATIVEDATE (BYDAY,0,1)
```

This specification will accumulate dumped information about the same data set using DISP=MOD:

```
//OUTDD1 DD DSN=DIANAG.SMFARCH.PERFORM3,DISP=(MOD,KEEP,KEEP)
//OUTDD2 DD DSN=DIANAG.SMFARCH.OTHER3,DISP=(MOD,KEEP,KEEP)
```

14.2.3 DUMP, ARCHIVE, and DELETE options with the LSNAME parameter

When you use log streams to record SMF data, you can use the ARCHIVE and DELETE options to delete data from the log stream.The ARCHIVE and DELETE options are changed to be able to operate on the entire log stream. The following options that are available with the SMFPRMxx parmlib member LSNAME parameter are processed as listed her:

- DUMP will simply dump information, but the log stream will still have this data there.
- ARCHIVE means data is dumped, and it will be deleted from the log stream.
- DELETE will simply delete information from the log stream.

Using the ARCHIVE and DELETE options

When you specify the ARCHIVE or DELETE options using the LSNAME keyword, pay special attention to the following items:

- The start date and time specified with the IFASMFDL program is always the start date and time of the log stream. The start time that is specified by the START parameter is ignored. The start date that is specified by DATE or RELATIVEDATE parameter will also be ignored.
- When using the ARCHIVE option, you can use any combinations of the OUTDD filters except for START and END.
- For an ARCHIVE request, you must write each record in the log stream until the end date and time to at least one OUTDD. Ensure that all the types and subtypes and all SIDs are being written to a data set or are specified on the OUTDD parameters. If any record fails to match any specified OUTDD, the request fails.
- If you specify the DELETE option, everything in the specified log stream is removed, from the start of the log stream up to the end date that is specified by the DATE or RELATIVEDATE parameter and the end time that is specified by the END parameter. The OUTDD statements are not permitted when the DELETE option is used.
- ARCHIVE and DELETE requests are performed on logger block boundaries. If the logger block time stamp matches the criteria that is specified on the ARCHIVE or DELETE request, all of the records in that block are included in the ARCHIVE or DELETE request. For an ARCHIVE or DELETE request, you might not get data in the time range that is written past the point of your end date and time, for example, if you are using the series of logger blocks where the time above them is the time they are written (Figure 14-6).

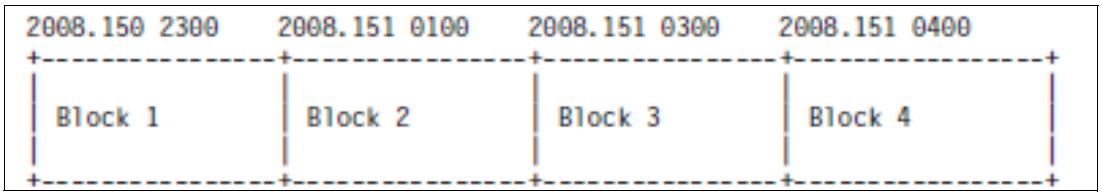


Figure 14-6 Block records in a log stream example

In the figure, each logger block has SMF records at the following times:

- Block 1 --> 2008.150 2200 to 2008.150 2259
- Block 2 --> 2008.150 2300 to 2008.151 0059
- Block 3 --> 2008.151 0100 to 2008.151 0259
- Block 4 --> 2008.151 0300 to 2008.151 0359
- When you specify an ARCHIVE or DELETE option on this log stream where the end date is 2008.150 and the end time is 2400, only the data from Block 1 is dumped and deleted for an ARCHIVE request, and only the data from Block 1 is deleted for a DELETE request.
- The MAXDORM option in SMFPRMxx parmlib member can be used to align the logger block boundaries with the date and time selection criteria of the ARCHIVE and DELETE options. For more information about the MAXDORM option, see the SMFPRMxx parmlib member in *z/OS MVS Initialization and Tuning Reference*, SA22-7592.

Note: When using the DELETE and ARCHIVE options, you must ensure that only one instance of the IFASMFDL program is running at a time for a given log stream. Running multiple instances of the dump program against the same log stream leads to unexpected results.

RELATIVEDATE parameter considerations

The date range specified through the RELATIVEDATE parameter is calculated starting at the previous full unit. For example, if you use the RELATIVEDATE parameter with the BYMONTH option on May 15, then the calculation of the time range starts at counting backwards from April, because April is the first previous full month.

To get the RELATIVEDATE parameter to resolve to the current date, you can use:

```
RELATIVEDATE(unit,0,1)
```

If the date range calculated by the RELATIVEDATE parameters exceeds the date and time that the job was submitted, the end date and time in use are set to the date and time that the job was submitted. In this case, return code X'04' is returned and message IFA834I is issued, indicating the end date and time for the dump.

Important: This support was rolled back to z/OS V1R11 and z/OS V1R12, and is available through the following PTFs for APAR OA34589:

- For z/OS V1R11, UA58910
- For z/OS V1R12, UA58911

14.3 Stop reading before end of log stream support

With previous releases, all log stream data was always read until the end of the log stream regardless of the specified end date and time specified. The SMF data set recording did not have this issue because each data set contained only a fixed amount of data.

z/OS V1R13 now provides a new option to allow the IFASMFDL dump program to stop reading the log stream before the end of the log stream. Depending on the configuration, this can greatly reduce the duration of IFASMFDL jobs.

z/OS V1R13 has two new options for the IFASMFDL program:

- SMARTENDPOINT
- SMARTEPOVER

14.3.1 SMARTENDPOINT processing

The SMF log stream dump program (IFASMFDL) SMARTENDPOINT keyword specifies that processing of input records in the log stream is to discontinue after it has been determined that records for all known SIDS contain a date and time that is two hours past the IFASMFDL specified date and time plus the specified SMARTEPOVER parameter value.

The default behavior is that IFASMFDL continues to read records all the way to the end of the log stream. In z/OS V1R9 through z/OS V1R12, this keyword only applies to the DUMP option.

Beginning with z/OS V1R13, this keyword applies to all the OPTIONS subparameters ARCHIVE, DELETE, and DUMP.

Notes:

- The SMARTENDPOINT keyword is available only for the DUMP option with z/OS V1R12 and below with APAR OA31737.

  These keywords are supported on lower releases with the PTFs for APAR OA31737 and OA34374 applied, as follows:

  - HBB7750 – UA58243 and UA55964
  - HBB7760 – UA58244 and UA55965
  - HBB7770 – UA58245 and UA55966
- With z/OS V1R12 and earlier, the SMARTENDPOINT parameter does not support ARCHIVE or DELETE processing. If it is used, those keywords will be ignored.

The IFASMFDL SMARTENDPOINT processing can dynamically detect the z/OS images while reading the log stream, or it can use the SID(xxxx) statement to specify the images. If there is the possibility that buffering occurred on an z/OS image that writes to this log stream, then IFASMFDL might not dynamically detect the image that was buffering data.

To ensure that all systems writing to a log stream are accounted for, despite any buffering that has occurred, use the SID(xxxx) option to specify each system of interest.

SMARTENDPOINT processing uses the following rules to find the logical endpoint in the log stream:

- Take the specified end time plus the SMARTEPOVER value. That is the logical smart endpoint time.

- For each SID found in a log stream, keep a table entry and mark that SID as complete when all SIDs have hit the smart endpoint time. After all SIDs are accounted for, stop reading.

This process occurs on a per-log stream basis.

14.3.2 SMARTEPOVER processing

To determine which value to specify for the SMARTEPOVER parameter, take into account the following considerations:

- If no SIDs are specified, you can set this value to double the maximum MAXDORM value of any image recording into the log stream. This allows for the best results from SID auto-detection.
  - MAXDORM is an SMFPRMxx parmlib member parameter. It applies to SMF data set recording and SMF log stream recording.

    MAXDORM specifies the amount of real time that SMF allows data to remain in an SMF buffer before it is written to a recording data set or a log stream, where mm is real time in minutes and ss is seconds.

    NOMAXDORM specifies that the data remains in the buffer until the buffer is full. The size of the non-full buffer written out when the MAXDORM reached is one control interval (CI).

    ```
    MAXDORM (mmss)|NOMAXDORM
    ```

    Value range: 0001-5959.

    Default: MAXDORM (3000). This indicates 30 minutes.
- If all SIDs are specified, this value can be minimized down to zero (0).
- If only a single SID is recorded into this log stream (for example, in a DASD-only log stream), this value can be minimized to zero.

This keyword only applies when SMARTENDPOINT is specified, as shown in Figure 14-7.

```
SMARTEPOVER(hhmm)
hhmm = 0000-0200 (0 to 2hs), default is 0200
```

Figure 14-7 SMARTEPOVER parameter

SMARTENDPOINT processing uses the following rules to find the logical endpoint in the log stream:

- Take the specified end time plus the SMARTEPOVER value. That is the logical smart endpoint time.
- For each SID found in a log stream, keep a table entry and mark that SID as complete when all SIDs have hit the smart endpoint time.
- After all SIDs are accounted for, stop reading.

This process occurs on a per-log stream basis.

IFA844I message

New message IFA844I is issued when SMARTENDPOINT processing is used. It lists the SIDs encountered in each log stream that was processed with SMARTENDPOINT.

```
IFA844I THE FOLLOWING SIDS ARE PRESENT IN lsname
sid1 sid2 sid3 sid4 ....
```

A multiple-line message is displayed listing the SID for each z/OS image that had records present in the log stream during the processing time period that is reported in the IFASMFDL program. The SMARTENDPOINT and SMARTEPOVER options for IFASMFDL can be used to control how much data is read from the log stream.

15

z/OS UNIX System Services

z/OS UNIX System Services, which is an element of z/OS, is a UNIX operating environment. It is implemented within the z/OS operating system and is also known as z/OS UNIX.

zFS can be used for all levels of the z/OS UNIX System Services hierarchy (including the root file system). zFS has higher performance characteristics than HFS, and is the strategic file system. For this reason HFS might not be supported in future releases, leading to migrating the remaining HFS file systems to zFS.

The z/OS support enables two open systems interfaces on the z/OS operating system:

- An application program interface (API)
- An interactive shell interface

This chapter provides information about changed and updated functions for z/OS UNIX System Services and zFS, as listed here:

- zFS installation changes with z/OS V1R13
- zFS internal restart
- zFS automatic re-enablement of disabled aggregates
- zFS statement of direction
- Add symbolic links to shared root filesystem
- Lost message detection
- Provide script utility to log session
- Enhancements to the **D OMVS,W** command
- RACF support for z/OS UNIX user mounts
- Enhancements for mount and unmount processing
- z/OS UNIX System Services file system access
- IPv4 IP_PKTINFO support for AF_INET UDP sockets
- Option on vi to edit ASCII files

15.1 zFS installation changes for z/OS V1R13

With z/OS V1R13, the following enhancements to zFS are:

- zFS Direct I/O (DIO)

  A new performance improvement for the UNIX System Services shared file system environment using zFS read-write sysplex-aware file systems is called zFS Direct I/O (DIO).

  This new support provides the ability to directly read and write zFS data from any system in a sysplex shared file system environment.

  In a sysplex shared file system environment, it should not matter where an application runs, or which system owns a file system. However, in current systems, location is important due to function shipping performance and file system ownership that needs to be managed.

- zFS internal restart

  There is an availability improvement for zFS internal failures.

- zFS automatic re-enablement of disabled aggregates

  There is an availability improvement for zFS aggregate failures.

Note: With z/OS V1R13, these enhancements especially mean much less concern about file system ownership and where applications execute in a sysplex shared file system environment.

15.2 zFS sysplex support with releases prior to z/OS V1R13

This section reviews a brief history of recent enhancements.

zFS in z/OS V1R11 introduced support for zFS read-write sysplex-aware file systems with a parameter setting of **`sysplex=on`**. This provided a performance improvement especially in the area of read performance. It also made file system ownership less important.

zFS R11 with APAR OA29619 improved the granularity of using zFS read-write sysplex-aware file systems with the new setting of **`sysplex=filesys`**. This allowed you to individually choose which zFS read-write file systems are sysplex-aware and which are not.

Important: Running zFS sysplex-aware on a file system basis, using **`sysplex=filesys`**, is the preferred option for a shared file system environment.

zFS in z/OS V1R13 introduces the new Direct I/O function. This enhances support for zFS read-write sysplex-aware file systems. When all systems are at level z/OS V1R13, zFS can directly read and write sysplex-aware file systems from all systems in the shared file system environment. This makes file system ownership much less important.

Note: zFS in z/OS V1R13 always runs with setting **`sysplex=filesys`** in a UNIX System Services shared file system environment.

15.2.1 File system ownership

IBM defines a file system owner as the system that coordinates sysplex activity for a particular file system. In a shared file system environment, there is also the concept of file system ownership. The owner of a file system is the first system that processes the mount. This system always accesses the file system locally. That is, the system does not access the file system through a remote system. Other non-owning systems in the sysplex access the file system either locally or through the remote owning system, depending on the PFS and the mount mode.

The file system owner is the system to which file requests are forwarded when the file system is mounted non-sysplex aware. Having the appropriate owner is important for performance when the file system is mounted read-write and non-sysplex aware. The term z/OS UNIX file system owner refers to the owner of the zFS file system as z/OS UNIX recognizes it. This is typically the system where the file system is first mounted, but it can differ from the zFS file system owner.

zFS file system owner

zFS has its own concept of file system ownership, called the zFS file system owner. This is also typically the system where the file system is first mounted in a sysplex-aware environment. File requests to sysplex-aware file systems are sent directly to the local zFS PFS, rather than being forwarded to the z/OS UNIX file system owner. This concept is shown in Figure 15-2.

The local zFS PFS forwards the request to the zFS file system owner, if necessary. The z/OS UNIX file system owner can be other than the zFS file system owner. (In reality, zFS owns aggregates. Generally, we simplify this to say zFS file system owner because, in most cases, zFS compatibility mode aggregates only have a single file system.)

z/OS UNIX file system owner

The term z/OS UNIX file system owner refers to the owner of the zFS file system as z/OS UNIX knows it. This is typically the system where the file system is first mounted.

15.2.2 Sysplex-unaware read-write file system

Figure 15-1 shows the original behavior of clients accessing a zFS file system through UNIX System Services function shipping using XCF and the XPFS file system interface. This is also the case if a zFS is started with sysplex=off in previous releases. This is an example of function shipping the requests, as follows:

Function shipping Function shipping means that a request is forwarded to the owning system and the response is returned back to the requestor through XCF communications.

Sysplex-unaware A file system is sysplex-unaware if the PFS (Physical File System) supporting that file system requires it to be accessed through the remote owning system from all other systems in a sysplex (allowing only one connection for update at a time) for a particular mode (read-only or read-write). The system that connects to the file system is called the file system owner. Access by other systems is provided through XCF communication with the file system owner.

For a non-sysplex aware zFS file system, file requests for read-write mounted file systems are function shipped to the owning system by z/OS UNIX. The owning system is the only system where the file

system is locally mounted and the only system that does I/O to the file system.

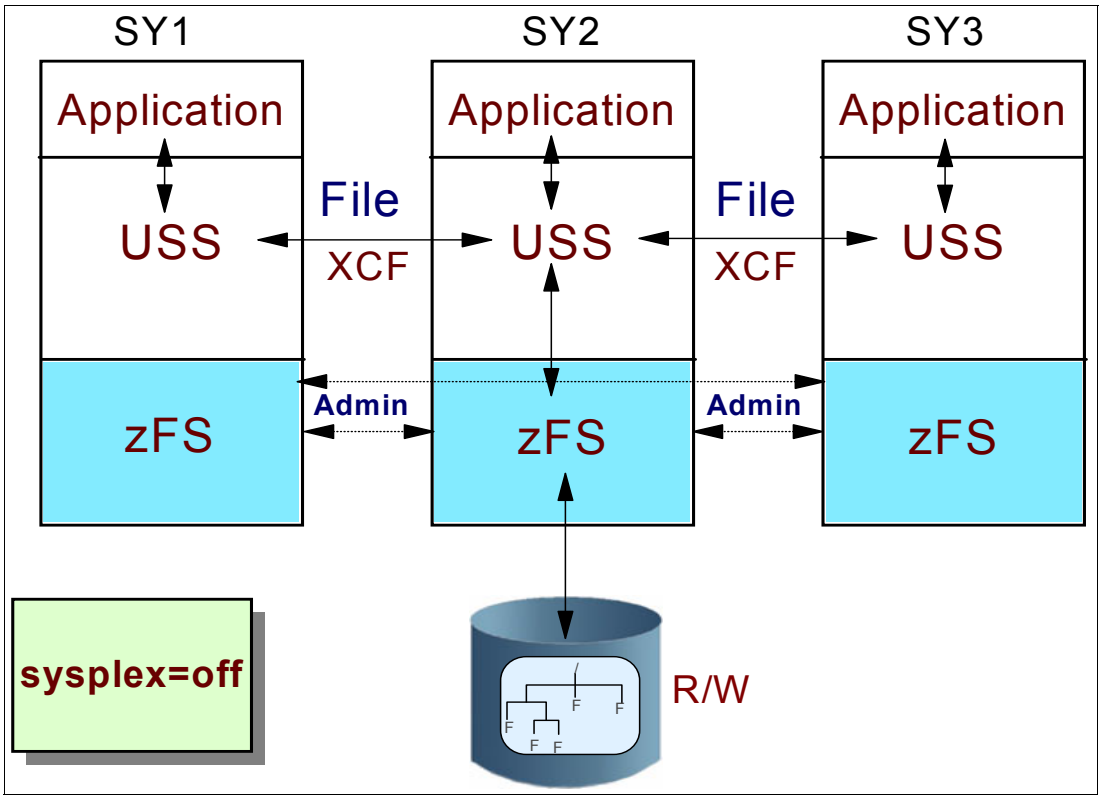


Figure 15-1 Sysplex-unaware read-write file system

Note: With function shipping, this means there is significant overhead. Thus, it is important that the system performing the most accesses is the owner of the file system.

15.2.3 z/OS V1R11 sysplex-aware read-write file system

Figure 15-2 shows the behavior as it was introduced with zFS in z/OS V1R11 with zFS started as sysplex-aware by using the setting **`sysplex=on`**.

- When zFS runs sysplex-aware (for read-write) on all systems, a read-write mounted sysplex-aware file system is locally mounted on all systems. There is still a z/OS UNIX owning system, but there is no z/OS UNIX function shipping to the owner.
- Instead, requests from applications on any system are sent directly to the local zFS on each system. This means it is now the responsibility of the local zFS to determine how to access the file system.
- One of the systems is known as the zFS owning system. This is the system where all I/O to the file system is done. zFS uses function shipping to the zFS owning system to access the file system.

- However, each zFS client system has a local cache where it keeps the most recently read file system information. Therefore, in many cases, when the data is still in the cache, zFS can avoid the zFS function shipping and satisfy the request locally.

Sysplex-aware Sysplex-aware pertains to a physical file system that handles file requests for mounted file systems locally instead of function shipping requests through z/OS UNIX.

Beginning with z/OS V1R11, zFS can be sysplex-aware for read-write mounted file systems. zFS can be configured to treat all zFS read-write mounted file systems owned on that system as sysplex-aware file systems.

Alternatively, zFS can be configured (zFS IOEFSPRM option sysplex=filesys) to allow certain zFS read-write mounted file systems owned on that system to be sysplex-aware file systems, and others to be non-sysplex aware file systems. zFS must be running sysplex-aware for read-write to allow a zFS read-write file system to be mounted as sysplex-aware.

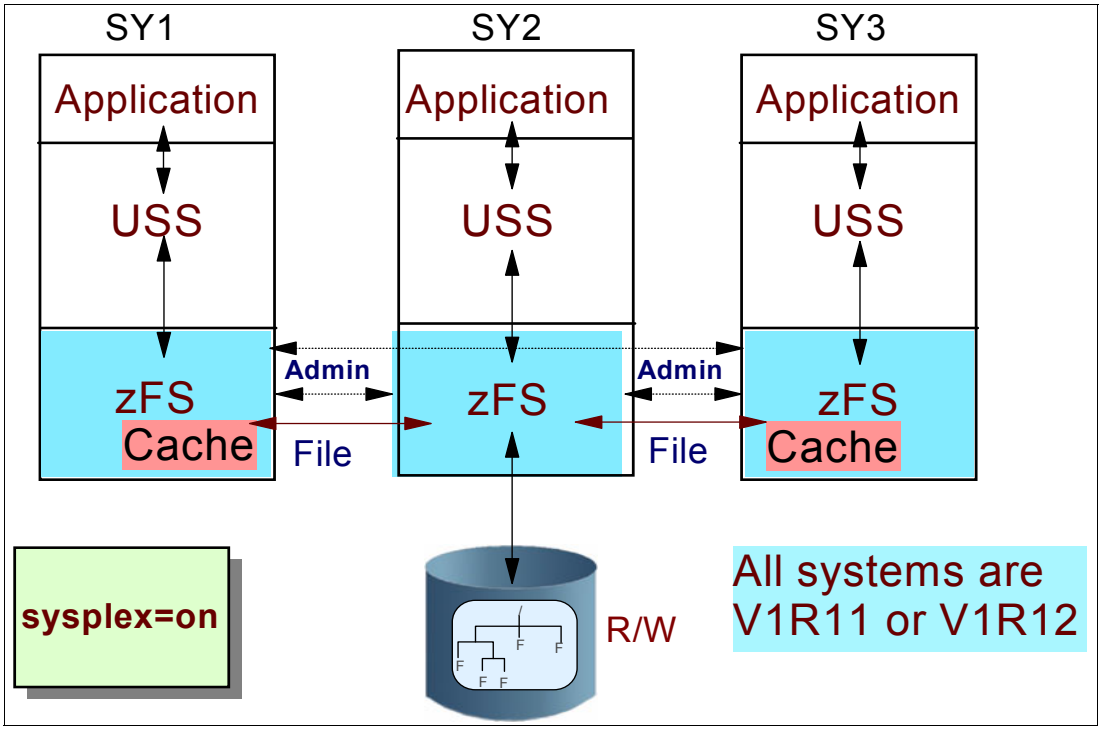


Figure 15-2 z/OS V1R11 sysplex-aware read-write file system

Note: Beginning with z/OS V1R13, zFS has enhanced its sysplex-aware support. For many file operations, zFS can now directly access zFS read-write mounted file systems in a shared file system environment from zFS client systems.

In z/OS V1R13, when zFS runs in a shared file system environment, zFS always runs sysplex-aware on a file system basis (sysplex=filesys).

15.2.4 zFS sysplex-aware on a file system basis

Figure 15-3 shows the new support of zFS being sysplex-aware on a file system basis that was introduced with APAR OA29619. Based on starting zFS with sysplex=filesys, this allows to decide whether a specific read-write file system is mounted sysplex-aware or not.

This is also the preferred way of starting zFS when the support is added.

- You see two zFS read-write file systems on a sysplex running zFS sysplex-aware on file system basis on all members.
- File system FS1 is mounted NORWSHARE and FS2 is mounted RWSHARE. They are both z/OS UNIX owned on SY2.
- The sysplex-unaware file system FS1 is only locally mounted on the z/OS UNIX owner SY2. Requests from z/OS UNIX clients SY1 and SY3 are function shipped to the z/OS UNIX owner SY2 by z/OS UNIX. A **`df -v`** command for file system FS1 from SY1 displays **`Client=Y`**.
- The sysplex-aware file system FS2 is locally mounted on all systems. z/OS UNIX never function ships requests to the z/OS UNIX owner SY2. A **`df -v`** command for the file system FS2 from SY1 displays **`Client=N`**.
- When you run zFS sysplex-aware on a file system basis on all your members, the zFS Physical File System initializes as sysplex-aware but it can individually determine which file system is sysplex-aware and which is not based on the system default setting for the file system sharemode (sysplex_filesys_sharemode) or the mount parameter RWSHARE or NORWSHARE.
- Based on file system activity, the zFS ownership of a sysplex-aware zFS file system can move to another system and thus differ from the z/OS UNIX ownership.

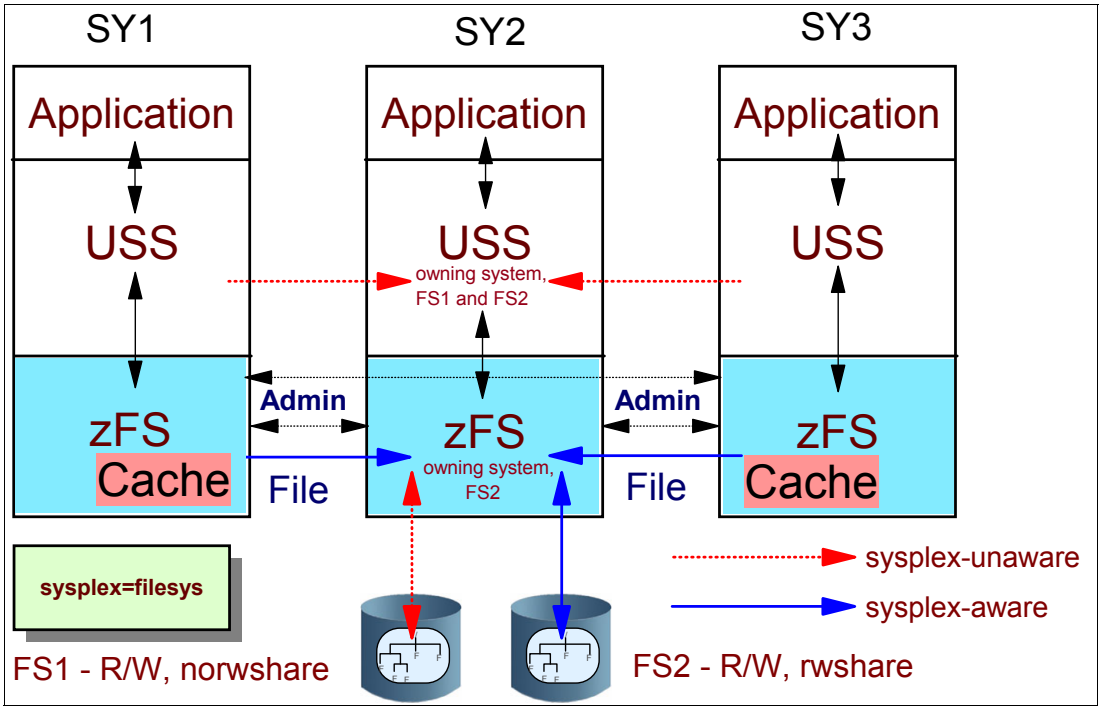


Figure 15-3 zFS sysplex-aware on a file system basis

15.3 z/OS V1R13 Direct I/O sysplex-aware read-write file system

Figure 15-4 shows that with this new support in z/OS V1R13, and if the zFS file system is mounted sysplex-aware, then any client zFS can directly access the file system for reading and writing data and partially also reading metadata.

Note: Metadata updates are always sent to the zFS owning system.

So, any user or application in a z/OS V1R13 environment can exploit the functionality for zFS read-write sysplex-aware file systems.

Note: The client cache is no longer used in z/OS V1R13 if the owner supports DIO. The client reply storage is still used and the new default for client_reply_storage is 10 M.

Nevertheless, if a zFS in z/OS V1R11 or z/OS V1R12 system is accessing data in a zFS file system owned at a z/OS V1R13 system level, it still uses the client cache and also cannot do a DIO. This is also true if the file system is owned by a z/OS V1R11 or z/OS V1R12 system.

Important:

- Aggregate movement is expected to happen less often. Only access operations to metadata, especially updates, are taken into account.
- In z/OS V1R13 with DIO being active, zFS does not automatically move a zFS file system to a down-level system for performance reasons.

In Figure 15-4, all systems (SY1, SY2 and SY3) are running z/OS V1R13. In this case, the sysplex-aware file system (FS2) is zFS owned by SY2 and is being directly accessed from SY1 and SY3.

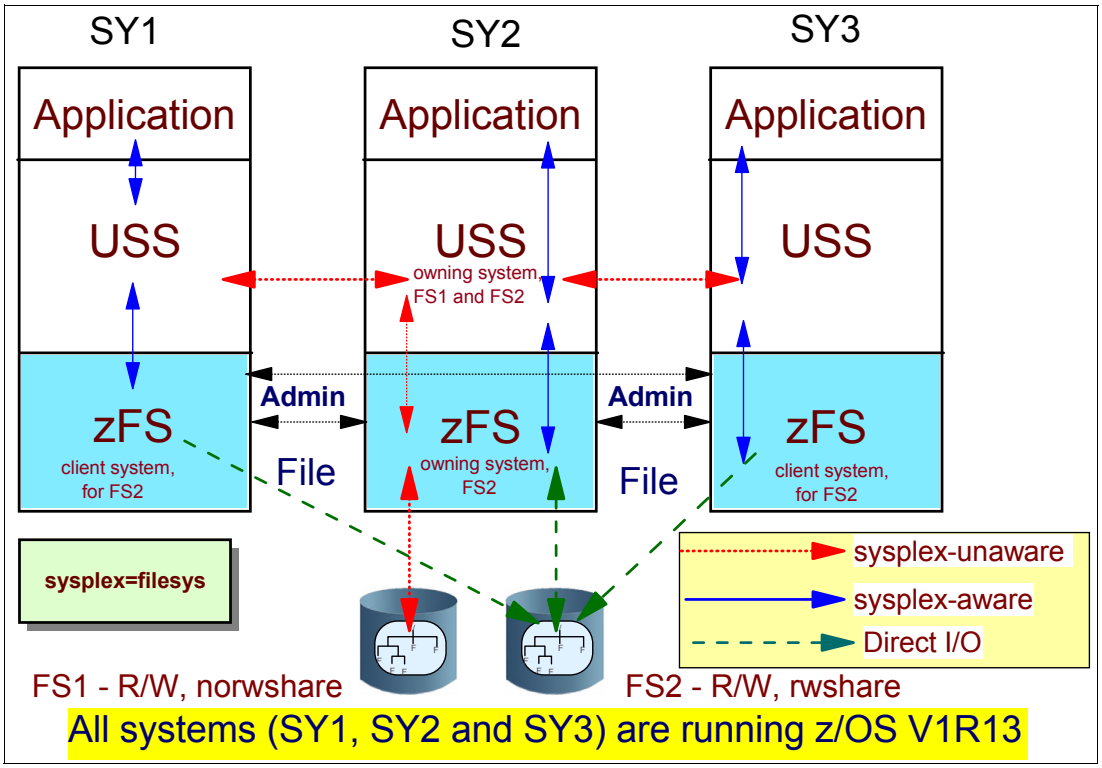


Figure 15-4 z/OS V1R13 Direct I/O sysplex-aware read-write file system

15.4 zFS internal restart

This is an availability improvement in case of zFS internal failures with z/OS V1R13.

In previous releases, when zFS had an internal failure, that required zFS to abort and have z/OS UNIX restart zFS. This cleared the failure but can cause several file systems to be unmounted.

zFS in z/OS V1R13 has been restructured so that internally, a controller task attaches the zFS kernel task. When an internal failure occurs in the zFS kernel, the zFS controller task can stop and restart the zFS kernel and internally remount the zFS file systems.

- zFS can recover from internal failures without losing mounts.
- zFS time to recovery is improved.

Previous behavior on zFS internal errors or abort

In previous releases, zFS restart can be explicitly invoked to resolve hangs involving zFS through the operator **`F ZFS,ABORT`** command.

Figure 15-5 shows the previous behavior and sequence of processing in the zFS address space.

- The zFS kernel is the main task in the zFS address space.
- When zFS has to go down based on a failure or an abort request, there is nothing left to initiate a restart.
- z/OS UNIX has to restart zFS.

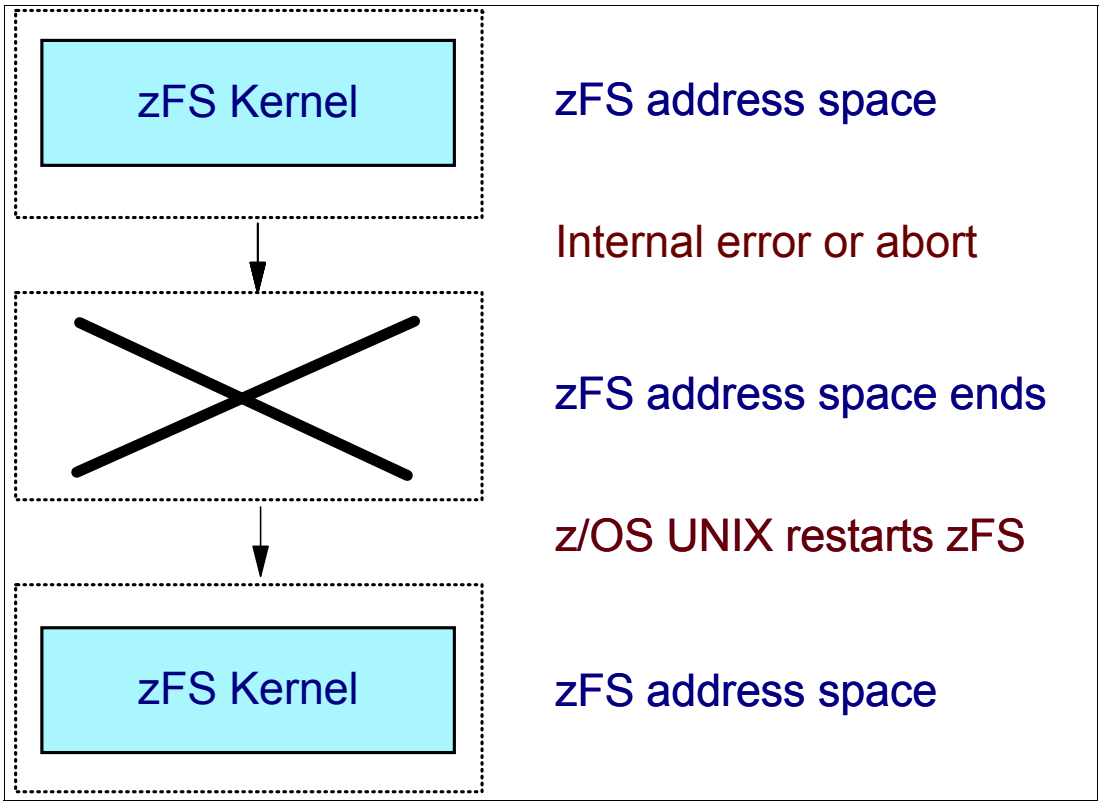


Figure 15-5 Previous behavior on zFS internal errors or abort

15.4.1 zFS internal restart processing with z/OS V1R13

With z/OS V1R13, zFS will attempt an internal restart rather than a stop and restart of the zFS address space. zFS file systems should not become unmounted due to an internal restart. In addition, a zFS internal error that might have been a terminating exception will attempt to perform an internal restart.

The following conditions occur after an internal zFS failure:

- zFS internal restart occurs automatically if zFS detects an internal failure.
- zFS ownership of zFS sysplex-aware file systems can change during internal restart.
- As before, applications can see failures as a result of zFS restart.

Note: You can query the number of zFS internal restarts that have occurred since zFS was started by using the `F ZFS,QUERY,STATUS` command.

Figure 15-6 shows the new behavior and sequence of processing in the zFS address space.

- The main task started in the zFS address space now is the controller task. This zFS controller then attaches the zFS kernel.
- On an internal error or abort, the zFS kernel is ended but the controller is still active.
- The controller then initiates the zFS kernel to be restarted. This recover processing avoids losing mounted zFS file systems.

Notes:

1. zFS does not read (again) the zFS parameter settings in the IOEPRMxx parmlib member on an internal restart.
2. If a zFS internal restart does occur, all zFS mounted file systems remain mounted.
3. The **F ZFS,HANGBREAK** command now causes a zFS internal restart. This produces the same result as issuing a **F ZFS,ABORT** command.

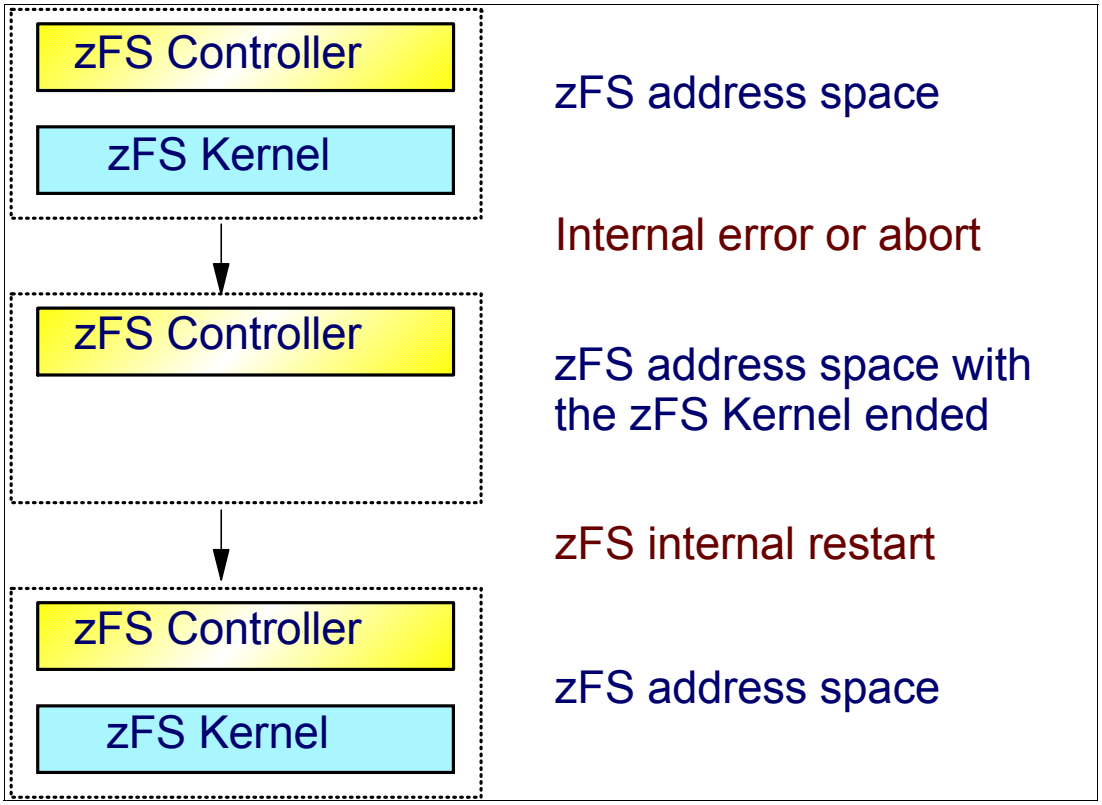


Figure 15-6 zFS internal restart

15.4.2 Forcing a zFS internal restart

Figure 15-7 shows information about the zFS environment on system SC75 from system SC74. Most zFS file systems are mounted at SC74.

```
$> echo This is system $(sysvar SYSNAME).
This is system SC74.
$> sudo /usr/sbin/mount -qv ~
----A- HERING.ZFS /u/hering
$> rxlsaggr | grep " SC75"
PLEX75.SC75.SYSTEM.ZFS                          SC75      R/W Sysplex-aware
HERING.ZFS                                      SC75      R/W Sysplex-aware
JES3.HFS                                        SC75      R/W Sysplex-aware
LUTZ.ZFS                                        SC75      R/W
$> zfsadm configquery -trace_table_size -system SC75
IOEZ00317I The value for configuration option -trace_table_size is 64M.
$> zfsadm configquery -trace_dsn -system SC75
IOEZ00317I The value for configuration option -trace_dsn is PLEX75.SC75.ZFS.TRACE.
$> cn "route sc75,f zfs,query,status"
 IOEZ00736I zFS kernel initiated at Apr 29 11:16:05 2011
   Current status: active
   Internal restart count 0
 IOEZ00025I zFS kernel: MODIFY command - QUERY,STATUS completed successfully.
$>
```

Figure 15-7 zFS information for system SC75

In Figure 15-8 an interactive REXX environment is started, a file in the home file system is opened, and a zFS restart is forced by running the **f zfs,hangbreak** command.

```
$> rexx
SH> "pwd"
/u/hering
SH> s "open test.zfs.internal.restart" o_creat+o_wronly 640
OMVS Return Value (retval) = 3
SH> data = "XxXxX"
SH> s "write 3 data"
OMVS Return Value (retval) = 5
SH> "cn route sc75,f zfs,hangbreak"
 IOEZ00338A zFS kernel: restarting exception 2C3 occurred, reason EA150384 abend
 psw 77C1000 9258961E
...
IOEZ00041I No start record, trace wrapped, total records 1677720, 67108800 bytesto
 format.
SH> s "write 3 data"
OMVS Return Value (retval) = 5
SH> s "write 3 esc_n"
OMVS Return Value (retval) = 1
SH> s "close 3"
OMVS Return Value (retval) = 0
SH> exit
$>
```

Figure 15-8 Forcing a zFS restart in system SC75

Figure 15-9 shows a first excerpt of the operlog with SC74 trace and SC75 dump information.

```
F ZFS,HANGBREAK
IOEZ00338A zFS kernel: restarting exception 2C3 occurred, reason
EA150384 abend psw 77C1000 9258961E
IOEZ00337E zFS kernel: non-terminating exception 2C3 occurred, reason
EA2F0385 abend psw 77C1000 9258961E
IOEZ00064I General Registers R0: 4000000 42C3000 71D6E000 124967EA
...
IOEZ00065I zFS kernel recovery: estae psw 70C0000 924FC7A8
...
IOEZ00034I zFS kern: printing contents of trace to dataset
<PLEX75.SC74.ZFS.TRACE(ZFSKNT01)>
...
IOEZ00036I Printing contents of table at address 7EC5FF50 name: Main Trace Table
IOEZ00041I No start record, trace wrapped, total records 1677720,
67108800 bytes to format.
IEA794I SVC DUMP HAS CAPTURED: 318
DUMPID=001 REQUESTED BY JOB (ZFS     )
DUMP TITLE=zFS abend 02C3 reason EA150384 May 13 18:47:57 in module IOEFSCM at
offset 0010A61E
IOEZ00334I Return code and reason code for dump is 40000000
```

Figure 15-9 Excerpt 1 of operlog during zFS restart

Figure 15-10 shows a second excerpt of the operlog with SC75 trace information. A dump for zFS in SC74 is forced by SC75 in addition.

```
IOEZ00034I zFS kern: printing contents of trace to dataset
<PLEX75.SC75.ZFS.TRACE(ZFSKNT01)>
IOEZ00036I Printing contents of table at address 7EC58F50 name: Main Trace Table
IEF196I IGD101I SMS ALLOCATED TO DDNAME (SYS00004)
IEF196I         DSN (DUMP.D110513.H18.SC75.ZFS.S00001            )
IEF196I         STORCLAS (DUMPS) MGMTCLAS (        ) DATACLAS (        )
IEF196I         VOL SER NOS= STPDP2
IOEZ00041I No start record, trace wrapped, total records 1677720,
67108800 bytes to format.
...
IOEZ00043I zFS kern: print of in-memory trace table has completed.
...
IOEZ00051I An error occurred in program IOEFSKN
IOEZ00357I Successfully left group IOEZFS.
IOEZ00057I zFS kernel program IOEFSKN is ending.
IOEZ00387E System SC75 has left group IOEZFS, aggregate recovery in progress.
...
IOEZ00579I Restarting zFS kernel, Restart count 1
```

Figure 15-10 Excerpt 2 of operlog during zFS restart

Figure 15-11 shows the completion of the zFS restart and the takeover of aggregate HERING.ZFS by SC74 with recovery and resume processing.

```
 IOEZ00388I Aggregate takeover being attempted for aggregate HERING.ZFS
...
 IOEZ00397I Recovery statistics for HERING.ZFS:
   elapsed time 6 ms 1 log pages
   2 log records, 1 data blocks modified
   1 redo-data, 0 redo-fill records
   0 undo records, 0 unwritten blocks
...
 IOEZ00559I zFS kernel: Initializing z/OS    zSeries File System 340
 Version 01.13.00 Service Level z130224 - HZFS3D0.
 Created on Thu Mar 3 09:24:20 EST 2011.
...
*IOEZ00727I Pre-processing 32768 vnodes for file system restart 352
 IOEZ00388I Aggregate takeover being attempted for aggregate JES3.HFS
 IOEZ00044I Aggregate HERING.ZFS attached successfully.
 IOEZ00044I Aggregate JES3.HFS attached successfully.
 IOEZ00617I zFS is running sysplex filesys,norwshare with interface level 4
...
*IOEZ00646I zFS Kernel is restarting 34 file systems 361
 IOEZ00055I zFS kernel: initialization complete.
...
IOEZ00728I Resuming user operations for file system HERING.ZFS
   Waking 0 waiting tasks
   Re-cached 4 vnodes for this file system
 IOEZ00728I Resuming user operations for file system PFA.HFS
   Waking 1 waiting tasks
   Re-cached 2011 vnodes for this file system
...
 IOEZ00731I zFS internal restart complete.
```

Figure 15-11 Excerpt 3 of operlog during zFS restart

In addition, note the following information.

Notes:

- Aggregate takeover took place for PLEX75.SC75.SYSTEM.ZFS and JES3.HFS in the same way as for HERING.ZFS.
- Recovery took place for PLEX75.SC75.SYSTEM.ZFS and for LUTZ.ZFS as well.
- Resuming user operations was done for most of the zFS file systems in the UNIX System Services sysplex file system sharing environment.

15.4.3 Results of the zFS restart

Figure 15-12 lists the essential results of the zFS restart.

```
$> cat test.zfs.internal.restart | od -tcx1 | head -12
0000000000    \0  \0  \0  \0  \0   X   x   X   x   X  \n
              00  00  00  00  00  E7  A7  E7  A7  E7  15
$> cn "route sc75,f zfs,query,status"
 IOEZ00736I zFS kernel initiated at Apr 29 11:16:05 2011
   Current status: active
   Internal restart count 1 (Last restart: May 13 18:48:08 2011)
 IOEZ00025I zFS kernel: MODIFY command - QUERY,STATUS completed successfully.
$> cn "f zfs,query,status"
 IOEZ00736I zFS kernel initiated at Apr 29 11:08:56 2011
   Current status: active
   Internal restart count 0
 IOEZ00025I zFS kernel: MODIFY command - QUERY,STATUS completed successfully.
$> rxlsaggr | grep " SC75"
LUTZ.ZFS                                        SC75      R/W
$> rxlsaggr | grep PLEX75.SC75.SYSTEM.ZFS
PLEX75.SC75.SYSTEM.ZFS                          SC74      R/W Sysplex-aware
$> rxlsaggr | grep HERING.ZFS
HERING.ZFS                                      SC74      R/W Sysplex-aware
$> rxlsaggr | grep JES3.HFS
JES3.HFS                                        SC74      R/W Sysplex-aware
$>
```

Figure 15-12 Results of the zFS restart

Note the following information for file systems and I/O processing:

- The zFS ownership of file systems previously owned by the system that performed a zFS restart might go to another system.
- If so, this happens independent of the automove setting and the UNIX System Services ownership of the file system.
- Active I/O operations can fail. Data that is not on DASD already can be lost.

15.5 zFS automatic re-enablement of disabled aggregates

This is an availability improvement for zFS aggregate failures. With z/OS V1R13 systems, if an internal problem occurs, zFS will go down and automatically restart. It will internally remount any zFS file systems that were locally mounted, without requiring any UNIX System Services support. The zFS ownership for aggregates that are owned on the system that is internally restarted will be moved (by zFS for sysplex-aware file system) to another system.

If zFS detects a problem on an aggregate that is mounted R/W, zFS will attempt to isolate the failure. As a result, zFS might mark an aggregate unavailable and issue message IOEZ00422E, as shown in the following example.

```
IOEZ00422E Aggregate PLEX.JMS.AGGR001.LDS0001 disabled
```

Disabled aggregates

When an aggregate is disabled, applications cannot read from, or write to, the aggregate. Other aggregates that are not involved in the failure will remain available. However, the disabled aggregate will be unavailable for reading and writing until it is unmounted and mounted. Beginning with z/OS V1R13, if the disabled aggregate is zFS owned on a zFS V1R13 system, zFS will attempt to automatically re-enable the disabled aggregate and make it available again for use.

An aggregate that has been disabled might potentially be corrupted, even if it has been disabled and remounted. To be sure the aggregate is internally consistent, you might want run the `ioeagslv` utility against the aggregate that was disabled.

To determine if the aggregate has been marked as disabled, use the **`zfsadm lsaggr`** command or the **`zfsadm aggrinfo`** command.

z/OS V1R13 and disabled aggregates

If the aggregate becomes disabled on a zFS R13 system, zFS will attempt to automatically re-enable the disabled aggregate. It will either request that another V1R13, V1R12, or V1R11 system in the shared file system environment take over the aggregate (if it is sysplex-aware), or it will attempt an internal remount samemode. This action recovers the aggregate and it will no longer be disabled.

Behavior in previous releases

In previous releases, when zFS had an internal failure that required it to disable an aggregate, you had two possibilities to solve the problem:

- The file system needed to be explicitly unmounted and then remounted to recover.
- Or you needed to perform remount samemode processing for the file system.

15.5.1 Automatic re-enablement of disabled aggregates considerations

With zFS R13, the implementation works as described here:

- zFS automatic re-enablement of disabled aggregates occurs automatically if zFS disables an aggregate.

  So, zFS can recover a disabled aggregate without administrator intervention.
- zFS ownership of a disabled zFS sysplex-aware file system can change during automatic re-enablement of the disabled aggregate.
- As in previous systems, applications can see failures as a result of zFS disabling an aggregate. Run the salvager utility IOEAGSLV against the aggregate at the earliest convenience.

Note: Automatic re-enablement of the disabled aggregate that becomes disabled again is tried up to three times.

If this occurs, file system must be manually unmounted and mounted again.

Sample messages on automatic re-enablement

Figure 15-13 shows automatic re-enablement messages when a zFS aggregates becomes disabled.

```
IOEZ00422E Aggregate HERING.TEST.ZFS disabled
IOEZ00548I Requesting that SC75 takeover aggregate HERING.TEST.ZFS
IOEZ00388I Aggregate takeover being attempted for aggregate HERING.TEST.ZFS
IOEZ00044I Aggregate HERING.TEST.ZFS attached successfully.
```

Figure 15-13 Messages on automatic re-enablement

Figure 15-14 shows the message that is displayed when automatic re-enablement of a zFS aggregate is halted after repeated failures.

```
IOEZ00422E Aggregate HERING.TEST.ZFS disabled
IOEZ00746E Automatic re-enablement of file system HERING.TEST.ZFS
halted after repeated failures
```

Figure 15-14 Halting automatic re-enablement after repeated failures

In this situation you can use one of the following commands, just as you were able to before this new function was introduced:

```
tsocmd "unmount filesystem('hering.test.zfs') remount(samemode)"
sudo rexx 's "unmount HERING.TEST.ZFS" mtm_samemode'
sudo /usr/sbin/unmount -f HERING.TEST.ZFS
```

Notes:

- A utility, `rexx`, allows you to run commands interactively. It is available from the ITSO tools disk at:

  `ftp://www.redbooks.ibm.com/redbooks/SG247035/`

- When an unmount is performed, a new mount is needed to make the file system available again.

15.6 Migration considerations

The migration to zFS R13 is a two-step process:

1. Install toleration APAR OA32925 (PTF UA55765) on all zFS R11 and R12 systems and make it active with a rolling IPL, for example.
2. Change your zFS IOEFSPRM file to specify **`sysplex=filesys`** on all systems and make it active with a rolling IPL, if not used already, as was recommended when this new setting was introduced by APAR OA29619.

At this point you can bring in z/OS V1R13 and zFS R13.

Unlike previous releases, zFS R13 does not store data in 1 K fragments. Instead, it is stored in 8 K blocks. zFS R13 can read data stored in fragments. However, when the data is updated, it is moved into 8 K blocks. See “DASD space with z/OS V1R13” on page 299 for more information about this topic.

Note: zFS user data is stored inline when the file is 52 bytes or less. Otherwise, 8 K blocks are used.

Because previous releases of zFS can read an 8 K block that is not full, no toleration support is required on those systems. Also in previous releases, when zFS stored data in fragments, data from multiple files typically resided in separate 8 K blocks.

However, there are certain cases when zFS R13 requires more DASD space than zFS in previous releases. This is a major reason to change the default for `aggrgrow`.

Important: The default for aggrgrow has been changed from **`aggrgrow=off`** to **`aggrgrow=on`**.

Installation considerations

Beginning with z/OS V1R13, zFS has enhanced its sysplex-aware support. For many file operations, zFS can now directly access zFS read-write mounted file systems in a shared file system environment from zFS client systems. With z/OS V1R13, when zFS runs in a shared file system environment, zFS always runs sysplex-aware on a file system basis (sysplex=filesys).

This offers the following possibilities:

- You can choose to have none of your zFS read-write file systems be sysplex-aware (this is the default).
- You can choose to have certain zFS read-write file systems be sysplex-aware and others be non-sysplex aware.
- You can choose to have all zFS read-write file systems be sysplex-aware.

15.6.1 zFS health check for sysplex=filesys

In both z/OSV1R11 and z/OSV1R12, zFS must run **`sysplex=filesys`** before zFS R13 can be brought into the shared file system environment.

Therefore, a migration health check has been provided in z/OS V1R11 and z/OS V1R12 to check whether zFS **`sysplex=filesys`** has been specified. This zFS R13 migration health check is provided as service to zFS in V1R11 and z/OS V1R12. The zFS APAR is OA35465 and PTF UA59383.

This helps to ensure that the migration action is executed, and to avoid a problem bringing zFS R13 into the shared file system environment.

Attention: If a problem occurs when zFS is running sysplex=filesys on a z/OS V1R11 or z/OS V1R12 system, you can perform the following steps:

1. Remove the sysplex specification from each system or specify sysplex=off on each system (this is equivalent to the old default prior to z/OS V1R13).
2. Perform a rolling IPL or restart zFS on each system.

However, be aware that this procedure cannot be performed after zFS on the z/OS V1R13 system has joined the sysplex.

Also, if you try to start zFS on another z/OS V1R13 system after you have changed zFS to sysplex=off on the z/OS V1R11 or z/OS V1R12 system, zFS on z/OS V1R13 will not start. This occurs because zFS on z/OS V1R13 requires all other systems to be zFS sysplex=filesys.

15.6.2 DASD space with z/OS V1R13

zFS R13 does not store data in 1 K fragments, as done in previous releases. z/OS V1R13 systems store data in 8 K blocks. zFS R13 can read data stored in fragments; however, when the data is updated, it is moved into 8 K blocks.

Note: Because previous releases of zFS can read an 8 K block that is not full, no toleration support is required on those systems. The zFS data that is stored in fragments typically resided in separate 8 K blocks.

The data for 1000 small files uses 1000 K in zFS R11, assuming they are perfectly packed into 8 K blocks. The data for 1000 small files uses 8000 K in R13. This means that you need about 9.5 cylinders more with zFS R13, if you take into account that there is 720 K data per cylinder.

Note: In R13, DASD space is also used for the directory names, the anodes, and so on. This is the same as in prior releases.

Nevertheless, a positive effect results from this change for the 1 K fragments, because zFS no longer uses them in zFS R13. It was always important to look for the aggregate full percentage, and not the file system full numbers. The file system full numbers can be misleading due to the use of the 1 K fragments.

DASD space considerations

With zFS R13, more DASD space can be required than in previous releases if zFS R13 is in a mixed-release sysplex and becomes the owning system of a file system. This can occur if every file in the file system is 1 K or less. zFS R13 can require up to four times the DASD storage as previous releases.

As another example, because HFS uses 4 K blocks to store data and zFS uses 8 K blocks, if every file in the file system is 4 K or less, zFS R13 can require up to twice as much DASD space to store these files.

Typically, however, any increase in the DASD storage used by zFS R13 will be negligible. For example, the R13 version root file system copied using zFS R13 takes approximately 2% more space than the same file system copied using zFS R11. Also consider that zFS R13 packs multiple ACLs and symbolic links into an 8 K block, which previous releases did not do.

Note: If a zFS aggregate has been completely rewritten or defined in z/OS V1R13, the information provided by a file system full percentage is equivalent to an aggregate full percentage. Also, the z/OS UNIX command **df -kP *mp_directory*** will then provide a more accurate capacity percentage.

zfsspace tool

Use the `zfsspace` tool to scan zFS file systems for small files to help determine if an existing file system might need more DASD storage when migrating to z/OS V1R13. A file system with a large number of small files and high space utilization might require more space.

The `zfsspace` tool is available in text mode at:

```
ftp://public.dhe.ibm.com/s390/zos/tools/zfsspace/zfsspace.txt
```

A sample call of `zfsspace` is shown in Figure 15-15.

```
tso zfsspace /u/hering/b1kfill
HERING.ONEK.FILES
  small files: 1000
  extents....: yes
  aggrgrow...: yes
```

Figure 15-15 Listing information about files less than 7 K

Note: Place the `zfsspace` tool in a place where you run REXX execs. This tool can run under the shell, TSO, and SYSREXX.

15.6.3 Changes in zFS installation

Note the following installation changes in zFS R13:

- The zFS load modules have been moved from a PDS library **`hlq.SIOELMOD`** to a PDSE library named **`hlq.SIEALNKE`**.
- In previous releases, the command **`zfsadm`** was a binary in the z/OS UNIX file system. Now it is a shell script that executes **`IOEZADM`**, which is an entry with the sticky bit set on, so the IOEZADM load module is executed; see Figure 15-16.

```
$> command -V zfsadm
zfsadm is cached /bin/zfsadm
$> cat /bin/zfsadm | tail -3
PATH=$PATH:/usr/lpp/dfs/global/bin

IOEZADM "$@"
$> ls -l /usr/lpp/dfs/global/bin/IOEZADM | awk '{print $1}'
-rwxr-xr-t
$>
```

Figure 15-16 The zfsadm command

Note: As of z/OS V1R13, when the **`zfsadm`** command displays help text or a syntax error message, it will show the name of the command as IOEZADM, instead of zfsadm. This occurs because the **`zfsadm`** command is not a binary in the z/OS UNIX file system. Instead, it is a shell script that invokes IOEZADM.

IOEZADM is an entry that has the sticky bit on in the permissions. The sticky bit means that the IOEZADM module is actually found and executed from the user's STEPLIB, link pack area, or link list concatenation. (IOEZADM is usually located in SYS1.SIEALNKE.) However, you cannot execute IOEZADM from the shell because IOEZADM is not normally in your PATH.

15.6.4 Changes in IOEPRMxx configuration options

There are several changes for the zFS parameter specifications.

- The setting for **`dir_cache_size`** is ignored in z/OS V1R13. It is no longer needed.
- The default value for the **`meta_cache_size`** has been changed from 32 M to 64 M. This is the sum of the previous defaults for the directory cache and the metadata cache.

  Note: The metadata cache is now used for caching both metadata and directory data.

- Client metadata can be stored in metadata backing cache and the size of the cache can be dynamically changed.

Note: Metadata backing cache cannot be dynamically defined in z/OS V1R13. It must be specified in the zFS parameter file with a small size. However, a metadata backing cache is normally not needed.

- The setting for **nbs** is ignored now. It is always on and there is no performance penalty.
- The default for **client_cache_size** has been reduced from 128 M to 32 M. It is only needed for zFS file system owners of sysplex-aware file systems that are of previous releases.

15.6.5 zFS ownership versus z/OS UNIX ownership of file systems

For zFS read-write sysplex-aware file systems there is a concept of file system ownership, as explained here:

zFS owner zFS takes responsibility for determining how to access the data. This means that zFS must have the concept of a file system owner to coordinate file requests. That system is the zFS owner.The zFS owner is the system that coordinates file access.

z/OS UNIX owner z/OS UNIX still has its indication of owner, which is called the z/OS UNIX owner. The zFS owner is independent of the z/OS UNIX owner.

The z/OS UNIX owner generally does not have any performance implications when zFS runs sysplex-aware because file requests are sent to the local zFS rather than being function shipped to the z/OS UNIX owner

As shown in Figure 15-17, you can now effectively change the zFS ownership of a read-write sysplex-aware file system by using these steps:

1. Change the z/OS UNIX ownership to the system that you want to have the zFS ownership.
2. Then do a remount samemode.

```
$> rxdowner -d ~

MP Directory : /u/hering
File System  : HERING.ZFS
PFS Type     : ZFS
Local Sysname: SC74     - File System local-client=N
USS Owner    : SC74     - File System read-only=N
zFS Owner    : SC74     - Aggregate read-only=N, sysplex-aware=Y

$> sudo /usr/sbin/chmount -d SC75 ~
$> rxdowner -d ~

MP Directory : /u/hering
File System  : HERING.ZFS
PFS Type     : ZFS
Local Sysname: SC74     - File System local-client=N
USS Owner    : SC75     - File System read-only=N
zFS Owner    : SC74     - Aggregate read-only=N, sysplex-aware=Y

$> sudo /usr/sbin/chmount -s ~
$> rxdowner -d ~

MP Directory : /u/hering
File System  : HERING.ZFS
PFS Type     : ZFS
Local Sysname: SC74     - File System local-client=N
USS Owner    : SC75     - File System read-only=N
zFS Owner    : SC75     - Aggregate read-only=N, sysplex-aware=Y

$>
```

Figure 15-17 Switching the zFS ownership in z/OS V1R13

15.7 zFS statement of direction

Note the following Statement of Direction in R13.

Statement of Direction: “z/OS V1R13 is planned to be the last release to support multi-file system zSeries File System (zFS) aggregates, including zFS clones. Support for the zfsadm clone command and mount support for zFS file system data sets containing a cloned (.bak) file system will be removed. IBM recommends that you use copy functions such as pax and DFSMSdss to back up z/OS UNIX file systems to separate file systems. Support for zFS compatibility mode aggregates will remain.”

As a result, zFS multifile system aggregates are being removed. The zFS clone function is also being removed.

15.8 Performance comparisons with sysplex-sharing

This section presents a simple test scenario that can be easily set up and run in a UNIX System Services sysplex file system sharing environment. You can use this scenario to gain an understanding of the functions provided in zFS to improve performance when accessing a file system. The results demonstrate the effectiveness of the enhancements that have been added in z/OS V1R13, named “the Direct I/O.”

15.8.1 Setup and description of the test environments

For this test scenario, two systems (SC74 and SC75) with UNIX System Services sysplex file system sharing were used. First two file systems were created on the same volume, one of type HFS and another of type zFS. Next, two 750 MB files were created and filled with data in both file systems, as shown in Figure 15-18.

```
#> mkdir -pm 755 /u/hering/r13test/hfs
#> /usr/sbin/mount -o "sync(60)" -f OMVS.R13TEST.LARGE.HFS -t HFS \
> /u/hering/r13test/hfs
#> chmod 755 /u/hering/r13test/hfs
#> mkdir -m 755 /u/hering/r13test/zfs
#> /usr/sbin/mount -o noreadahead -f OMVS.R13TEST.LARGE.ZFS -t ZFS \
> /u/hering/r13test/zfs
#> /usr/sbin/mount -qv /u/hering/r13test/
----A- OMVS.R13TEST.LARGE.ZFS /u/hering/r13test/zfs
----A- OMVS.R13TEST.LARGE.HFS /u/hering/r13test/hfs
----A- HERING.ZFS          /u/hering
#> # Now creating and filling the files...
#> ls -l /u/hering/r13test/hfs
total 3072000
-rw-rw-rw-   1 HERING   SYS1     786432000 Mai  5 13:54 large_file_1
-rw-rw-rw-   1 HERING   SYS1     786432000 Mai  5 14:06 large_file_2
#> ls -l /u/hering/r13test/zfs
total 3073568
-rw-rw-rw-   1 HERING   SYS1     786432000 Mai  5 23:24 large_file_1
-rw-rw-rw-   1 HERING   SYS1     786432000 Mai  5 23:24 large_file_2
#> rexx 'Say "File size=" 786432000/(1024*1024)||"MB"'
File size= 750MB
```

Figure 15-18 Creating and filling the two HFS and the two zFS files

Note: For more detailed information about the setup, JCL, and procedures used, see *z/OS Distributed File Service zSeries File System Implementation*, SG24-6580.

User and client cache with V1R13

The `user_cache` and the old `client_cache` are shown in Figure 15-20. In z/OS V1R13, the cache is one cache for all situations, as explained here:

1. For any requests (at the zFS owner, also called the server, and at zFS clients), the `user_cache_size` cache is used.

   The default is 256 MB. The maximum size is 65536 MB (which is 64 GB).

2. For requests at a system that is not the zFS owner (these systems are called clients), the client cache that was used in z/OS V1R11 and V1R12 is no longer needed and has been removed. The user cache is used in this case now, as well.

For reference, Figure 15-19 shows a basic picture for zFS cache structures in z/OS V1R13. The **`F ZFS,QUERY,ALL`** command is detailed in Appendix B, “zFS commands” on page 809.

Note: Beginning with z/OS V1R13, the optional metadata backing cache cannot be created dynamically using the **`zfsadm config -metaback_cache_size`** command. You must specify the **`metaback_cache_size`** configuration option in your IOEFSPRM file to have a metadata backing cache.

You can still use the **`zfsadm config`** command to dynamically change the size of the metadata backing cache, when the metadata backing cache exists.

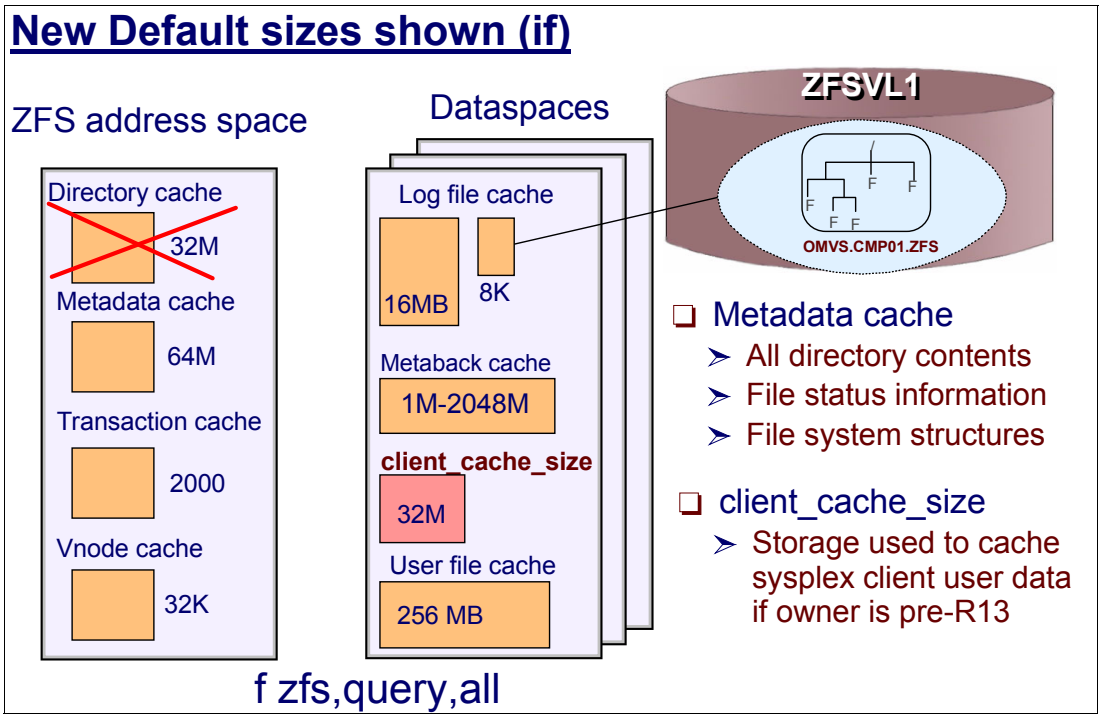


Figure 15-19 R13 zFS cache types

The HFS environment

The HFS file system was owned by system SC74, and SC75 is the client system. Filling the files was performed on system SC74. HFS was used with the default cache setting.

Sysplex-unaware zFS file system mount

In the first case, the zFS file system was mounted sysplex-unaware. The file system owner was SC74, as for HFS. Filling the files was performed on SC74, as well. For both systems, the user cache size was set to the default value of 256 MB.

Sysplex-aware zFS file system mount

The second zFS environment was set up to use the file system mounted sysplex-aware.

Note: In z/OS V1R13, zFS mounts file systems sysplex-aware (for read-write mounted file systems) by default if the actual setting of *sysplex_filesys_sharemode* is `rwshare`.

The same zFS files as before were used, along with the same user cache size of 256 MB. Figure 15-20 illustrates this environment.

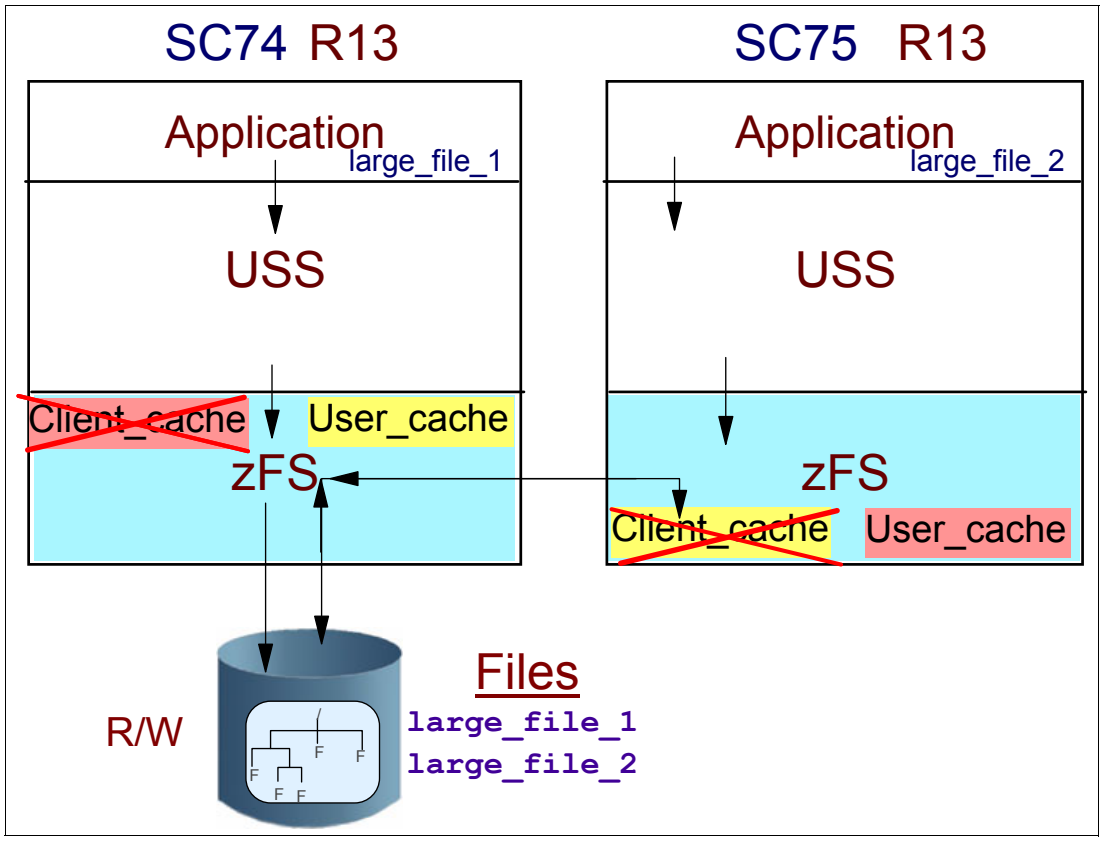


Figure 15-20 Sysplex-aware zFS file system with owner SC74 and client SC75

Accessing the file system files in the tests

The tests were set up as follows.

- Two jobs were set up to run in parallel on separate systems for every environment tested. One job accessed large_file_1 in system SC74. The other job accessed large_file_2 in system SC75.
- Each job locally started 20 processes running in parallel and performing 200 I/Os to read or write 1 MB blocks.
- 70% of the I/Os were reads and 30% were writes. This is a percentage of reads that is seen to be realistic for applications.
- The sequence of I/Os was based on a set of predictable random numbers for offsets into the large files when doing an I/O access. The term “predictable” in this case means that all tests used exactly the same set of random numbers, which made the results directly comparable.

15.8.2 Test results

The following single tests provide the necessary information to clearly identify what is set in the environment.

HFS numbers

Table 15-1 lists the results for the HFS tests, locally in SC74 and remote on SC75. There are no useful numbers for the client system because all client requests must be forwarded to the owning system for access to the file system.

Table 15-1 HFS test results

	AVG sec per process	MIN sec per process	MAX sec per process
HFS on owning system	67.39	66.61	68.28
HFS on client system	780.41	778.60	781.93

Results for zFS being sysplex-unaware

Table 15-2 lists the results for the old zFS behavior and shows the setup described in “Sysplex-unaware zFS file system mount” on page 305. The results on the zFS owning system are good. The numbers for the zFS client are slightly better than for HFS, but still much less than optimum due to sending function shipping requests to the owning system.

Table 15-2 zFS test results with file system mounted sysplex-unaware

	AVG sec per process	MIN sec per process	MAX sec per process
zFS on owning system	12.30	11.18	12.87
zFS on client system	481.75	479.26	483.30

Results for zFS being sysplex-aware

Table 15-3 lists the results if the zFS is mounted sysplex-aware. In this test, both user cache sizes are set to 256 MB. The zFS client system shows improved performance because it can access and change data without having to go to the owning system for data.

Important: The local user cache is used for caching if it is the zFS owning system and also if it is a client system.

In this case, there is no difference between the owner and the client results.

Table 15-3 zFS test results with file system mounted sysplex-aware

	AVG sec per process	MIN sec per process	MAX sec per process
zFS on owning system	21.45	18.87	22.25
zFS on client system	21.45	20.21	22.39

Comparing the owner numbers, the sysplex-unaware case looks better but this is a result of the client now also performing well; see the results shown in Table 15-4 for reference.

Table 15-4 zFS owner-only test results with file system mounted sysplex-aware

	AVG sec per process	MIN sec per process	MAX sec per process
zFS on owning system	11.91	11.30	12.42

These results shows that you can share now zFS file systems mounted in read-write mode in a UNIX System Services sysplex file system sharing environment with no performance disadvantage for a client system on data access. Therefore, the old requirement MR1211023911 addressed to zFS is solved.

zFS metadata considerations

Although data access from a client system is no longer a problem, restrictions still apply to complex access operations to metadata.

Note: It is important to run commands resulting in complex access to metadata (for example, **`find $MPDIR -xdev -exec ls -Ed {} ";"`**) on the zFS owning system.

If the UNIX file structure is large, you might see the zFS ownership moving if you start the command on a client system in such a situation.

15.9 Add symbolic links to shared root filesystem

z/OS V1R13 offers a new feature that provides benefits for symbolic links in the version root file system. In previous releases it was not possible to mount a version-root file system read-only without having to take post-installation actions for several utilities such as cron, mail and uucp after each new release.

- New support in z/OS V1R13 eliminates the post-installation actions by defining symbolic links for these utilities to allow using them in a read-only version root file system at installation time.
- A new migration health check is provided to determine whether the system environment will be affected by a one-time migration action due to this change.

Note: In previous systems, for example, the following post-installation actions had to be implemented:

- This member needs to be run on all systems to make sure the mountpoints get set in all version roots. */

```
/* Build symbolic links for cron */
BPXBATCH SH rm -r /usr/lib/cron
BPXBATCH SH rm -r /usr/spool
BPXBATCH SH rm -r /usr/local
BPXBATCH SH ln -s /etc/cron /usr/lib/cron
BPXBATCH SH ln -s /etc/spool /usr/spool
/* Set the program control bit for crontab */
BPXBATCH SH extattr +p /bin/crontab
```

15.9.1 Using the new support

A ServerPac delivery for z/OS V1R13 is provided with the /usr/lib/cron, /usr/mail, and /usr/spool directories as symbolic links to /var. There is no need to take any action to create these symbolic links. The required directories and symbolic link structure are created during installation by the BPXKMDIR REXX exec in SYS1.SAMPLIB. The exec is invoked by BPXISMKD and can also be invoked at other times as needed.

Note: Typically, there is no action needed for the BPXKMDIR exec to set up the required directories and symbolic links. However, if files have been generated by a previous customization of the `cron`, `uucp`, and `mail` utilities, they must be moved to the appropriate `/var` directories before BPXMKDIR can create the necessary files and symbolic links.

For a CBPDO installation, the required directories and symbolic link structure are created during execution of the BPXMKDIR REXX exec in SYS1.SAMPLIB.

The symbolic links are directed to `/var` directories as shown in Table 15-5 on page 309.

Table 15-5 New symbolic links for directories

Directory	--->	Linked to ...
/usr/lib/cron	--->	../../var/cron
/usr/spool	--->	../var/spool
/usr/mail	--->	../var/mail

For files used by `uucp`, these files are delivered as symbolic links that are directed to the `/var` directories as shown in Table 15-6.

Table 15-6 New symbolic links for uucp files

File name	--->	Linked to ...
/usr/lib/uucp/Systems	--->	../../../var/uucp/System
/usr/lib/uucp/Devices	--->	../../../var/uucp/Devices
/usr/lib/uucp/Dialers	--->	../../../var/uucp/Dialers
/usr/lib/uucp/Dialcodes	--->	../../../var/uucp/Dialcodes
/usr/lib/uucp/Permissions	--->	../../../var/uucp/Permissions
/usr/lib/uucp/config	--->	../../../var/uucp/config

The `/var` file system provided by ServerPac does not contain these files. They are provided when the utilities are configured and used.

15.9.2 CBPDO considerations

The following issues will cause a failure in the BPXMKDIR REXX exec:

1. Symbolic link targets created ahead of time must be created exactly as shown in Table 15-5 and Table 15-6. Otherwise, BPXMKDIR and the health check will fail.

   Note: Only relative symbolic links are accepted. Absolute symbolic links are not allowed. Thus, /usr/mail must point to ../var/mail and not to /var/mail.

2. Installation-specific content must be moved before running the BPXMKDIR REXX exec.
   - Directory /usr/lib/cron must be empty.
   - Directory /usr/mail must be empty.

– Directory /usr/lib/uucp should contain the IBM files uucc, uucico, uuxqt, and the IBM subdirectory.
– Directory /usr/spool may contain the following empty IBM subdirectories: cron, locks, cron/atjobs, cron/crontabs, uucp, uucppublic, uucp/.Xqtdir, uucp/.Sequence and uucp/.Status.

Note: The migration health check verifies both issues.

15.9.3 Migration considerations

The verification used in the z/OS V1R13 BPXMKDIR REXX exec and the z/OS V1R11 and V1R12 migration health check is essentially the same.

The migration health check is written in System REXX. Therefore, the System REXX service, the health check started task, and z/OS UNIX System Services must be available on the system to run this check.

The health check notifies when a one-time migration action needs to be taken if either of the following is true:

1. You have performed the post-installation activities to make uucp, cron, or mail supported for a read-only z/OS version root. You do not necessarily have to be running with the z/OS version root as read-only, but have performed the post-installation customization as described in *z/OS UNIX System Services Planning*, GA22-7800.
2. You have used uucp, cron, or mail facilities and have not performed the post-installation customization as described in *z/OS UNIX System Services Planning*, GA22-7800. Although this migration action is to be performed before the first IPL of z/OS V1R13, the changes to use /var for this support can be done at any time.

Note: Although previous documentation had shown the use of /etc in examples, it is preferable to use /var for these utilities.

Using /etc or another directory instead of /var

If /etc or another directory is currently used for post-installation customization, you may continue to use that directory. However, this is not a practical approach.

It is best to use the /var structure that is provided with z/OS V1R13 for the following reasons:

- You can minimize any subsequent post-installation customization, because the symbolic links to /var are provided for you by IBM.

  Continued use of directories other than /var can result in post-installation work for every new release, to remove the delivered structure and replace it with your own.

  Doing so negates the benefits of the provided symbolic links, because the post-installation actions are changed, but not eliminated.

- The continued use of /etc or another directory requires you to manage and maintain the symbolic links required from /var to that directory, which is “double symlinking.” This double symlinking might be confusing for the personnel who maintain the system.

  To use a directory other than /var, you have to create an additional symbolic link from the /var directory to your chosen target directory. For example, if you chose to use a directory other than /var/mail, such as /etc/mail, you need to create an additional symbolic link from /var/mail to this directory. These double symlinks can be reused for

future releases, provided that you continue to provide the intermediate links for each release.

Other possible migration steps

The following steps are needed if you have not taken actions yet to create symlinks in the version root pointing to locations under /var or another directory to support the version root being used read-only.

- If there are user files under `/usr/spool`, `/usr/lib/cron`, or `/usr/mail` in the version root file system, then those files must be moved to `/var/spool`, `/var/cron`, or `/var/mail`, or to a directory other than `/var` that you choose.
- If you have any files except `uucc`, `uucico`, `uuxqt` and subdirectory `IBM` under `/usr/lib/uucp` in the version root file system, then those files must be moved to `/var/uucp`, or to a directory other than /var that you choose.

15.9.4 Installation considerations

For both ServerPac and CBPDO installations, use the health check that is provided in APAR OA35605 and APAR OA35636 and named ZOSMIGV1R13_RO_SYMLINKS to check whether a one-time migration action will be needed when migrating from z/OS V1R11 or V1R12.

- Install the APARs OA35605 and OA35636 on z/OS V1R11 or V1R12.
- Activate the ZOSMIGV1R13_RO_SYMLINKS health check.
- Examine the results.

Successful health check messages

These are the successful health check messages:

```
BPXH910I The directory /usr/mail is not customized.
BPXH920I The directory /usr/lib/cron is customized to the ../../var/cron
directory.
```

This is the overall success message:

```
BPXH913I All directories verified were found to be acceptable for the new
symlinks added in z/OS V1R13. A migration action is not required.
```

Health check failure messages

These health check messages indicate failure situations:

```
BPXH912I The directory /usr/lib/cron has additional files, directories, or
symbolic links found as follows:
my_file_1
my_file_2
my_directory

BPXH911I The directory /usr/mail has a symlink to /var/mail.
```

This is the overall failure message:

```
BPXH915E One or more of the directories verified were found to contain
post-install customization that is expected to be affected by the new symlinks
added in z/OS V1R13, or there were problems accessing the directory. A
migration action is required.
```

15.10 Lost message detection

A change has been made to z/OS UNIX System Services message processing in a UNIX System Services shared file system configuration environment. In previous releases you needed increased diagnostic capability to detect lost XCF messages between z/OS UNIX members.

Now, in z/OS V1R13, XCF ordered message delivery is utilized along with assigning sequence numbers to messages. Each message sequence number is checked. A lost or duplicate message results in a two-system dump.

There is a new BPXPRMxx parmlib member keyword for this support:

```
LOSTMSG(ON|OFF)
```

There is also a SETOMVS command update:

```
SETOMVS LOSTMSG=ON|OFF
```

Note: Using LOSTMSG(ON) can incur a performance penalty in high z/OS UNIX traffic environments.

However, using most read-write file systems mounted as sysplex-aware reduces z/OS UNIX traffic.

15.11 Provide script utility to log session

There is a new shell script command to record shell session activity so you can use the recorded data for troubleshooting and documenting purposes. With z/OS V1R13, the script command makes a typescript of everything displayed on the terminal.

Note: The script command records shell session activity in a way that is similar to other UNIX and Linux platforms.

- The script command works with the `/bin/sh` and `/bin/tcsh` shells.
- It works in the OpenSSH, rlogin, telnet and OMVS shell environments.

15.11.1 Using the script command

As mentioned, the new script command makes a typescript of everything displayed on the terminal. This typescript is written to the file specified by the file parameter. If no file name is given, the typescript is saved in the current working directory with the file name `typescript`.

If the file exists, the default behavior is to overwrite its contents. The script command is not listed in the UNIX standards SUSv2 or SUSv3.

Note: SUS is the short form of Single UNIX Specification.

- SUSv2 is UNIX98; see `http://www.unix.org/version2/` for more information.
- SUSv3 is UNIX03; see `http://www.unix.org/version3/` for more information.

The z/OS shell script command is similar to the AIX script command.

Command usage:

```
script [-aq] [file]
```

Options and parameters are:

-a This option appends the typescript to the file. The default is to overwrite data.

-q In quiet mode, all diagnostic messages are suppressed.

file Typescript is written to the file specified by the file parameter. The default name is `./typescript`.

15.11.2 Usage considerations

The script command writes everything in the typescript to the file including backspaces, prompts, and so on. The script command forks and executes a shell according to the value of the SHELL environment variable. If the environment variable is not set, the script uses the `/bin/sh` shell. The script ends when the shell process exits. Use either **`exit`** or Ctrl-D to exit the shell process.

Note: Terminal-modifying commands such as **`vi`** can create unexpected data in the typescript.

Note the following considerations:

- Ensure that access to the file containing the typescript is properly controlled, because the file can contain sensitive data.
- The script command does not support setting 3270 pass-through mode during the shell session. As a result, OEDIT, OBROWSE, and other utilities requiring 3270 pass-through mode will fail.
- The script command creates a new session and controlling terminal for the shell process.
- The command is provided as part of z/OS V1R13 as a binary installed in the `/bin` directory.

Figure 15-21 shows an example of using the script command.

```
$> script
Script command is started. The file is typescript.
$> id
uid=888(HERING) gid=0(SYS1)
$> pwd
/u/hering
$> command -V script
script is cached /bin/script
$> exit
Script command is complete. The file is typescript.
$> cat ~/typescript
Script command is started on Mo 16 Mai 20:38:43 2011.
$> id
uid=888(HERING) gid=0(SYS1)
$> pwd
/u/hering
$> command -V script
script is cached /bin/script
$> exit
Script command is complete on Mo 16 Mai 20:39:58 2011.
$>
```

Figure 15-21 Using the script command

15.12 Enhancement for the D OMVS,W command

Following are the improvements that have been made to the **D OMVS,WAITERS** command.

- File latch contention is now displayed in a new table, namely the file latch activity table.
- Filtering options have been added to limit the amount of data being displayed.

In previous releases, the following restrictions or missing functionality exist:

- File latch contention is not displayed in its own table like file system latch contention. File latch activity will currently only show up in the other waiters table. This makes it difficult to correlate the waiters and holders causing file latch contention. The file latch holder is not necessarily displayed either unless it is waiting on something else.
- The information displayed on the **D OMVS,W** command can include many waiters. There is no way to limit the output being displayed.

Now, the **D OMVS,W** command display has been enhanced to include a table for file latch activity. This table is similar to the existing table for file system latches. The file latch activity table displays the user, the ASID and TCB of the holder or waiter, whether the latch is held exclusive or shared, the age, and so on.

A set of filtering options has also been added to the command to limit what is displayed.

- It is now easier to identify and debug file latch contention.
- You can easily limit the amount of data being displayed to help in problem determination.

15.12.1 Showing file latch activity

Figure 15-22 shows sample output of a **D OMVS,WAITERS** command.

```
BPXO063I 16.22.32 DISPLAY OMVS 586
OMVS     000E ACTIVE             OMVS=(2W,DN)
MOUNT LATCH ACTIVITY: NONE

FILE SYSTEM LATCH ACTIVITY:
    USER   ASID     TCB             SHR/EXCL                    AGE
-------------------------------------------------------------------
Latch 14 FILE SYSTEM: ZOS113.VAR.ZFS
HOLDER(S):
  TCO      0028  008E6D90             EXCL                   00.00.15
     TIME: 2011/02/08 16.22.17
     IS DOING: Running
WAITER(S):
  TCO      0029  008E6D90             SHR                    00.00.07
     TIME: 2011/02/08 16.22.24

FILE LATCH ACTIVITY:
    USER   ASID     TCB             SHR/EXCL                    AGE
-------------------------------------------------------------------
LATCH 66 LSET 01 TYPE REGFILE   DEVNO 2 INO 204
FILE: myfile
FILE SYSTEM: ZOS113.ETC.ZFS
HOLDER(S):
  TCO      0026  008E6D90             EXCL                   00.00.56
     TIME: 2011/02/08 16.21.36
     IS DOING: ZFS MKDirCall
WAITER(S):
  TCO      0027  008E6D90             SHR                    00.00.53
     TIME: 2011/02/08 16.21.38

OTHER WAITING THREADS:
    USER   ASID     TCB               PID                      AGE
-------------------------------------------------------------------
  CEA      0016  008E2D90        16777218                   00.06.51
     TIME: 2011/02/08 16.15.40
     IS DOING: Osi Wait
```

Figure 15-22 Excerpts from the output of a D OMVS,W command

The fields and values shown in Figure 15-22 have the following meaning:

DEVNO This indicates the device number in decimal of the file system in which the file resides.

INO This indicates the inode number in decimal of the file.

LSET This indicates the latch set identifier. File latches are created in the SYS.BPX.A000.FSLIT.FILESYS.LSN.*XX* latch set where *XX* corresponds to LSET.

TYPE This indicates the type of the file. It may be DIR, CHARSPEC, REGFILE or FIFO.

FILE This indicates the name of the file, if it is known. The name can be up to 16 characters long and is truncated on the left, if it is longer.

FILE SYSTEM This indicates the file system that owns the file.

Note: CINET sockets use the same latch set as files. This display will not show any sockets.

15.12.2 Output filtering

A set of filtering options limits the information that is displayed. If a section of the display is filtered out, the display does not show the section instead of displaying NONE. You can combine options to display certain sections.

For example, specify the **D OMVS,W,xxx** command where xxx is one or more of the following options:

LATCHES | L This will only display the latch activity tables. These include the mount, file system and file latch activity tables. The cross-system messages and other waiters tables will be suppressed.

MESSAGES | M This will only display the sent and received cross-system messages table. The mount latch, file system latch, file latch and other waiters tables will be suppressed. This option is only valid in a shared file system environment.

OTHER | O This will only display the other waiters table. The mount latch, file system latch, file latch and cross-system messages tables will be suppressed.

AGE | A This will only display waiters that have been waiting for more than five minutes. A table will be displayed if there are waiters that meet the criteria.

SPECIAL | S This will allow special files to be displayed in the other waiters table that are otherwise filtered out. These are character special files, pipes and sockets. This option will be allowed by itself, with the **A** option or when the **O** option has also been specified.

Filtering examples

Following are examples commands for using the filtering options.

D OMVS,W,L,O This displays only the latch and other waiters tables.

D OMVS,W,A This displays any activity in any of the tables that have been waiting for more than five minutes.

D OMVS,W,L,A This displays only activity in the latch tables that have been waiting for more than five minutes.

D OMVS,W,S This displays any waiters in all tables and allows the other waiters table to also include waiters on character special files, sockets and pipes.

D OMVS,W,O,S This displays only waiters in the other waiters table and also includes waiters on character special files, sockets and pipes

Figure 15-23 shows an example output using filtering with the **D OMVS,W** command.

```
D OMVS,W,O
BPXO063I 16.22.32 DISPLAY OMVS 586
OMVS     000E ACTIVE             OMVS=(2W,DN)
OTHER WAITING THREADS:
    USER   ASID     TCB               PID                        AGE
-------------------------------------------------------------------------
  CEA      0016  008E2D90        16777218                     00.06.51
     TIME: 2011/02/08 16.15.40
     IS DOING: Osi Wait
```

Figure 15-23 Using D OMVS,W,O for filtering data of normal D OMVS,W command

15.12.3 Using an assembler program

Data from a **D OMVS,W** command can be obtained from using the assembler BPXEKDA macro and the BPXZODMV mapping. You can set the new filters in ODMVINBYTE2 when ODMVWAITERS has been set in ODMVINBYTEM3.

```
ODMVWLATCHES EQU X'80'   D OMVS,W,LATCHES
ODMVWMESSAGES EQU X'40'  D OMVS,W,MESSAGES
ODMVWOTHER EQU X'20'     D OMVS,W,OTHER
ODMVWAGE EQU   X'10'     D OMVS,W,AGE
ODMVWSPECIAL EQU X'08'   D OMVS,W,SPECIAL
```

The output data is mapped by the ODMVDWHEADER and ODMVDWELEMENT.

Note: The BPXEKDA macro provides an interface for an authorized report application to retrieve kernel data.

15.13 RACF support for z/OS UNIX user mounts

This RACF function is delivered in support of z/OS UNIX System Services non-privileged user mounts. Mounting a z/OS UNIX file system is a privileged operation that requires superuser authority. However, it is useful to be able to move the management of user data away from system administrators and to the users who own the data.

With z/OS V1R13, the capability to permit nonprivileged users to mount and unmount file systems based on the user's authority to the mount point and the file system root is implemented. This is only supported in the BPX2MNT interface, for example:

```
/usr/sbin/mount
TSO mount commands
```

System administrators will no longer have the burden of user file system data management. Application developers and users will now be able to manage their own data.

15.13.1 Mount security requirements

To exploit the new functions for mount processing the following security requirements must be met:

- The user must have read access to profile `SUPERUSER.FILESYS.USERMOUNT` in the UNIXPRIV class.

- The user must have Read-Write-Execute(RWX) access permission to the mount point directory.
- If the sticky bit is set, then the user must be the owner of mount point directory.
- The mount point directory must be empty.
- The user must have Read-Write-Execute(RWX) access permission to the file system root directory.
- If the sticky bit is set, then the user must be the owner of the file system root directory.

Note: If any one of the requirements or restrictions are violated, a unique return code and reason code will be returned to identify the problem along with the audit failures.

Unmount security requirements

To use the new functions for unmount processing the following security requirements must be met:

- The user must have read access to profile SUPERUSER.FILESYS.USERMOUNT in the UNIXPRIV class.
- The user must have been the one that mounted the file system.

Using the new support

Figure 15-24 shows the RACF commands to authorize users or groups by defining the profile and give READ access permission to them. This allows nonprivileged users to mount and unmount file systems with the **nosetuid** option.

```
RDEFINE UNIXPRIV SUPERUSER.FILESYS.USERMOUNT UACC(NONE)
PERMIT SUPERUSER.FILESYS.USERMOUNT CLASS(UNIXPRIV) ID(userid) ACCESS(READ)
SETROPTS RACLIST(UNIXPRIV) REFRESH
```

Figure 15-24 Setting the RACF authorization

Note the following points:

- z/OS UNIX System Services calls **ck_priv** (IRRSKP00) for mount and unmount requests. If the user is not in superuser mode and audit function code indicates a user mount or unmount, then **ck_priv** checks access to the SUPERUSER.FILESYS.USERMOUNT profile.
- The ck_priv service interface is unchanged, but it supports new audit function codes:

```
AFC_MOUNT_NA    (0076x)         mount no audit
AFC_MOUNT_U     (0077x)         user mount
AFC_MOUNT_UNA   (0078x)         user mount no audit
AFC_UNMOUNT_U   (0079x)         user unmount
AFC_UNMOUNT_UNA (007Ax)         user unmount no audit
```

- To initiate and allow the new support, there are no RACF interface changes.
- New audit function codes are defined.

Note: Audit records are always written when a user who is not a superuser or not permitted to the SUPERUSER.FILESYS.USERMOUNT RACF profile tries to mount or unmount a file system.

The file to be unmounted must have been mounted by that nonprivileged user.

15.13.2 Mount and unmount processing

z/OS V1R13 provides the capability to permit nonprivileged users to perform mount and unmount file system operations based on their authority to the mount point and the file system root. In addition, several changes for mount and unmount processing are made.

Mounting a z/OS UNIX file system is a privileged operation. In previous releases there was no way to move the management of user data away from system administrators and to the users who own the data.

Permitting nonprivileged users to mount and unmount file systems based on the user's authority to the mount point and the file system root has the following considerations:

- It is supported with the `__mount()` interface that is used in **`/usr/sbin/mount`** or the TSO **`MOUNT`** command, for example.
- System administrators will no longer have the burden of user file system data management.
- In z/OS V1R13 with this support, permitted nonprivileged users will be able to perform mount and unmount file system operations and manage their own data requirements.

Notes:

- Significant restrictions and explicit enablement are placed on this capability so that security characteristics are not affected.
- This is a user productivity enhancement because it allows the application and the user to control their data.

The automount facility must be running to mount a HSM-migrated file system using a nonprivileged user mount. Otherwise, the request will fail with the `JrFileSystemMigrated` reason code.

Note: Users can manually HSM recall the file system and use it to mount the file system.

Restrictions

The following restrictions apply to this support:

- Supported file system types are HFS, zFS, and NFS.
- The `SYSNAME` option is not supported.
- The `NOSECURITY` option cannot be specified.
- The `NOSETUID` option must be specified.
- Using **`chmount`** is not supported for nonprivileged users.
- Performing a `remount` is not supported for nonprivileged users.
- Using `///` as a file system name placeholder is not supported.
- The `BPX1MNT` callable service is not supported for nonprivileged users.
- The mount operation is bounded by `MAXUSERMOUNTSYS` and `MAXUSERMOUNTUSER` BPXPRMxx parmlib member limits.

15.13.3 Setup for the new support

The following actions are needed to activate the support.

- Permit users to the new UNIXPRIV profile SUPERUSER.FILESYS.USERMOUNT as shown in Figure 15-24.
- Set appropriate values in BPXPRMxx parmlib member for the new keywords:

  `MAXUSERMOUNTSYS(xx) and MAXUSERMOUNTUSER(yy)`

These new keywords have the following meaning:

MAXUSERMOUNTSYS The `MAXUSERMOUNTSYS` statement specifies the maximum number of nonprivileged user mounts in the system or in shared file system configuration.

MAXUSERMOUNTUSER The `MAXUSERMOUNTUSER` statement specifies the maximum number of nonprivileged user mounts allowed for any nonprivileged user in the system or in shared file system configuration.

Using the parameters:

- The accepted value range for MAXUSERMOUNTSYS and MAXUSERMOUNTUSER is 0 to 35000.
- The MAXUSERMOUNTSYS and MAXUSERMOUNTUSER values can be dynamically increased or decreased using **`SET OMVS`** or **`SETOMVS`** commands.
- The default value for MAXUSERMOUNTSYS and MAXUSERMOUNTUSER is zero (0), which indicates that no nonprivileged mount is allowed in the system or in the shared file system configuration.

Important: The most recent specification of these values will prevail for all of the systems participating in the shared file system configuration.

15.13.4 Authorizing a nonprivileged user mount

The system administrator must authorize a user to be able to perform a nonprivileged user mount. Figure 15-25 is a JCL example of an administrator submitting a job to create a zFS file system for a nonpriviledged user ID (CESAR) to be the owner of the file system. The file system is also formatted and attached.

The user CESAR is permitted to mount his file system because of the profiles defined in “Using the new support” on page 318.

```
//DEFZFS1 JOB (999,POK),'CONWAY',CLASS=A,REGION=0M,
//     MSGCLASS=T,TIME=10,MSGLEVEL=(1,1),NOTIFY=&SYSUID
/*JOBPARM SYSAFF=SC74
//*********************************************************************
//* this job - defines and formats a zFS data set for a user
//*********************************************************************
//DEFINEZ  EXEC   PGM=IDCAMS
//SYSPRINT DD     SYSOUT=*
//SYSUDUMP DD     SYSOUT=*
//AMSDUMP  DD     SYSOUT=*
//SYSIN    DD     *
     DEFINE CLUSTER (NAME(CESAR.TEST.ZFS) -
            LINEAR CYLINDERS(5 5) SHAREOPTIONS(3)    -
            VOLUME(BH5HF2))
/*
//CREATEZ  EXEC   PGM=IOEAGFMT,
//           PARM=('-aggregate CESAR.TEST.ZFS -compat
//             -owner CESAR -group SYS1 -perms o755')
//SYSPRINT DD     SYSOUT=*
//STDOUT   DD     SYSOUT=*
//STDERR   DD     SYSOUT=*
//SYSUDUMP DD     SYSOUT=*
//CEEDUMP  DD     SYSOUT=*
//*
```

Figure 15-25 JCL to create a file system for user ID CESAR

Displaying information about user mount support

Figure 15-26 shows an example sequence of commands for performing a nonprivileged user mount after displaying a situation that does not allow the mount.

Following are examples of displaying nonprivileged user mount information:

- The **D OMVS, O** command displays the current system settings for MAXUSERMOUNTSYS and MAXUSERMOUNTUSER as shown in Figure 15-26.
- The **D OMVS,F,UID=PRIV|USER|*uid*** command displays information for file system mounted:

  USER This displays all file systems mounted by nonprivileged users.

  PRIV This displays all file systems mounted by privileged users.

  uid This displays all file systems mounted by the user whose effective UID is *uid*.
- A **D OMVS,USERMOUNTS** command displays summary information for nonprivileged user mounts.
- A **F BPXOINIT,FILESYS=DISPLAY,GLOBAL|FILESYSTEM=*fsn*|ALL** command displays current sysplex settings and mounted file system information.
- The **ISHELL** shows nonprivileged user mount information in the file system mount table and in the file system attributes panels.
- The updated z/OS UNIX shell command **df -v** displays the user ID and the effective UID for a nonprivileged user mounted file system.

```
$> cn d omvs,o | grep MAXUSERMOUNTSYS
 MAXUSERMOUNTSYS =          5    MAXUSERMOUNTUSER=          5
$> /usr/sbin/mount -f HERING.TEST.ZFS test
FOMF0504I mount error: 79 119B063B
EINVAL: The parameter is incorrect
JrNoSetUID: NOSETUID was not specified on the nonprivileged user mount interface.
$> /usr/sbin/mount -s nosetuid -f HERING.TEST.ZFS test
FOMF0504I mount error: 88 119B063C
ENOTEMPTY: The directory is not empty
JrNonEmptyMntPtDir: The mount point directory is not empty.
$> mkdir -m700 testumnt
$> /usr/sbin/mount -s nosetuid -f HERING.TEST.ZFS testumnt
$>
```

Figure 15-26 Performing a nonprivileged user mount

Examples of displaying nonprivileged user mount information

Figure 15-27 displays nonprivileged user mount information for a specific file system.

```
$> cn "d omvs,f,n=hering.test.zfs"
 BPXO045I 00.48.37 DISPLAY OMVS 101
 OMVS     0010 ACTIVE             OMVS=(3A)
 TYPENAME   DEVICE ----------STATUS----------- MODE  MOUNTED    LATCHES
 ZFS            89 ACTIVE                      RDWR  05/20/2011  L=74
   NAME=HERING.TEST.ZFS                              22.56.19    Q=74
   PATH=/u/hering/testumnt
   UID=888
   OWNER=SC74     AUTOMOVE=Y CLIENT=N
$> cn "f bpxoinit,filesys=display,filesystem=hering.test.zfs"
 BPXM027I COMMAND ACCEPTED.
 BPXF040I MODIFY BPXOINIT,FILESYS PROCESSING IS COMPLETE.
 BPXF035I 2011/05/21 00.50.23 MODIFY BPXOINIT,FILESYS=DISPLAY
 -------------NAME-------------------------        DEVICE  MODE
 HERING.TEST.ZFS                                       89  RDWR
   PATH=/u/hering/testumnt
   UID=888
   STATUS=ACTIVE                 LOCAL STATUS=ACTIVE
   OWNER=SC74        RECOVERY OWNER=SC74     AUTOMOVE=Y PFSMOVE=Y
   TYPENAME=ZFS      MOUNTPOINT DEVICE=      42
   MOUNTPOINT FILESYSTEM=HERING.ZFS
   ENTRY FLAGS=90074800  FLAGS=40000400  LFSFLAGS=00000000
   LOCAL FLAGS=40000400  LOCAL LFSFLAGS=02000000
   ACTIVECHK  =00000000  LFSFLAGS2     =D8000000
$>
```

Figure 15-27 Displaying nonprivileged user mount information for a specific file system

Displaying all file systems mounted by nonprivileged users

Figure 15-28 displays all file systems mounted by nonprivileged users.

```
$> cn "d omvs,f,uid=user"
 BPXO045I 00.54.39 DISPLAY OMVS 114
 OMVS     0010 ACTIVE             OMVS=(3A)
 TYPENAME   DEVICE ----------STATUS----------- MODE  MOUNTED    LATCHES
 ZFS            89 ACTIVE                      RDWR  05/20/2011  L=74
   NAME=HERING.TEST.ZFS                              22.56.19    Q=74
   PATH=/u/hering/testumnt
   UID=888
   OWNER=SC74     AUTOMOVE=Y CLIENT=N
 ZFS            79 ACTIVE                      RDWR  05/19/2011  L=73
   NAME=PRICHAR.HFS1                                 03.35.17    Q=0
   PATH=/u/prichar/dir1
   UID=37
   MOUNT PARM=FSFULL(70,10)
   OWNER=SC74     AUTOMOVE=Y CLIENT=N
$>
```

Figure 15-28 Displaying all file systems mounted by nonprivileged users

Displaying summary information for nonprivileged user mounts

Figure 15-29 displays the output of the **D OMVS,USERMOUNTS** command.

```
$> cn "d omvs,usermounts"
 BPXO072I 00.57.28 DISPLAY OMVS 118
 OMVS     0010 ACTIVE             OMVS=(3A)
 NONPRIVILEGED USER MOUNTS SUMMARY
           UID   CURRENT MOUNTS
           888           1
            37           1
$>
```

Figure 15-29 Displaying summary information for nonprivileged user mounts

Displaying nonprivileged user mount settings and highwater marks

Figure 15-30 displays nonprivileged user mount settings and highwater marks.

```
$> cn "d omvs,limits" | head -15
 BPXO051I 00.40.50 DISPLAY OMVS 097
 OMVS     0010 ACTIVE             OMVS=(3A)
 SYSTEM WIDE LIMITS:         LIMMSG=NONE
                    CURRENT  HIGHWATER     SYSTEM
                      USAGE      USAGE      LIMIT
$> cn "d omvs,limits" | grep MAXUSERMOUNT
 MAXUSERMOUNTSYS         2          2          5 *
 MAXUSERMOUNTUSER        1          1          5 *
$> cn "f bpxoinit,filesys=display" | grep MAXUSERMOUNT
 MAXUSERMOUNTSYS=             5   HIWATER MAXUSERMOUNTSYS=    2
 MAXUSERMOUNTUSER=            5   HIWATER MAXUSERMOUNTUSR=    1
$>
```

Figure 15-30 Displaying nonprivileged user mount settings and highwater marks

ISHELL displays for nonprivileged user mounted file systems

Figure 15-31 displays the ISHELL mount table with the new user column.

```
                        Work with Mounted File Systems

Select one or more file systems with / or action codes.
U=Unmount   A=Attributes   M=Modify   R=Reset unmount or quiesce
  Type  Owner     A/M  User      Mount Point           File syst  Row 9 of 49
_ ZFS   SC74      Yes            /u/haimo              HAIMO.ZFS
_ ZFS   SC74      Yes            /u/hering/onek.r12    HERING.ONEK.FILES
_ ZFS   SC74      Yes            /u/hering/onek.r13    HERING.ONEK.FILES.COPY
a ZFS   SC74      Yes  HERING    /u/hering/testumnt    HERING.TEST.ZFS
_ ZFS   SC74      Yes            /SC74/var/zosmf/data  IZU.SIZUDATA
_ ZFS   SC74      Yes            /u/jes2               JES2.ZFS
_ ZFS   SC74      Yes            /u/kappele            KAPPELE.HFS
_ ZFS   SC74      Yes            /u/lafitte            LAFITTE.ZFS
_ HFS   SC74      Yes            /pp/db2v8/UK05586     OMVS.DB2V8.UK05586.HFS
_ ZFS   SC74      Yes            /pp/db2v9/D080325/db  OMVS.DB2V9.D080325.HFS
```

Figure 15-31 ISHELL panel of mounted file systems

Figure 15-32 displays the updated ISHELL panel of file system attributes for a specific file system.

```
                        Work with Mounted File Systems
 ____________________________________________________________
S |                  File System Attributes                    |
U |                                                            |
  | File system name:                                          | Row 9 of 49
_ | HERING.TEST.ZFS                                            |
_ | Mount point:                                               | .FILES
_ | /u/hering/testumnt                                         | .FILES.COPY
a |                                          More:   - +       | .ZFS
_ | Blocks in use  . . . . : 35231                             | A
_ | Ignore SETUID  . . . . : 1                                 |
_ | Bypass security  . . . : 0                                 |
_ | Automove . . . . . . . : Yes                               |
_ | Owning system  . . . . : SC74                              | UK05586.HFS
_ | Data blocks read . . . : 0                                 | D080325.HFS
_ | Data blocks written  . : 0                                 |
_ | Dir blocks r/w . . . . : 0                                 | TC
_ | Char Set ID/Text flag  :                                   | DSSRV1.HFS
_ | Seclabel . . . . . . . :                                   | AR
_ | User . . . . . . . . . : HERING(888)                       | AR.WBEM
_ |                                                            | EB.PRINTWAY.Z
_ |  F1=Help       F3=Exit       F4=Name       F6=Keyshelp   | .Z1DRA1.JAVA3
_ | F12=Cancel                                                 | .Z1DRA1.JAVA3
_ '____________________________________________________________' .Z1DRA1.JAVA6
_ ZFS   SC74      Yes           /Z1DRA1/usr/lpp/java  OMVS.ZOSR1D.Z1DRA1.JAVA6
```

Figure 15-32 ISHELL panel of file system attributes for a specific file system

15.13.5 Additional mount and unmount enhancements

In z/OS V1R13, several additional enhancements have been added for mount and unmount processing.

BPXPRMxx parmlib statement for non-empty mount point directories

A new BPXPRMxx PARMLIB statement is provided to control non-empty mount point directory contents overlay during mount operations:

```
NONEMPTYMOUNTPT (NOWARN|WARN|DENY)
```

NOWARN This causes the system to mount any file system on mount points without any warning message when the mount point is a non-empty directory. The contents of that directory will be hidden for the duration of the mount.

WARN This causes the system to mount any file system on mount point with a warning message when the mount point is a non-empty directory. The contents of that directory will be hidden for the duration of the mount.

DENY This causes the system not to mount any file system when the mount point is a non-empty directory.

Note: This setting can be dynamically changed using the **SET OMVS** or **SETOMVS** commands.

Figure 15-33 displays example commands regarding this NONEMPTYMOUNTPT setting.

```
$> cn "d omvs,o" | grep NONEMPTYMOUNTPT
 SWA             = ABOVE         NONEMPTYMOUNTPT = NOWARN
$> sudo /usr/sbin/mount -f hering.test.zfs test
$> sudo /usr/sbin/unmount test
$> cn "setomvs nonemptymountpt=deny"
 BPXO015I THE SETOMVS COMMAND WAS SUCCESSFUL.
$> sudo /usr/sbin/mount -f hering.test.zfs test
FOMF0504I mount error: 88 55B063C
ENOTEMPTY: The directory is not empty
JrNonEmptyMntPtDir: The mount point directory is not empty.
$> cn "setomvs nonemptymountpt=warn"
 BPXO015I THE SETOMVS COMMAND WAS SUCCESSFUL.
$> sudo /usr/sbin/mount -f hering.test.zfs test
$>
```

Figure 15-33 Example commands regarding the NONEMPTYMOUNTPT setting

If **NONEMPTYMOUNTPT** is set to **WARN**, then a warning message is provided as shown in Figure 15-34.

```
BPXF263I FILE SYSTEM 502
HERING.TEST.ZFS
HAS BEEN MOUNTED ON A NONEMPTY DIRECTORY
```

Figure 15-34 Warning message on mounting on a non-empty directory

Updates for the shell command /usr/sbin/mount

z/OS V1R13 adds several enhancements to the shell command **/usr/sbin/mount**.

- If waiting for an asynchronous mount to complete, failures are now reported to the user.

Note: This also applies to the TSO **MOUNT** command.

- The file system type is dynamically determined if the type option **-t** is not used and file system option **–o** was specified.

  Note: In this case the mount of a zFS file system can start to fail if file system options specified were HFS parameters. This can happen by mistake or because the mount originally was to mount an HFS that has been migrated to zFS.

- The file system name length is now verified and the mount fails if the name is larger than 44 characters.
- The file system name is changed to uppercase letters if the type option **–t** is not used and the type is determined to be zFS.

Examples of new mount behavior are shown in Figure 15-35. Type option **-t** is not used and the zFS file system name is not specified in uppercase. Nevertheless, you can successfully use zFS file system options.

```
$> sudo /usr/sbin/mount -f hering.test.zfs test
$> zfsowner hering.test.zfs
zFS Owner    : SC74     - Aggregate read-only=N, sysplex-aware=Y
$> mkdir -m755 test/test2
$> sudo /usr/sbin/mount -f hering.test2.zfs -o norwshare test/test2
$> zfsowner hering.test2.zfs
zFS Owner    : SC74     - Aggregate read-only=N, sysplex-aware=N
$>
```

Figure 15-35 Demonstrating new mount behavior in various cases

Updates for the shell command /usr/sbin/unmount

z/OS V1R13 adds a significant change to shell command **/usr/sbin/mount** in using the path specified to identify the file system to be unmounted.

- The default behavior has been changed to unmount a file system only if the path specified is a mount point.
- A new option **–m** has been created to retain the original behavior. This means that the path specified can be any file or directory contained in the file system.

Note: z/OS V1R12 added new function to clearly identify the file system to be unmounted by specifying option **-f** together with the file system name.

The new changes added in z/OS V1R13 are shown in the sequence of commands displayed in Figure 15-36.

```
$> sudo /usr/sbin/mount -qv test
----A- HERING.TEST2.ZFS /u/hering/test/test2
----A- HERING.TEST.ZFS  /u/hering/test
$> sudo /usr/sbin/unmount test/test2
$> sudo /usr/sbin/unmount test/test2
FOMF0512I Path is not a mountpoint: test/test2
$> sudo /usr/sbin/unmount -m test/test2
$> sudo /usr/sbin/unmount test
FOMF0512I Path is not a mountpoint: test
$>
```

Figure 15-36 New unmount processing in using the path specified

Installation reference information

For reference purposes, the following updates to the BPXPRMxx parmlib member statements have been introduced in z/OS V1R13:

```
MAXUSERMOUNTSYS(xx)
MAXUSERMOUNTUSER(xx)
NONEMPTYMOUNTPT(NOWARN|WARN|DENY)
```

15.13.6 z/OS UNIX System Services file system access

The new file system access control in the RACF setup is described as to how a RACF administrator can now control zFS and NFS file system access by managing profile permissions in the new RACF FSACCESS class.

Previously, there was no ability for the RACF administrator to control access to z/OS UNIX mounted file systems without needing to use UNIX commands.

This new support has been implemented as a new FASTAUTH call in **ck_access** that checks a user's permission to a file system profile in the new FSACCESS class.

Note: This new interface for authorization checking is supported for zFS and NFS.

This provides several benefits:

- It provides the RACF administrator with coarse-grained file system access control.
- For administration it does not require UNIX command expertise, for example in using the **setfacl** command.
- It provides compliance and audit verification for RACF-centric installations.

Using the new interface

A mount point traversal triggers a one-time check to the container:

- An access failure prevents any operation within the file system, regardless of permission bits, acls, file ownership or UID(0).
- Successful access or no covering profile simply continues with existing UNIX-style checks that might or might not allow access to file system object.
- The RACF AUDITOR attribute bypasses the FSACCESS check.
- UPDATE access is required to the FSACCESS class resource name that equals the containing data set name.

Note: FSACCESS class checking is only performed if the FSACCESS class is active.

The RACF setup is shown in Figure 15-37.

```
RDEFINE FSACCESS IBMUSER.ZFS.SRC UACC(NONE)
PERMIT IBMUSER.ZFS.SRC CLASS(FSACCESS) ID(IBMUSER) ACCESS(UPDATE)
SETROPTS CLASSACT(FSACCESS)
SETROPTS RACLIST(FSACCESS)
```

Figure 15-37 Setting up class FSACCESS and a profile in this class

Installation considerations

The following set of APARs are required:

- APAR OA35973 for RACF
- APAR OA35974 for SAF
- APAR OA35970 for z/OS UNIX

Note: This new function is available in z/OS V1R12 and z/OS V1R13.

15.14 IPv4 IP_PKTINFO support for AF_INET UDP sockets

This section describes the new capability to obtain the AF_INET UDP request's inbound interface information and thus allow it to be used on the reply UDP packets.

In previous releases, under a CINET environment with multiple TCP/IP stack configuration, when a server system has multiple home addresses with multiple routes back to the client, the UDP reply packet might not flow on the same interface where the UDP request packet arrived.

z/OS V1R13 provides a new z/OS IPv4 external interface to obtain a request's inbound interface information and use it on the reply.

Note: This new interface is similar to the existing IPv6 external interface.

Based on the information now available with the data, a server can send the UDP reply packet to a client request out on the same inbound physical interface on which the client's UDP request packet arrived.

Important: In security environments where clients will not trust replies that do not seem to come from the same server to which the request was sent, the server system must use the interface used by the client to send the UDP request packet.

15.14.1 Using the new interface

The new interface follows this sequence:

1. The application requests inbound interface information to be part of the **recvmsg()** data by using **SetsockOpt()** with the new `IP_RECVPKTINFO` option.

2. The TCP/IP stack includes the inbound interface information in a new IN_PKTINFO structure as an ancillary data item on **recvmsg()**.
3. The returned inbound interface information is used untouched on the subsequent **sendmsg()**.

Notes:

- The z/OS unique IPv4 external interface is non-portable because it is not stated in the standard specification.
- Other platforms have also extended the IPv6 IP_PKTINFO external interface outside of the standard specification to the IPv4 environment.

15.14.2 IPv4 IP_PKTINFO external interface

Figure 15-38 shows the definitions for sending packets on same inbound interface. You can find this in /include/netinet/in.h.

```
/* Definitions for sending packets on same inbound interface   */
/* as request arrived on.                                      */

#define  IP_PKTINFO     101 /* Ancillary data type             */
#define  IP_RECVPKTINFO 102 /* setsockopt(), receive inbound   */
                           /* interface information as        */
                           /* ancillary data on recvmsg()     */

struct in_pktinfo {
  struct in_addr  ipi_addr; /* src/dst IPv4 address           */
  unsigned int ipi_ifindex; /* send/recv interface index      */
};
```

Figure 15-38 Definitions for sending packets on same inbound interface

This provides the following ancillary data mapping structure:

```
IN_PKTINFO     DS  0F
IPI_ADDR       DS  CL4   IPv4 Address
IPI_IFINDEX    DS  F     Interface index
```

15.15 Option on vi to edit ASCII files

This feature describes new support for editing ASCII files. UNIX commands **vi** and **ex** now provide more detailed control of text conversion to assist users when editing ASCII files.

In previous releases there was no manual control of the text conversion status of files that were edited or browsed using the **vi** and **ex** editors.

z/OS V1R13 provides two new options for editing ASCII files:

- Option –W filecodeset=*codeset*,pgmcodeset=*codeset* enables text conversion on the **vi** and **ex** editors.
- Option –B on the vi and ex editors disables automatic text conversion.

Note: This is consistent with other commands that already have this override support.

The new support provides the following benefits.

- It allows a more detailed control of text conversion:
  - No file tagging is required.
  - No environment or system setup is required.
- You can easily override the system's automatic text conversion.

15.15.1 Using the enhancements

The new option **–W**, which has been added to the commands **vi** and **ex**, specifies z/OS-specific options. The option keywords are case sensitive. Possible options are **filecodeset=*codeset*** and **pgmcodeset=*codeset***.

The new option **–B**, which has also been added to the commands **vi** and **ex**, disables the automatic text conversion of tagged files. This option is ignored if the **filecodeset** or **pgmcodeset** options are specified with **-W**.

Using option –W

Using **–W filecodeset=*codeset*** performs text conversion from one code set to another when reading from or writing to the file. The code set of the file is *codeset*. The **filecodeset** and **pgmcodeset** options can be used on files with any file tag. The value for **codeset** can be a character code set name known to the system or a numeric coded character set identifier (CCSID).

If **pgmcodeset** is specified but **filecodeset** is omitted, then the default file code set is ISO8859-1, even if the file is tagged with another code set.

Restriction: The only supported values for **filecodeset** are ISO8859-1 and 819.

Option **–W pgmcodeset=codeset** works similar to the **filecodeset=codeset** option except for the following items:

- It represents the editor program's code set.
- The default program code set is IBM-1047.
- The only supported values for **pgmcodeset** are IBM-1047 and 1047.

Precedence rules

Note the following precedence rules:

- The **–W filecodeset=codeset,pgmcodeset=codeset** option overrides the **–B** option and the system's automatic text conversion.
- The **–B** option disables the system's automatic text conversion.
- If the **–W filecodeset=codeset,pgmcodeset=codeset** and **–B** options are not specified, then the system's automatic text conversion rules apply.
- The **vi** and **ex** option **–r** for recovering a file must use the same text conversion options to ensure proper file recovery.

Note: See *z/OS V1R13 UNIX System Services Command Reference* for the full technical specifications.

15.15.2 Example of using the new support

Figure 15-39 shows an example of browsing a file containing ASCII characters using the **vi** editor. The file is untagged and must be kept as is for certain reasons (for example, the file system can be mounted read-only).

```
$> cat test.ascii | iconv -f 819 -t 1047
# ------------------------------------------------------------ #
# This file contains ASCII data.                               #
# ------------------------------------------------------------ #
$> ls -T test.ascii
- untagged    T=off test.ascii
$> cat test.ascii | iconv -f 819 -t 1047 | wc -c
    198
$> vi -RW filecodeset=819 test.ascii
"test.ascii" [Read Only]
# ------------------------------------------------------------ #
# This file contains ASCII data.                               #
# ------------------------------------------------------------ #
~
~
...
~
~
:q
$>
```

Figure 15-39 Browsing a file containing ASCII data with vi

For reference, Figure 15-40 displays the initial panel on starting the **vi** session for the ASCII file.

```
# ------------------------------------------------------------ #
# This file contains ASCII data.                               #
# ------------------------------------------------------------ #
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"test.ascii" [Read Only] 3 lines, 198 characters
```

Figure 15-40 Initial panel on starting the vi session for the ASCII file

If option **-W** is not specified, then the data displayed by **vi** is not readable and the status line at the bottom of the initial panel displays the following message:

```
"test.ascii" [Read Only] [Incomplete last line]  1 lines, 198 characters
```

This is similar to what a **cat** output without conversion shows.

Note: The number of characters is the same as in the good case but the data is seen as one single incomplete line.

16

z/OS UNIX-related applications

Distributed File Service Server Message Block (SMB) support provides a server that makes Hierarchical File System (HFS) files and data sets available to SMB clients. The data sets supported include sequential data sets (on DASD), partitioned data sets (PDS), partitioned data sets extended (PDSE), and Virtual Storage Access Method (VSAM) data sets.

IBM Ported Tools for z/OS is a collection of several open source products like Opens, Perl, PHP, bzip2 or cURL that have been ported to be used just as other IBM program products with z/OS.

The open source tool `sudo` (su “do”) allows system administrators to delegate authority to give certain users or groups the ability to run some or all commands as a superuser or another user.

The utility `dbx` is a source-level debugger for z/OS UNIX System Services. It provides an environment to debug and run C and C++ programs, and to perform machine-level debugging.

This chapter provides new information and describes functions with z/OS V1R13 that are related to the following z/OS UNIX System Services-related applications:

- Windows 7 support for DFS/SMB
- Ported Tools sudo utility
- Hookless debug support for dbx

16.1 Windows 7 support for DFS/SMB

The z/OS SMB server did not previously support Microsoft Windows 7 as a z/OS SMB client. This ability was requested to access shares from Windows 7 clients.

With z/OS V1R13, this requirement has been implemented.

Notes:

- This implementation solves marketing requirement MR072709212.
- Compatibility was tested using Windows 7 Professional, Windows 7 Enterprise, and Windows 7 Ultimate.
- Support is being rolled back to z/OS V1R11 and z/OS V1R12.

With these changes, and by having the supported Windows 7 software installed on your SMB client, you can use Windows 7 clients to access z/OS data through the z/OS SMB server.

For more information, see Windows 7 under the topic about accessing data in *z/OS Distributed File Service SMB Administration*, SC24-5918.

16.2 Ported Tools sudo utility

As mentioned, the `sudo` utility is an open source tool that allows system administrators to delegate authority to provide certain users or groups of users with the ability to run some or all commands as a superuser or another user. It also provides an audit trail of the commands and their arguments. It is a command-line UNIX application. The name derives from `su` "do," meaning to run one command in superuser mode.

Previously, z/OS system administrators did not have a method that was granular or flexible enough to minimize user privileges while allowing users to perform necessary tasks. Now, having `sudo` ported to z/OS provides a useful solution to this challenge in many situations.

Note: The `sudo` utility is commonly available on other UNIX/Linux platforms.

16.2.1 Benefits of using sudo

Using `sudo` provides the following benefits:

- The utility allows system administrators to minimize user privileges while allowing users to get their work done.
- Using `sudo` is preferred over using `su` because it does not require giving the invoking user "open" access to run as the target user.

Without `sudo`, z/OS system administrators can provide high or higher authorization as described in each of the following cases:

- Case 1: The system administrator can allow users to be a permanent superuser by setting the OMVS UID to 0.
- Case 2: The user can get authorized to be a surrogate user of someone else and can use command **`su -s newuser`** to act as that user.

- Case 3: A user can get another user's password and use command **su *newuser*** to work with the authorization of that new user.
- Case 4: The administrator can give users read access to BPX.SUPERUSER and thus give them authority to use command **su** for acting in superuser mode.
- Case 5: A user can get selected UNIXPRIV authorities.

However, each of these options have inherent risks associated with them. They can provide a user with more privileges than the system administrator wants to provide. These risks can be avoided when using `sudo`.

Notes:

- In Case 1 and Case 4, the user can act as a superuser. However, the invoking user's MVS identity is not changed.
- In Case 2 and Case 3, the z/OS UNIX rights of another user can be used. Additionally, the MVS identity is switched to that user.
- In Case 5, no change of a user's z/OS UNIX or MVS identity is performed, Instead, the user is given authority to perform certain operations, for example to mount or unmeant file systems.

The `sudo` utility does not require sharing of UNIX UIDs or passwords, creating surrogates, or granting excessive authority to allow users to perform their work.

Tip: The `sudo` utility supports built-in logging of commands that are being run under `sudo` authority.

16.2.2 More information about sudo

For more information about sudo, including maintenance, refer to the following sources.

- For details about the open source version of `sudo`:

  `http://www.sudo.ws/`

- For the z/OS version of `sudo`:

  `http://www-03.ibm.com/systems/z/os/zos/features/unix/ported/suptlk/index.html`

Note:

IBM ported open source `sudo` version 1.7.2p2 to z/OS and modified the port for better z/OS integration.

This addresses two requirements requesting `sudo` for z/OS UNIX, MR0402036647 and MR08024061647.

16.2.3 Using sudo

Figure 16-1 illustrates accessing and using the sudoers file `/etc/sudoers`.

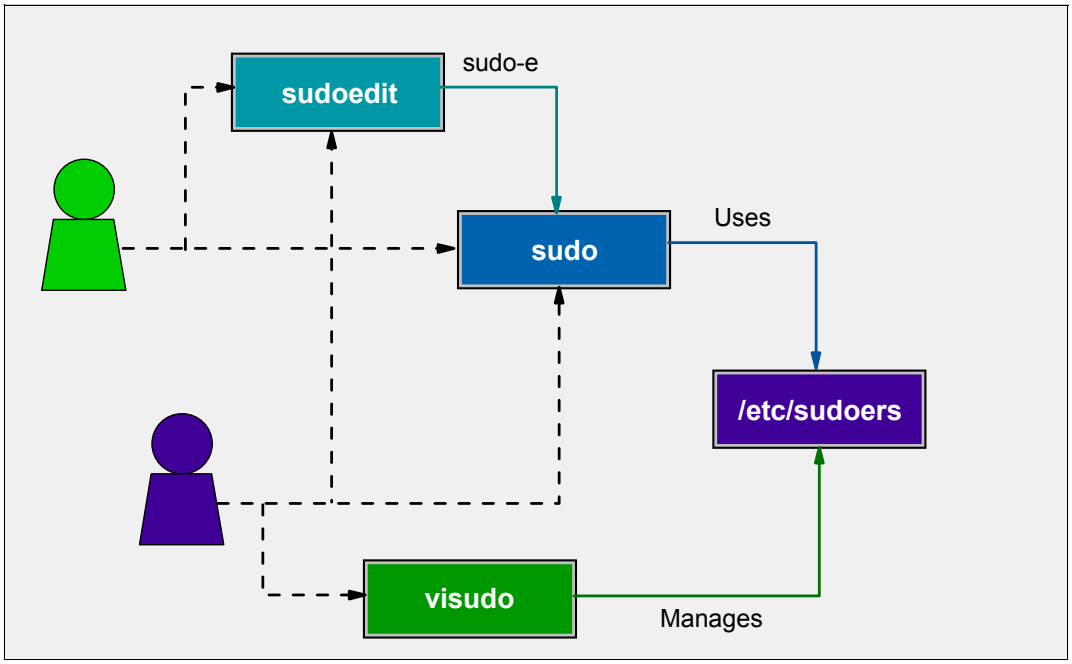


Figure 16-1 Accessing and using the sudoers file /etc/sudoers

Note the following points:

- The `sudo` utility allows a permitted user to execute a command as a superuser or another user, as specified in the sudoers file. The real and effective UID and GID are set to match those of the target user as specified in the user database. The group vector is initialized based on the group file, unless the **`-P`** option was specified.
- The MVS identity may also be changed to correspond to the target user. There are additional authority requirements unique to z/OS that must be met to change the MVS identity.

  Note: See the sudoers `zos_set_mvs_identity` option for more information.

- Using the **`sudoedit`** command is the same as running `sudo` with the **`–e`** option.
- The `sudoers` file is composed of two types of entries, namely *aliases* (which are basically variables), and *user specifications* (which determine which commands a user may run).
- The **`visudo`** command edits the `sudoers` file in a safe fashion. It locks the `sudoers` file against multiple simultaneous edits, provides basic sanity checks, and checks for parse errors. If the `sudoers` file is currently being edited, you will receive a message to try again later.
- By default, all users are allowed to attempt to use `sudo` or `sudoedit`. However, only superusers are allowed to use `visudo` and manage the `/etc/sudoers` file.

16.2.4 Examples of using sudo

Figure 16-2 shows several ways to use the **`sudo`** command.

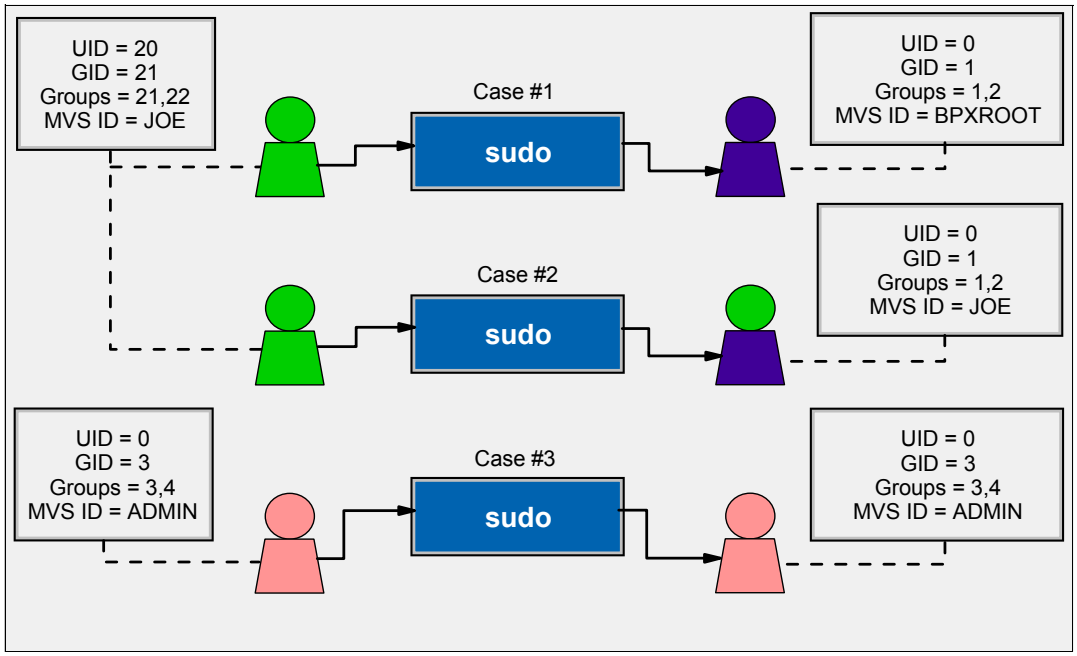


Figure 16-2 Usage examples for sudo

Use cases - sudo command

This section explains how the **sudo** command works in each case that is illustrated in Figure 16-2.

Case 1 - Full identity change

User JOE uses `sudo` to run fully as user BPXROOT. This is how the `su` command works when a user is specified.

Case 2 - Partial identity change (the default behavior)

User JOE uses `sudo` to run partially as user BPXROOT. That is, JOE has the z/OS UNIX identity of BPXROOT. This means it switches the UID, and groups accordingly. However, JOE does not get the MVS identity of BPXROOT. This is similar to how the `su` command works for a `BPX.SUPERUSER` switch.

Case 3 - No identity change (using sudo for auditing)

User ADMIN uses `sudo` to run fully as user ADMIN. This technique is often used by system administrators as an easy way to log the commands and arguments used on commands that they themselves run as superusers. `sudo` can be used in this way, as is often done on other UNIX platforms.

Other cases

There are other, less common and so-called “runas” use cases for the **sudo** command that are not referenced in Figure 16-2. For more detailed information about using sudo for z/OS, see *z/OS IBM Ported Tools for z/OS: Supplementary Toolkit for z/OS Feature User's Guide and Reference*, SA23-2234.

16.2.5 Comparing several z/OS UNIX authority interfaces

Table 16-1 compares the authority change and actions allowed or denied for various authority modifying mechanisms on z/OS. The column titled “Command control” refers to the ability to control the commands and their arguments run by a user after an ID change.

Table 16-1 Comparing several z/OS UNIX authority interfaces

Command/Authority	z/OS UNIX id change	MVS id change	Shell access	Command control
sudo	Optional	Optional	Optional	Optional
su <user>	Yes	Yes	Yes	No
su-s<user> (that is, SURROGAT)	Yes	Yes	Yes	No
su (that is, BPX.SUPERUSER)	Yes	No	Yes	No
UNIXPRIV	No	No	No	Partial

16.2.6 Installation considerations regarding security

Note the following security recommendations for user specifications:

- The `sudoers` grammar is described in Extended Backus-Naur Form (EBNF). However, this can be confusing. Therefore, using examples can be helpful.
- Make user specifications as specific as possible.
- Minimize the use of the `ALL` alias and `sudo` “chaining.”
- Specify commands with arguments or use double quotes (“ ”) to ensure that commands are run without arguments.
- Subtracting commands from the `ALL` alias using the exclamation (`!`) point operator is generally not effective.
- Minimize shell access and shell escapes.

Note: It is useful to refer to *z/OS IBM Ported Tools for z/OS: Supplementary Toolkit for z/OS Feature User's Guide and Reference*, SA23-2234, before you start to use the **`sudo`** command. The following topics are especially informative:

- *Preventing shell escapes* for the `sudoers` command
- *Security notes* for the `sudo` command
- *Security notes* for the `sudoers` command

Default option value differences between z/OS and open source

Note the following differences between the sudoers `ignore_dot` option defaults:

- The z/OS default is on.
- The open source default is off.

Note the following differences between the sudoers `runas_default` and `mailto` options defaults:

- The z/OS default is `BPXROOT`.
- The open source default is `root`.

Note the following differences between the sudoers `path_info` option defaults:

- The z/OS default is `off`.
- The open source default is `on`.

Unsupported open source functionality on z/OS

sudo options:	-A askpass, -a type, -c class, -r role, -t type
sudoers options:	askpass, ignore_local_sudoers, insults, long_opt_prompt, noexec, noexec_file, passprompt_override, pwfeedback, role, rootpw, stay_setuid, sudoers_locale, type, use_loginclass, visiblepw
sudoers specifications:	netgroup, nonunixgroup, NOEXEC or EXEC

New z/OS-specific functionality

Environment variables:	`_ZOS_SUDO_NOMSGID` and `_ZOS_SUDO_DEBUG`
sudoers options:	`zos_set_mvs_identity`
sudoers specifications:	`ZOS_SET_MVS_IDENTITY` or `NO_ZOS_SET_MVS_IDENTITY`

Specific sudo settings and comments

This section briefly describes several specific `sudo` settings and provides examples.

- Using `sudo` option **`-E`** to preserve the environment variables overrides the `env_reset` option in the `sudoers` file.
  - It is only available when either the matching command has the SETENV tag, or the setenv option is set in the `sudoers` file.
  - In addition, it cannot be used in conjunction with the **`-i`** option.
- By default, `sudo` will initialize the group vector to the list of groups that the target user is in. When `preserve_groups` is set in the `sudoers` file, the user's existing group vector is left unaltered, as shown here:

```
Defaults preserve_groups
```

- In the same way, when using `sudo` option **`-P`**, this preserves the invoking user's group vector unaltered.

> **Note:** In both cases, however, the real and effective group IDs are set to match the target user.

- Using the following specification for environment variables allows you to replace using **`vi`** for editing with **`sudo -e`** or **`sudoedit`** by setting the environment variable `SUDO_EDITOR`:

```
Defaults env_keep=SUDO_EDITOR
```

- The following user privilege specification in the `sudoers` file achieve a behavior similar to allowing the user read access to `BPX.SUPERUSER`:

```
HERING ALL = NOPASSWD: ALL
```

16.2.7 Additional sudo command examples

Following are two additional examples demonstrating how sudo works.

Example 1

This example shows how to allow users on a team, named BACKUPS in the sudoers file, the ability to run a specific **pax** command as a specific superuser and administrator named "admin" with specific arguments determined by the administrator.

```
/etc/sudoers file entries:
   Defaults umask=077
   User_Alias BACKUPS = june, fred, mary
   BACKUPS ALL = (admin) /bin/pax -x pax -wf /u/code/src.pax /u/code/src
```

This is the **sudo** command that is used:

```
sudo –u admin pax -x pax -wf /u/code/src.pax /u/code/src
```

Using sudo provides the following benefits:

- The backup team is not allowed to view the data they back up as an administrator.
- The backup team is not allowed to run other **pax** commands or change **pax** options as an administrator.
- An audit trail is provided for every backup performed by the backup team.

Example 2

Log all commands run by superuser "admin".

```
/etc/sudoers file entries:
   admin ALL=(admin) ALL
```

This is the example **sudo** command that is used:

```
sudo rm –rf /u/baduser
```

Using sudo provides the following benefit:

- An audit trail is provided for every command run by the superuser admin using sudo.

Here is a sample syslog audit entry created by sudo:

```
Aug 25 08:00:04 SY1 sudo:   admin : TTY=ttyp0000 ; PWD=/SYSTEM/tmp/syslogd ;
USER=admin ; GROUP=admingrp ; COMMAND=/bin/rm -rf /u/baduser
```

Note that sudo can log both successful and unsuccessful attempts, and errors, to syslog, a log file, or both. By default, sudo will log using syslog, but this is changeable by way of the sudoers file.

16.2.8 Migration and coexistence considerations

There are unlikely to be migration and coexistence considerations because there is no previous version from IBM. This can be considerations if a previous version from a non-IBM source was used.

Note: For more information, refer to the topic about migrating from previous versions *z/OS IBM Ported Tools for z/OS: Supplementary Toolkit for z/OS Feature User's Guide and Reference*, SA23-2234.

Installation considerations

Note the following migration and installation considerations for coexistence:

- The `sudo` command for z/OS is provided through APAR OA34949 and PTF UA59179; see *z/OS IBM Ported Tools for z/OS: Supplementary Toolkit for z/OS Feature User's Guide and Reference*, SA23-2234, FMID HPUT110.
- The utility `sudo` for z/OS is supported for z/OS V1R10 or higher.
- For z/OS V1R10 and z/OS V1R11, the PTF for APAR OA32470 must be applied.

Note: For more information, see the topic about installing the Supplementary Toolkit for z/OS in *z/OS IBM Ported Tools for z/OS: Supplementary Toolkit for z/OS Feature User's Guide and Reference*, SA23-2234.

Pre-installation planning

For pre-installation planning purposes, note the following points:

- For the Apart, new directories must be created before installation.
- For the Supplementary Toolkit, all directories are created during installation.
- New and updated files and required links will be created during installation.
- Verify the z/OS release requirements as described.

Note: For more information, see the topic about pre-installation planning in *z/OS IBM Ported Tools for z/OS: Supplementary Toolkit for z/OS Feature User's Guide and Reference*, SA23-2234.

- setup and verification

The following steps are required for post-installation setup and verification:

- Enable sudo for z/OS. The SAMPLIB member HPUTIFA provides an example.
- Copy the `sudoers` file to `/etc/sudoers`.

```
# sudoers must have mode 0440 (i.e. read for owner and group).
# sudoers must be owned by UID 0 and GID 0.
cp -p /usr/lpp/ported/samples/sudoers /etc/sudoers
```

- Customize the `/etc/sudoers` file for your installation using `visudo`.

```
# By default, there's no sudo authority.
# By default, BPXROOT is the default runas and mailto user.
visudo
```

Note: For more information, see the topic about post-installation setup and verification in *z/OS IBM Ported Tools for z/OS: Supplementary Toolkit for z/OS Feature User's Guide and Reference*, SA23-2234.

The following steps are recommended.

- Add a symbolic link to the man pages, if necessary.

```
/usr/man/C/man1/hpuza200.book
# symlink --> /usr/lpp/ported/man/C/man1/hpuza200.book
```

- Add a symbolic link to the message catalog:

```
/usr/lib/nls/msg/C/hpusudo.cat
# symlink --> /usr/lpp/ported/lib/nls/msg/C/hpusudo.cat
```

- Add a symbolic link to the binaries:

```
/usr/bin/sudo      # symlink --> /usr/lpp/ported/bin/sudo
/usr/bin/visudo    # symlink --> /usr/lpp/ported/bin/visudo
/usr/bin/sudoedit  # symlink --> /usr/lpp/ported/bin/sudoedit
```

Verify the sudo for z/OS installation:

- Command `sudo` must be owned by UID 0.
- Command `sudo` must have mode 4111. This means the execute bit is on for all and the set-user-ID bit is on.
- Command `sudo` must not have set the shareas extended attribute.
- Command `sudo` must have the program control extended attribute.

Note: For more information, see the topic about post-installation setup and verification in *z/OS IBM Ported Tools for z/OS: Supplementary Toolkit for z/OS Feature User's Guide and Reference*, SA23-2234.

Further installation information about sudo for z/OS

Additional installation information about sudo for z/OS is available as listed here.

Updated toolkit parts for sudo for z/OS:

```
/usr/lpp/ported/Ported_Tools_License.readme
/usr/lpp/ported/man/C/man1/hpuza200.book
SYS1.SAMPLIB(HPUTIFA)
SYS1.SAMPLIB(HPUTMKDR)
```

New sudo for z/OS parts:

```
/usr/lpp/ported/bin/base/sudo-1.7.2p2
/usr/lpp/ported/bin/base/visudo-1.7.2p2
/usr/lpp/ported/samples/sudoers
/usr/lpp/ported/lib/nls/msg/C/hpusudo.cat
# Supporting directories (lib/nls/msg/C) are also new.
```

New sudo for z/OS symbolic links:

```
/usr/lpp/ported/bin/sudo      # --> base/sudo-1.7.2p2
/usr/lpp/ported/bin/sudoedit  # --> base/sudoedit-1.7.2p2
/usr/lpp/ported/bin/visudo    # --> base/visudo-1.7.2p2
```

New sudo for z/OS hard links:

```
/usr/lpp/ported/bin/base/sudoedit-1.7.2p2  # --> ./sudo-1.7.2p2
/usr/lpp/ported/IBM/HPUDXSUD  # --> ../bin/base/sudo-1.7.2p2
/usr/lpp/ported/IBM/HPUDXVIS  # --> ../bin/base/visudo-1.7.2p2
/usr/lpp/ported/IBM/HPUDUERS  # --> ../samples/sudoers
/usr/lpp/ported/IBM/HPUDRCAT  # --> ../lib/nls/msg/C/hpusudo.cat
```

16.3 Hookless debug support for dbx

When code is compiled for debug, it results in a program that is larger, runs slower, and sometimes does not allow you to reproduce problems seen in production code.

In z/OS V1R13, you can use the **dbx** command to debug a program compiled with the `-qdebug=nohook` or `-Wc,"DEBUG(NOHOOK)"` options, which provides the following benefits:

- It allows you to debug unoptimized code that is closer to production code.
- It helps you to better reproduce problems seen in production code.
- Code compiled without EX hooks will run significantly faster.
- Not compiling with EX hooks will result in smaller programs.

16.3.1 Using the hookless debug enhancement

Figure 16-3 shows an example of compiling a test program without EX hooks and then debugging it.

```
$> xlc -g -qdebug=nohook test.c
$> cc -g -Wc,"DEBUG(NOHOOK)" test.c
$> dbx a.out
(dbx64) s
stopped in main at line 13 in file "test.c"  ($t1)
   13       foo();
(dbx64) s
stopped in foo at line 5 in file "test.c"  ($t1)
    5       i = i+3;
(dbx64)
```

Figure 16-3 Compiling a test program without EX hooks and debugging it

Notes:

- You do not have to do a stop in `main` and then continue. You can just step.
- The `dbx` utility can only stop on lines for which there is code generated.
- Stepping into `foo` does not stop on the first line of `foo`, but on the first statement in `foo`.

16.3.2 List instructions from the program

Figure 16-4 shows an example with no EX hooks.

```
(dbx64) listi
0x22393974 (foo+0x4c)  5800d098     L       R0,152(,R13)
0x22393978 (foo+0x50)  a70a0003     AHI     R0,3
0x2239397c (foo+0x54)  5000d098     ST      R0,152(,R13)
0x22393980 (foo+0x58)  5800d098     L       R0,152(,R13)
0x22393984 (foo+0x5c)  a70a0001     AHI     R0,1
0x22393988 (foo+0x60)  5000d098     ST      R0,152(,R13)
0x2239398c (foo+0x64)  5800d098     L       R0,152(,R13)
0x22393990 (foo+0x68)  a70a000a     AHI     R0,10
0x22393994 (foo+0x6c)  5000d098     ST      R0,152(,R13)
```

Figure 16-4 Listing instructions in case of no EX hooks

Figure 16-5 shows the same situation when not compiled with the options `-qdebug=nohook` or `-Wc,"DEBUG(NOHOOK)"`.

```
(dbx64) listi
0x22393978 (foo+0x50)  4400c1ac     EX      0,428(,R12)
0x2239397c (foo+0x54)  5800d098     L       R0,152(,R13)
0x22393980 (foo+0x58)  a70a0003     AHI     R0,3
0x22393984 (foo+0x5c)  5000d098     ST      R0,152(,R13)
0x22393988 (foo+0x60)  4400c1ac     EX      0,428(,R12)
0x2239398c (foo+0x64)  5800d098     L       R0,152(,R13)
0x22393990 (foo+0x68)  a70a0001     AHI     R0,1
0x22393994 (foo+0x6c)  5000d098     ST      R0,152(,R13)
0x22393998 (foo+0x70)  4400c1ac     EX      0,428(,R12)
0x2239399c (foo+0x74)  5800d098     L       R0,152(,R13)
```

Figure 16-5 Listing instructions if not compiled with -qdebug=nohook or -Wc,"DEBUG(NOHOOK)"

You can debug programs whether or not they are compiled with EX hooks.

- When `dbx` turns them on, it stops at every one.
- With this support we never turn them on. They are like noops to `dbx`.

17

Resource Recovery Services

Resource recovery refers to the protection of resources. Resource recovery consists of the protocols and program interfaces that allow an application program to make consistent changes to multiple protected resources.

z/OS, when requested, can coordinate changes to one or more protected resources, which can be accessed through several resource managers and reside on several systems. z/OS ensures that all changes are made or no changes are made. Resources that z/OS can protect include:

- Hierarchical databases
- Relational databases
- Product-specific resources

This chapter describes the new functional enhancement to RRS with z/OS V1R13 known as automated shutdown for RRS.

17.1 Automated shutdown

In prior releases of z/OS there was no way to terminate RRS in a clean way after an RRS failure. Starting with z/OS V1R13, RRS has enhanced the ATR246I message and the ATR247E message that appears when RRS has an abend with the shutdown option. Figure 17-1 shows the new syntax of the changed messages.

```
*SY1  ATR246I RRS HAS DETECTED A SHE CONTROL BLOCK ERROR - UNEXPECTED ERROR
DUMP REQUESTED
*SY1 *23 ATR247E RRS HAS DETECTED A SEVERE ERROR - TERMINATE RMS AND
*OPTIONALLY REPLY SHUTDOWN TO SHUTDOWN RRS.
```

Figure 17-1 New updated ATR247E message

The figure shows that RRS processing has detected a problem with the identified control block that might potentially need to be investigated. In this message text, the SHE control block contains an error. The failing control block might not be the same in the message.

RRS processing continues but individual transaction results might be impacted and monitoring them is a good practice. If the processing has determined that the error is severe, then the ATR247E message will be issued to signal that action needs to be taken.

If you reply SHUTDOWN to the ATR247E message, RRS tries to attempt a clean shutdown of RRS. If a clean shutdown does not occur within 30 seconds, a forced shutdown is attempted.

You can collect the available diagnostic information, including the unexpected error dump and the associated symptom records, and then contact IBM Service. The dump for this message might have been suppressed by DAE if there was a prior occurrence of the error for the same control block in the same module.

In addition, there can be multiple messages reporting errors for the same control block depending on what the problem is.

Important: Check your system automation in regard to the enhanced message, because prior versions of this WTOR did not have the SHUTDOWN option. Therefore, existing automation based on that prior version will function as it did previously, if the automation looks at the message number (ATR247E) rather than the exact text.

If there is no response from the operator, then RRS keeps running, with the terminated thread halted. Alternatively, you can be shut down by other means such as by the **SETRRS SHUTDOWN** command. That was the only option on prior RRS releases.

Note: Issuing the **SETRRS SHUTDOWN** command provides a normal shutdown command to bring down RRS without resulting in a X'058' abend or X'0D6' abend.

To notify RRS resource managers that RRS is terminating, all currently active resource managers will be unset. After unset processing is completed, the RRS jobstep task and all of its subtasks will normally be terminated to clean up the address space.

This new implementation informs the operator that a shutdown is necessary and provides a straightforward way to perform it. It also gives the operator a chance to terminate the resource managers.

Important: The IBM default in the AUTOR feature is SHUTDOWN as an answer to the ATR247E message.

18

IBM z/OS Management Facility

IBM z/OS Management Facility (z/OSMF) provides a framework for managing various aspects of a z/OS system through a web browser interface. By streamlining various traditional tasks and automating others, z/OSMF can help to simplify areas of system management and reduce the level of expertise needed for managing a system.

z/OSMF provides a single platform for hosting the web-based administrative console functions of IBM server, software, and storage products. With z/OSMF, you manage solutions rather than specific IBM products.

This chapter describes the following topics:

- IBM z/OS Management Facility (z/OSMF)
- Focus areas for z/OSMF V1R13
- z/OSMF new system management tasks with z/OS V1R13, namely Capacity Provisioning, DASD Management, Deployment, and ISPF
- New base capabilities and enhancements in z/OSMF V1R13 for managing tasks and keeping up with browsers and configuration setup
- z/OSMF configuration and setup specifics

18.1 IBM z/OS Management Facility (z/OSMF)

In z/OS releases prior to z/OS R11, there is no central system management portal for z/OS. Instead, many interfaces are somewhat unfamiliar to users who are new to the System z platform, performing manual tasks can require extensive documentation, and overall it can require significant z/OS experience to be completely productive.

IBM z/OS Management Facility (z/OSMF), introduced with z/OS V1R11 at no additional charge, simplified, optimized, and modernized the z/OS systems programmer experience in the following areas:

- z/OSMF delivers solutions in a task-oriented, web browser-based user interface with integrated user assistance.
- z/OSMF eases the management of daily operations and administrative tasks of the mainframe z/OS systems for both new and experienced systems programmers.
- z/OSMF facilitates systems programmer productivity, making the functions easier to understand and use.

18.1.1 Introduction to the new faces for z/OS

z/OS V1R9 introduced the first steps of a series of ongoing efforts designed to simplify the management, administration, and configuration of the system through the following enhanced areas:

- New user interfaces (UIs)
- The enablement of remote access technologies on z/OS
- Extensions to base operating system components to allow for such management

18.1.2 z/OS ease-of-use enhancements

To address the skill base for z/OS, a new approach has been taken to develop and increase the z/OS skills of less experienced members of the zSeries IT community in the following areas.

z/OS basics

In collaboration with the ITSO, z/OS development has created an IBM Redbooks publications “basics” library for z/OS. If you are new to z/OS systems programming and have recently assumed the role of systems programmer or system analyst, using the *z/OS Basics:* series of publicaitons can help you develop your understanding of the various aspects of the z/OS system.

Configuration and operations

The IBM OMEGAMON® z/OS Management Console is a no-charge availability monitoring product that includes a GUI for z/OS management. It is designed to help the new generation of IT workers automate, eliminate, and simplify many z/OS management tasks. The OMEGAMON z/OS Management Console helps deliver real-time, health check information provided by the IBM Health Checker for z/OS, and configuration status information for z/OS systems and sysplex resources.

IBM Health Checker for z/OS is a base function for z/OS V1R7. It provides a foundation to simplify and automate the identification of potential configuration problems before they impact

system availability. It compares active values and settings to those suggested by IBM or defined by your installation.

Software maintenance

SMP/E V3.4 and later has been enhanced to provide Internet Service Retrieval. This capability allows you to automate ordering and delivery of PTFs and HOLDDATA. The PTFs and HOLDDATA can be processed in the same job step. This can help eliminate manual tasks required for ordering and delivery of IBM PTFs using current methods.

Networking

The z/OS Network Security Configuration Assistant GUI can dramatically reduce the amount of time needed to create configuration files. It can be used to generate the configuration files for both Application Transparent-Transport Layer Security (AT-TLS) and IP Security (IPSec).

The z/OS Network Security Configuration Assistant is a stand-alone application that runs under the Windows operating system and requires no network connectivity or setup. The GUI can be downloaded from the Communications Server family downloadable tools web page. Through a series of wizards and online help panels, the GUI can be used to create both AT-TLS and IPSec configuration files for any number of z/OS images with any number of TCP/IP stacks per image.

IBM Academic Initiative

The IBM Academic Initiative brings together interested colleges and universities, zSeries client partners, and IBM resources to provide a link between the academic world and the business world and potential future career opportunities. The goal is to increase student awareness of the z/OS platform while developing zSeries skills to meet the current and future programmer needs of our zSeries clients.

Membership in the Scholars zSeries Program includes access to a native z/OS system through the Internet for course exercises, systems programmer assistance to instructors for setting up and teaching their classes on z/OS, access to zSeries course material, and opportunities for faculty education. A focus of the program is working with zSeries clients to help them find schools to partner with to develop zSeries skills.

LibraryCenter

The LibraryCenter provides a view of IBM z/OS BookManager® documentation and presents the information in a Windows Explorer format. The LibraryCenter contains documentation for z/OS elements, features, software products and selected z/OS-related IBM Redbooks publications. There are multiple library centers, one for each release, and also one for z/VM.

Security

These RACF-based products can assist in managing security and monitoring compliance:

- Partnership with Vanguard Integrity Professionals, Inc. including the Administrator, Advisor, Analyzer, Enforcer, and SecurityCenter products
- IBM Tivoli administration for RACF, which is a lower function alternative

18.1.3 The “new face” of z/OS

Beginning with z/OS, various steps have been taken to lay the foundation for simplified access to the z/OS platform. These “new faces” are embodied under the cover of different functions in z/OS V1R9 with a set of new tools enhancing the “MVS face” of z/OS, as shown in Figure 18-1, to provide the following new functions. In addition, with z/OS V1R11, IBM introduced z/OSMF to the z/OS platform.

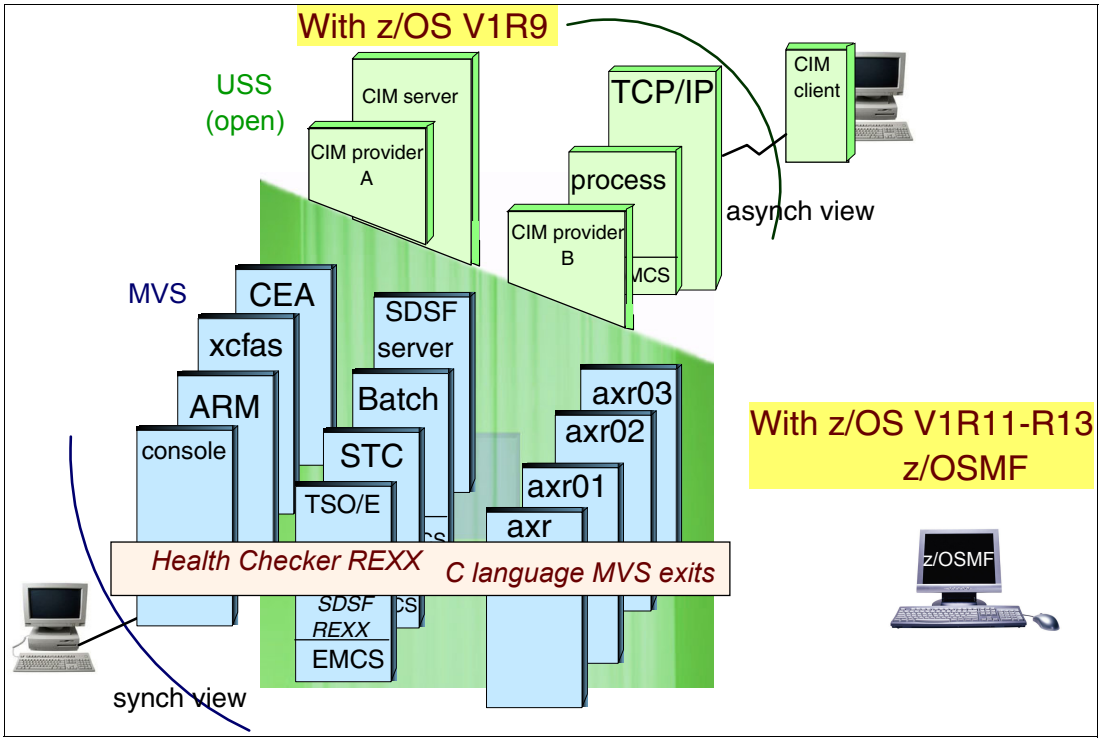


Figure 18-1 z/OS V1R9 and z/OS V1R11 - R13 z/OSMF

System REXX

System REXX is a z/OS component that allows REXX execs to be executed outside of conventional TSO/E and batch environments. REXX has long been considered one of the fastest development languages for system exit and utilities work on z/OS. The possibilities for exploiting REXX code through the use of System REXX are vast, whether to provide operator assists or to provide an easy way to process files and strings.

The System REXX environment provides a function package that allows a REXX exec to invoke the system commands and to return results back to the invoker in a variety of ways. System REXX execs may be initiated through an assembler macro interface called AXREXX, or through an operator command.

SDSF REXX

SDSF was enhanced to add the capability to provide access to SDSF functions through REXX variables. The variables are loaded with data from the SDSF panels. This enables them to be processed by REXX execs. The data can also be changed, which provides capabilities similar to those provided in the SDSF dialog by action characters and overtyping.

Starting with z/OS V1R9, you can access SDSF function with the REXX programming language. Using SDSF REXX provides a simpler and more powerful alternative to using SDSF in batch.

Using REXX to write health check routines

IBM Health Checker for z/OS supports checks that are written in REXX using SYSREXX facility made available with z/OS V1R9. This SYSREXX facility makes it easier for you to write your own health checks.

XL C Metal compiler option

Prior to z/OS V1R9, all z/OS XL C compiler-generated code required Language Environment. In addition to depending on the C runtime library functions that are available only with Language Environment, the generated code depended on the establishment of an overall execution context, including the heap storage and dynamic storage areas. These dependencies prohibit you from using the XL C compiler to generate code that runs in an environment where Language Environment did not exist.

With z/OS V1R9, the XL C Metal compiler option generates code that does not have access to the Language Environment support at run time. Instead, the Metal option provides C language extensions that allow you to specify assembly statements that call system services directly. Using these language extensions, you can provide almost any assembly macro, and your own function prologs and epilogs, to be embedded in the generated HLASM source file. When you understand how the Metal-generated code uses MVS linkage conventions to interact with HLASM code, you can use this capability to write freestanding programs.

Common event adapter

Common event adapter (CEA) is a component introduced in the z/OS base with z/OS V1R9. Its must be up and running for the CIM server to properly operate. It provides the ability to deliver z/OS events to C language clients, such as the z/OS CIM server.

A CEA address space is started automatically during initialization of every z/OS V1R9 (or higher) system. CEA has two modes of operation, full function and minimum:

- Full function mode

  In this mode, both internal z/OS components and clients such as CIM providers using the CEA application programming interface can use CEA functions.

- Minimum mode

  In this mode, only internal z/OS components can use CEA functions.

The EA) provides the ability to deliver z/OS events to C language clients, such as the z/OS CIM server. The CEA address space is started automatically during z/OS initialization and does not terminate.

Common Information Model

The Common Information Model (CIM) is a standard data model developed by a consortium of major hardware and software companies including IBM. This consortium, known as the Distributed Management Task Force (DMTF), is part of the Web Based Enterprise Management (WBEM) initiative. CIM was introduced in z/OS with z/OS V1R7.

The WBEM initiative includes a set of standards and technologies that provide management solutions for a distributed network environment. Interoperability is a major focus of WBEM, and using WBEM technologies can help you develop a single set of management applications for a diverse set of resources.

18.2 Introduction to z/OSMF V1R13

As mentioned, z/OSMF is a no charge IBM separately licensed program product for z/OS. Structurally, z/OSMF is a set of web applications hosted on the z/OS system. Depending on the system management task to be performed, z/OSMF interfaces with other z/OS components to offer a simplified interface for performing tasks.

These components make up the environment necessary for using the z/OSMF functions. z/OSMF does not provide a separate client installation. You will need to provide a compatible browser to access the z/OSMF web application.

Software delivery options for z/OSMF

For ServerPac and CBPDO users:

- If you select the full system replacement installation type, a default instance of z/OSMF is set up for you. Here, z/OSMF is configured through a ServerPac post-installation job, using mostly IBM-supplied defaults.

  If you use full system replacement, review the setup steps in this document to ensure that z/OSMF is configured correctly for your installation.
- If you select the software upgrade installation type, you require the planning and configuration information. Your installation's systems programmer must set up the product run-time, IBM WebSphere Application Server OEM Edition for z/OS, and z/OSMF, through shell scripts that are provided with the product.
- For a software upgrade installation, if you accept the system defaults, ServerPac provides customization guidance for configuring z/OSMF. See the copy of “ServerPac: Installing Your Order” that is supplied with your order.
- If you receive z/OSMF in a Custom-Built Product Delivery Option (CBPDO) software delivery package, you require the planning and configuration information. Your installation's systems programmer must set up the product run-time, IBM WebSphere Application Server OEM Edition for z/OS, and z/OSMF, through shell scripts that are provided with the product.

Note: For z/OS V1R13 delivery options, see *z/OS IBM z/OS Management Facility Configuration Guide*, SA38-0652.

z/OSMF includes the following software

IBM WebSphere Application Server OEM Edition for z/OS Version 7 provides a native application server runtime environment for z/OSMF, and the tools for configuring z/OSMF.

A set of administration and system management tasks run on IBM WebSphere Application Server OEM Edition for z/OS. There are technologies for serving the web browser interface, such as JavaScript and a Dojo framework.

The goal of this architecture is to provide simplified systems management function through a common, easy-to-use, graphical user interface. Figure 18-2 shows a typical architecture and flow, starting with the user's browser session and continuing through z/OSMF and IBM WebSphere Application Server OEM Edition for z/OS, with information passed to various z/OS system components as needed.

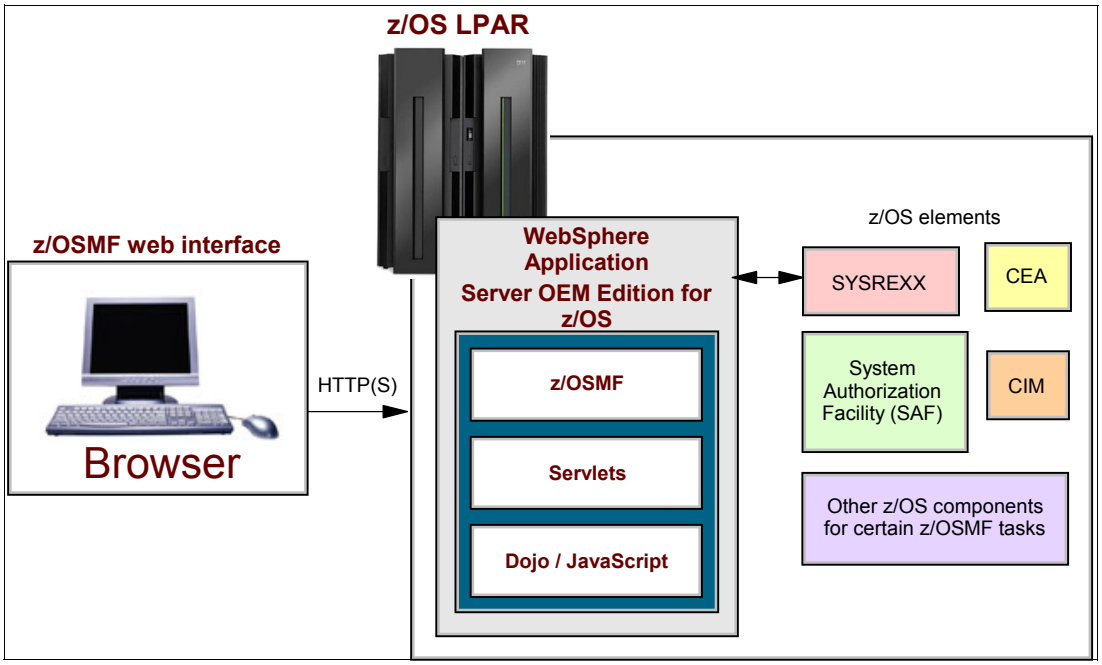


Figure 18-2 z/OSMF, WebSphere Application Server OEM, and related system components

As previously mentioned, and depending on the particular system management task being performed, z/OSMF (that is, WebSphere Application Server OEM) uses new enabling technologies on z/OS, such as the following z/OS components:

- Common Information Model (CIM) server running on the host z/OS system

  This component provides the z/OS data and administrative capability.

- Common event adapter (CEA)

  This component enables CIM providers to identify, receive, and process selected z/OS events.

- System Authorization Facility (SAF)

  This component enables programs to use system authorization services to control access to resources, such as data sets and MVS commands. SAF either processes security authorization requests directly, or works with RACF or other security products, to process them.

- System REXX (SYSREXX)

  This component provides an infrastructure through which programs written in the REXX language can be run outside the normal TSO/E or batch environments, using a programming interface.

Supported browsers

To work with z/OSMF, your workstation requires a supported operating system and web browser. Table 18-1 displays the supported web browsers and workstation operating systems for use with z/OSMF.

Table 18-1 Supported web browsers for z/OSMF

Browser	Windows 7 Professional 64-bit	Windows XP 32-bit	Windows Vista Business Edition 32-bit	Windows 7 Professional 32-bit
Firefox 3.5	Yes	Yes	Yes	Yes
Firefox 3.6	Yes	Yes	Yes	Yes
Internet Explorer 7	Yes	Yes	Yes	Yes
Internet Explorer 8	Yes	Yes	Yes	Yes
Internet Explorer 64-bit	Yes	No	No	No

Note: z/OSMF provides an environment checker tool to help you verify your browser and workstation settings at any time. The environment checker tool, for example, can verify that the resolution is 1024 by 768 or higher, that cookies are enabled, and that JavaScript is enabled. For more information, see the help topic “Verifying your workstation with the environment checker,” as shown in Figure 18-39 on page 414.

18.3 z/OSMF administrator with z/OS V1R13

z/OSMF V1R13 enhances its security model, enables cross-application linkage, and introduces a new HTTP interface for submitting and accessing job information.

The z/OSMF administrator ID is created with both a TSO and an OMVS segment because it is expected someone will logon to TSO with this ID. The TSO logon processing sets up a user environment, giving access to the necessary resources.

Review the existing logon procedure that your systems programers use to log on to TSO. You can either use that procedure or customize a procedure based on it. The z/OSMF scripts or ServerPac jobs can reference the procedure libraries and build the RACF ALTUSER statement with the procedure name. The scripts do not generate the procedure. If you use another security product, you can our commands as a template and translate them to the appropriate equivalent.

z/OSMF user authorizations

In previous releases, user authorizations were defined by the z/OSMF administrator in the product repository and enabled on the host system by your security administrator. This earlier form of authorization, now referred to as Repository Authorization Mode, remains a supported option in this release. However, SAF Authorization Mode is the default for z/OSMF, so plan according for this changed mode.

This release includes tools to help with converting your existing z/OSMF user authorizations to SAF profiles and groups, which is required for using SAF mode.

SAF Authorization With this mode, z/OSMF user authorizations are managed by your security management product, such as RACF. Access to z/OSMF links and tasks is maintained through profiles in the ZMFAPLA resource class profiles, which is new in z/OS V1R13. Here, authorizations are enabled when your security administrator defines groups, connects users to those groups,

and permits those groups to the profiles that represent the z/OSMF tasks. This comprehensive form of security is called SAF Authorization Mode because it relies on traditional SAF interfaces to manage user authorizations, as is done today for most products and subsystems that run on z/OS. SAF Authorization Mode is the recommended and default mode of authorization for z/OSMF.

In SAF Authorization Mode, user authorization to resources (tasks and links) is based on SAF users and groups in your security management product.

Repository Authorization

With this mode, z/OSMF user authorizations are managed internally by z/OSMF. This form of security is called Repository Authorization Mode because user authorizations are maintained in the z/OSMF repository, rather than in your security management product. Here, authorizations are created when the person who is designated as the z/OSMF Administrator assigns tasks to roles and roles to users through the Users and Roles panels. Your installation is limited to the predefined roles of the z/OSMF Administrator, Storage Administrator, or z/OSMF user.

In Repository Authorization Mode, user access to tasks is administered based on user roles, which are explicitly assigned to users by the z/OSMF Administrator.

18.3.1 SAF Authorization Mode

As an aid to your security administrator, z/OSMF provides a set of REXX execs with RACF commands for creating a secure environment. These execs are created during the z/OSMF configuration process, and are customized for your installation, based on the plug-ins you have selected to configure.

During the configuration process, your security administrator runs the REXX exec izuconfig1.cfg.rexx. This exec contains commands for setting up a default security environment through the RACF security product. The REXX exec contains RACF commands for the following tasks:

- Activating the ZMFAPLA resource class and enabling it for RACLIST processing.
- Creating ZMFAPLA resource class profiles for each of the z/OSMF tasks to be enabled on your system.

Note: To allow users in your installation to access z/OSMF, your security administrator must authorize the users to resources on the z/OS system. As an aid to your security administrator, z/OSMF includes sample REXX programs with RACF commands for authorizing users.

Figure 18-3 shows an example of the RACF commands that your security administrator can use to define a profile for a task (in this case, the Configuration Assistant task), and permit a group to that task.

```
//AUTHZMF JOB X,CAC,MSGLEVEL=(1,1),CLASS=A,
// MSGCLASS=A,NOTIFY=&SYSUID
//*-------------------------------------------------
//* The following RACF commands setup a basic z/OSMF
//* environment in SAF Authorization Mode
//* with profiles in the ZMFAPLA class
//*-------------------------------------------------
//TSORACF EXEC PGM=IKJEFT01
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
setr classact(zmfapla) generic(zmfapla) raclist(zmfapla)
addgroup IZUADMIN OMVS(GID(9003))
addgroup IZUUSER OMVS(GID(9004))
addgroup IZUSTGA OMVS(GID(9005))
connect (zosmfad) group(IZUADMIN) auth(use)
connect (zosmft2) group(IZUUSER) auth(use)
rdefine zmfapla BBNBASE.ZOSMF.** uacc(none)
rdefine zmfapla BBNBASE.ZOSMF.ADMINTASKS.** uacc(none)
PERMIT BBNBASE.ZOSMF.** CLASS(ZMFAPLA) ID(IZUADMIN) ACCESS(READ)
PERMIT BBNBASE.ZOSMF.** CLASS(ZMFAPLA) ID(IZUUSER) ACCESS(READ)
PERMIT BBNBASE.ZOSMF.** CLASS(ZMFAPLA) ID(IZUSTGA) ACCESS(READ)
PERMIT BBNBASE.ZOSMF.ADMINTASKS.** CL(ZMFAPLA) ID(IZUADMIN) ACC(READ)
setr raclist(zmfapla) refresh
/*
```

Figure 18-3 RACF commands for setting up security in z/OSMF

Using z/OSMF SAF authorization

z/OSMF exploits new resource classes, profiles, and groups. Plug-ins register resource names that are associated with each of their tasks, and all task resource names with the ZMFAPLA resource class.

IBM reserved resources are introduced:

- ZOSMF.<plugin-name>.<task-name>.<action-control-qualifier>
- The plug-in profile is being defined at deployment time.
- For delayed deployment and activation of plug-ins, reserved resources creates profiles when the plug-in is deployed.

z/OSMF V1R13 provides support for custom roles through the creation of additional SAF groups at the systems programmer's discretion, as shown in Figure 18-4. Granularity of access is determined by z/OSMF resource profile permissions for a given group.

There is an option to stay with repository mode or convert to SAF mode. You can decide to switch to SAF mode at a more convenient time.

z/OSMF has created a worksheet for the DASD Management task variables as shown in Figure 18-5 as follows:

- Group for authorizing users to the DASD Management task

  Group name to use for authorizing user access to the DASD Management task. The configuration process permits this group to the DASD Management task.

  Variable Name: IZU_STORAGE_GROUP_NAME **Default name:** IZUSTGA **Your value:**

- DASD group GID or AUTOGID

  DASD group GID to use for the DASD group administrator identity. Instead of specifying the GID value, you can enter AUTOGID to have RACF automatically generate a unique ID. **Variable name:** IZU_STORAGE_GROUP_GID **Default value:** 9005 **Your value:**

ZMFAPLA resource class

Configuration support is provided for conversion to SAF mode through scripts. This requires the activation of the ZMFAPLA resource class and is enabled for generic profiles, if needed. You have the ability to switch back to repository mode if needed, though it is inadvisable to switch back and forth repeatedly.

SAF mode is definitively the strategic destination. Users, roles. groups and task authorization are managed differently, depending on the chosen mode.

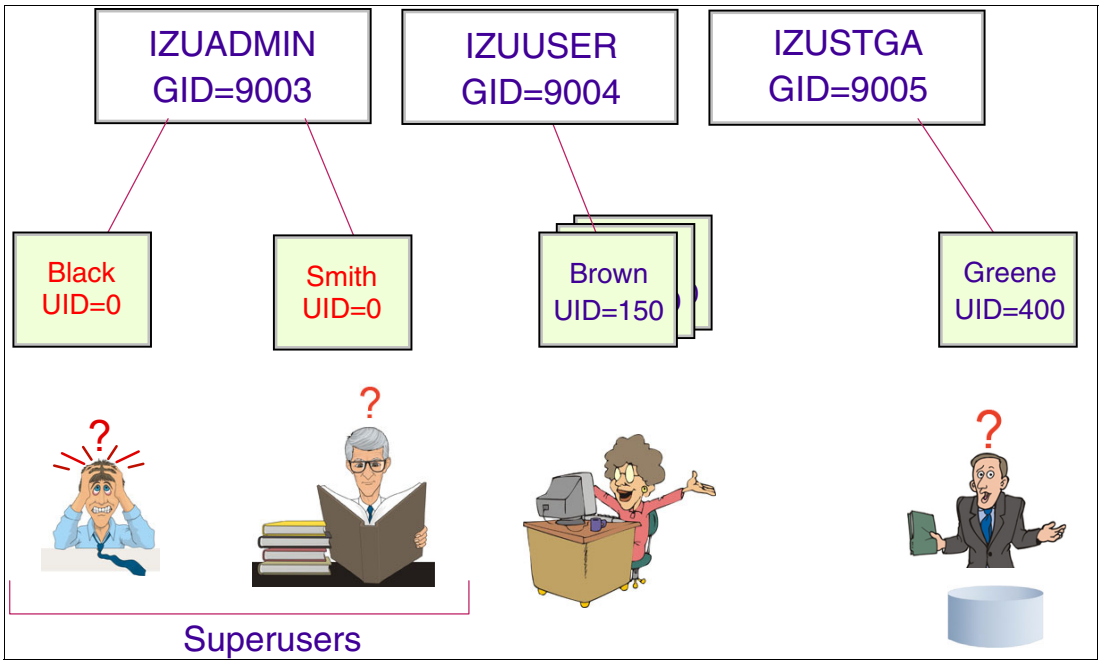


Figure 18-4 Administrators for z/OSMF with SAF Authorization Mode

SAF groups and connection of z/OSMF users are being defined to these new groups, as shown in Table 18-2 on page 360. Groups are permitted to z/OSMF resource profiles (in ZMFAPLA) to facilitate the z/OSMF role support.

Note: When introducing the z/OSMF product to your environment, use the concept of “roles” to group similar users for managing user access to tasks. z/OSMF supports your installation's security requirements for specific user permissions for each of the tasks.

Depending on the authorization mode that you select for the product (more on this later), role definitions can be managed entirely through your security product (SAF authorization mode) or by the z/OSMF administrator through the z/OSMF interface (repository authorization mode).

Table 18-2 z/OSMF support - new classes

ZMFAPLA	GZMFAPLA
POSIT=592 ID=1 MAXLNTH=246 RACLIST=ALLOWED RACLREQ=NO GENLIST=DISALLOWED DFTRETC=4 DFTUACC=NONE GROUP=GZMFAPLA SLBLREQ=NO OPER=NO PROFDEF=YES FIRST=ANY OTHER=ANY CASE=ASIS	RACLIST=DISALLOWED MEMBER=ZMFAPLA

Note: The ZMFAPLA class requires the RACLIST option to ensure optimal performance through the caching of profiles. If you make changes to the profiles, you must refresh the ZMFAPLA class to have the changes take effect.

Converting to SAF Authorization Mode

If your installation is currently administering user authorizations through the z/OSMF administration tasks, you can convert your existing security setup to SAF Authorization Mode. Doing so will require you to repeat the steps of the z/OSMF configuration process, supplying your current configuration file as input. The z/OSMF configuration process generates new REXX execs, which your security administrator can use to set up security for z/OSMF and authorize additional users to the product.

Attention: To switch your current configuration to SAF Authorization Mode, follow the steps in “Converting to SAF Authorization Mode” in *z/OS IBM z/OS Management Facility Configuration Guide*, SA38-0652.

18.3.2 z/OSMF administrator user ID and group ID

The values for properties IZU_ADMIN_PROC and IZU_ADMIN_ACCOUNT are used in creating the z/OSMF administrator ID and are provided by the installation. The other properties are used in the resultant RACF exec that is generated at the end of the -config step and they are the new variables for z/OS V1R13 as follows:

IZU_ADMIN_PROC This specifies the name of the z/OSMF administrator's default logon procedure when logging on through the TSO/E logon panel.

IZU_ADMIN_ACCOUNT This specifies the z/OSMF administrator's default TSO account number when logging on through the TSO/E logon panel.

Note: The values map to the TSO login information for "PROC" and "ACCTNUM". These properties are described in *z/OS Security Server RACF Command Language Reference*, SA22-7687, under the **ADDUSER** command syntax.

z/OS V1R13 provides the following new variables names and their defaults in the izuconfig1.cfg file. Gathering this information can require the assistance of your installation's security administrator.

IZU_DATA_DIR Mount point for the z/OSMF data file system. The data file system is identified by the IZU_DATA_FS_NAME variable.

Default: /var/zosmf/data

IZU_ADMIN_NAME z/OSMF administrator user ID.

Default: User ID for the z/OSMF administrator is ZOSMFAD.

IZU_DATA_FS_SIZE Initial space allocation, in cylinders, for the z/OSMF data file system data set. The script uses 90% of this value for the primary allocation and 10% for the secondary allocation.

Default: 200 cylinders

IZU_ADMIN_NAME z/OSMF administrator user ID. Default user ID for the z/OSMF administrator.

Default: zosmfad

IZU_ADMIN_PROGRAM z/OSMF administrator shell program path. This program path for the z/OSMF administrator identity for z/OS UNIX system services. This value is used for the OMVS segment.

Default: /bin/sh

IZU_ADMIN_REGION z/OSMF administrator region size is for z/OSMF administrator identity. Decreasing this value from the default can result in configuration errors.

Default: 2096128

These new variables are used when you select SAF Authorization Mode with z/OS V1R13:

IZU_ADMIN_GROUP_NAME The default group used when creating the z/OSMF administrator user ID. If your installation has selected SAF Authorization Mode for this configuration, this group name is used in association with the z/OSMF Administrator.

Default: IZUADMIN

IZU_USERS_GROUP_NAME Primary group for z/OSMF users. This group is used in both Repository Authorization Mode and SAF Authorization mode. The use varies depending on the mode; for more information, see “Planning security for z/OSMF” in *z/OS IBM z/OS Management Facility Configuration Guide*, SA38-0652.

18.4 z/OSMF system management tasks with z/OS V1R12

With z/OSMF V1R12, the following system management tasks are available for use:

Configuration Assistant This task provides functions under the Configuration categoryas:

- IKE Version 2.
- New cryptographic algorithms for IPSec and IKE.
- FIPS 140 cryptographic mode for IPSec.
- Certificate trust chains and certificate revocation lists.
- Enforcement of RFC4301 compliance for IPSec filter rules.

Incident Log The Incident Log task under the Problem Determination category provides functions such as:

- Encryption of the incident files, including dumps, and transmission of these files to IBM in parallel through FTP to save time. To do so, the host and destination must have the z/OS Problem Documentation Upload Utility installed.
- Sending additional documentation with an incident to an FTP destination.
- Providing free-form notes or comments for each incident.

The Incident Log task supports the creation of diagnostic log snapshots based on the SYSLOG and LOGREC data sets, and the OPERLOG and LOGREC sysplex log streams. If your installation has determined that a single-system scope for message and error log data collection is sufficient, you do not need to set up OPERLOG and the logrec log stream to use the Incident Log task. Instead, z/OSMF uses your system's SYSLOG or logrec data set, or both, as the source for creating diagnostic log snapshots.

Links task The Links task also includes a text entry window that requires the z/OSMF administrator to further qualify the link resource name with a suffix, which can be used if a discrete profile is required for the link. The z/OSMF administrator provides the fully qualified resource name to the security administrator to use to create the user authorizations for the link.

Monitoring Desktops This task was made available in z/OS V1R12.

The task monitors the performance of the z/OS sysplex and Linux images (System z and Intel in your installation). Specifically, you can monitor most of the metrics supported by the RMF Monitor III, create and save custom views of the metrics, and display real-time performance data as bar charts.

Sysplex Status This task was made available in z/OS V1R12.

This task allows you to assess the performance of the workloads running on the z/OS sysplexes in your environment. The Sysplex Status task also provides a single location where you can define the z/OS sysplexes and Linux images to be monitored in the Monitoring Desktops task.

Workload Management This task was made available in z/OS V1R12.

The Workload Management task with z/OS V1R12 provides a shared location (Settings tab) where you can specify how long to keep the service definition history, and also define the code page and backup sequential data set for the sysplex.You can also enable consistency checking between z/OSMF and the WLM couple data set, and indicate whether you want the Workload Management task to display or suppress information messages.

18.4.1 z/OSMF new system management tasks with z/OS V1R13

Management of z/OS is made easier with z/OS V1R13 for new and experienced system programmers with the following new system management tasks:

Capacity Provisioning	This task is included under the Performance category. Use this task to view the status of the Capacity Provisioning Manager, and to view information about the active domain configuration and provisioning policy.
DASD Management	This task is included under the Storage category. Use this task to perform storage management tasks, including adding storage to a System Managed Storage (SMS) pool storage group, defining reserved storage pools, adding volumes to a reserved storage pool, viewing and managing volumes, and validating and activating source control data sets.
Deployment	This task is included under the Software category. Use this task to deploy any SMP/E packaged and installed software and to identify software requisites and possible regressions.
ISPF	This task is included under the z/OS Classic Interfaces category. Use this task to access the ISPF applications on your host system through z/OSMF.

z/OSMF V1R13 will make the day-to-day operations and administration of the mainframe z/OS systems easier to manage for both new and experienced system programmers.

The focus is to help improve system programmer productivity, and make the functions easier to understand and use.

18.5 Installation considerations for z/OSMF V1R13

With z/OS V1R13, the following z/OS system components might require customization to get z/OSMF running:

- CIM
- CEA
- SYSREXX
- RMF
- Capacity Provisioning

Important: See Appendix A, “Setting up WebSphere OEM, z/OSMF, CIM, and Capacity Provisioning” on page 765, for details about the setup process for those components.

18.5.1 Customization requirements

This section describes the customization process for z/OS system prerequisites that are used to get z/OSMF running for certain components. The CIM server is needed for the Incident Log and WLM only, so it is not always required. CEA and SYSREXX are needed for the Incident Log only.

CIM server

In our environment, CIM customization was done following the instructions in *z/OS Common Information Model User's Guide,* SC33-7998. The default directory locations, along with default RACF user ID (CFZSRV for the server) and groups (CFZADMGP and CFZUSRGP)

were used, meaning that supplied jobs CFZSEC and CFZRCUST can be run with minor modification.

If you choose to customize the CIM server using this method, reply `N` (the default) to the following prompt when running the z/OSMF configuration script (`izusetup.sh`):

```
"Do you need assistance in setting up security for the Common Information Model
(CIM) server?"
```

The CFZSEC and CFZRCUST jobs might have already been customized and run as part of base z/OS installation.

Note: Although z/OSMF documentation refers to using the “CIM server quick setup and verification” process, we found that further customization was required in our environment; specifically, the must-stay-clean feature, which was enabled in our environment by the existence of the RACF FACILITY class BPX.DAEMON profile. RACF program control profiles were required for z/OSMF non-core functions, as described in *z/OS Common Information Model User’s Guide*, SC33-7998.

CEA address space

Customization of the CEA address space might be required in your environment. The CEA started task uses the IBM-supplied CEAPRM00 parmlib member. However, you will need to provide additional information for the CEA parameters when running the z/OSMF configuration script (`izusetup.sh`). This is to provide client-specific information required for the Incident Log function of z/OSMF.

SYSREXX

The SYSREXX component provides an infrastructure through which programs written in the REXX language can be run outside the normal TSO/E or batch environments, using a programming interface. The default IBM-supplied parmlib member is found in SYS1.SAMPLIB(AXR00). It can be used without having additional REXXLIB data sets specified.

See *MVS Authorized Assembler Services Guide,* SA22-7608, for guidance if the System REXX address space, AXR, is not started.

RMF for Linux

For Linux images or guests, z/OSMF uses input from the RMF Linux data gatherer. Each Linux image needs to have an active data gatherer. The RMF Linux data gatherer is an optional tool that you can download and install in your installation. The tool is provided as-is at no additional charge. It is not supported by IBM, and it is not shipped with z/OSMF. For more details about the tool or to download a copy, see the RMF PM with Support for Linux Enterprise Server web page.

RMF for z/OS

RMF is used for the performance monitoring feature of z/OSMF. For z/OS sysplexes, z/OSMF uses input from a single data server on one system in the sysplex. This data server collects data from the RMF Monitor III data gatherer on each image in the sysplex using the Distributed Data Server (DDS) function. For complete information about setting up the Distributed Data Server (DDS), see the following documentation:

- *z/OS Resource Management Facility User's Guide*, SC33-7990
- *z/OS IBM z/OS Management Facility Configuration Guide*, SA38-0652

Following is a summary of the setup tasks:

- Ensure RMF, RMF Monitor III, and DDS are set available. The started task name for DDS is GPMSERVE.
- Configure the resource monitoring plug-in through the IZU_RMF_ CONFIGURE parameter during z/OSMF configuration.
- Configure your DDS security environment.
- Set up the started task definitions for the Security Server (RACF):

```
RDEFINE STARTED RMF.* STDATA(USER(RMF) TRUSTED(YES))
RDEFINE STARTED RMFGAT.* STDATA(USER(RMFGAT) TRUSTED(YES))
RDEFINE STARTED GPMSERVE.* STDATA(USER(GPMSERVE) TRUSTED(YES))
SETROPTS RACLIST(STARTED) REFRESH
```

- Allow access to BPX.WLMSERVER profile:

```
PERMIT  BPX.WLMSERVER CLASS(FACILITY) ID(GPMSERVE) ACCESS(READ)
```

- Define GPMSERVE to program control:

```
PERMIT  BPX.DAEMON CLASS(FACILITY) ID(GPMSERVE) ACCESS(READ)
RDEFINE PROGRAM GPM* ADDMEM('SYS1.SERBLINK'//NOPADCHK) UACC(READ)
RDEFINE PROGRAM ERB* ADDMEM('SYS1.SERBLINK'//NOPADCHK) UACC(READ)
RDEFINE PROGRAM CEEBINIT ADDMEM('CEE.SCEERUN'//NOPADCHK) UACC(READ)
RDEFINE PROGRAM IEEMB878 ADDMEM('SYS1.LINKLIB'//NOPADCHK) UACC(READ)
SETROPTS WHEN(PROGRAM) REFRESH
```

- Ensure RACF access to the Distributed Data Server through the RACF APPL class and the GPMSERVE profile in this class. A user must have access to this.

```
RDEFINE APPL GPMSERVE UACC(READ)
```

- Configure PassTicket support for the Distributed Data Server (optional).
- If DDS is configured to require authentication, then a user ID and a PassTicket can be supplied instead of a user ID and a password.
- A PassTicket is validated against an application name. The RACF application name of the DDS is GPMSERVE. Define a DDS application profile with an associated encryption key, where <key> is a user-supplied 16-digit value used to generate the PassTicket:

```
RDEFINE PTKTDATA GPMSERVE SSIGNON(KEYMASKED(<key>)
SETROPTS RACLIST(PTKTDATA) REFRESH
```

  You can specify a value of your choice. Valid characters are 0 - 9 and A - F.

- Define a profile in the PTKTDATA class controlling access to the PassTicket services and explicitly set the universal access authority to NONE.

```
RDEFINE PTKTDATA IRRPTAUTH.GPMSERVE.* UACC(NONE)
```

- The user ID connecting to the DDS needs update permission to the newly created profile:

```
PERMIT IRRPTAUTH.GPMSERVE.* CLASS(PTKTDATA) ID(CFZSRV,WSSRU1) ACCESS(UPDATE)

For z/OSMF access to DDS IDs CFZSRV and WSSRU1 require access.
```

- Activate the changes with the following command:

```
SETROPTS RACLIST(PTKTDATA) REFRESH
```

18.5.2 Active log medium

If your installation collects messages about programs and system functions (the hardcopy message set) on a sysplex-wide basis, z/OSMF uses the operations log (OPERLOG) as the source for its message data. Otherwise, z/OSMF uses the system log (SYSLOG) as the source for message data. To display the active medium where messages are recorded, enter the command shown in Figure 18-6.

```
D C,HC
SC74      CNZ4100I 08.11.18 CONSOLE DISPLAY 354
SC74      CONSOLES MATCHING COMMAND: D C,HC
SC74      MSG:CURR=0    LIM=1500 RPLY:CURR=3    LIM=999   SYS=SC74     PFK=00
SC74      HARDCOPY  LOG=(SYSLOG,OPERLOG)  CMDLEVEL=CMDS
SC74            ROUT=(ALL)
SC74      LOG BUFFERS IN USE: 0       LOG BUFFER LIMIT: 6000
```

Figure 18-5 Display active log medium

Using the SYSLOG to create diagnostic log snapshots requires one of the following minimum levels of JES on your z/OS system:

- JES2 V1R11, or
- JES3 V1R11 (with the PTF for APAR OA29534)

18.6 Migrating to z/OSMF V1R13

Migrating to a new release of z/OSMF from an older release is a two-step process. Start by migrating your existing configuration file and override file to the latest format. Then, configure the product as you do normally, supplying the updated configuration and override files as input to the z/OSMF configuration process.

Depending on your current release of z/OSMF, you might also need to perform additional migration actions. For the considerations specific to your particular migration path, see one of the following sections in *z/OS IBM z/OS Management Facility Configuration Guide*, SA38-0652, as appropriate:

- Migrating from z/OSMF V1R12 to z/OSMF V1R13
- Migrating from z/OSMF V1R11 to z/OSMF V1R13

After you have completed these actions, you are ready to configure the new release of z/OSMF on your system.

18.7 Installing and customizing z/OSMF

Whether you are migrating from a previous release of z/OSMF, or installing it for the first time, be aware that z/OSMF requires careful planning to ensure a smooth installation and configuration process. Note the following points:

- z/OSMF, the product, requires the SMP/E installation of several FMIDs, according to *Program Directory for z/OS Management Facility*, GI11-2886.
- IBM WebSphere Application Server OEM Edition for z/OS must be configured, if not already done. This process requires certain z/OS resources to be set up, shell scripts to

be run, and jobs to be run to set up security for RACF or an equivalent security management product.

- z/OSMF, the application, requires certain z/OS resources to be set up, shell scripts to be run, and security set up performed for RACF (or the equivalent). Using z/OSMF requires sufficient authority in z/OS. Specifically, on the z/OS system to be managed, the resources to be accessed on behalf of z/OSMF users (data sets, operator commands, and so on) are secured through the security management product at your installation; for example, Resource Access Control Facility (RACF). Your installation's security administrator must create these authorizations.

Note: If you are migrating to a new release of z/OSMF, you can reuse much of the customization from your current configuration.

18.7.1 z/OSMF plug-ins

In z/OSMF, a “plug-in” is a collection of one or more system management tasks. Most functions in the z/OSMF navigation area are provided through plug-ins, which you enable when you configure the product.

In general, configure all of the available plug-ins during the z/OSMF configuration process. Table 18-3 on page 367 lists which plug-ins are available for configuration in z/OSMF V1R13. The plug-in names that are highlighted in bold are new with V1R13.

Important:

- Review all of the steps for installing plug-ins before performing the configuration.
- If your installation is upgrading from a previous release of z/OSMF, you must migrate your existing configuration file and, if applicable, override file to the latest formats before configuring the new release of z/OSMF. Your upgraded files are used as input to the z/OSMF configuration process.

Table 18-3 Plug-ins and associated tasks in z/OSMF V1R13

Plug-in name	Tasks provided by the plug-in	Associated category in the z/OSMF navigation area
Capacity Provisioning	Capacity Provisioning	Performance See“Capacity Provisioning” on page 385.
Configuration Assistant	Configuration Assistant	Configuration See “Configuration Assistant for z/OS Communications Server” on page 408.
DASD Management	**DASD Management**	**Storage** See “DASD Management task” on page 382.
Incident Log	Incident Log	Problem Determination “Incident Log enhancements” on page 401.
ISPF	**ISPF**	**z/OS Classic Interfaces** See “ISPF task with z/OSMF” on page 376.

Plug-in name	Tasks provided by the plug-in	Associated category in the z/OSMF navigation area
Resource Monitoring	► Resource Monitoring ► System Status	Performance See “Resource Monitoring task” on page 407.
Software Deployment	**Deployment**	**Software** See “Software category and Deployment task” on page 368.
Workload Management	Workload Management	Performance See “Workload Management and z/OSMF” on page 397.

Note: Configuring a plug-in is optional. It is generally advised to configure the set of plug-ins that you are interested in during the initial z/OSMF configuration. You can always add more plug-ins later.

To add a plug-in after the configuration process is completed, you will repeat most of the z/OSMF configuration steps.

Installing z/OSMF plug-ins

In terms of security setup, the amount of work needed will depend in part on the authorization mode that will be used for your configuration. In SAF Authorization Mode, which is the default for new z/OSMF V1R13 instances, enabling a task will require that you customize the security management product on your host system (for example, make updates to the RACF database). The configuration process provides a REXX exec with RACF commands to help with performing these changes. In Repository Authorization Mode, security product customization is less likely because authorizations are maintained through z/OSMF.

Important: Various z/OSMF tasks require the Common Information Model (CIM) server to be running on the host z/OS system. Using these tasks will require that you ensure that the CIM server is configured on your system, including security authorizations and file system customization:

- Capacity Provisioning
- Incident Log
- Workload Management

See *z/OS IBM z/OS Management Facility Configuration Guide*, SA38-0652, for the installation details of all plug-ins.

18.8 Software category and Deployment task

Software deployment is the process of making software available to be used on a system by users and other programs. A *deployment* is the software deployment workflow, and it is the object in which z/OSMF stores your input and any output generated during the workflow. You can use a deployment to deploy one software instance onto one system at a time.

The Deployment task provides a web-based, user-interface that contains a checklist, wizards, and property sheets designed to guide you through the software deployment process and help reduce common deployment errors. Select this category to view tasks that can help you manage your z/OS software. This category contains the Deployment task, which lets you

deploy any software that is installed using SMP/E, organize your software instances and deployments, and follow IBM recommendations for software deployment.

Software Deployment will make deployment of installed software simpler and safer. It replaces manual and error-prone procedures with a user-friendly application, and codifies IBM advice for software deployment. To assist you with performing these tasks, z/OSMF offers a software deployment solution, the Deployment task.

After you finish installing software, you might need to deploy it to perform one or more of the following tasks:

- Create a backup copy
- Move the software to another system
- To create another SMP/E-serviceable copy for installing service or other products

Requirement: In z/OSMF V1R13, you require SMP/E V3R6 (new with z/OS V1R13), FMID HMP1J00 to install the Deployment manager plug-in.

Software deployment key functions

Software deployment key functions consist of the following tasks:

- Verify that cross-system and cross-product software requisites are satisfied.
- Verify that software fixes are not regressed.
- Clone all parts of the software.
- Clone the inventory (SMP/E CSI) along with the software.

A z/OSMF software deployment scope is to manage the following areas:

- All SMP/E-installed software
- IBM and ISV software
- z/OS operating system and related products
- Subsystems and related products
- Individual products
- Service upgrades for all of these areas (through complete replacement)

Software deployment will clone software, as follows:

- Locally, either on a single system or system-to-system within a sysplex
- Remotely, system-to-system across a network and multiple sysplexes

Software deployment uses a checklist approach to guide you through all the steps of a deployment:

1. Select the software to deploy (a software instance).
2. Report missing requisites and possible regressions.
3. Select the deployment objective.
4. Configure the target software instance.
5. Validate the configuration against the target system, summarize the deployment actions, and generate the deployment jobs.
6. Execute the deployment jobs.

18.8.1 Software instance

A software instance, shown in Figure 18-6, is a collection of one or more SMP/E target and distribution zone pairs, the related libraries, and any additional data sets associated with a product set. To define the software in your enterprise to z/OSMF, create a software instance that describes the software libraries and data sets associated with the software.

A software instance may contain non-SMP/E managed data sets such as sequential, PDS(E), VSAM, HFS, or zFS data sets.

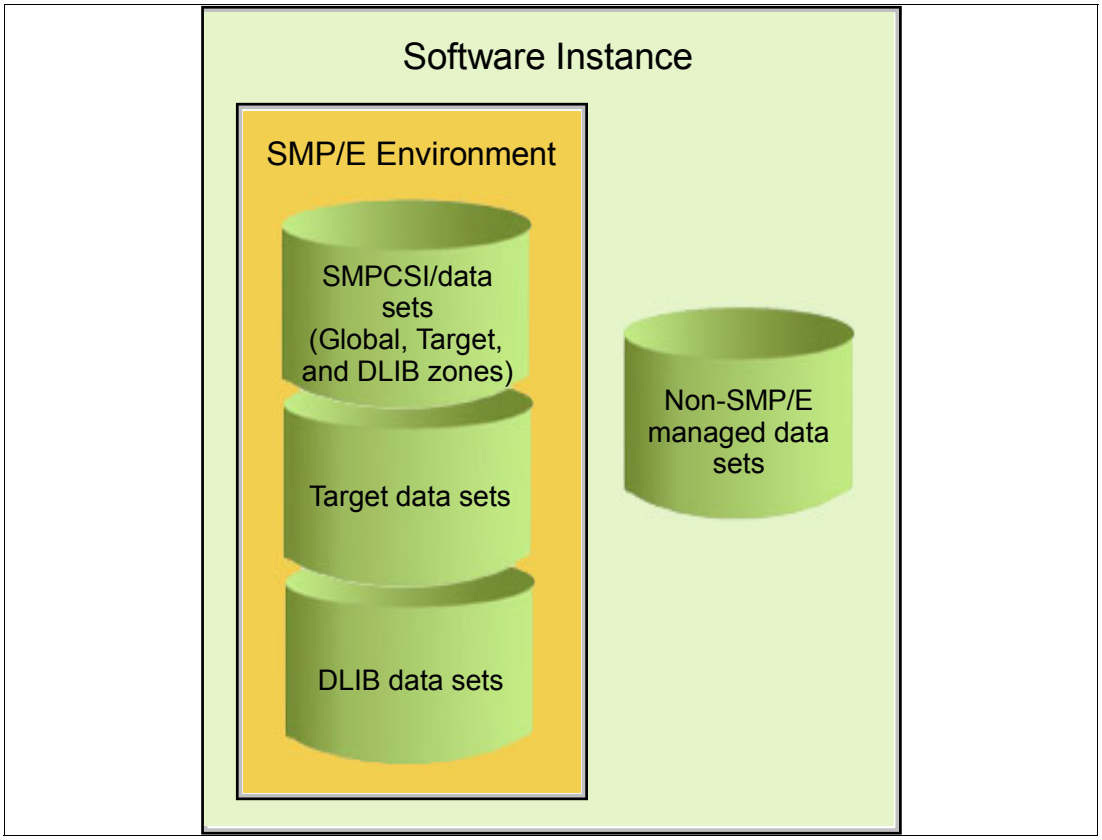


Figure 18-6 Software instance

During the deployment workflow, you will work with at least two software instances:

- The source instance
- The target instance

The source software instance is the input for the deployment workflow. It is the software instance to be deployed.

The target software instance is the output of the deployment workflow. It is the resulting software instance.

When a deployment is complete, you will have two copies of the software instance, as shown in Figure 18-7:

- The source copy
- The target copy

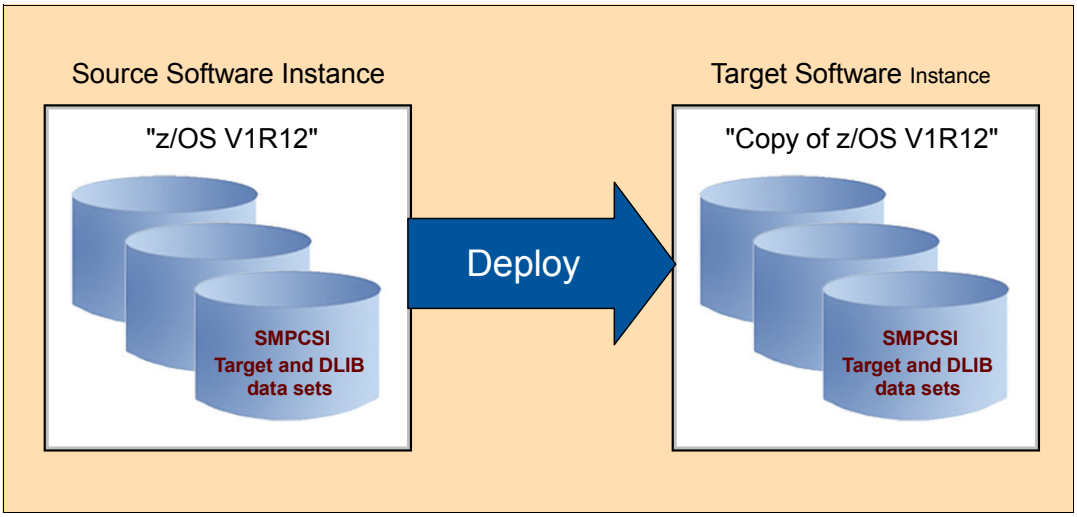


Figure 18-7 Source and target instances

18.8.2 Deployment task

After you finish installing software, you might need to deploy the software to create a backup copy, to move the software to another system, or to create another SMP/E-serviceable copy for installing service or other products. To assist you with performing these tasks, z/OSMF offers a software deployment solution, the Deployment task. Selecting the **Deployment** task from the Software category displays Figure 18-8.

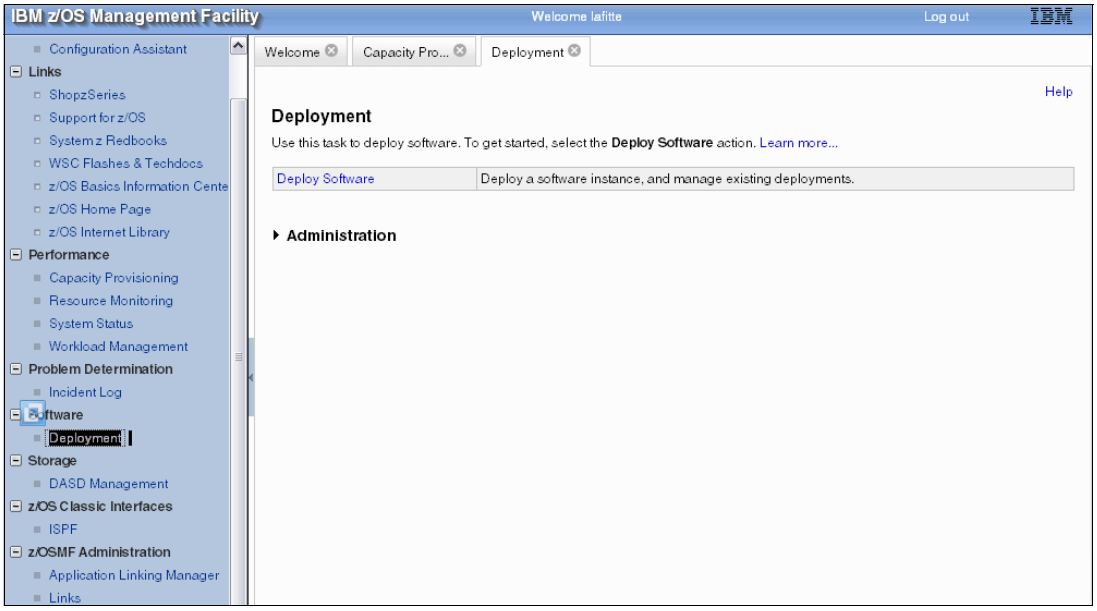


Figure 18-8 Deployment task

When you select the **Administration** button pointer, Figure 18-9 is displayed where you can select each of the administrative functions shown.

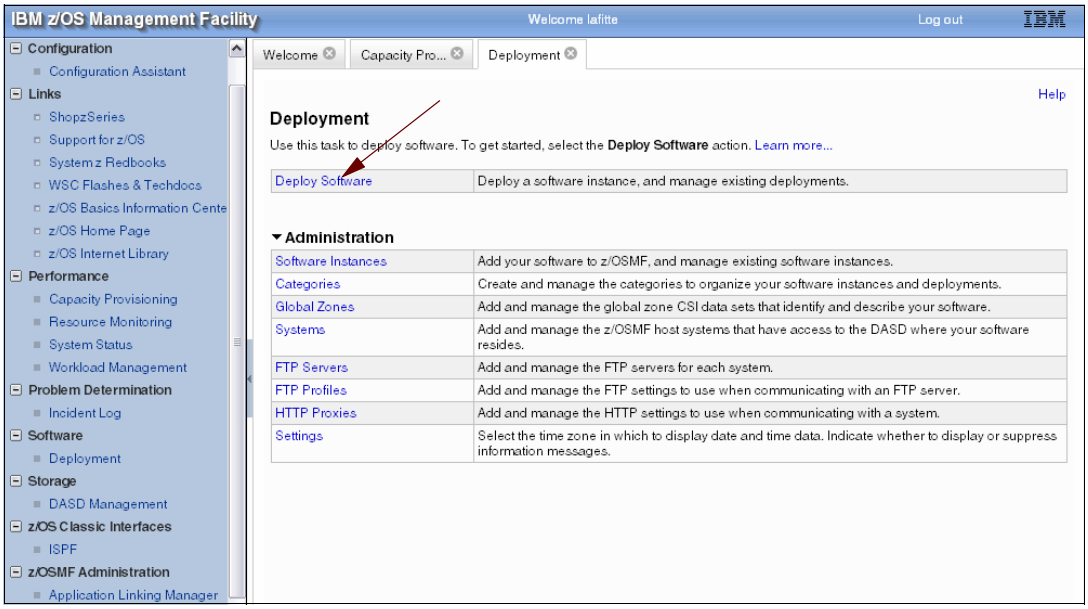


Figure 18-9 Deployment administration

18.8.3 Deployment software

In the window shown in Figure 18-9, select **Deploy Software** to define a software instance. Then the window shown in Figure 18-10 is displayed.

To deploy a software instance, you must define a new deployment. To do so, select the **New** action provided in the Deployments table, as shown in Figure 18-10.

Software deployment is the process of making software available to be used on a system by users and other programs. A deployment is the software deployment workflow, and it is the object in which z/OSMF stores your input and any output generated during the workflow. You can use a deployment to deploy one software instance onto one system at a time.

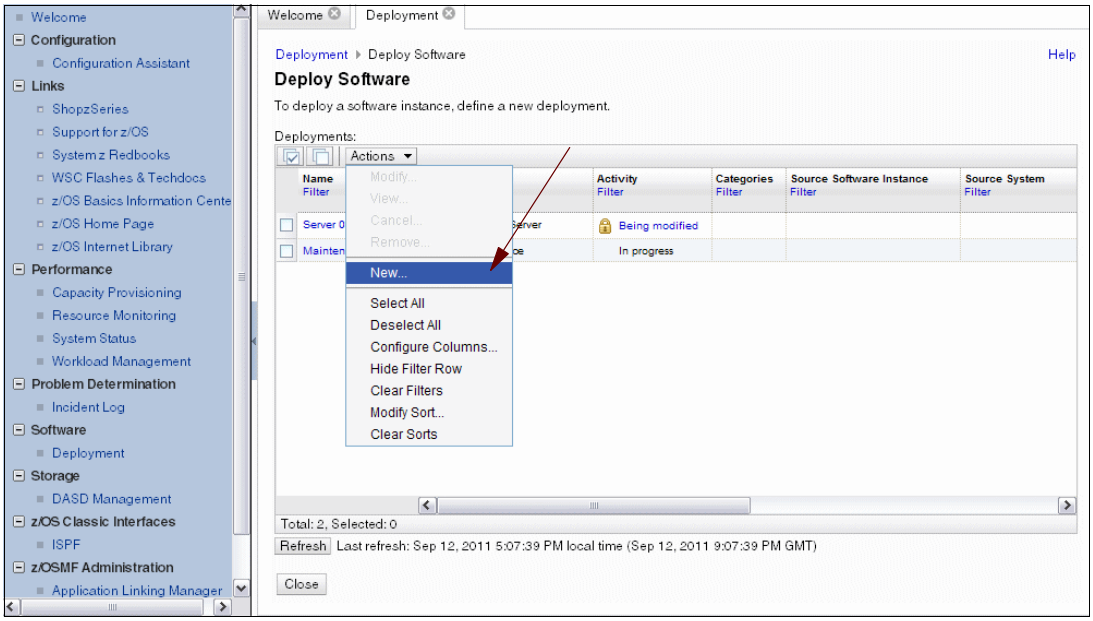


Figure 18-10 Deploy Software to define a new deployment

Note: Verify that the software you want to deploy is SMP/E-packaged and installed. Installation of software or service upgrades is outside the scope of the Deployment task. You can use SMP/E, CustomPac dialogs, or a similar product to assist with the installation process.

Ensure that any associated z/OS UNIX file system data sets are mounted at the mount points specified by the target zone's DDDEF entries. The z/OS UNIX file system data sets mounted at the specified mount points are included in the deployment.

After you select New in Figure 18-10, the Deployment Checklist window shown in Figure 18-11 is displayed.

18.8.4 Deployment Checklist page

You can use the Deployment Checklist page, shown in Figure 18-11, in the Deployment task to modify existing deployments or to define new deployments.

The Checklist table lists the steps included in the deployment workflow. For more details about the workflow, see the Help topic “Defining new deployments.” For more information about deployments, see the Help topic “Deploying software.”

Ensure that any associated z/OS UNIX file system data sets are mounted at the mount points specified by the target zone's DDDEF entries. The z/OS UNIX file system data sets mounted at the specified mount points are included in the deployment.

During the deployment workflow, your input is saved after you complete each step in the checklist. Therefore, you can exit the workflow after completing any step. To exit the workflow, click **Close** on the Deployment Checklist page.

Select deployment objectives

The deployment objective is shown as Step `3. Select the objective for this deployment` in the Deployment Checklist in Figure 18-11.

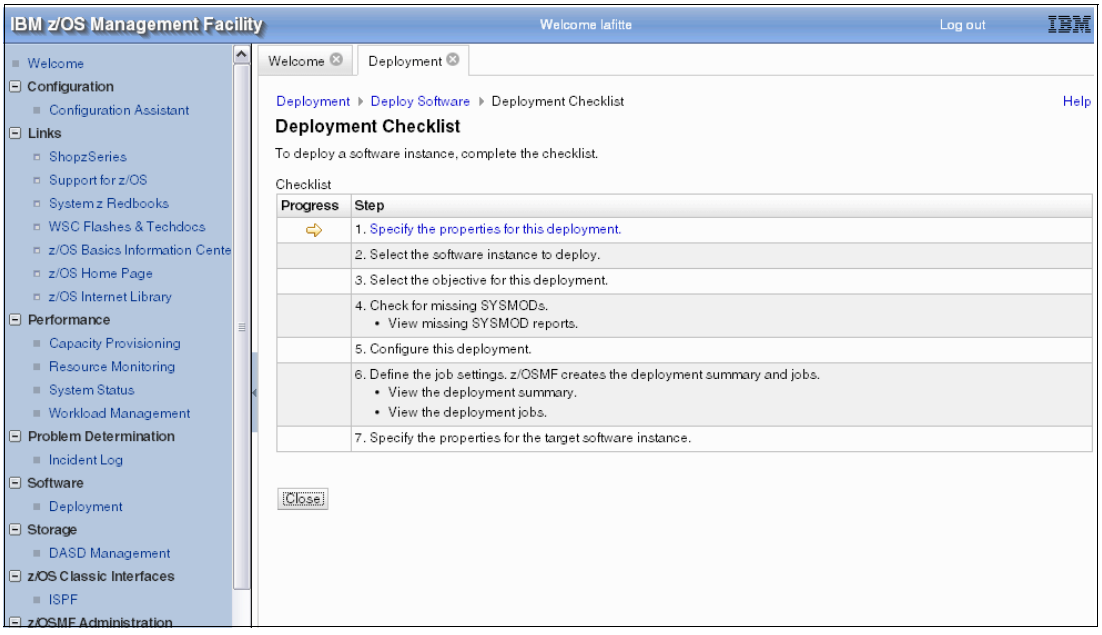


Figure 18-11 Deployment Checklist page

The objective is to create a copy of the source software instance; see Figure 18-12. The resulting copy is referred to as the “target software instance.” Use this page to indicate whether you want the target software instance to be a new software instance on the target system, or to replace an existing software instance. Also, select the global zone to use for the target software instance.

If you select the option `Create a new software instance and connect it to the following global zone CSI.`, then select the global zone CSI to use for the target software instance.

If you select the option `A new global zone CSI`, the deployment will create a new global zone CSI for the target software instance. If you select this option, then in the `Target system:` field, select the system where the target instance will reside. This field lists the systems defined to z/OSMF. You can select a system from the list, or click Select and select a system from the table. The target system is required.

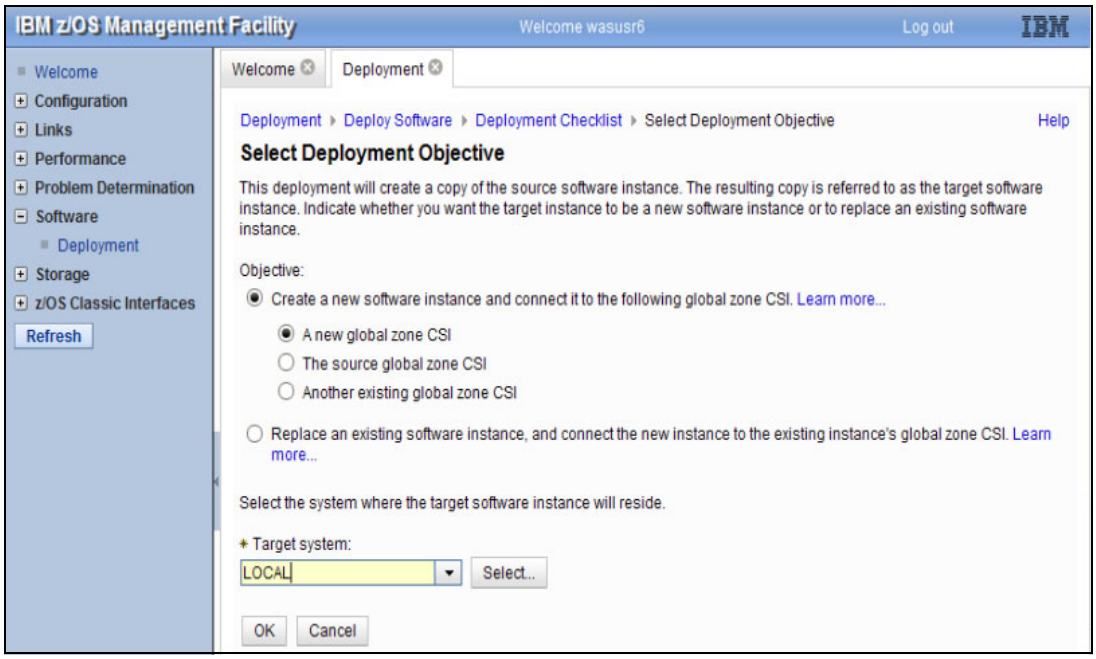


Figure 18-12 Select Deployment Objective

18.8.5 z/OSMF deployment topology

Figure 18-13 shows the deployment topology. The Deployment task supports the deployment of software to DASD volumes shared within the same sysplex (local deployment) or to DASD volumes accessible to another sysplex (remote deployment).

Important: Adhere to IBM recommendations for software deployment because many of those recommendations are integrated into the deployment workflow. For example:

- The Deployment task uses SMP/E DDDEF entries to automatically locate data sets, such as SMP/E data sets, target libraries, and distribution libraries.
- The task deploys all of the software included in a target zone, and optionally, in the related distribution zone.
- The task also copies the SMP/E consolidated software inventory (CSI) with the software. If you currently do not copy your SMP/E CSIs, you will see a slight increase in DASD usage per target software instance.

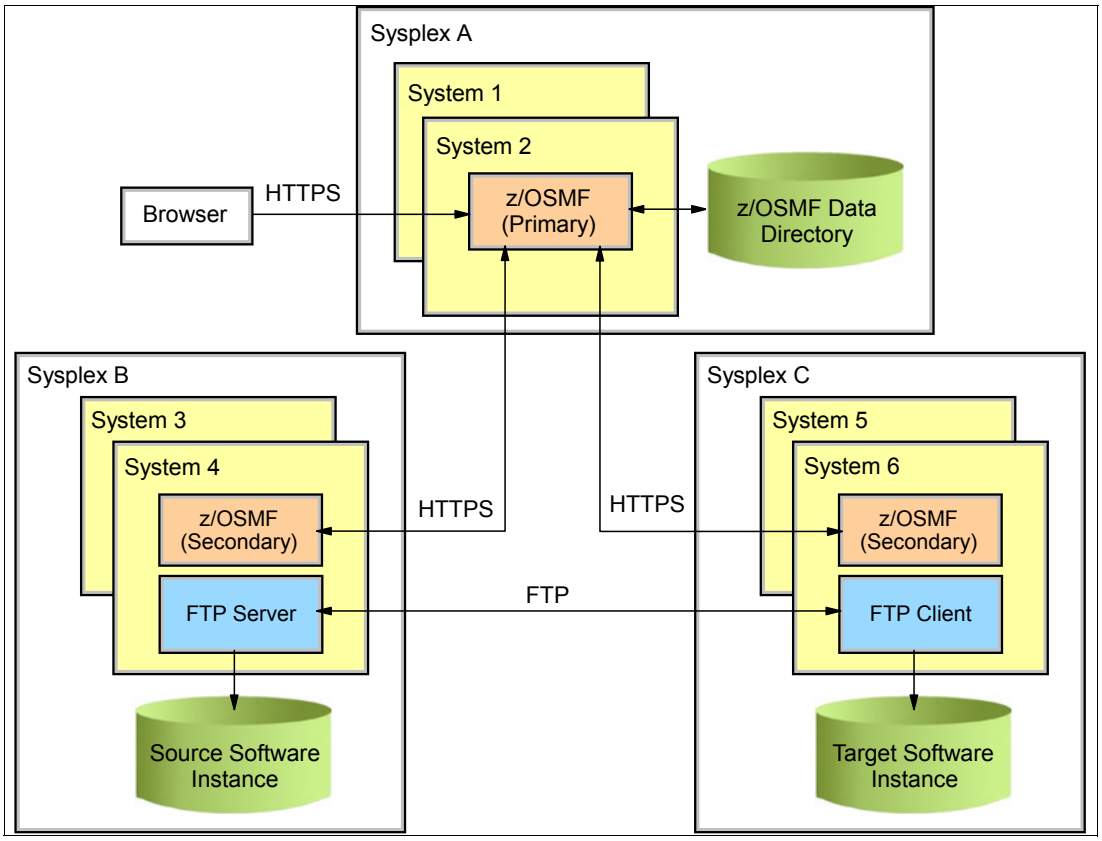


Figure 18-13 z/OSMF deployment topology

Configure the target software instance

To configure the target software instance, specify the following items:

- SMP/E zone names
- Data set names and locations: volume or storage class
- Catalog state for data sets, by HLQ: use default catalog, new catalog alias, or do not catalog.
- Target volume properties: initialize the volume, symbol for indirectly cataloged data sets on that volume.
- Mount points for z/OS UNIX file system data sets

z/OSMF validates the target configuration to ensure there are no unintended data set collisions on the target volumes, and in the target system catalog.

18.8.6 Software deployment summary

z/OSMF Software Deployment will provide rigor in the deployment of SMP/E-installed software. It will ensure that all parts of a software instance are copied, and that the CSI is carried forward with the software.

It will also help to ensure that cross-system requisites are satisfied (coexistence and preconditioning); cross-product requisites on the same system are satisfied; and that software fixes are not regressed.

z/OSMF Software Deployment can be used to create a clone for installation or for execution. User-specified information will be persisted and available for reuse. Subsequent deployment operations of the same software instance can require little or no user input.

18.9 ISPF task with z/OSMF

The ISPF task is included under the z/OS classic interfaces category. It can be used to access the ISPF applications on the host system through z/OSMF. The ISPF task allows you to access your host system ISPF applications from z/OSMF. For system administrators, the ISPF task provides a web-based alternative to using traditional, 3270-based ISPF.

Through the ISPF task, you can access any applications that you usually access through z/OS ISPF on the host system, such as System Display and Search Facility (SDSF) and Hardware Configuration Definition (HCD). You can issue TSO commands and have multiple sessions in parallel (split-screen mode), as shown in Figure 18-18. You can customize the ISPF settings as you do with ISPF on the host system, and use dynamic areas in ISPF and attributes such as color highlighting.

Customizing for profile sharing

If you plan to allow the use of multiple ISPF sessions, the user's logon procedure must be configured to allow profile sharing. This option avoids enqueue lockouts and loss of profile updates when the same profile data set is used for concurrent ISPF sessions.

With profile sharing enabled, the user's logon procedure is required to allocate ISPF profile data sets with the disposition SHARED, rather than NEW, OLD, or MOD, and the data sets must already exist. Or, these data sets must be temporary data sets.

18.9.1 Logon to ISPF

Figure 18-14 shows the ISPF logon window. From here you can specify logon parameters for ISPF.

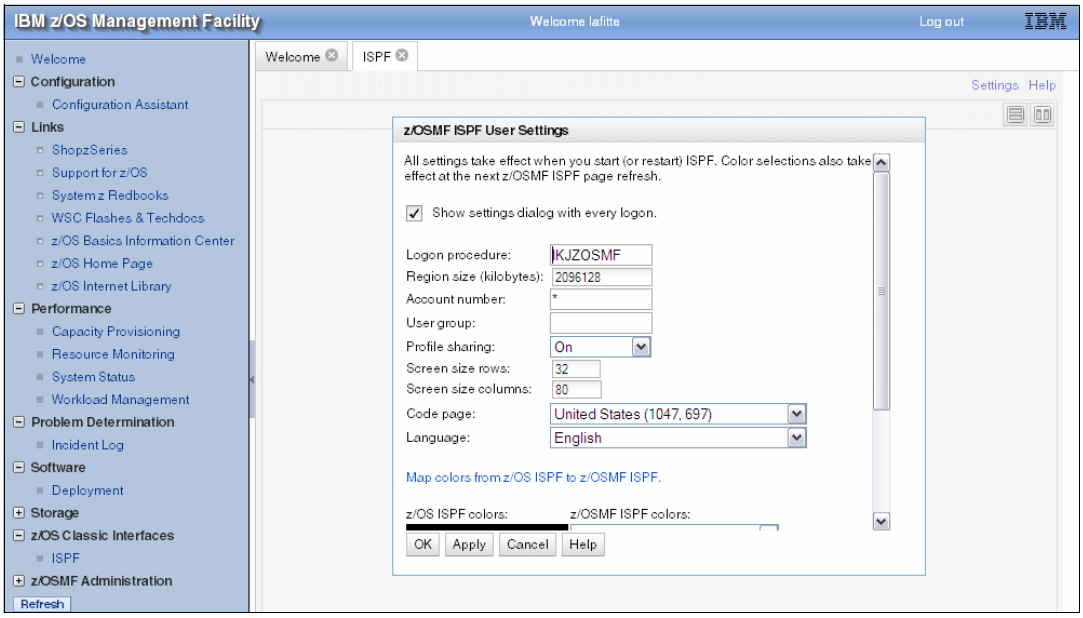


Figure 18-14 z/OS Classic Interfaces: z/OSMF ISPF User Settings

Color settings

You can choose and customize your color settings by adjusting the color map at the bottom of the panel in Figure 18-15. You can change the colors that are displayed in a z/OSMF ISPF panel.

To change a color, click the z/OSMF ISPF color and select the replacement color from the drop-down list. Then click **OK** to save the color mapping, and close the Settings window. If you are in the process of starting z/OSMF ISPF, it starts with the color mapping you have selected. If you are already in ISPF, the color mapping is applied when you change the ISPF window.

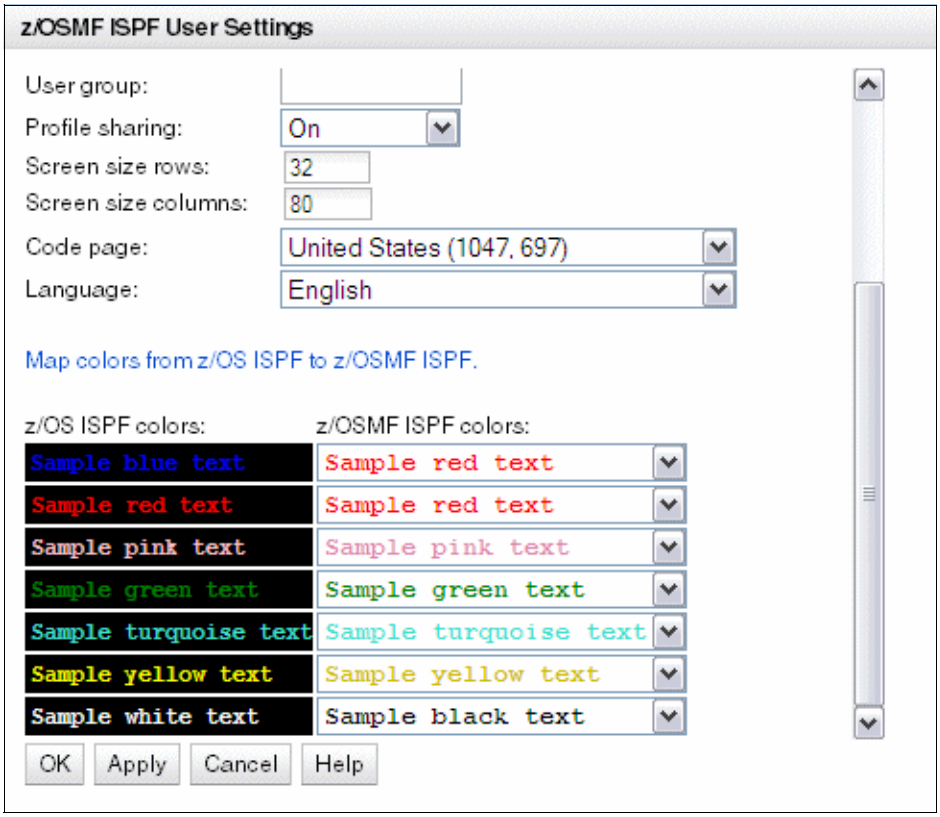


Figure 18-15 Mapping colors (continuation of previous window, Figure 18-14)

When you access ISPF from the z/OSMF panel, messages similar to those shown in Figure 18-16 appear in the syslog. The ISPF task works with ISPF on your host z/OS system. User access to ISPF applications is controlled through the same authorizations that exist for your z/OS system.

```
LOGON
 $HASP100 LAFITTE  ON TSOINRDR
 $HASP373 LAFITTE  STARTED
 +ISPWB000  Client requested ISPF session initialization 449
            Userid: LAFITTE ASIDX: 0058
            Message Queue: 0000000013 CCSID: 01047
```

Figure 18-16 ISPF access in the syslog for the logon

ISPF Primary Option Menu

After the logon completes, click **OK** (shown in Figure 18-15) to complete the logon.

Figure 18-17 is displayed, showing the ISPF web panel for the Primary Option Menu.

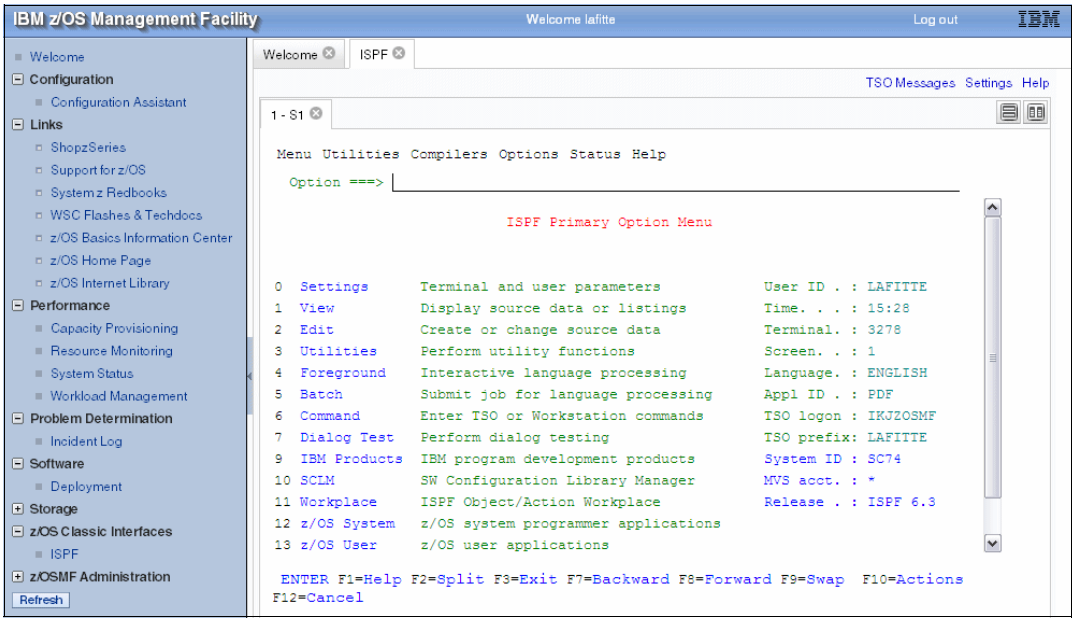


Figure 18-17 ISPF web panel for Primary Option Menu

Multiple ISPF sessions

Figure 18-18 shows multiple ISPF sessions. The z/OSMF ISPF task interface has been enhanced to take advantage of the web browser environment. In particular, this means that you can use the mouse tool more often, for example, to select “point and shoot” menu options and function keys. You can also define long (numerous rows) logical screens, and scroll within these screens.

When you split a 3270 z/OS ISPF application window (by pressing F2), a “split” dotted line appears horizontally across the window. To switch sessions, press Swap (F9).

When you split a z/OSMF ISPF task panel (by pressing or clicking F2), z/OSMF creates a tabbed panel under the ISPF task. To switch panels, press Swap (F9), or click the relevant tab.

In the z/OSMF ISPF task, pressing F2 generates, at most, two sessions. You can split a pane by clicking the split buttons to generate up to four sessions.

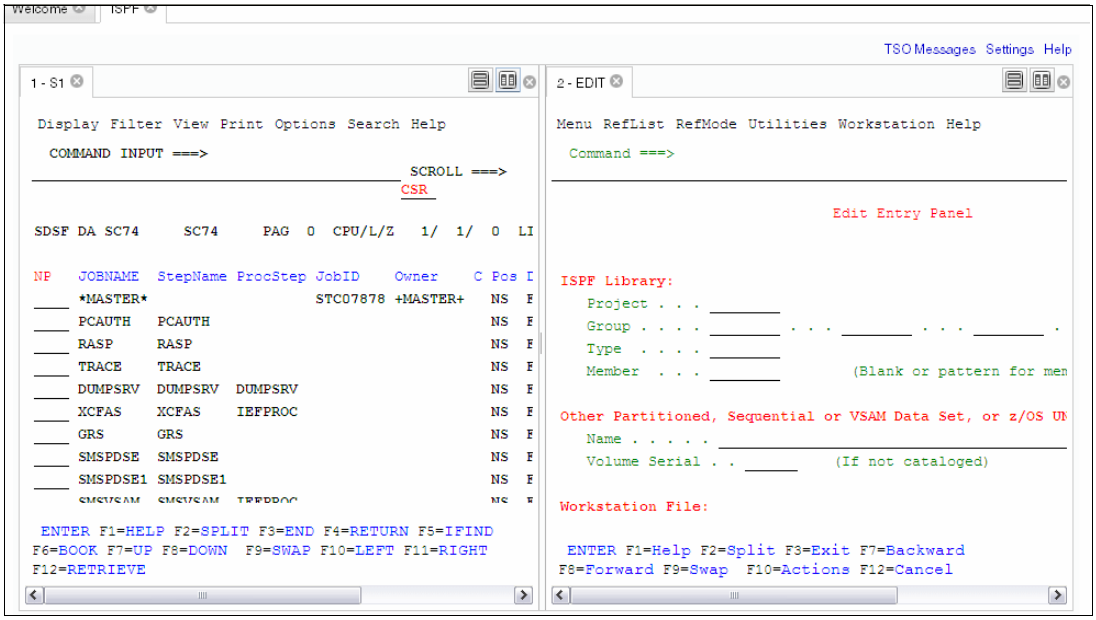


Figure 18-18 Multiple ISPF sessions in parallel

Splitting panels

You can split a panel by clicking the horizontal and vertical split buttons at the top right side of the panels shown in Figure 18-18. Note that this function is built into z/OSMF. It is not available in 3270 z/OS ISPF. When the split buttons are displayed and active, you can continue to split panels. Each click opens another ISPF panel.

You can resize a panel by clicking and dragging the divider between panels. You can move panels around by clicking and dragging the tab header for the panel to a new position. z/OSMF displays a green arrow when you move the tab header to a legitimate position.

ISPF and SDSF

The ISPF task allows you to access your host system ISPF applications from z/OSMF.

To do so, from SPF Option 6, type `SDSF` on the command line; Figure 18-19 is displayed.

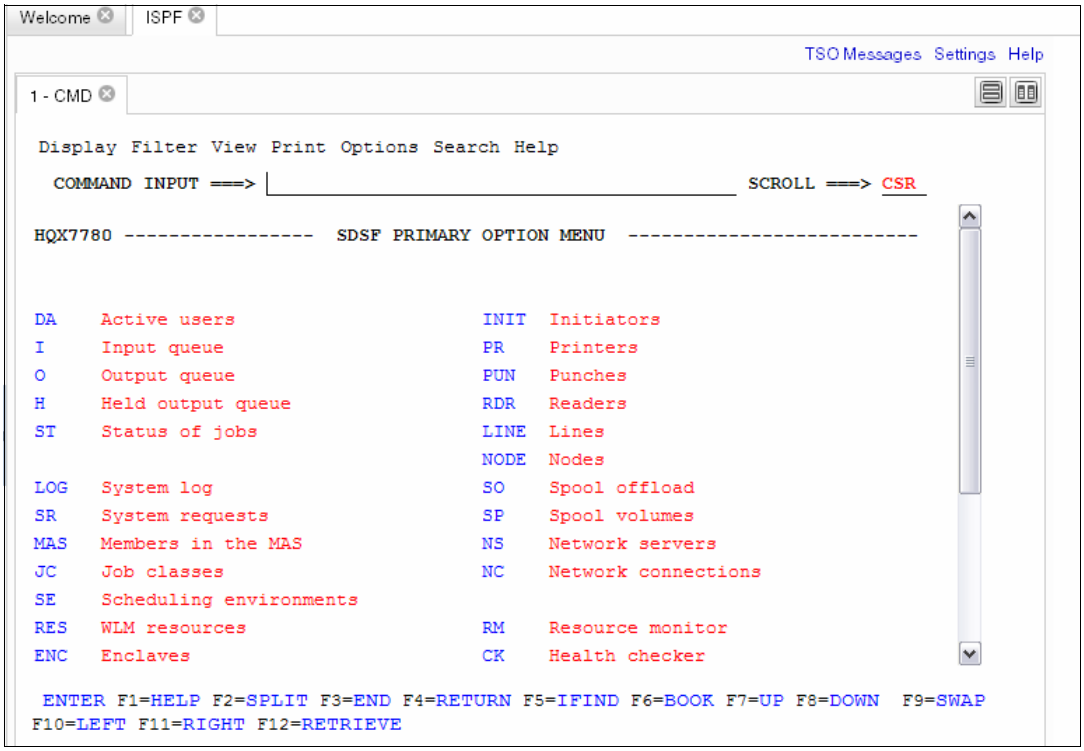


Figure 18-19 From ISPF Option 6 command line, type SDSF

When you select this ISPF category to access ISPF and traditional ISPF applications, you then have access to SDSF, as shown in Figure 18-20.

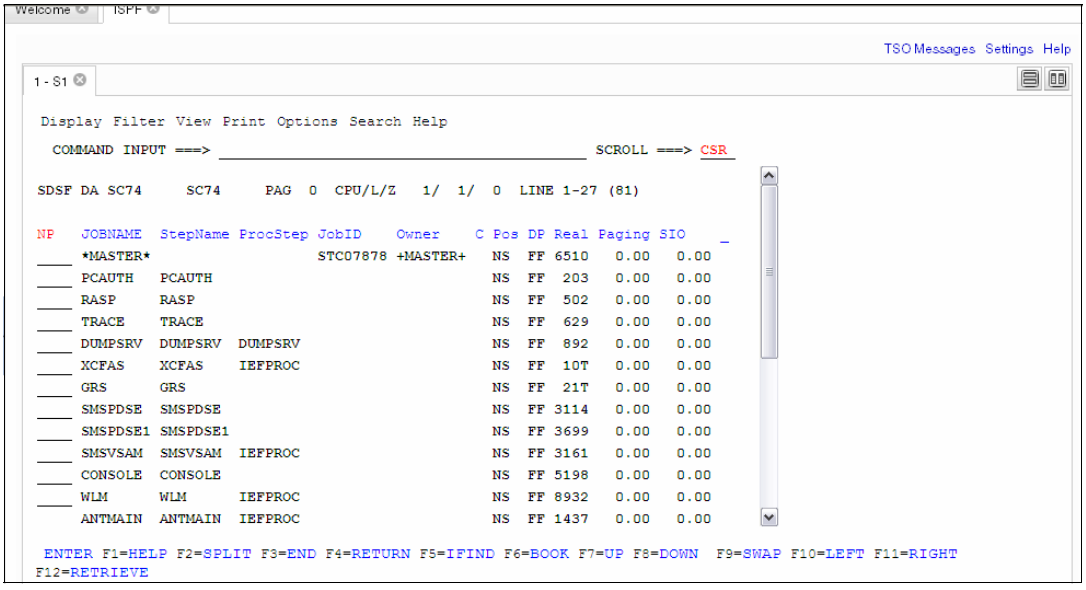


Figure 18-20 SDSF DA command example

18.10 DASD Management task

When you select the storage category, you can view tasks that can help you manage a subset of your storage on z/OS.The DASD Management task is included under the storage category. It can be used to perform storage management tasks, including adding storage to an SMS pool storage group, defining reserved storage pools, adding volumes to a reserved storage pool, viewing and managing volumes, and validating and activating source control data sets.

Note: This function is planned for delivery through APAR PM40869. This enablement APAR PM40869 is scheduled to be available 1Q2012.

Figure 18-21 shows the DASD Management window.

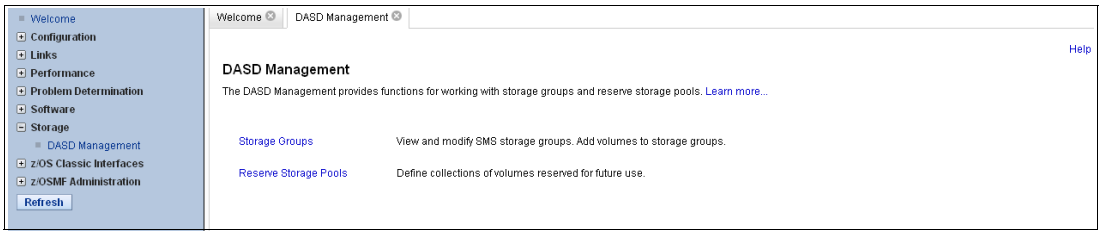


Figure 18-21 DASD Management

Today, the storage administrator must determine when a storage group is near its capacity, and identify how much storage to add and what volumes to add. After that is determined, multiple steps across various user interfaces are required to make the added capacity available to SMS:

- ISMF to add volume entries to the storage group definition (update SCDS)
- ICKDSF to initialize volumes
- Operator command to vary volumes online
- ISMF to activate the updated SCDS

When the DASD Management page is displayed, you can select the following options:

Storage Groups Use this option to view and add volumes to SMS pool storage groups. Use the Storage Groups page to work with SMS pool storage groups. You can view and add volumes to storage groups.

Reserve Storage Pools Use this option to view collections of storage volumes that you reserve for future use. Use the Reserve Storage Pools page to display reserve storage pools and list the volumes that make up the pools. Reserve storage pools provide the source of volumes that you can add to a storage group with the Add Storage action for storage groups. In addition, they can simplify the management of volumes that are defined but unused.

DASD Management task functions

The z/OSMF DASD Management task is designed to provide the storage administrator the ability to perform the following actions with a simplified single user interface, which is also designed to incorporate policy that can be defined at the storage group level.

The DASD Management task supports the new z/OSMF predefined role of Storage Administrator, and it provides the following functionalities:

- Manage containers of predefined available volumes with the introduction of the reserve storage pool resource
- Define new pool storage group SMA attributes to be used as policy with this task
- View storage group and volume information associated with the active configuration
- Add storage to an SMS pool storage group through a new wizard

18.10.1 Reserve storage pools

DASD management manages containers of predefined available volumes with the introduction of the reserve storage pool resource. A reserve storage pool refers to a group of volumes that are available for future use, whether they will eventually be used for SMS storage groups or for other reasons. The reserve storage pool resource is designed to replace the need for a storage administrator to manually maintain a list of defined but unused volumes.

Using reserve storage pools can make it easier to manage your defined but unused volumes. In addition, the DASD Management task of z/OSMF uses reserve storage pools to simplify adding storage to a pool storage group.

To belong to a reserve storage pool, a volume must have been initialized as reserve storage pool volume. A reserved volume has a volume serial number but cannot be brought online. It remains offline until it is reinitialized as a non-reserve storage pool volume.

With the DASD Management task, users can:

- Discover reserve storage pools which exist in the current system
- List volumes in a reserve storage pool

Assigning a reserve storage pool to a storage group

You can define a reserve storage pool as the default reserve storage pool for a pool storage group. Assigning a default reserve storage pool to a storage group simplifies adding volumes to the storage group with z/OSMF.

Assigning volumes to a reserve storage pool

To assign a volume to a reserve storage pool, you initialize the volume as reserved, using the **`ICKDSF INIT`** command with these parameters:

RESERVED This marks the volume as reserved.

OWNERID This specifies the name of the reserve storage pool, in the form IBMRSPrspname where rspname is the name of the reserved storage pool (up to 8 characters).

Note: The volume must be offline to all systems.

Listing reserve storage pools and volumes

To list the reserve storage pools in the I/O configuration for a system, you use z/OSMF. The Reserve Storage Pool function of the DASD Management task lets you display the reserve storage pools and list the volumes assigned to them.

You can then list volumes in a reserve storage pool for a particular reserve storage pool by using the Reserve Storage Pool function of the DASD Management task of z/OSMF.

Although the volumes in a reserve storage pool cannot be brought online, they can be viewed through ISMF functions or system commands that display offline volumes.

18.10.2 Defining pool storage group SMA attributes

With z/OS V1R13, the Storage Management Application (SMA) attributes are used by the DASD Management task of z/OSMF. They provide default values for adding storage to a pool storage group, which you do with the Add Storage action of the Storage Groups page.

Note: When defining SMA attributes for a pool storage group, you can use ISMF to define SMA attributes for a pool storage group. The SMA attributes include the name of a default reserve storage pool, a storage utilization goal, and the maximum amount of storage to be added to the storage group at one time. They are used by the DASD Management task of z/OSMF to simplify adding volumes to the storage group.

To define SMA attributes for a pool storage group, specify Y for DEFINE SMA Attributes on the Pool Storage Group Define panel in ISMF.

SMA attributes

When you define SMA attributes, the following parameters are used:

Storage Notification Threshold This specifies the point at which storage use in the storage group, as a percentage of total storage, exceeds the acceptable level.

Note: Total storage includes volumes ineligible for data set allocation because of their SMS status. Therefore, consider setting this to a lower value if not all of the online volumes have an SMS status of ENABLE.

Storage Utilization Goal This specifies the ideal storage use in the storage group, as a percentage of total storage.

Note: Total storage includes volumes ineligible for data set quitallocation because of their SMS status. Therefore, consider setting this to a lower value if not all of the online volumes have an SMS status of ENABLE.

Max Storage Size This specifies the maximum amount of storage to be added to the storage group at one time, in gigabytes.

Default Volume Size This specifies the typical capacity of any new volume added to the storage group, in gigabytes.

Default VTOC Size This specifies the default size of the VTOC for any new volume added to the storage group, in tracks. Enter a value from 1 through 65535.

Default Reserve Storage Pool This specifies the name of the default reserve storage pool for this storage group.

Volser Name Generation This specifies a pattern that defines a default naming convention used to generate volume serial numbers for new volumes to be added to the storage group.

In the pattern, you can use mask characters that will be replaced by other characters, as follows:

- **!** - Any alphabetic character (A-Z) or @,# or $
- **%** - Any numeric character (0-9)

- **+** - Any hexadecimal character (0-F)

Note: A valid volume serial number consists of 1 to 6 alphanumeric characters plus the special characters @, # and $.

Storage group and volume information

With the DASD management task, users can:

- View the list of pool storage groups associated with the active configuration
- View an alert when the Storage Utilization Notification Threshold is exceeded
- Display storage group level attributes
- Update storage group SMA attributes
- View volumes associated with a storage group
- Display volume level attributes.

AddStorage wizard

Users can select the AddStorage action against a storage group that invokes a wizard. The wizard guides users through steps that simplify the task of adding storage to a storage group

- Identifies the amount of storage to add (specified manually or through policy)
- Selects volumes from a reserved storage pool (based on policy and volumes found in reserve storage pool)
- Updates the SCDS with the selected volumes
- Initializes the selected volumes to match the storage group naming convention
- Optionally, varies the volumes online
- Optionally, activates the changes to the SCDS to make the added capacity available for SMS use

18.11 Capacity Provisioning

This task is included under the performance category. It can be used to view the status of the Capacity Provisioning Manager, and to view information about the active domain configuration and provisioning policy. You select this category to view tasks that can help you manage and monitor the performance of the workloads in your installation.

The following key functions are available in the Capacity Provisioning task in z/OSMF:

- Manage CIM connections to your Capacity Provisioning Manager

  In the Capacity Provisioning task you can create, modify, and delete CIM connections that you can use to connect to your Capacity Provisioning Manager. You can use both local CIM servers that run on the same system as z/OSMF, and connect to remote CIM servers.

- View a domain status report

  In the Capacity Provisioning task, you can view the status of a domain. This report contains information about the current setup of the domain managed by the Capacity Provisioning Manager. It matches the data displayed on the z/OS console in response to a REPORT DOMAIN command.

- View an active configuration report

In the Capacity Provisioning task, you can view the active configuration for a domain. The report contains information about the active domain configuration and the status of its elements. In addition to the name and status of the active configuration, you can inspect details about the CPCs and systems that belong to the active configuration. This report matches the data displayed on the z/OS console in response to a REPORT CONFIGURATION command.

- View an active policy report

  In the Capacity Provisioning task, you can view the active policy for a domain. The report contains information about the active policy and its status. You can view detailed information about each policy element. This report matches the data displayed on the z/OS console in response to a REPORT POLICY command.

Capacity Provisioning task

The Capacity Provisioning task in z/OSMF provides a browser-based user interface for working with the Capacity Provisioning Manager on your z/OS system. Through this task, you can request various reports of the status of the Capacity Provisioning Manager.

Capacity Provisioning Control Center

The Capacity Provisioning Control Center (CPCC) is the user front-end for administering Capacity Provisioning policies. CPCC is available as a separate Windows-based stand-alone client. Part of the functionality is integrated into z/OSMF V1R13, which will ease the monitoring of CP status for other domains.

18.11.1 z/OSMF Capacity Provisioning task

The z/OSMF Capacity Provisioning task simplifies the work of a z/OS CP administrator and provides functionality to manage connections to CPMs view reports for domain status, active configuration and active policy. It is designed to simplify the management of temporary capacity.

The scope of z/OS Capacity Provisioning is to address capacity requirements for relatively short-term workload fluctuations for which On/Off Capacity on Demand is applicable. It is not a replacement for the Capacity Management process.

If selected, the arrow that points to “use same system as z/OSMF” in Figure 18-22 indicates a connection to a Provisioning Manager running on the same system as z/OSMF is to be defined. On that page you can manage connections to the Provisioning Manager. When selecting such a connection the host address, protocol and port properties are hidden because they are determined automatically.

Provisioning Manager

The Provisioning Manager is the component of Capacity Provisioning which monitors the domain and performs activation and deactivation requests for temporary capacity, based on your active policy and domain configuration.

Connecting to the Provisioning Manager on a domain has the following prerequisites:

- You must be a member of the Provisioning Manager query security group.
- You can only communicate with a Provisioning Manager that delivers report information, if it runs on a system with z/OS V1R12 or higher.

You can use the Provisioning Manager page in the Capacity Provisioning task in IBM z/OS Management Facility (z/OSMF) to manage the communication to Capacity Provisioning

Managers through CIM connections. Specifically, you can add, modify, or remove connection entries.

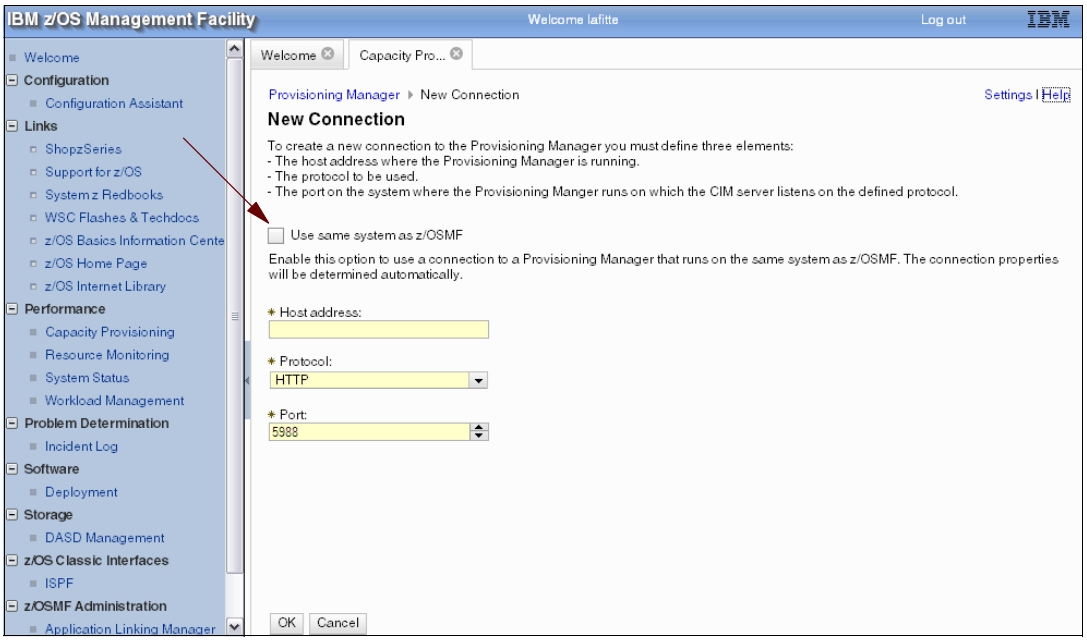


Figure 18-22 Capacity Provisioning primary panel

In Figure 18-22 on page 387:

Host address This is the symbolic host name or IP address where the CIM server is running that is used to communicate with the Provisioning Manager. The host name or IP address is required and allows up to 255 characters.

Protocol This indicates whether HTTP or HTTPS is used to establish a connection to the CIM server.

Port This is the port number where the CIM server is listening for incoming HTTP or HTTPS requests. The port number is required, and it must be an integer between 1 and 65535. The default port number for HTTP is 5988. The default port number for HTTPS is 5989.

Using Capacity Provisioning with z/OSMF

As shown in Table 18-4, availability of z/OSMF Capacity Provisioning introduced with z/OS V1R13 provides benefits that are quantified based on an IBM laboratory environment. This can differ in other MVS environments

Table 18-4 z/OSMF Capacity Provisioning (V1R13) benefits

	Without Capacity Provisioning with z/OSMF	With Capacity Provisioning with z/OSMF[a]
View active CP policy and compare with data provided by RMF and WLM.	Start stand-alone Windows-based client (CPCC) for Capacity Provisioning, connect to CPM and display active configuration report. Open z/OSMF in a browser and inspect RMF and WLM data. 5-10 minutes until all tasks are completed	Use integrated z/OSMF GUI to work with CP. RMF and WLM and compare data provided by each exploiter. 2-3 minutes until all tasks are completed
Operator needs to reuse existing connection.	Connection information like host name, protocol, and port needs to be manually gathered from a primary person. Available domains must be known. Up to 5 minutes.	Usage of shared connection repository in z/OSMF. List of available domains is retrieved from server and shown to user. No extra time needed.
Installation of the capacity provisioning UI application[b]	Install Windows client (CPCC) on workstation. Difficult to install on managed clients; 20 minutes otherwise.	Centrally managed z/OSMF application available to all authorized users. No extra time needed.

a. Based on IBM laboratory results, your results can vary.
b. Complete set of CPCC functionality is not provided in z/OS V1R13.

This new function manages CIM connections to access Capacity Provisioning Manager in a central shared repository. It can create, modify, and delete CIM connections to CIM servers which can be either local and remote. Various reports are also provided, as described here:

- The domain status report views the status of a domain. The same data is displayed that is retrieved when a **REPORT DOMAIN** command on the z/OS console is issued.
- The active configuration report views the active configuration of a domain. The same data is displayed that is retrieved when a **REPORT CONFIGURATION** command on the z/OS console is issued.
- The active policy report views the active policy of a domain. The same data is displayed that is retrieved when a **REPORT POLICY** command on the z/OS console is issued.

Figure 18-23 shows the Capacity Provisioning active configuration.

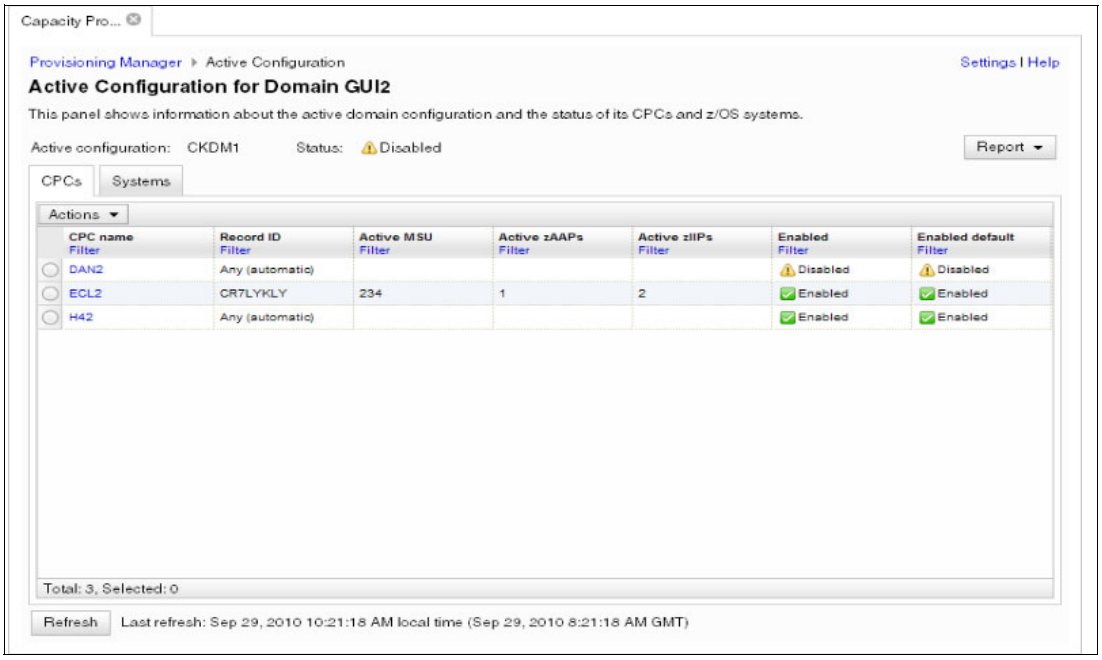


Figure 18-23 z/OSMF Capacity Provisioning active configuration

You can drill down to the CPC or system, view information about the active configuration for a domain, and view CPC details, system details, or active policy details, as shown in Figure 18-24.

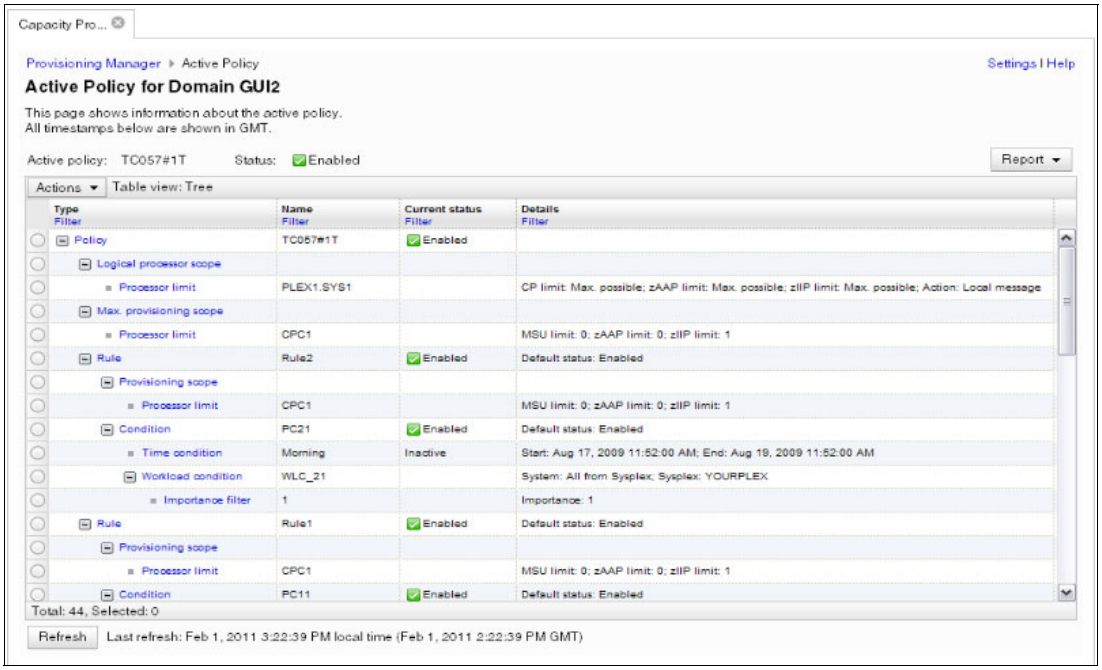


Figure 18-24 z/OSMF Capacity Provisioning active policy

18.12 z/OSMF base enhancements

z/OSMF with z/OS V1R13 includes information about the use of the following application programming interfaces (APIs) in z/OSMF:

- Application Linking Manager interface is a new API that allows a client application to define event types and event handlers to z/OSMF.
- z/OS jobs REST interface is a new API that allows a client application to perform operations with batch jobs on a z/OS system.

Note: In SAF Authorization Mode, the z/OSMF administrator has access to the following tasks:

- Manage links through the Links task
- Manage task associations through the Application Linking Manager task

In Repository Authorization Mode, the z/OSMF Administrator has access to the following tasks:

- Assign users to roles through the Users task
- Assign tasks to roles through the Roles task
- Manage links through the Links task
- Manage task associations through the Application Linking Manager task.

18.12.1 Application Linking Manager task

To perform z/OS management tasks, typically you interact with various interfaces, such as the TSO command line, graphical user interfaces, and web-style interfaces. To help you link or connect your z/OSMF tasks and external applications, z/OSMF provides the Application Linking Manager task.

To open the Application Linking Manager task, in the navigation area, expand the z/OSMF Administration category and select Application Linking Manager, as shown in Figure 18-25. The task provides a web-based, user interface that you can use to:

- Define new event types, and view and delete existing event types.
- Define new handlers; view, enable, disable, and delete existing handlers; and make a handler the default handler.

Note: The Application Linking Manager task also provides an application programming interface (API) that you can use to complete these actions.

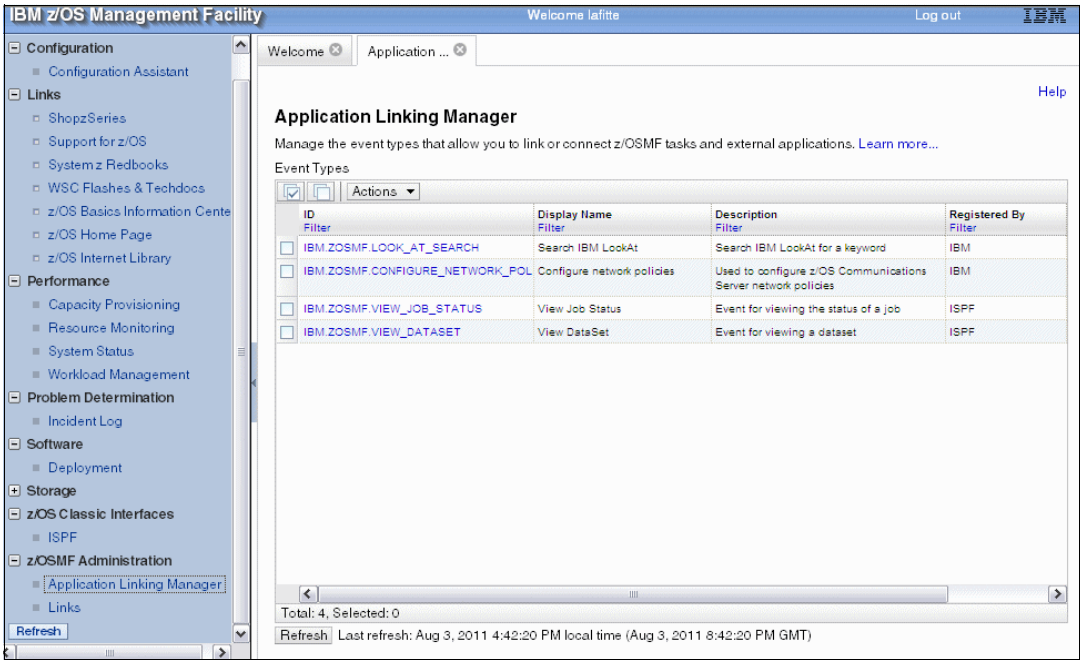


Figure 18-25 Application Linking Manager task

Application Linking Manager components

The key components of the Application Linking Manager task are shown in Figure 18-26 and are listed here:

- Event requestor - z/OSMF task or external application that requests the launch of a specific function within another task or external application.
- Event - Action requested by the event requestor. It includes the type of event and the event parameters.
- Event type - Object that connects an event requestor to an event handler. It identifies the handlers that can process an event and the possible parameters that can be supplied with an event.
- Event handler - z/OSMF task or external application that can process the event parameters and display the requested information.

Application linking process

As shown in Figure 18-26, the process begins with a user action, such as clicking a link as the event requestor. In response to this action, the event requestor creates an event and sends it to the Application Linking Manager.

The Application Linking Manager searches the set of known event types for the type identified by the event. If a match is found, the Application Linking Manager searches for event handlers that are registered for this event type. If only one handler is found, it is launched. Otherwise, the user is prompted to select the handler to launch.

The Application Linking Manager provides the handler with the parameters that were supplied with the event. The event handler processes the parameters and displays the requested information.

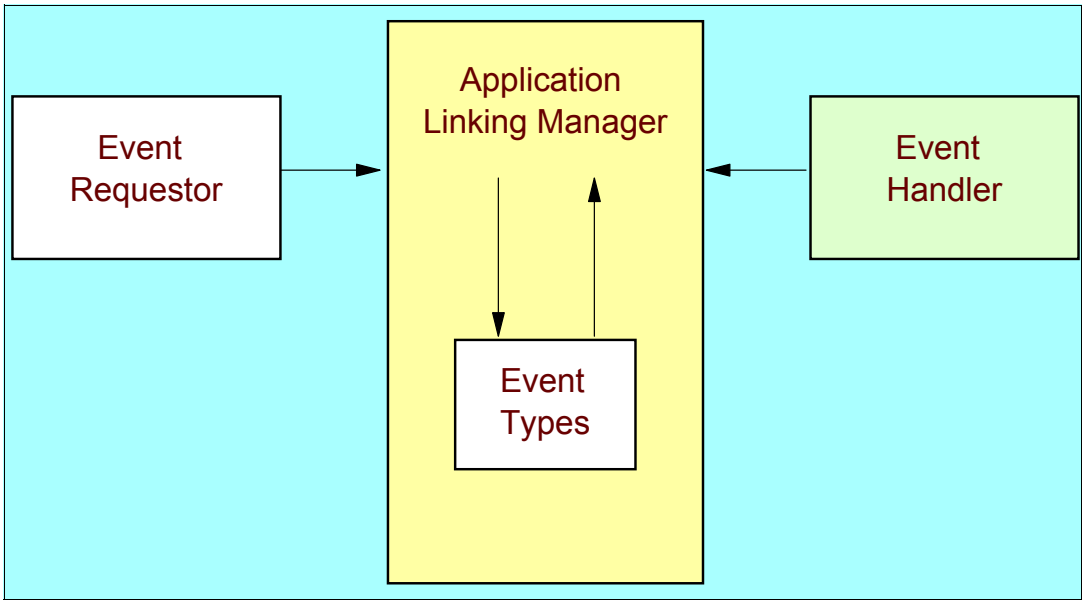


Figure 18-26 Key components in the application linking process

18.12.2 z/OSMF Application Linking Manager interface

The Application Linking Manager interface services can be invoked by any HTTPS client application running on the z/OS local system or a remote system; for example:

```
https://{host}:{port}/zosmf/izual/rest/{resource}?{parm}
```

Your program (the client) initiates the HTTPS request to the Application Linking Manager interface. If the interface determines that the request is valid, it performs he requested service. After performing the service, the Application Linking Manager interface creates an HTTP response. If the request is successful, this response takes the form of an HTTPS 200 (OK) response and, if applicable, an object containing a result set, which it passes back to your program. Results are returned in the form of a JavaScript Object Notation (JSON) object, which must be parsed by your program. If the request is not successful, the response consists of a non-OK HTTPS response code and a JSON object.

Note: For requests that return data, the services define an expected response in the form of a JavaScript Object Notation (JSON) object or another type of response data. Errors can also include a returned JSON object that contains a message that describes the error.

Events

z/OSMF will maintain a repository of Events, which are definitions of services that can be invoked (that is, View abc). An event has a type and a set of parameters (key value pairs). The values must be representable in a URL. The type describes the action requested by the user. The parameter values are all of type string; there is no distinction between optional and required parameters.

Each plug-in or external application can use the registered event types, or define new event types to send or handle events. The event types are dynamic. Each exploiter (external or internal) can define new event types at run-time. The process of defining of a new type is called “registering”. For the registration, the plug-in has to provide a name (string) and a map of parameters the events of this type will contain.

Event handlers

If an application (external or internal) provides functionality to process an event, it can register as a handler for the given event type in the application linking manager. Thus, it can be invoked when the respective event is sent. An application can be a handler of multiple event types.

Multiple applications can register as handlers for a given event type. In this case, the application linking manager will display a dialog prompting the user to select a handler for the executed action. Non-z/OSMF applications (IBM or ISVs) can register their own links, which provide functionality defined by z/OSMF link interfaces. Registration is possible through servlet invocation or can be performed on a new z/OSMF task (also known as GUI). The servlet API will provide a JSON-based RESTful interface. For each registration, the ID, Name, and URL need to be provided.

Launching an event from z/OSMF

When an event is launched from z/OSMF into an external link definition:

- The code attempting to launch will supply the name to be invoked and the parameters defined by the respective event to use when launching.
- The URL will be launched and the parameters will be passed through the POST method.
- Each attempt to launch will launch a new browser window.
- No identity will be passed in a parameter. It is assumed that the launched URL will address prompting for the user for an ID and password, and then manage that from that point forward.

The URL will be the same regardless of which link is to be invoked and the respective link interface parameters will need to be provided, as follows:

```
https://host:port/zosmf/LinkManagerLaunch)
```

When a z/OSMF provided link interface is launched:

- The parameters to the URL will be event type and link parameters.
- If the browser session has already been authenticated to z/OSMF, the existing identity will be used.
- If the browser session has not already been authenticated to z/OSMF, a dialog will be displayed for users to provide their user ID and password to authenticate to z/OSMF.
- After the user is authenticated to z/OSMF, the link will be launched in a tab in the work area. There will be no other tabs open in the work area. However, the navigation tree will display on the left side so that the user can utilize other functions of z/OSMF that they are authorized to.

Table 18-4 lists the application linking servet APIs.

Note: There are no strict validity checks for parameters. That is, if a handler registers an event type that already exists with other parameters, an exception will be thrown only for information.Whether the event type registration is consistent or not, events will be delivered to the registered handler at run-time. It is then up to the handler to check the validity of the parameters. If, for instance, important parameters are missing, the handler cant choose to display the welcome page and show an error message. If there are too many parameters in the event, they will probably be ignored.

Table 18-5 Application linking servlet APIs

Method	Servlet	API	Method
	URL	HTTP method	Request Parameters
registerEventType	/zosmf/applinker/eventtype/<id>	POST/	String params String description String eventTypeId (in the URL)
registerExternalHandler	/zosmf/applinker/handler/<id>	POST	String eventTypeId, String handlerID String handlerName, String url, JSO NObject opt
getHandlersForEventType	/zosmf/applinker/handler	GET	String eventTypeId
unregisterHandler	/zosmf/applinker/handler/<id>	DELETE	String handlerId (in the URL) String eventTypeId
unregisterEventType	/zosmf/applinker/eventtype/<id>	DELETE	String eventTypeId (in the URL)

The new event type can defined as displayed in Figure 18-27.

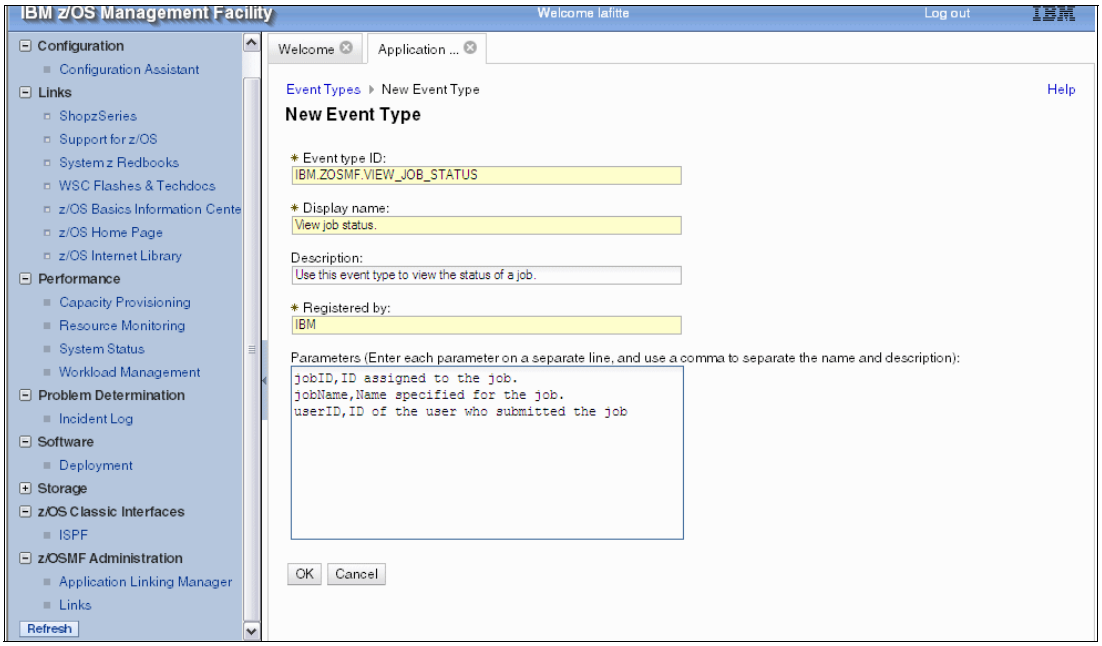


Figure 18-27 Defining new event type

18.12.3 REST API for job management

z/OSMF V1R13 is introducing a new HTTP(s) interface to z/OS for submitting and accessing job information. This interface is an open API that can be driven locally on a z/OS system or more likely driven from a remote system. The remote system only needs to the HTTP(s) protocol.

The web services are easily consumable by:

- Such web applications as javascript, AJAX, or Flex(Flash)
- Other web service clients, such as Java, PHP, and Perl

The web service supports both JES3 and JES2 as the primary JES subsystem, and secondary subsystems, though the support is currently limited.

Representational State Transfer APIs

z/OSMF supports the use of Representational State Transfer (REST) APIs, which are public APIs that your application can use to work with system resources and extract system data. As with implementations of REST services on other platforms, the z/OSMF APIs allow for easy-to-use HTTP services that are language- and platform-independent, stateless, scalable, and easily parsed.

The format for the z/OSMF APIs requests and responses is based on the HTTP/1.1 protocol. Conceptually, your application (the client) issues requests to the target system (z/OS) in the form of request messages. Each request message consists of a request line, optionally followed by request headers (HTTP headers), an empty line, and an optional message body. The request line includes the HTTP method, such as GET, a Universal Resource Identifier (URI) and, where appropriate, parameters that further qualify the request. For requests that return data, the services define an expected response in the form of a JavaScript Object Notation (JSON) object or another type of response data.

Using the REST interface with batch jobs

With the z/OS jobs REST interface, an application can use REST services to perform the following operations with batch jobs:

- Obtain the status of a job
- List the jobs for an owner, prefix, or job ID
- List the spool files for a job
- Retrieve the contents of a job spool file
- Submit a job to run on z/OS
- Cancel a job
- Change the job class of a job
- Purge a job from the JES spool

Important: Using the z/OS jobs REST interface requires one of the following minimum levels of JES on your z/OS system: JES2 V1R13 or JES3 V1R13.

The z/OS jobs REST interface services can be invoked by any HTTP client application running on the z/OS local system or a remote system.

Accessing the REST interface

Your program (the client) initiates an HTTP request to the z/OS jobs REST interface. If the interface determines that the request is valid, it performs the requested service. After performing the service, the z/OS jobs REST interface creates an HTTP response. If the request is successful, this response takes the form of an HTTP 2nn response. If applicable, a result set that is passed back to your program.

The URLs of the z/OS jobs REST interface have the following format:

```
https://{host}:{port}/zosmf/restjobs/jobs/{resource}?{parm}
```

Note: The URL requires the following formats:

- `https://{host}:{port}` specifies the target system address and port.
- `/zosmf/restjobs/jobs` identifies the z/OS jobs REST interface.
- `{resource}?{parm}` represents the resource, such as a job name and job ID, and optionally one or more parameters, to qualify the request.

Depending on which service was requested, the result set can be returned in a format that requires parsing by your program, for example, a JSON object. In other cases, results can be returned in another format, such as plain text or binary data. If the request is not successful, the response consists of a non-OK HTTP response code with details of the error provided in the form of a JSON object.

Security considerations

Users need to authenticate to z/OSMF. Additional system security requirements exist such as:

- To get the contents of a spool file for a job, users must have access to the JESSPOOL profile associated with the spool data set.
- To submit a job, s must be authorized to run jobs on the system and be able to access any protected resources that the job might require.
- To cancel a job, change a job's class, or purge a job, users must be authorized to use the Common Information Model (CIM) server and be permitted to the JES3 or JES2 Jobs CIM provider.
- To cancel a job or purge a job, users must be authorized to cancel the job.

Table 18-6 lists the RESTful services for job management.

Table 18-6 RESTful services

Service	URL	HTTP method
Get job status	`https://host}:{port/zosmf/restjobs/jobs/jobname/jobid`	GET
Get job status - secondary JES	`https://host:port/zosmf/restjobs/- jesb/jobname/jobid`	GET
Get a list of spool files for a job	`https://host:port/zosmf/restjobs/jobname/jobid/files/`	GET
Get the contents of a particular spool file for a job	`https://host:port/zosmf/restjobs/jobs/jobname/jobid/files/nnn/records`	GET
Get a list of jobs, optionally based on owner or prefix parameters	`https://host:port/zosmf/restjobs/jobs`	GET
Submit a job	`https://host:port/zosmf/restjobs/jobs`	PUT
Cancel a job	`https://host:port/zosmf/restjobs/jobs/jobname/jobid`	DELETE

18.13 Workload Management and z/OSMF

The Workload Management task in IBM z/OS Management Facility (z/OSMF) provides a browser-based user interface that you can use to manage z/OS Workload Manager (WLM) service definitions and to provide guidelines for WLM to use when allocating resources. Specifically, you can define, modify, view, copy, import, export, and print WLM service definitions. You can also install a service definition into the WLM couple data set for the sysplex, activate a service policy, and view the status of WLM on each system in the sysplex.

To display the Workload Management task, expand the Performance category in the navigation area and select Workload Management. The Workload Management page is displayed, as shown in Figure 18-28.

Note: If you are working with more than one service definition, several New, View, Modify, Copy, and Print Preview tabs can be displayed. You can display up to 20 tabs at a time.

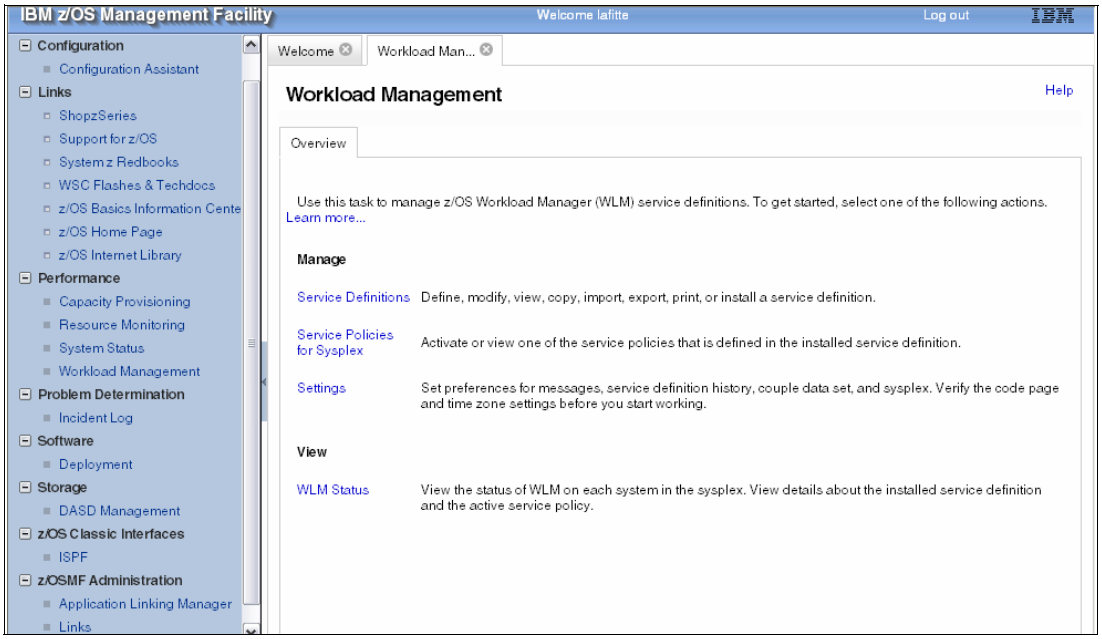


Figure 18-28 Workload Management page

18.13.1 z/OSMF Workload Management V1R13 new features

Separate authorization levels are provided for the following tasks:

- Viewing service definitions, service policies, and WLM status
- Installing and activating service policies
- Modifying service definitions

Group name

With z/OS V1R13, you must supply the name of the security group that your installation has created for authorizing users to WLM resources on your system, such as the WLM couple data set.

The planning worksheet shown in Table 18-7 includes the variable name and default for this value.

Note: If you choose not to deploy the Workload Management plug-in, you can skip this planning worksheet.

Table 18-7 Worksheet for the Workload Management task variable

Input	Description	Variable name	Default value	Your value
Group for accessing workload manager (WLM) resources.	Group name to use for allowing the Workload Management task to access WLM resources on your system This group requires UPDATE access to facility MVSADMIN.WLM.POLICY. Your installation must create this group outside of the z/OSMF configuration process. If you do not plan to create this group, specify NO.KNOWN.VALUE for this group name.	IZU_WLM_ GROUP_NAME	WLMGRP	

New system prerequisite for the Workload Management plug-in

With z/OS V1R13, there is a new system prerequisites for the Workload Management plug-in. To use the Workload Management task, your installation must perform an additional system setup step, as listed in Table 18-8.

Table 18-8 z/OS setup actions for the Workload Management task

z/OS setup action	Where described
Ensure that library BLDUXTID in SYS1.MIGLIB is program controlled. For example, in a RACF system, you can use the following commands to ensure that a library is program controlled: RDEFINE PROGRAM BLSUXTID RALT PROGRAM BLSUXTID + ADDMEM('SYS1.MIGLIB'/'******'/NOPADCHK) + UACC(READ) SETROPTS WHEN(PROGRAM) REFRESH	This step is performed in the CIM provided job CFZSEC. See the chapter on customizing the security for the CIM server in *z/OS Common Information Model User's Guide*, SC33-7998.

Additional authorizations for the Workload Management task

With z/OS V1R13, the following additional steps are required to authorize users to specific functions within the Workload Management task.

Users of the Workload Management task require access to the RACF facility class profile MVSADMIN.WLM.POLICY. If your installation runs the CIM security setup job CFZSEC as part of setting up the CIM server for z/OSMF, then all users who are authorized for the CIM server will also be authorized for this RACF facility. If this set of authorizations is acceptable in your environment, no further steps are needed.

This example shows how you can create a separate RACF WLM user group and authorize it for the RACF facility MVSADMIN.WLM.POLICY:

```
ADDGROUP "WLMGroupName" OMVS(GID("WLMGroupGID"))
PERMIT MVSADMIN.WLM.POLICY CLASS(FACILITY) ID("WLMGroupName") ACCESS(UPDATE)
SETROPTS RACLIST(FACILITY) REFRESH
```

Note: If not all CIM server users are to have access to facility MVSADMIN.WLM.POLICY, you have to perform additional customization to avoid creating unwanted authorizations.

Viewing WLM status

The Workload Management task provides an HTML-formatted view (WLM Status tab) of the same data that is retrieved when you enter the **`D WLM,SYSTEMS`** command on the z/OS console. Specifically, the WLM Status tab displays the status of WLM on each system in the sysplex, and lists details about the installed service definition and the active service policy.

To define settings, the Workload Management task provides a shared location (Settings tab) where you can specify how long to keep the service definition history and define the code page, time zone, and backup sequential data set for the sysplex. You can also enable consistency checking between z/OSMF and the WLM couple data set, and indicate whether you want the Workload Management task to display or suppress information messages.

Figure 18-29 shows the WLM properties of a z/OSMF user.

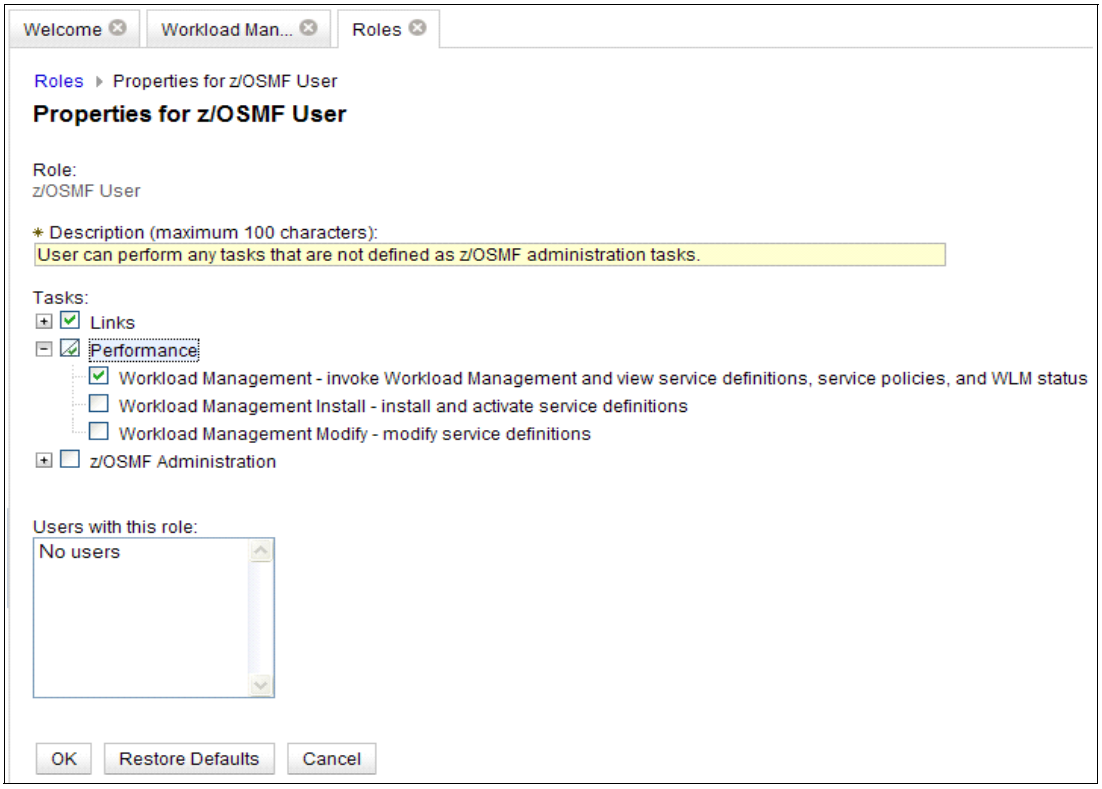


Figure 18-29 WLM properties of a z/OSMF user

SAF Authorization Mode

In SAF Authorization Mode, the WLM authorization of roles is controlled through the SAF resource names:

- ZOSMF.WORKLOAD_MANAGEMENT.WORKLOAD_MANAGEMENT.VIEW
- ZOSMF.WORKLOAD_MANAGEMENT.WORKLOAD_MANAGEMENT.INSTALL
- ZOSMF.WORKLOAD_MANAGEMENT.WORKLOAD_MANAGEMENT.MODIFY

Note: To enable a role to launch the Workload Management task, it is not sufficient to provide authorization for simply “installation” or “modification”; the role must also be authorized for “viewing.”

Repository Authorization mode

In Repository Authorization Mode, the WLM authorization of roles is controlled by three tasks on the roles panel:

- Workload Management
- Workload Management install
- Workload Management modify

Note: If your installation is running z/OSMF in Repository Authorization Mode, your installation manages user authorizations through the Roles task. For Workload Management, assign tasks to roles as described here.

Required authorization level of role Task to be selected in the Roles page:

View Workload Management
Install Workload Management Install
Modify Workload Management Modify

By default, z/OSMF administrators are authorized for the View, Install, and Modify functions, which is equivalent to a WLM policy administrator. z/OSMF users are authorized for the function View, which is equivalent to a WLM performance analyst.

SAF Authorization Mode

If your installation chooses to use z/OSMF in SAF Authorization Mode, you control the authorization through your security management product, for example, RACF. Grant access authority to the appropriate users or groups, as shown in Table 18-9.

Table 18-9 Workload Management task authorizations - z/OSMF running in SAF Authorization Mode

Required authorization level of user or group	Required SAF access authority
View	READ access for profile <SAF-prefix>.ZOSMF.WORKLOAD_MANAGEMENT.WORKLOAD_MANAGEMENT.VIEW
Install	READ access for profile <SAF-prefix>.ZOSMF.WORKLOAD_MANAGEMENT.WORKLOAD_MANAGEMENT.INSTALL
Modify	READ access for profile <SAF-prefix>.ZOSMF.WORKLOAD_MANAGEMENT.WORKLOAD_MANAGEMENT.MODIFY

Attention: If these default settings do not meet your needs, you can change the SAF authority of these respective groups for the profiles shown in Table 18-9, or you can define new custom groups:

- <IZU_ADMIN_GROUP_NAME>
- <IZU_USERS_GROUP_NAME>)

Migration consideration for the Workload Management task

z/OSMF V1R12 supports only one authorization level for the Workload Management task. However, z/OSMF V1R13 supports several authorization levels, as described previously. For z/OSMF V1R13 migration consideration, see the information about verifying user authorizations for the Workload Management task in *z/OS IBM z/OS Management Facility Configuration Guide*, SA38-0652.

18.14 Incident Log enhancements

Using the Problem Determination task allows you to view a task that can help you manage problems on z/OS. This category contains the Incident Log task, which provides a consolidated list of system problems, along with the details and the diagnostic data captured and saved with each problem.

This task also helps you send the diagnostic data to IBM or a vendor for further diagnosis. Specifically, z/OSMF and the Incident Log task interact with z/OS system functions in the following areas:

- Common Information Model (CIM) server for handling requests made by z/OSMF
- SDUMP component for managing the capture of OPERLOG, SYSLOG, and logrec snapshots
- IPCS dump directory services for managing the inventory of dumps related to incidents
- System Logger to capture log snapshots when sysplex-scope recording is requested through the OPERLOG or logrec system logger streams
- Dump analysis and elimination (DAE) for enabling the “take next dump” feature of the Incident Log task
- Environmental Record Editing and Printing (EREP) program for formatting the logrec data
- Common Event Adapter (CEA) for providing the data that is subsequently displayed in the Incident Log task user interface.

Figure 18-31 shows a number of base z/OS functions are involved when the Incident Log task is used to manage diagnostic data for your system.

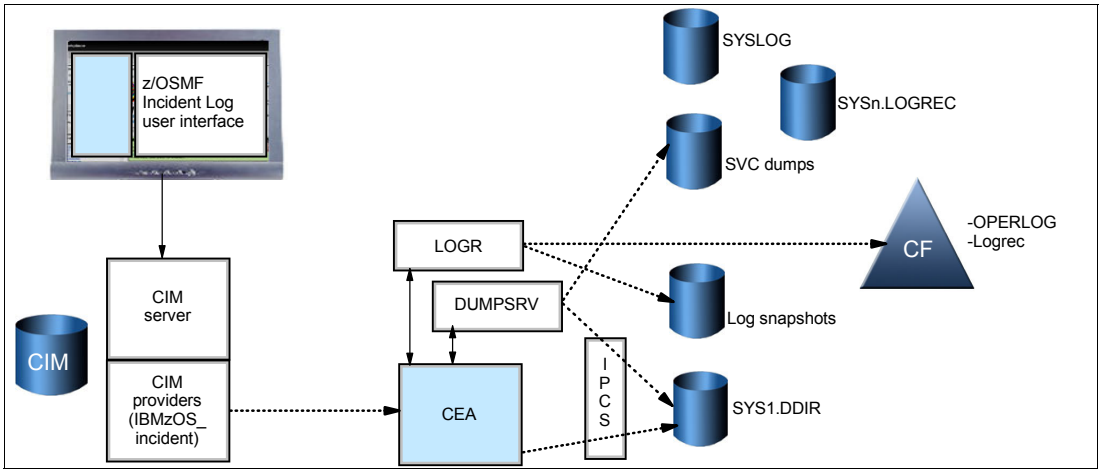


Figure 18-30 z/OS components used in Incident Log task processing

Setting up the Incident Log

Enabling your z/OS system for the Incident Log task requires initial setup work on the z/OS host system, as described here. The following system components interact and play a role in support of problem data management at your installation.

Incident Log interactions with other z/OSMF components

z/OSMF and the Incident Log task interact with z/OS system functions in the following ways:

CIM The Common Information Model (CIM) server handles requests made by z/OSMF.

SDUMP The SDUMP component manages the capture of OPERLOG, SYSLOG, and logrec snapshots.

IPCS The dump directory services are used for managing the inventory of dumps related to incidents.

System Logger The System Logger is used to capture log snapshots when sysplex-scope recording is requested through the OPERLOG or logrec system logger streams.

DAE The Dump analysis and elimination (DAE) component enables the “take next dump” feature of the Incident Log task.

EREP The Environmental Record Editing and Printing (EREP) program is used to format the logrec data.

CEA The Common Event Adapter (CEA) is used to provide the data that is subsequently displayed in the Incident Log task user interface. CEA helps to coordinate these system functions on behalf of z/OSMF incidents, in both single system and sysplex environments.

Incident Log and SYSLOG

If your installation collects messages about programs and system functions (the hardcopy message set) on a single-system basis, the Incident Log task uses the system log (SYSLOG) as the source for diagnostic log snapshots. This function requires one of the following minimum levels of JES on your system: JES2 V1R11 or JES3 V1R11 (with the PTF for APAR OA29534).

Here you must ensure that the proper security permissions exist, so that the JES subsystem can access SYSLOG on behalf of the CEA component of z/OS. For example, in a system with RACF as the security management product, your security administrator can issue RACF commands as follows:

```
RDEFINE JESSPOOL SY1.+MASTER+.SYSLOG.*.* UACC(NONE)
PERMIT SY1.+MASTER+.SYSLOG.*.* CLASS(JESSPOOL) ID(CEA_userid) ACC(ALTER)
SETROPTS RACLIST(JESSPOOL) REFRESH
```

Incident Log and CEA

The role of CEA in z/OSMF processing is summarized here:

- When CEA becomes active, it establishes an association with your installation's sysplex dump directory (typically SYS1.DDIR). This directory contains the inventory of SVC dumps taken in your sysplex and relevant information about each dump incident. This processing is performed for SVC dumps taken on behalf of system abends and those taken through the **DUMP** command and SLIP traps.
- Whenever an SVC dump is written to a data set, the DUMPSRV address space (on behalf of SVC dump processing) creates a new entry in the sysplex dump directory and informs CEA that the new incident has arrived.

Incident Log page

You can use the Incident Log page in the Incident Log task to manage the incidents that have occurred on a system or in a sysplex.

z/OSMF retrieves the incidents from the system and displays them in the Incident Log table. A maximum of 500 of the incidents that match the date and time filter criteria are retrieved and are available to be displayed in the table. By default, only the incidents that occurred in the past three days are displayed, and they are sorted in descending order according to the Date and Time column.

You can change the default of three days by selecting Date and Time and changing the number of days in the dialog that displays on the panel; see Figure 18-31.

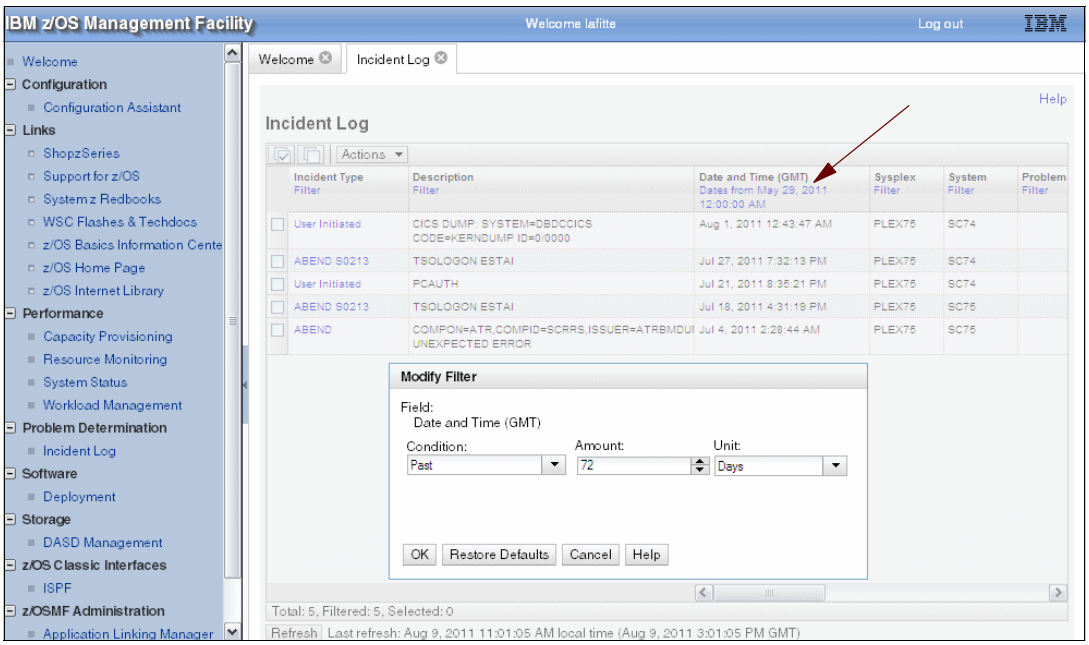


Figure 18-31 Incident Log panel to modify date and time values

View Diagnostic Details page

You can use the View Diagnostic Details page in the Incident Log task to view the properties of an incident, to view the system-supplied diagnostic data files, and to add additional files to send with an incident.

To select this page, click the **Actions** bar shown in Figure 18-32 and a pull-down menu will display.

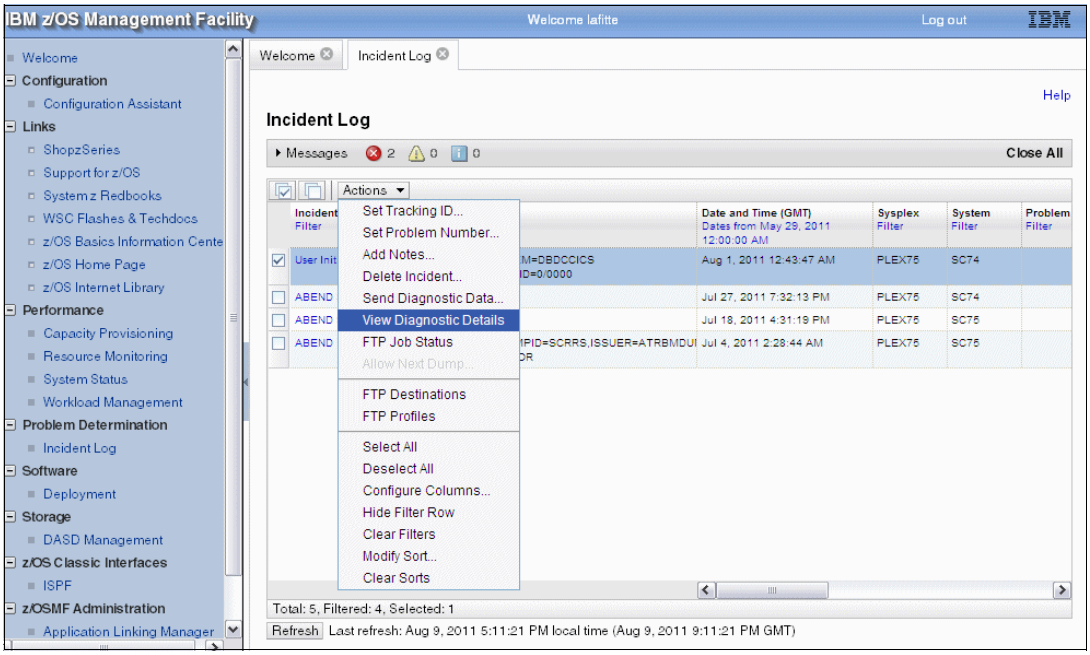


Figure 18-32 Incident Log display showing the pull-down menu for Actions

Selecting **View Diagnostic Details** from the pull-down menu then displays the window shown in Figure 18-33.

You can use the Diagnostic Data tab on the View Diagnostic Details page, shown in Figure 18-34, to see a list of the system-supplied diagnostic data files, browse the log snapshots, and specify additional files to send with the incident.

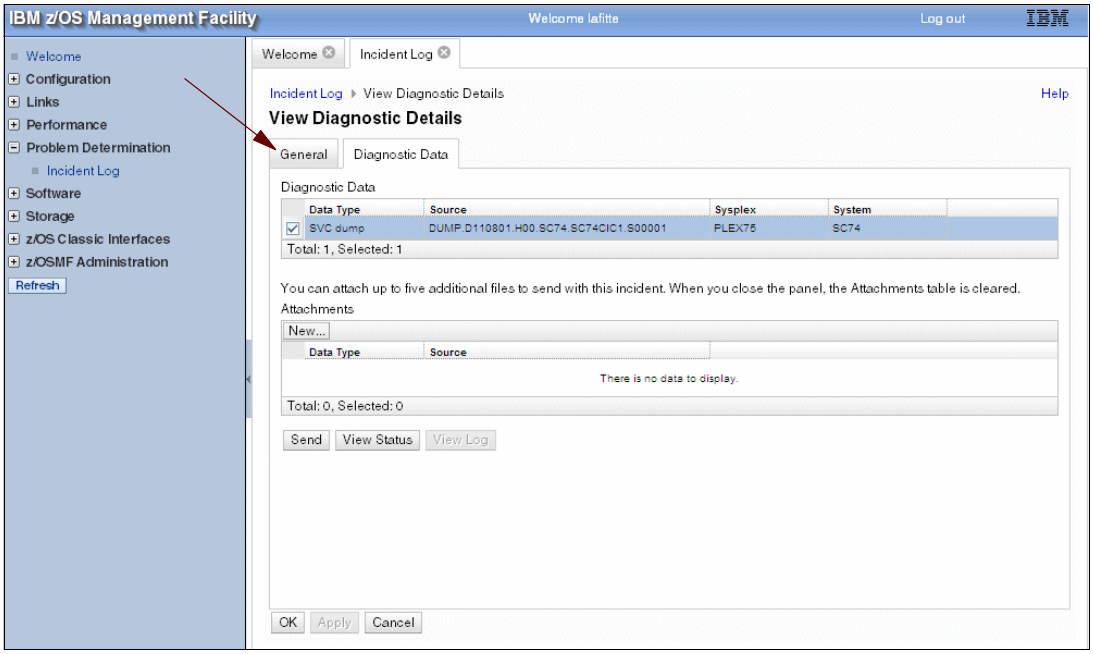


Figure 18-33 View Diagnostic Details panel

Properties of incidents

Use the General button, shown in Figure 18-33, to view the properties of an incident, to set the tracking ID and problem number of an incident, and to document additional information about an incident.

Figure 18-34 is then displayed and it contains the following information.

Date and time This is the date and time on the system when the incident occurred. The date and time is in Greenwich Mean Time (GMT).

Sysplex name This is the name of the sysplex in which the incident occurred. If the system is not in a sysplex, this field is blank.

System name This is the name of the system on which the incident occurred.

Problem number Enter the problem number to associate with the incident. For more information about problem numbers, see the help topic “About problem numbers.”

If a problem number is associated with the incident, it is displayed. Otherwise, this field is blank. You can modify the existing problem number, or enter a new number.

The problem number is required only in the Send Diagnostic Data wizard.

After all the fields are filled in and you enter the panel again, Figure 18-38 is shown. At this point you can enter a Retain search string to the incident (the figure shown is for another incident).

View Diagnostic Details window

In Figure 18-38, the arrow points to a search string that you can use to search the RETAIN database to find z/OS APARs related to the incident. RETAIN is the database used by IBM Support Centers to record all known problems with IBM licensed programs. To search the RETAIN database, visit the IBMLink/ServiceLink website at:

`http://www.ibm.com/ibmlink/servicelink)`

If the search does not return a match, omit several search terms and retry the search.

Note: This field is displayed only when the incident type is ABEND and a symptom string is provided in the correct format.

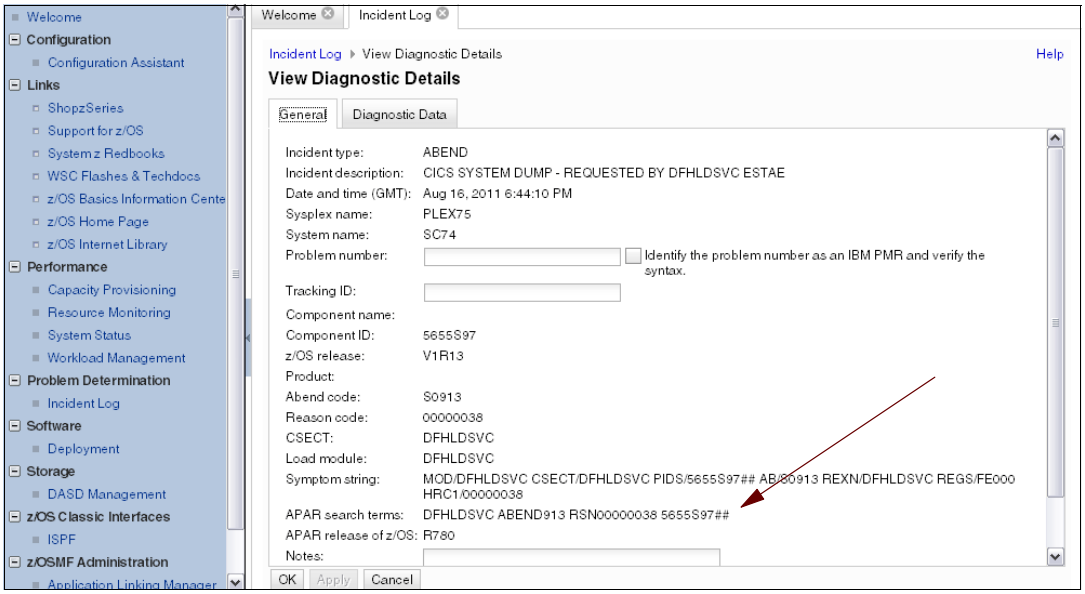


Figure 18-34 Attach retain search string to an incident

18.14.1 Installation considerations for the Incident Log

Over time, your sysplex dump directory can become full with the dumps you have saved. To create more space for dumps, you can delete old dumps from the directory. If you must retain the saved dumps, however, you can instead migrate your existing dumps to a larger sysplex dump directory.

To establish a larger sysplex dump directory, consider the following:

- Create a new sysplex dump directory data set through the BLSCDDIR CLIST, for example:

  ```
  EXEC 'SYS1.SBLSCLIO(BLSCDDIR)'
  'DSNAME(new.DDIR) VOLUME(volser) RECORDS(25000)'
  ```

  If your existing dump directory was created with the default size of 15000 records, you might want to specify a larger size. Approximately 50 directory entries are used for each incident and more are used for multisystem dumps.

- Use the **IPCS COPYDDIR** command to copy the old directory entries to the new directory data set:

  ```
  COPYDDIR INDSNAME(SYS1.DDIR) DSNAME(new.DDIR)
  ```

- Update BLSCUSER with the new dump directory name (but make note of the old dump directory name).
- Recycle the DUMPSRV address space (CANCEL DUMPSRV; it restarts automatically). This action registers the new dump directory name to DUMPSRV.
- Run BLSJPRMI (START BLSJPRMI). This action updates the in-storage copy of the dump directory name. Your new sysplex dump directory now contains the old dumps and can be used to store new dumps.

18.15 Resource Monitoring task

The Resource Monitoring task in z/OSMF provides a web-based user interface that you can use to monitor the performance of the z/OS sysplexes, AIX system complexes (such as IBM System p®), Linux system complexes (System z and System x), or Linux images (System z and System x) in your environment.

With the Resource Monitoring task, you can monitor most of the metrics supported by the IBM Resource Measurement Facility™ (RMF) Monitor III, create and save custom views of the metrics, and display real-time data as bar charts.

In z/OS V1R13, the names of the tasks provided through the Resource Monitoring plug-in are changed, as follows:

- The Monitoring Desktops task is now called the Resource Monitoring task.
- The Sysplex Status task is now called the System Status task.
- The Desktops have been renamed to Dashboards, as shown in Figure 18-35.

  You can use the Dashboards tab in the Resource Monitoring task to define new dashboards and to open or delete existing dashboards.

  The Dashboards tab is displayed by default and cannot be closed. It contains a table that lists the names of the dashboards that you have defined and the names of the dashboards that are supplied with z/OSMF by default. The table can contain a maximum of 100 dashboards.

Note: The predefined dashboards are provided so that you can quickly start monitoring the performance of the sysplexes and workloads in your installation.

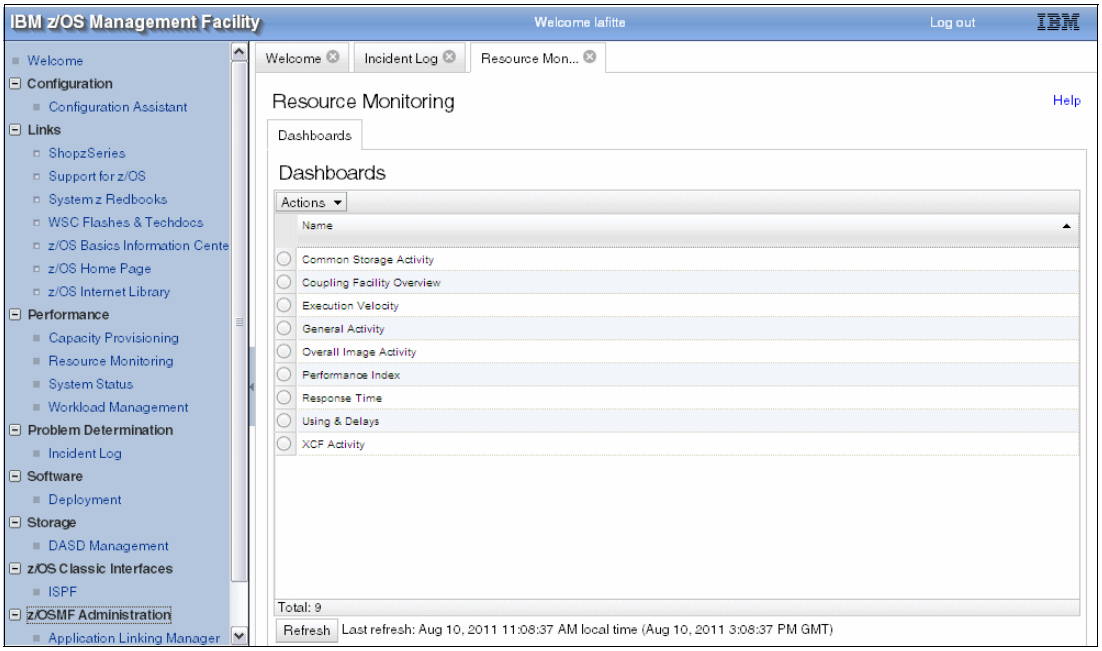


Figure 18-35 Resource Monitoring task

Resource Monitoring task

For z/OS sysplexes, the Resource Monitoring task takes its input from a single data server on one system in the sysplex. That data server collects data from the RMF Monitor III data

gatherer on each image in the sysplex. This function is called the Distributed Data Server (DDS).

With z/OS V1R13 and z/OSMF V1R13, RMF provides new CIM-based performance data gatherers for Linux on System z, Linux on System x, and AIX systems to provide a consistent monitoring solution for zEnterprise ensembles.

Along with the resource monitoring plug-in for the z/OSMF, first made available with z/OSMF V1R12, this is intended to display performance metrics from those platforms and combine them with z/OS metrics in common graphic views. For further details, see “Integrated ensemble performance monitoring” on page 736.

The Resource Monitoring task includes the following key functions:

- Create monitoring dashboards

  You can create monitoring dashboards or custom views that you can use to monitor the performance of the sysplexes, system complexes, or images in your environment.

- Save monitoring dashboards

  You can save monitoring dashboards. Doing so allows you to reuse the monitoring dashboard or template so that you can easily view performance data for your monitored sysplexes, system complexes, or images from the same angle.

- Work with multiple monitoring dashboards

  You can work with multiple monitoring dashboards simultaneously. To do so, open the dashboards with which you want to work in a new tab in the z/OSMF work area or in a new browser tab or window.

- Monitor multiple resources simultaneously

  You can collect data for multiple resources at the same time. To do so, associate the metrics in a dashboard with other resources.

- Create dashboards that are not associated with a specific sysplex

  Doing so streamlines the number of dashboards that you have to create because you can create one dashboard and use it for all of the sysplexes in your installation.

- Monitor the performance over time

  The Resource Monitoring task provides controls that you can use to browse through the samples that have been collected for the metric groups contained in a monitoring dashboard. Up to 100,000 samples are collected for a dashboard. To browse the samples, use the slider and the backward and forward arrows provided in each metric group.

18.16 Configuration Assistant for z/OS Communications Server

IBM provides a configuration Graphical User Interface (GUI) that you can use to generate configuration files for Communications Server for z/OS. Through a series of wizards and online help panels, you can use the Configuration Assistant to create configuration files for any number of z/OS images with any number of TCP/IP stacks per image.

The Configuration Assistant reduces configuration complexity by providing a consistent and easily manageable interface. It can dramatically reduce the amount of time required to generate and maintain policy files for Communications Server disciplines. The Configuration Assistant is intended to replace manual configuration of the policy disciplines, but it can also incorporate policy data directly from the Policy Agent.

z/OS V1R13 Configuration Assistant enhancements

Beginning in z/OS V1R13, Configuration Assistant makes it easier to manage a diverse configuration by supporting the configuration of multiple z/OS Communications Server releases. You no longer have to maintain multiple installations of z/OSMF to manage multiple releases.

In z/OS V1R13, Configuration Assistant supports both z/OS V1R13 and z/OS V1R12 configuration. You can select the release level for a new image and change the release level for an existing image.

In z/OS V1R13 Communications Server, the IBM Configuration Assistant for z/OS (Configuration Assistant) supports the following actions:

- z/OSMF System Authorization Facility (SAF) mode authorization and registration of the Configuration Assistant home page with the z/OSMF application linking function.
- You can use your security product instead of z/OSMF to authorize users.
- Other applications can link to z/OSMF applications.

In addition, you can perform the following tasks:

- Use password phrases when you are using FTP from the Configuration Assistant
- Delete backing-store files when you are running in z/OSMF mode
- Enable a default AT-TLS rule to support the RACF Remote Sharing Facility (RRSF)

Restriction: You can delete backing-store files from z/OSMF only; you cannot delete backing-store files from the Windows client.

18.16.1 Configuration Assistant primary panel

Configuration Assistant for z/OS Communications Server is a z/OSMF task that simplifies the configuration of the TCP/IP policy-based networking functions. The Configuration Assistant task provides centralized configuration of TCP/IP networking policies and can help reduce the amount of time required to create network configuration files. Figure 18-36 is the primary panel displayed when Configuration Assistant is selected.

Use this panel to access the technologies you want to configure and to create new z/OS images.

To configure a specific technology:

1. Select the technology you want to configure in the table shown in the figure.
2. Click **Configure**. This changes the main perspective of the console to the technology you selected.

Adding z/OS images

To add a new z/OS image, follow these steps:

1. Click **Add a New z/OS Image**. (In this example, two images were added, SC74 and SC75, as shown in Figure 18-36).
2. A panel is immediately shown. Respond as directed by the panel.

To work with a specific z/OS image:

1. Select the image in the navigation tree. This changes the console to show the settings specific to that image.

18.16.2 Support configuration for new IPSec enhancements

Beginning in V1R13, Configuration Assistant makes it easier to manage a diverse configuration by supporting the configuration of multiple z/OS Communications Server releases. You no longer have to maintain multiple installations of z/OSMF to manage multiple releases. In V1R13, Configuration Assistant supports both V1R13 and V1R12 configuration.

Select the **IPSec** button and then use the **Select Action** pull-down menu to **Configure**, as shown in Figure 18-36.

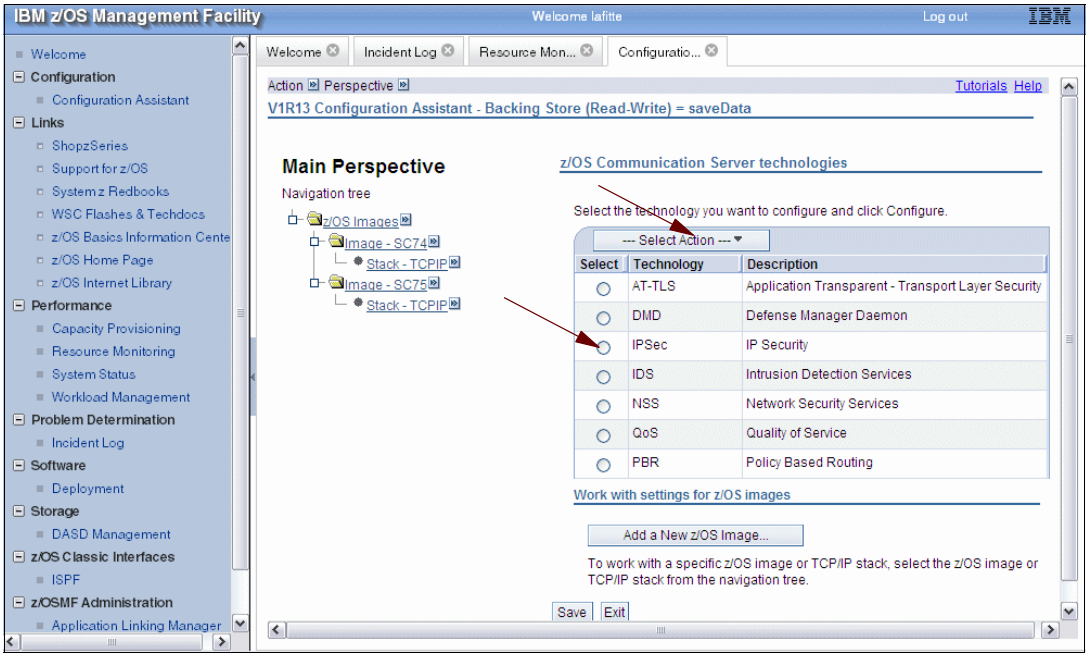


Figure 18-36 Configuration Assistant primary menu after defining images for the sysplex LPARs

z/OSMF IPSec enhancements

For each TCP/IP stack, configure the set of local IP addresses and assign names to each address by doing one of the following tasks:

- In the IPSec perspective, select a stack from the navigation tree. Click the **Local Addresses** tab. Click **Add** to add an IP address and assign a name to it.
- With z/OSMF, in the IPSec perspective, select a stack from the navigation tree.
  - Click the **Local Addresses** tab.
  - Select **Discover from the Select Action** table menu to query the TCP/IP stack and automatically populate the panel with the stack's local addresses.

IPSec is configured by creating a set of rules for each TCP/IP stack. Each stack can have a large number of rules, and many of these rules are exactly the same on each stack except for the local IP addresses. This can result in a signifcant number of rules to manage and if an update is required, it might need to be done for all stacks. Figure 18-37shows the IPSec Perspective.

Beginning in V1R13, the Configuration Assistant supports common configuration of multiple stacks. This introduces a new reusable object called *rules*. Reusable rules are created a single time and assigned to TCP/IP stacks. If a reusable rule needs to be updated, only a single rule needs to be modified to have the changes propagated to all stacks.

In certain cases, as with a DVIPA or distributed DVIPA, local IP addresses and IKE identities can be shared among multiple stacks. In other cases, local IP addresses and IKE identities can differ from stack to stack. Reusable rules can reference variable names for both local IP addresses and IKE identities, and these names can be assigned specific values for each stack.

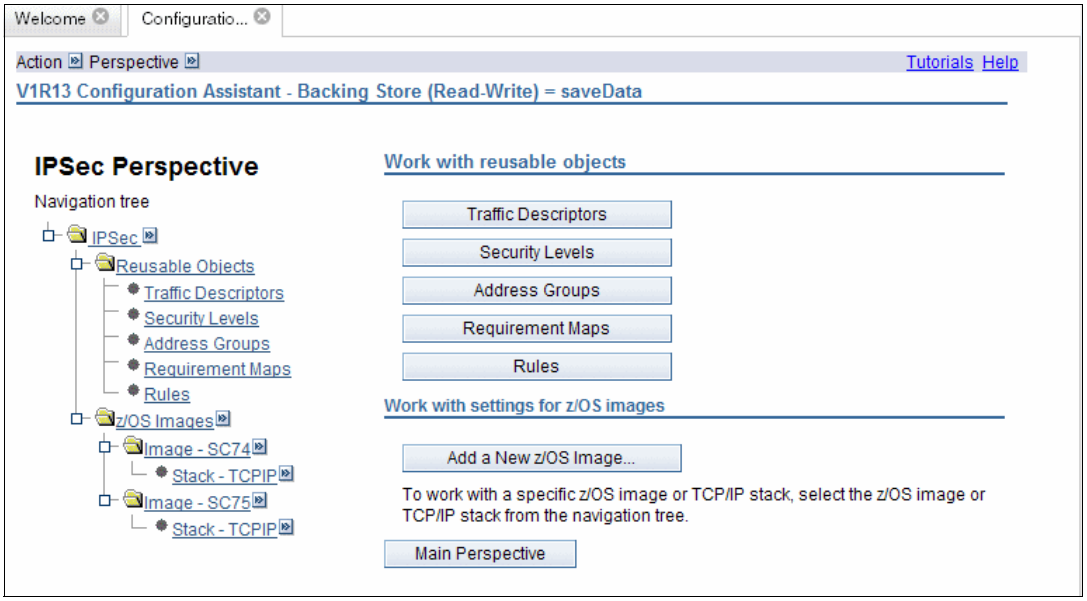


Figure 18-37 IPSec Perspective

18.16.3 Support for reusable rules for IP security

The IBM Configuration Assistant for z/OS Communications Server is enhanced to allow a policy rule to be defined a single time for multiple stacks. The IBM Configuration Assistant for z/OS Configuration Assistant with z/OS V1R13 makes it easier to create policy rules by discovering the local IP addresses for a TCP/IP stack and importing them into the Configuration Assistant. After you associate local addresses with the TCP/IP stack, you can use the addresses in IP address groups or in places where IP addresses are specified. By using the discovery function, you do not have to remember your IP addresses and manually enter them when you are creating rules or IP address groups.

Before z/OS V1R13 Communications Server, IP security administrators had to create a set of rules for each TCP/IP stack. Many of these rules were identical across stacks, and various settings were specific to the stack. An example of stack-specific settings is the settings for the stack's local addresses.

In z/OS V1R13 Communications Server, you can use the IBM Configuration Assistant for z/OS (Configuration Assistant) to create common configuration objects by using existing reusable configuration objects: address groups, traffic descriptors, security levels, and requirement maps, and new reusable rule configuration objects.

Reusable rules allow you to reduce the number of configuration tasks that apply to multiple TCP/IP stacks. You can create a single rule and assign it to multiple stacks. You define the rule only a single time and you modify it in a single location. To make the rule reusable across TCP/IP stacks, you can use symbols or names. IP security rules are now reusable because you can configure the local addresses as a name, rather than as a specific IP address. These names resolve to the correct IP address on each stack to which you assign the rule.

This permits more efficient policy configuration for multiple systems without having to individually define every policy rule for every stack. Configure rules are defined a single time and apply to existing or new stacks. This removes the need to “copy” and changes are automatically available to all stacks that use the rule.

This requires support for symbolic naming for IP Addresses and IKE identities.

Note: Reusable rules appear in a stack’s list of connectivity rules alongside stack-specific connectivity rules. You can change the order of reusable rules just like stack-specific connectivity rules.

To help you migrate from stack-specific rules to reusable rules, you can change a stack rule into a reusable rule. Select a stack-specific rule and click **Make Reusable**. A copy of the stack rule is added to the reusable rule set and the stack rule will remain in place, but is marked as derived from a reusable rule.

18.16.4 Support for configuration of multiple z/OS releases

In z/OS V1R13 Communications Server, you can use the IBM Configuration Assistant for z/OS Communications Server (Configuration Assistant) to configure multiple z/OS releases (V1R12 and V1R13). You can configure multiple LPARs that are running other releases by using a single instance of the Configuration Assistant.

Using Configuration Assistant to manage multiple releases

To use the Configuration Assistant to manage multiple z/OS Communications Server releases, perform the appropriate task as listed here:

- Specify the z/OS release level for images (LPARs). When you create a new image by using the Configuration Assistant, specify the z/OS release level on the “New z/OS Image” panel. The default release level is the current release.
- Change the z/OS release level for images (LPARs). To change the z/OS release level for an image by using the Configuration Assistant, click the image name in the navigation tree. Then select the z/OS release level from the drop-down list on the “Image Information” panel.

18.17 Getting started with z/OSMF V1R13

After z/OSMF has been set up and configured and started on a system, point your browser to the URL for the z/OSMF instance. This is basically the host name, the secure port, and the context root for z/OSMF. You reach the z/OSMF welcome page and login panel. Our location is listed here in bold:

```
https://your-ip-address:32208/zosmf/

https://wtsc74.itso.ibm.com:32208/zosmf/
```

The navigation panel is displayed on the left side of the panel. The login is at the top. The large center panel is where the tasks will open, as shown in Figure 18-38. The only function you can perform at this time is logging in.

To log in you need a valid z/OS user ID that has been defined and enabled to for z/OSMF and the WebSphere runtime environment. See A.1, “Setting up WebSphere Application Server

OEM” on page 766 for information about defining those values. z/OSMF can use RACF or equivalent security products.

18.17.1 Customizing the Welcome panel for guest users

Your installation can customize the z/OSMF Welcome panel with its own information for guest users. You might do so, for example, to provide users with tailored information or instructions specific to your company. You can even customize the Welcome panel with a small image or graphic, such as your company logo. After the user authenticates, the Welcome panel is replaced with the standard z/OSMF Welcome panel.

Note: In our case, we customized the Welcome panel using these steps:

1. We copied from `/usr/lpp/zosmf/V1R13/samples/customWelcome.properties` to `/var/zosmf/data/customWelcome.properties` (we used the default target location `/var/zosmf/data` when building z/OSMF).
2. We edited `/var/zosmf/data/customWelcome.properties`. We updated the last two lines, as shown:

   ```
   header=Welcome to z/OSMF on WTSC80
   footer=Brought to you by the ITSO
   ```

3. We copied the ITSO `logo .gif` file to `/var/zosmf/data/cutsomLogo.gif`, as shown at the bottom of Figure 18-38.

Regarding the size of the image, you do not need to know the size of the area for the image to fit, as explained here:

- If the image is smaller than the area, it will retain its original dimensions.
- If the image is larger than the area, it will be scaled down to fit in the area. Due to this scaling, the image will always “fit” although there can be distortion from this scaling.
- However, to avoid image distortion, clearly state that this area is 120 x 40 pixels.

Welcome window

Figure 18-38 on page 414 shows the Welcome window after being customized. The arrows point to the ITSO customization changes discussed.

Figure 18-38 Customized Welcome window (ITSO)

Logged-in user

After the user is logged in, the multiple categories and tasks under these categories can be viewed in the Navigation panel, as shown in Figure 18-39. z/OSMF supports role-based authorization, so the user sees only the tasks that they are authorized to. z/OSMF supports SAF-based authentication and the standard granular z/OS-level authorization.

Figure 18-39 z/OSMF list of functions

Getting started with z/OSMF

Installations can selectively configure tasks to use. To find information about using z/OSMF, click **Getting Started with z/OSMF**; see Figure 18-39.

Figure 18-40 is then displayed, showing the help information.

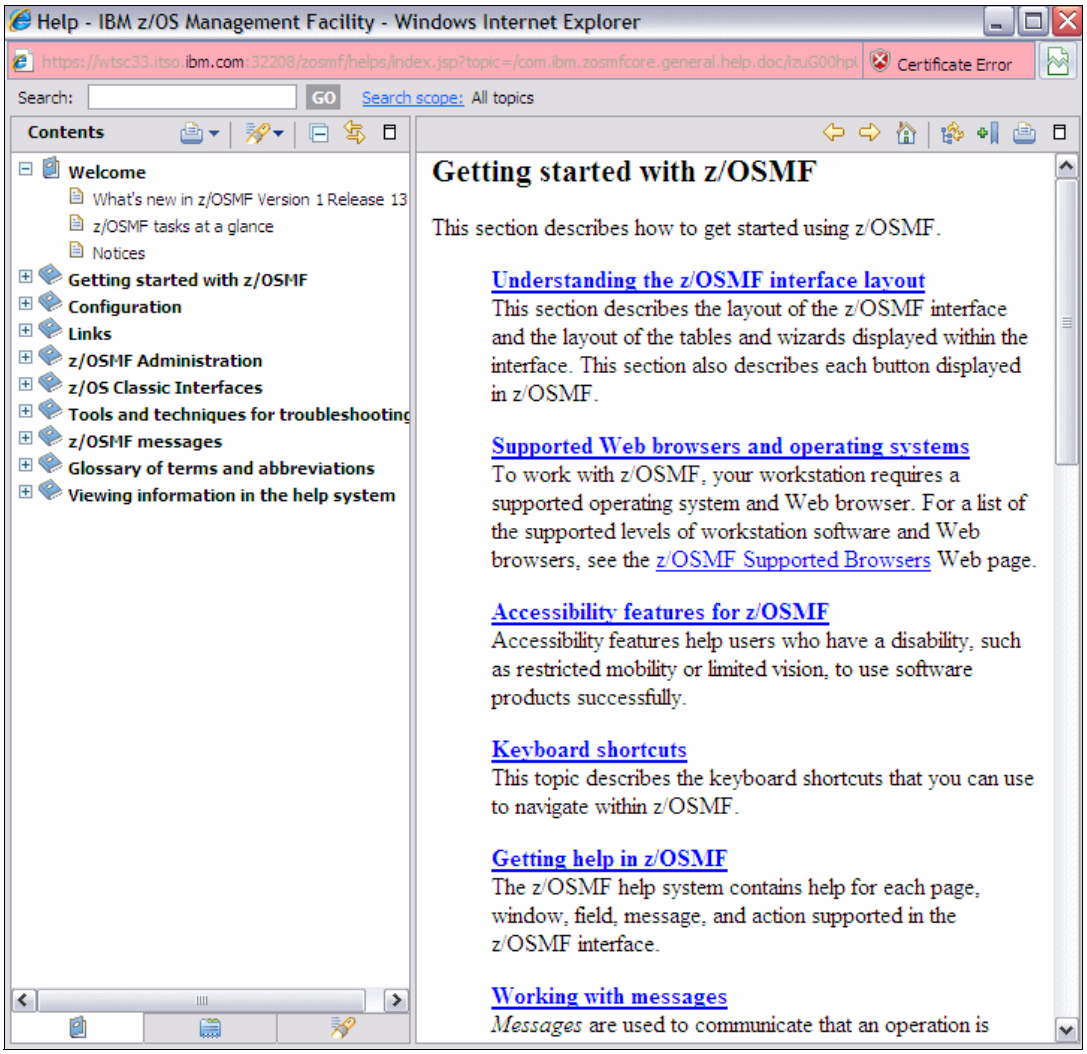


Figure 18-40 Getting started with z/OSMF help information

z/OSMF tasks at a glance

z/OSMF provides several tasks that you can use to manage various aspects of your z/OS systems, as explained here.

Figure 18-41 provides the name, description, and category for each z/OSMF task. If the task is new for the current release, its name is followed by the new icon (). You need to scroll down to see all of the tasks that are available.

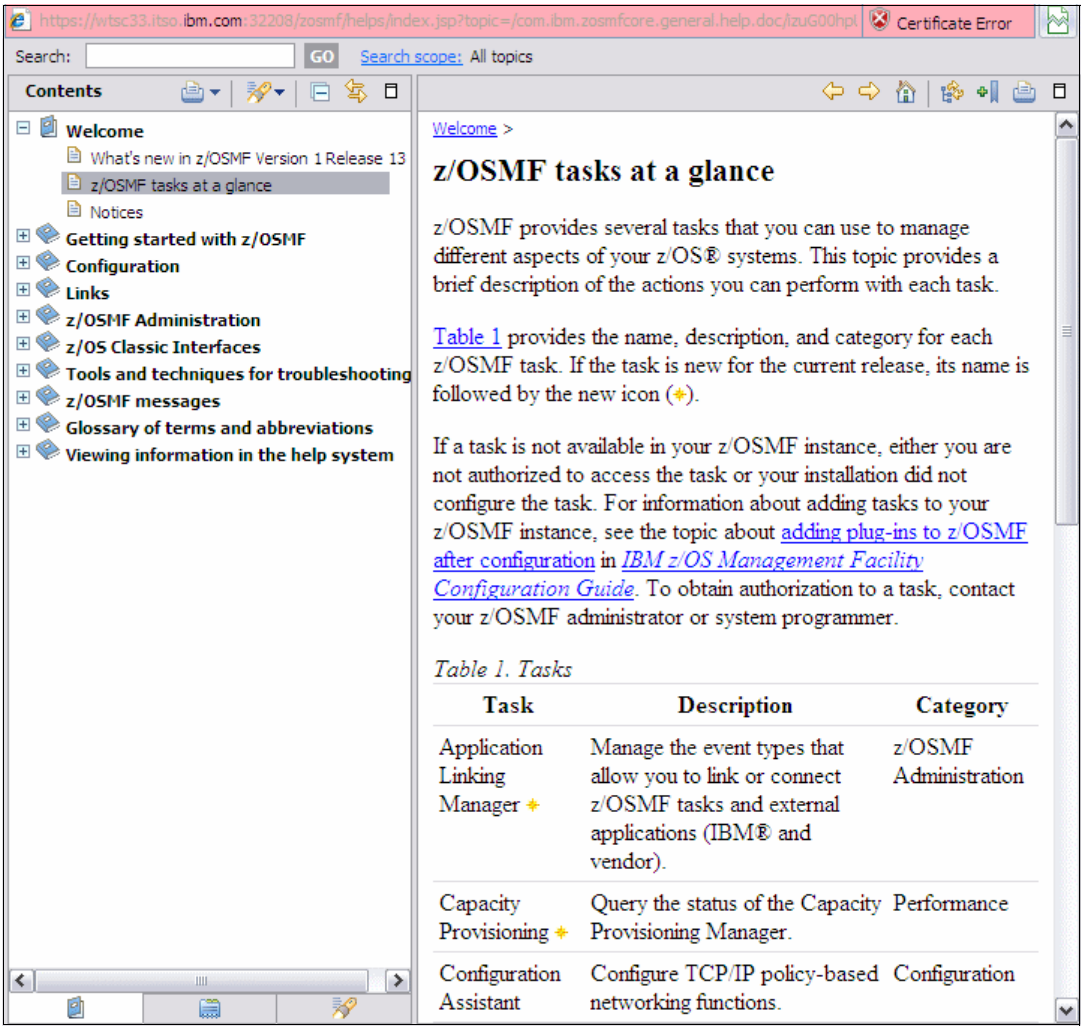


Figure 18-41 Tasks at a glance

If a task is not available in your z/OSMF instance, either you are not authorized to access the task or your installation did not configure the task.

For information about adding tasks to your z/OSMF instance, see the topic about adding plug-ins to z/OSMF after configuration in *z/OS IBM z/OS Management Facility Configuration Guide*, SA38-0652.

To obtain authorization to a task, contact your z/OSMF administrator or systems programmer.

New tasks and enhancements in z/OS V1R13

Figure 18-42 displays the new tasks and enhancements provided in z/OSMF Version 1 Release 13.

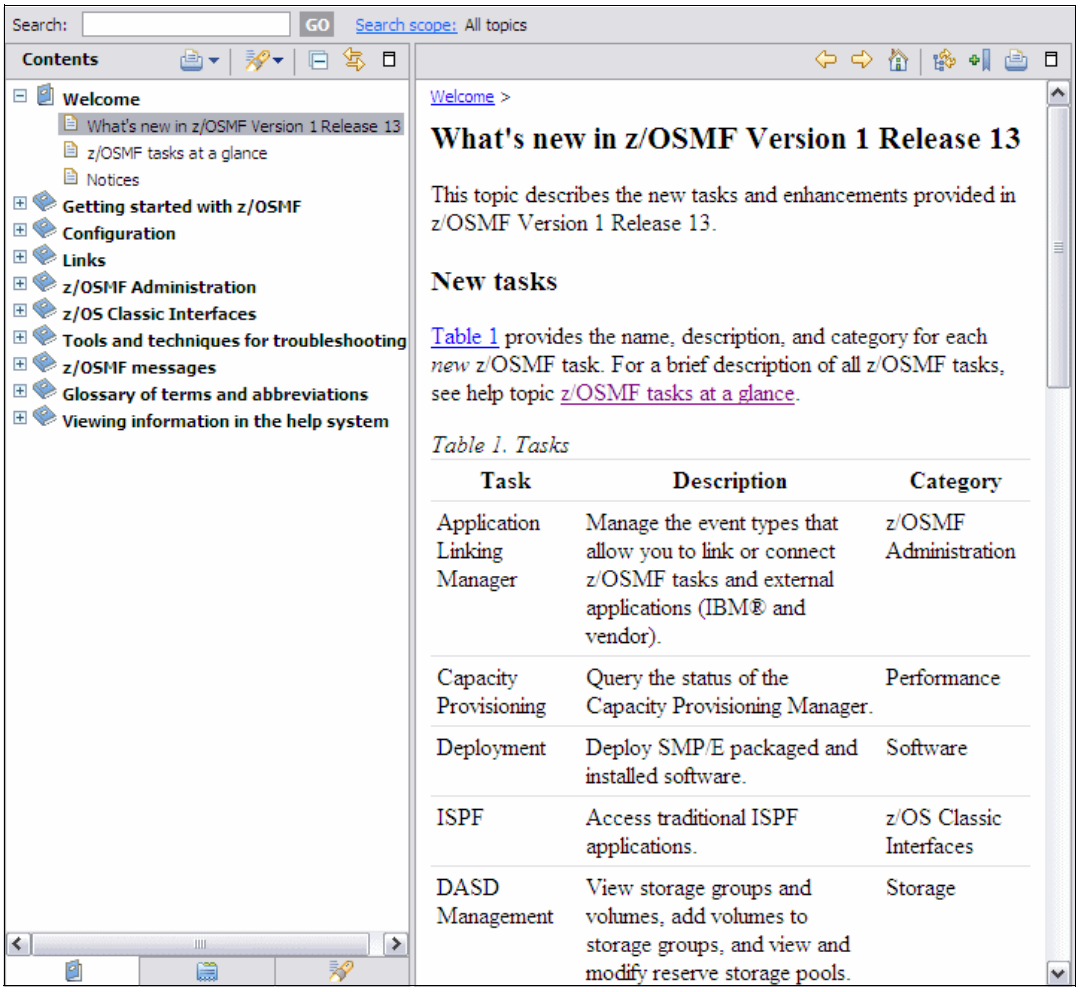


Figure 18-42 What's new in z/OSMF V1R13

As shown in the figure, the section labelled `Table 1. Tasks` provides the name, a brief description, and the category for each new z/OSMF task.

18.17.2 z/OSMF in a monoplex or sysplex

An instance of z/OSMF can manage only one local system or sysplex. Multiple users may log into the same instance of z/OSMF from other workstations or browsers. Based on client feedback, the expectation is to support up to 15 concurrent users that are simultaneously logged in and actively working and driving functions at the same time. However, many more can be authorized to work with z/OSMF.

Although only one z/OSMF instance can be active in a sysplex at any time using the same configuration and the same z/OSMF data repository, an additional instance may be created (for example, for test or service update or backup).

However, it is not to be actively managing the systems at the same time (for example, working on the same incident concurrently from two separate instances of z/OSMF) or using the same data repository. This is enforced by z/OS through global enqueues.

If you have multiple sysplexes, you will want to manage all those sysplexes. Users can manage additional sysplexes in their enterprise from a single client system, by opening new browser windows (or tabs) and logging into the z/OSMF instance installed on those sysplexes (one browser per system or sysplex).

Requirement: A z/OSMF instance must be active on every sysplex that a user wants to manage through this interface.

19

z/OS Hardware Configuration Definition and Hardware Configuration Manager

With Hardware Configuration Definition (HCD), you perform the hardware and software I/O configuration processes through a single, interactive user interface. As you enter data, HCD performs validation checking to help avoid data entry errors before you attempt to use the I/O configuration.

The output of HCD is an I/O definition file (IODF), which contains I/O configuration data. An IODF can define multiple hardware and software configurations to the z/OS operating system. When you activate an IODF, HCD defines the I/O configuration to the channel subsystem and the operating system, or both.

Hardware Configuration Manager (HCM), an optional feature of z/OS, works with HCD to provide a set of graphical and text configuration reports. HCM complements HCD by adding information about the physical properties of the processors and I/O configurations to the logical information that HCD provides. For example, the physical information about processors and I/O configurations includes details about the IBM ESCON® and FICON infrastructure such as cables, distribution panels, and patch panels.

This chapter describes the following enhancements provided with HCD and HCM in z/OS V1R13:

- Support for new hardware
- Support of z/VM V6R2
- Support of Windows 7
- Quality improvements

19.1 HCD and HCM enhancements with z/OS V1R13

The z/OS V1R13 enhancements to HCD and HCM improve quality and ease of use. This is achieved by providing extended information in dialogs or reports, introducing additional checks to warn users of definition errors, enhancing the performance and productivity of definitions in HCM, and improving the detection and repair capability of corrupted IODFs.

The enhancements are as follows:

- HCD support for new hardware
- Information about used and available unit addresses and control units on a CHPID
- Emphasize CF partition in CF Channel Path Connectivity List/Report
- Introduce PPRC usage type NONSYSPLEX
- Check of PPRC secondary device
- Warning with device subchannel set mismatch
- IODF consistency check when building a production IODF
- Message improvements
- Export/import of unconnected control units and devices
- Flag tasks that require a prerequisite product to be available
- Warning when changing local system name
- Reject CF CIB connections between two z9 processors
- Warning if more than 4 CIB CHPIDs are defined on the same AID port
- Performance improvement in HCM when working with CHPIDs
- Rework of multiple control unit connect dialog in HCM

19.2 HCD support for new hardware

This section introduces the HCD support for zEnterprise CPCs (z196 and z114):

- Changes to the z196 machine
- Support of the z114 machine
- Support of four AID ports
- Support of more than 7 subchannels for a CIB connection
- Support of IQDX communication

In z/OS V1R13, support for the z196 is upgraded and new support for the MR is provided. Therefore, you can define the I/O configuration for the zEnterprise 196 and for the MR.

Table 19-1 lists the HCD z196 support ** model changes.

*Table 19-1 HCD z196 support ** model changes*

	z196	**z114**
Type and models	2817-M15/M32/M49/M66/M80	2818-M05/M10
Number of channel subsystems/partitions	4/60	2/30
Number of subchannel sets	3	2
Number of ports/AID	4	4

	z196	z114
Number of ChPIDs - FICON - ESCON - OSA - Coupling - HiperSockets	1024 368 360 48 128 32	512 160 240 48 128 32
IQDX support	Yes	Yes
Number of subchannels per CIB connection	32	32

As shown in the table, zEnterprise 196 HE GA-2 differs from the GA-1 support in the following areas:

- The maximum number of FICON CHPIDs is increased from 336 to 368.
- There is support of AID PORT values 1 to 4 instead of 1 to 2.
- There is support for 32 subchannels (devices) per coupling CIB CHPID for InfiniBand.
- There is support for IQDX communication.

The IBM zEnterprise 114 consists of the following elements:

- Machine type 2818 with two models, namely M05 and M10
- Two subchannel sets
- 2 CSSs/30 LPs
- Max PCHID value: 2FF
- Max AID value: 0B
- Max AIDs: 8 (not checked in HCD)
- Allowed PORT values: 1 - 4
- Max number of CHPIDs: 512
- Max number of ESCON CHPIDs: 240
- Max number of FICON CHPIDs: 160
- Max number of FCP CHPIDs: 160
- All other CHPID maximums are the same as the z196 model
- Support for 32 subchannels (devices) per coupling CIB CHPID
- Support for IQDX communication
- CF connect dialog per CIB CF link

This processor support is transparent to HCM.

When creating a processor definition, HCD and HCM define the maximum channel subsystems, the maximum number of devices in all possible subchannel sets, and the maximum number of partitions as reserved partitions.

HCD and HCM allow you to perform changes on the maximum configuration. At Build Production IODF time, it is again assured that the maximum configuration is defined.

19.2.1 Support of 32 subchannels per CIB CF link

Licensed Internal Code (LIC) at driver level 93 will exploit more than 7 subchannels per CHPID on long range HCA (InfiniBand) adapters. Because this is unique firmware that will only be on IBM zEnterprise 196, this exploitation is only on a z196 - z196 long range connection. In all other cases, only 7 subchannels per CHPIDs can be used.

HCA2-0 LR and HCA3-0 LR (long reach) features can exploit 32 subchannels, providing better performance for long-range CIB connections between two zEnterprise CPCs.

When connecting CIB CHPIDs between two zEnterprise CPCs, HCD and HCM will present a default number of 32 devices for the connection. It allows overwriting the default value to 7, and it accepts only 32 or 7.

HCD in z/OS V1R13 provides help information for the number of devices that say that a value of 7 is to be used for shorter distances, and a value of 32 is to be used for greater distances. It adds a warning message if both ends of the CIB connection will not specify the same number of devices:

```
CBDG544I CIB CF connection between channel paths @1 and @2 does not specify the
same number of devices on both sides.
```

The number of devices can be only changed by breaking the CIB connection and reconnecting it with the appropriate value.

Note the following points:

- When CIB connections involve any other machine family (z9 or z10), the number of devices is fixed at 7, and there is no override allowed.
- When any other type of CF peer connections (that is, CFP or ICP) are made, regardless of machine type, the number of devices is fixed at 7, and there is no override allowed.

CF channel path connectivity

The panel “Add CF Control Unit and Devices” now displays `Number of Devices` as an input/output field; see Figure 19-1. Users specifying a CF Link through a CIB channel path between two zEnterprise CPCs will be able to specify the number of devices used on panel CBDPCF10.

The `Number of devices` field on CBDPCF10 will be changed to an input/output field. On entry, the value shown here is set by default:

- For timing-only (STP) connections (earlier or peer), the value zero (0) is displayed and cannot be modified.
- For other, earlier connections, the value is set to 2 (only the sender side is be shown) which cannot be changed.
- For other peer connections, the value is set to 7 if the processors do not both have the extended CF support of more than 7 devices for the selected channel path type. The value cannot be changed in that case.

  If both processors do have the extended support for the channel path type, then value 32 is presented and can be changed to 7.

```
                    Add CF Control Unit and Devices
CBDPCF10

Confirm or revise the CF control unit number and device numbers
for the CF control unit and devices to be defined.

Processor ID . . . . . . . : GRYPH1
Channel subsystem ID . . . : 0
Channel path ID  . . . . . : F3          Operation mode . . : SHR
Channel path type  . . . . : CIB

Control unit number  . . . . FFF4  +

Device number  . . . . . . . FEEF
Number of devices  . . . . . 32
```

Figure 19-1 New HCD Add CF panel

As shown in Figure 19-2, the CF Channel Path Connectivity List now has a new column -#-Dev, which shows the number of defined message devices for each CF connection.

```
  Goto  Filter  Backup  Query  Help
------------------------------------------------------------------------------
CBDPCFF0            CF Channel Path Connectivity List               Row 1 of 4
Command ===> ____________________________________ Scroll ===> PAGE

Select one or more channel paths, then press Enter.

Source processor ID . . . . . : GRYPH1
Source channel subsystem ID . : 0
Source partition name . . . . : *

  -------Source--------   -----------Destination--------   -CU-  -#-
/ CHPID  Type  Mode Occ    Proc.CSSID   CHPID  Type  Mode   Type Dev
_ F0     CIB   SHR  N      GRYPH1.0     F0     CIB   DED    CFP  7
_ F1     CIB   SHR  N      GRYPH2.0     F1     CIB   DED    CFP  32
_ F2     CIB   SHR  N       P2098.0      F2     CIB   DED    CFP  7
_ F3     CIB   SHR  N      G30.0        FA     CIB   SPAN   STP
****************************** Bottom of data ********************************



 F1=Help       F2=Split      F3=Exit       F4=Prompt     F5=Reset      F7=Backward
 F8=Forward    F9=Swap       F10=Actions   F12=Cancel    F13=Instruct  F22=Command
```

Figure 19-2 CF Channel Path Connectivity List

The number of used devices for a connection is shown in the CF Channel Path Connectivity List. If each side of the CF connection has defined a different number of devices, XCF uses the lower number of devices for the connection.

Migration

For the migration function, HCD extends the TPATH keyword of the CHPID I/O control statement by an optional fifth parameter per CHPID that specifies the number of devices defined for a connection; see Figure 19-3. If this parameter is not specified, the default value is applied.

```
TPATH=((proc,chpid[,CFP CU[,CFP device[,nodevs]]]),(proc,chpid[,CFP CU[,CFP
device[,nodevs]]]))
```

Figure 19-3 TPATH keyword

The TPATH operand of the CHPID statement allows the optional specification of the number of devices for a CIB connection. The value of attribute hcdConnChannelPath of LDAP class hcdChannelPath is extended by the specification of the number of devices; see Figure 19-4.

```
dn:hcdChannelPathID=FE,hcdProcessorConfigID=GRYPHON1,
   hcdiodfid=weid.IODF00.LDAP,cn=HCD
   changetype:modify
   replace:x
   hcdConnChannelPath:GRYPHON2.0.FF.FF04.FF0C.32
   hcdDescription:CIB CF connect to FF on 2817processors
```

Figure 19-4 LDAP class hcdChannelPath

The number of devices defined for a CF CIB connection is only written with the TPATH statement if both sides of the CF connection support more than 7 CF devices, regardless of whether 7 devices or 32 devices are defined.

In all other cases (for other CF channel path types, or if at least one of the processors involved does not support more than 7 CF devices on CIB channels), the TPATH statement is built as before without specifying the number of devices.

As a result, for a search on channel path information, the number of defined CF devices is only written for processor connections where both sides support more than 7 CF devices.

Note: The zEnterprise is a heterogeneous hardware infrastructure that consists of a zEnterprise CEC and an attached IBM zEnterprise BladeCenter® Extension (zBX) managed as a single logical virtualized system (ensemble) by the zEnterprise Unified Resource Manager.

z/VM enables access to the private IBM zEnterprise System intraensemble data network (IEDN) and the intranode management network (INMN) through both its real and virtual networking capabilities. As a result, the deployment and management lof z/VM network topology is fully integrated into the zEnterprise System ensemble environment.

19.2.2 Internal Queued Direct I/O Extensions for IEDN support

This section explains the new use of IQDIO known as Internal Queued Direct I/O Extensions for IEDN (IQDX).

zEnterprise CPCs created a new internal network for intraensemble communications known as the intraensemble data network (IEDN). The primary purpose of the IEDN was to provide a dedicated and secure communications path from System z to the zBX. However, the IEDN

spans the entire ensemble. The HMC Unified Resource Manager provides the access controls, virtualization, and management functions necessary to secure and manage the IEDN.

Within a System z, the existing CPC internal network called HiperSockets or Internal Queued Direct I/O (IQDIO) was excluded from zManager-related functions and is not part of the IEDN or ensemble management.

zEnterprise 196 with driver 93 is enhanced to extend the IQDIO functions to the IEDN and zManager functions by allowing the two functions to be used together. This new use of IQDIO is referred to as Internal Queued Direct I/O Extensions for IEDN (IQDX). The IQDX function is similar to the functions provided by OSA configured as an OSX CHPID. There are several dissimilarities, however.

IQDX is a new channel parameter. It has an OSX-like interface but can exchange data between two LPARs in the same CEC by a simple data move, similar to HiperSockets; see Figure 19-5.

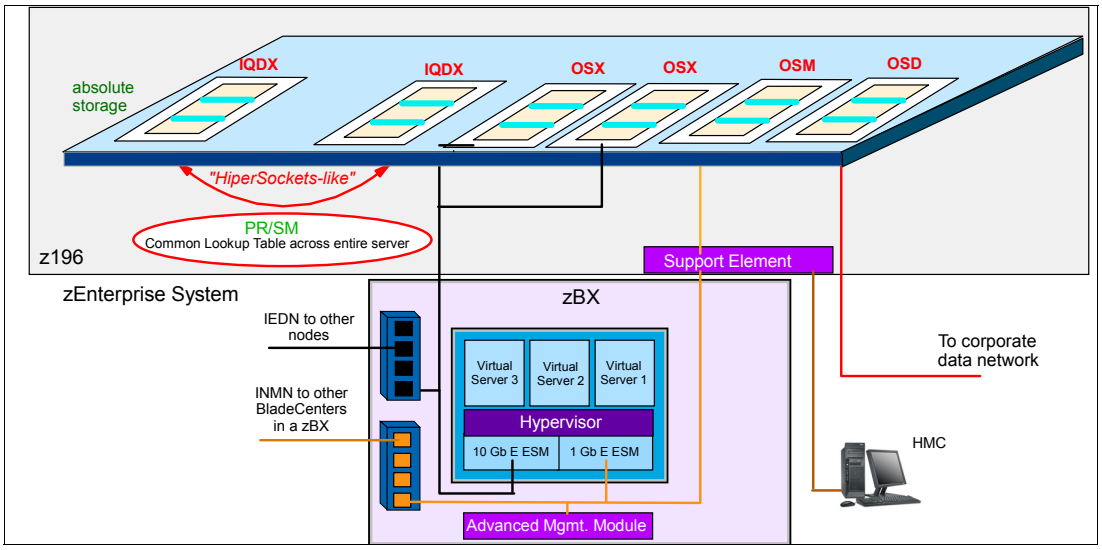


Figure 19-5 IQDX in a zEnterprise System

The new IQDX CHPID type can be accessed in two ways:

- It can be accessed by a z/VM environment (shown as LPAR1 in Figure 19-6) that provides, through its z/VM Virtual Switch, a transparent IQD interface to its virtual machines, mostly running zLinux.

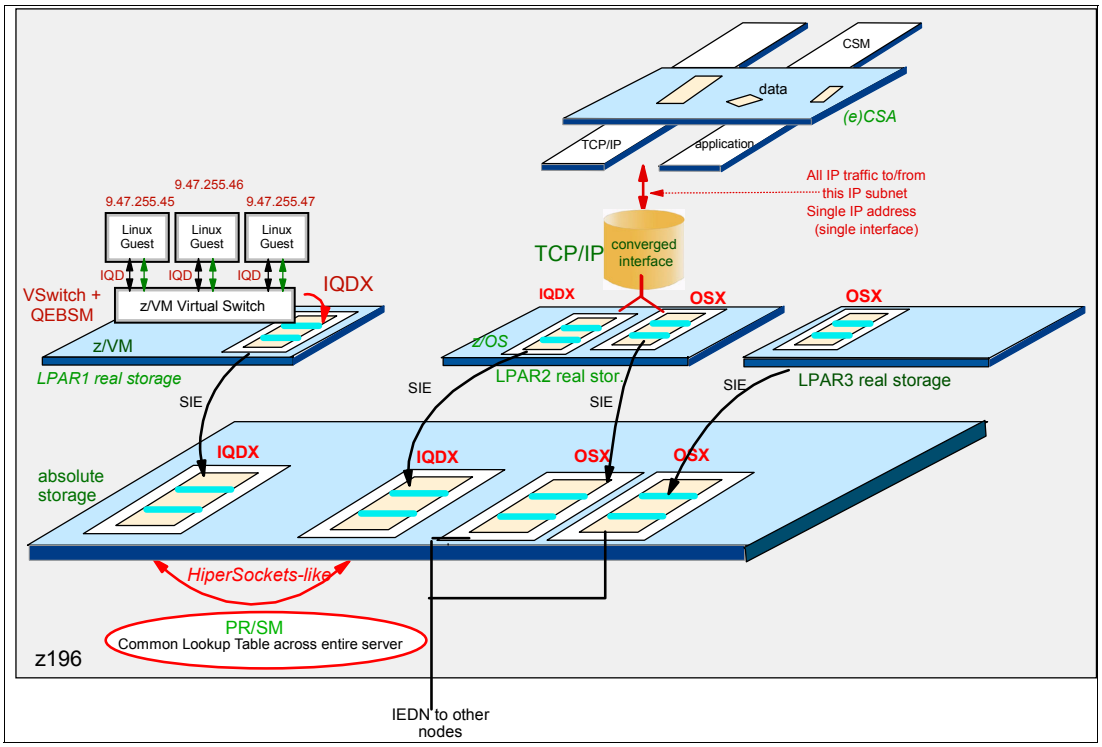


Figure 19-6 Converged IQDX link

- Or it can be accessed from a z/OS environment that is using it in a native way.

  A converged interface, provided within the TCP/IP stack, re-directs from a single IP address onto either:

  - An IQD interface (if the traffic is with a nearby LPAR connected with HiperSockets, (IQD)
  - Or an OSX interface, if the traffic is with an LPAR (either local or remote) along the IEDN network of the zEnterprise System.

There are two distinct approaches or system methods to provide the ability to connect or extend HiperSockets to the IEDN. The two methods allow an operating system running in an LPAR to access both HiperSocket and the external IEDN networks through a single host IP interface.

Initially (the first deliverable) the method used to provide this single interface solution is based on the environment of the virtual server (z/VM guest versus native LPAR) where each environment has unique objectives (transparent versus non-transparent mode):

z/VM Virtual Guest Transparent mode, in which the operating system is IQDX-unaware. This is a classic HiperSocket exploitation with no operating system awareness whatsoever, or software changes required where HiperSockets can be exploited “as is” and is can be transparently bridged to the IEDN through the z/VM Virtual Switch bridge support. Initially, only z/VM Linux guests using QEBSM are eligible to exploit this support.

Native LPAR Virtual Guest Non-Transparent mode, in which the operating system is IQDX-aware. The operating system provides a native operating system-unique solution where IQDX and OSX interfaces are logically “converged” or combined into a single host interface

whereby the communications stack (upper layers) only sees the single (OSX) network interface with a single IP address (initially, only z/OS will implement this).

For defining IQDX in HCD, the IQD channel path type specifies new channel parameter (CHPARM) values:

00 Basic HiperSockets function

02 IQD for IEDN (IQDX)

04 IQD bridged to an external (OSD) network

The parameter values go with all possible maximum frame sizes and result in the following CHPARM bits:

00, 40, 80, C0 Basic HiperSockets function with 16 K, 24 K, 40 K, 64 K maximum frame size

02, 42, 82, C2 IQDX function with 16 K, 24 K, 40 K, 64 K maximum frame size

04, 44, 84, C4 Bridged function with 16 K, 24 K, 40 K, 64 K maximum frame size

In the HCD dialog, the “Specify Maximum Frame Size” panel is changed to “Specify IQD Channel Parameters”; see Figure 19-7.

```
            Specify IQD Channel Parameters

Specify or revise the values below.

Maximum frame size in KB . . . . . 16  +

IQD function . . . . . .  1   1. Basic HiperSockets
                                2. IEDN Access (IQDX)
                                3. External Bridge

 F1=Help      F2=Split    F3=Exit     F4=Prompt
 F5=Reset     F9=Swap   F12=Cancel
```

Figure 19-7 IQD channel parameters

For the definition of an IQD channel path on a z196 processor, an IQD function other than the default function Basic HiperSockets can be selected.

Unused unit addresses

The task of defining unit addresses and control units to a CHPID is done by the system programmer. The number of devices (unit addresses) and control units on a CHPID are limited. However, it is difficult to obtain the number of unused resources on a CHPID before the maximum values are exceeded.

With z/OS V1R13, HCD provides a new action list “View unused resources” on the Channel Path List. It shows a new message containing the number of used and available unit addresses and Us for the selected CHPID. The maximum values are defined in the processor support module.

This provides the number of recognized unit addresses and control units in relation to the maximum allowed value. The new action code “u” or “U” is implemented as a group action.

The following informational message is added for each selected CHPID to give users the ability to obtain the number of free resources on a channel path:

```
CBDA377I 'For channel path chpid of processor proc, currently act_ua_num of
max_ua_num available unit addresses and act_cu_num of max_cu_num available
control units are used.'
```

Note: If specified for a system that does not support IQDX functions, the following error message is issued:

```
CBDA205I Specified IQD function is not supported by channel path xx of
processor procid.
```

The Channel Path Summary Report is changed so that the IQD function is put as an indication directly behind the maximum frame size value; see Figure 19-8.

```
                                                    --SWITCH---
      PCHID       MFS DIS        I/O      DYN                        PARTITION NUMB
CHPID AID/P TYPE KB  QP  MNGD CLUSTER   ID  ID PORT  MODE |1|2|3|4|5|6|7|8|9|A|
_____ _____ ____ ___ ___ ____ ________  ___ __ ____  ____  ____________________
 00         FC             NO          10           DED   | |A| | | | | | | | |
 01         IQD  16        NO                       DED   | | | |A| | | | | | |
 02         IQD  24X       NO                       DED   |A| | | | | | | | | |
 03         IQD  40B       NO                       DED   |A| | | | | | | | | |
 07   110   CFP            NO                       SPAN  | | | |A| | | | | | |


Blank: indicates the HiperSockets function
X:     indicates IEDN Support (IQDX)
B:     indicates Bridge Support
```

Figure 19-8 Channel Path Summary Report

The Channel Path Compare Report includes the comparison of the IQD function when comparing two IQD CHPIDs.

19.2.3 Support of IQDX in HCM

In z/OS V1R13, the following changes are introduced to HCM to support IQDX.

HCM dialog: Create CHPIDs

The title of the “Edit Maximum Frame Size” dialog is changed from “Edit Maximum Frame Size” to “Edit IQD Channel Parameters”; see Figure 19-9.

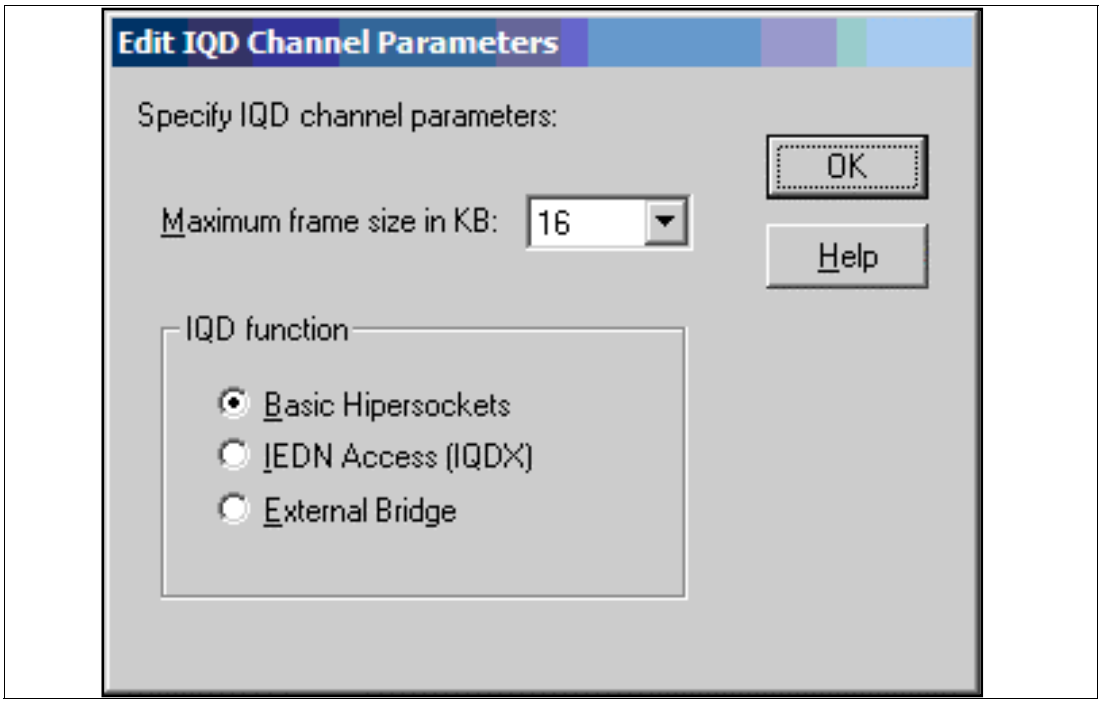


Figure 19-9 Edit IQD channel parameters

Additionally, a group of radio buttons is displayed to select an IQD function.

If HCM connects to a HCD version that does not support IQDx CHPIDs, the group of radio buttons will be disabled.

HCM dialog: Edit CHPID

For channel paths of type IQD, the “Edit CHPID” dialog is enhanced with a group of radio buttons to select an IQD function; see Figure 19-10. The dialog for all other channel types (for example, OSD with “more than 160 TCP/IP stacks” is unchanged.

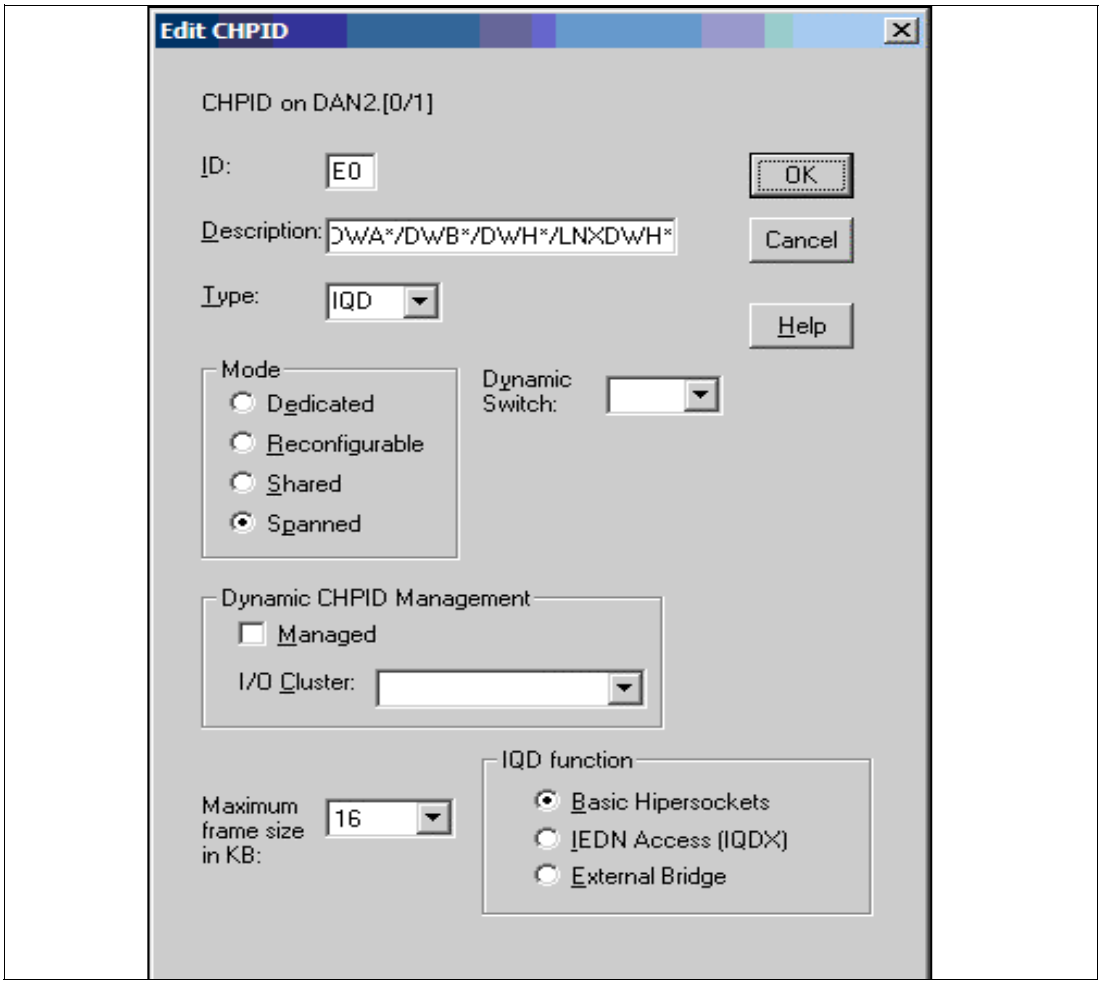


Figure 19-10 Edit CHPID dialog

If HCM connects to a HCD version that does not support IQDx CHPIDs, the group of radio buttons is disabled.

HCM dialog: CHPIDs

An additional column is introduced displaying the value of the IQD function; see Figure 19-11.

- Basic HiperSockets
- IEDN Access (IQDX)
- External Bridge

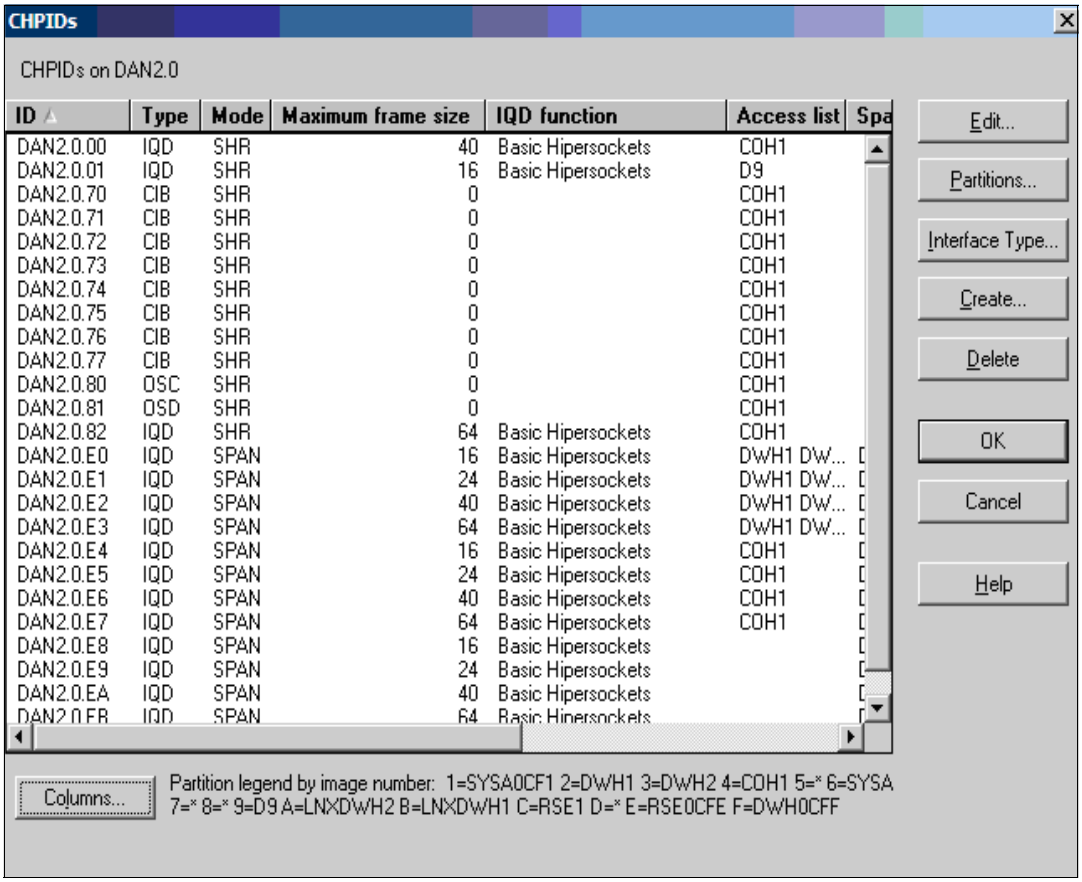


Figure 19-11 CHPID table

For non-IQD channel types, a blank is displayed.

Depending on the HCM `eeqhcm.ini` file, the new column is initially hidden and can be displayed by selecting the **Columns** button.

Coexistence

Note the following coexistence considerations regarding these enhancements:

- Exploitation support of z196 GA2 functions is rolled back to z/OS R11 and R12 through HCD SPE OA32576 and HCM SPE IO13473.
- Coexistence support for the z196 HE and MR processor definition without exploitation of IQDX functions is rolled back to z/OS R7 to z/OS R10 (OA35390 for z/OS R7 to R9, and OA32576 for z/OS R10).
- All z/OS versions of HCD to be used must be on the z196 GA2 level to work with the z196 GA2 functions.
- When a back-level HCM accesses an IQD channel definition with IQDX function, HCD will reject that request with error message CBDA609I `Channel path xx of processor proc1 uses IQDX functions which are not supported by the used HCM version`.
- HCM with z196 GA2 support does not accept the definition of IQDX functions when connected to a back-level HCD.
- When opening an HCM configuration file containing IQDX function definitions, a back-level HCM is forced into read-only mode.

19.3 Support of z/VM V6R2

HCD and HCM will ship new releases in z/VM V6R2 which include the functions of z/OS V1R13. In addition, HCD will support z/VM Single System Image (Sysplex Support) in z/VM V6R2 through a new parameter in the device support of z/VM.

New functions that are supported in HCD and HCM are:

- Support of configuration packages
- Support of IPv6

19.3.1 Support of configuration packages in z/VM

Configuration packages are supported in z/VM HCD through a new REXX utility, CBDSDPKG. The utility allows the HCD user to specify a source production IODF, a new target production IODF, target user ID and node ID, and control files for processor, operating system and switch configurations that are to be extracted from the source IODF and packaged in the target IODF. The target IODF is then sent to the specified node/user ID. This function does not require the definition of a configuration package in the IODF.

Support of configuration packages in HCM

In HCM, configuration packages for z/VM are supported through the Edit Configuration Packages dialog. One or more configuration packages can be defined in the IODF. They allow the generation of subset production IODFs that are sent to a target node/user ID on an attended z/OS or z/VM system, or to an unattended z/OS system.

In the Create Configuration Packages dialog, shown in Figure 19-12, you must specify the target IODF name, at a minimum.

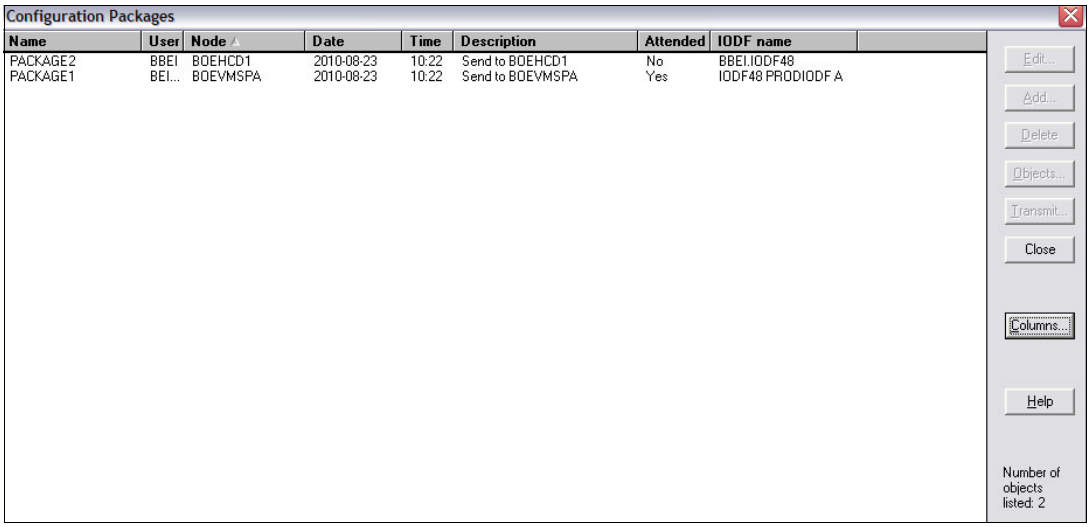


Figure 19-12 Configuration Packages dialog

This Configuration Packages dialog lists the existing definitions and provides Add, Edit, Delete, and Transmit actions.

HCM accepts both MVS and z/VM IODF names because it might be determined later whether the IODF is sent to a z/VM system or to an attended or unattended z/OS system. If a file type

of PRODIODF is given, the syntax is validated against z/VM IODF name conventions. Otherwise, the syntax is validated against MVS IODF name rules.

The HCM dialog for configuration packages is the same for a target z/VM and z/OS system, independent of whether HCD is running on z/OS or z/VM. Figure 19-13 shows the Edit Configuration Package panel. To edit a defined configuration package, select the package and press the Edit button in the Configuration Package dialog to display the Edit Configuration Package dialog which is similar to the Create Configuration Package dialog. You can edit all package attributes except for the package name.

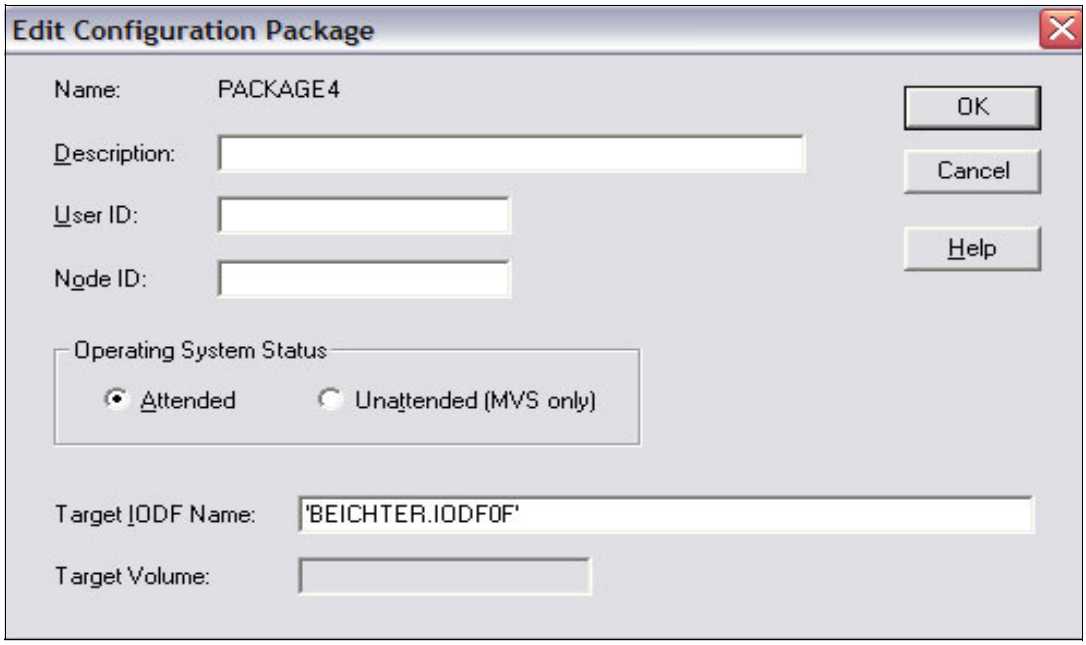


Figure 19-13 Edit Configuration Package dialog

In z/VM, the IODF name (IODFxx) is used for building the production IODF that is sent to the target system. Therefore, make sure there is no IODF `iodfxx WORKIODF` existing on the A-disk. Otherwise, the package IODF cannot be built during the transmit task.

User ID and Node ID are optional fields that have to be specified, at the latest, when the transmit task is performed. The default target is an attended system (MVS or z/VM). A target MVS system (but not a target z/VM system) might be unattended, which means that you can receive the target IODF on the target system through an import JCL. If the target system is attended, the target IODF is received at the target system in the spool reader, from where it has to be explicitly received.

The transmit task can only be entered from the source production IODF, which must have the configuration package defined. Figure 19-14 shows the Transmit Configuration Package panel.

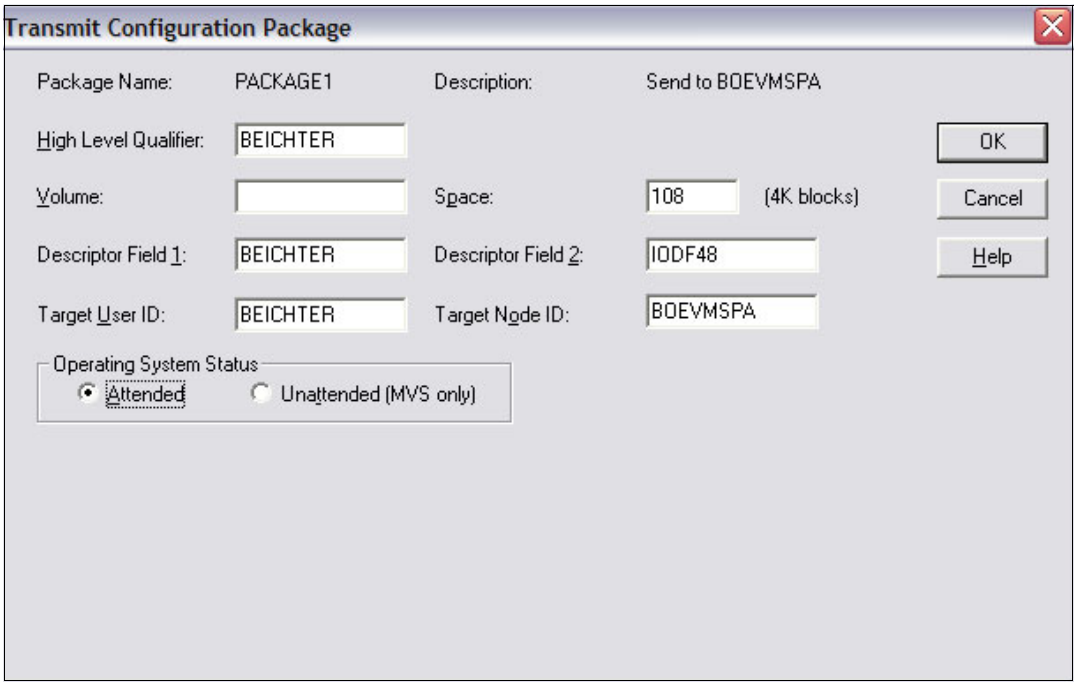


Figure 19-14 Transmit Configuration Package

The `High Level Qualifier` field shows the user ID. Volume is only applicable when sending the IODF to an unattended MVS system; otherwise, it is ignored. Space shows the size of the IODF in 4 K blocks. This is used to allocate the new target IODF. Descriptor Field 1 contains, as a default, the user ID. Descriptor Field 2 contains the IODF name of the target IODF.

When transmitting the package IODF to an unattended MVS system, you must provide the Import JCL for the job that runs on the target MVS system to import the IODF there.

19.3.2 Support of z/VM Single System Image

Single System Image in z/VM is supported through a new operating system device parameter EQID in the z/VM Unit Information Modules (UIMs). This support is also available in z/OS V1R13 HCD. It allows the assignment of the device equivalency ID to a real device (RDEV) or to an emulated device (EDEV).

For the z/VM sysplex support, each RDEV and EDEV device statements gets a new parameter EQID that defines the equalized device ID between separate z/VM systems within the same sysplex.

For HCD, EQID is a new operating system parameter that can be specified for each device. It assigns the device equivalency ID **`eqid`** to a real device (RDEV) or to an emulated device (EDEV). The parameter **`quid`** is a string of 1 to 8 alphanumeric characters. If this parameter is not specified, any previously assigned EQID by the system will be left unchanged. The system-generated name can be overwritten by explicitly specifying a name to this parameter.

When a device is defined to a z/VM operating system configuration, the new parameter EQID is shown; see Figure 19-15.

It allows the assignment of a unique ID to a device that is shared by several system images. Initially, the value is empty. In that case, a system-generated EQID is used. Value NOEQID is reserved. It has the meaning that any previously assigned EQID will be removed.

```
                    Define Device Parameters / Features
CBDPDV13                                                        Row 1 of 5
Command ===> _____________________________________________ Scroll ===> PAGE

Specify or revise the values below.

Configuration ID . : VM9
Device number  . . : 1000         Number of devices  : 16
Device type  . . . : 3390B

Parameter/
Feature    Value +          R Description
OFFLINE    No                 Device considered online or offline at IPL
SHARED     No                 Device shared between multiple real systems
UIRATE     DEFAULT            Hot I/O Recovery Rate
MDC        DFLTON             Device to be cached in minidisk cache
EQID       _______________    Device equivalency ID
****************************** Bottom of data ******************************
```

Figure 19-15 Support of z/VM Single System Image

HCD supports this parameter when defining a device to an operating system of type z/VM. The parameter will show up in the HCD or HCM dialogs, in HCD/HCM reports, and in exported operating system device data. A change in this parameter will result in a delete or add device action during dynamic reconfiguration.

To support this function, all z/VM UIMs that are shipped within HCD had to be extended by the new parameter EQID. The parameter is optional.

The support is only necessary in HCD. When this support is available, an HCM user will see the new parameter and can use it.

Migration and coexistence

The z/VM device support for Single System Image will be rolled back to all z/OS and z/VM systems that are in service at the time of z/VM V6R2 GA, namely V1R11, VR1R12 and z/VM 6.1. It will be rolled into z/OS HCD V1R13 and z/VM HCD V6R2.

Single System Image support for z/VM will be provided in z/OS V1R11 and z/OS V1R12 with APAR OA32576. The support will be provided in z/VM V6R1 with APAR VM64856.

19.4 Quality enhancements

z/OS V1R13 provides the following quality enhancements in HCD and HCM to reduce client PMRs:

- Extended information in dialogs or reports

  Information about used and available unit addresses and control units on a CHPID is now provided. The number of devices (unit addresses) and control units on a CHPID are limited. However, it is difficult to obtain the number of unused resources on a CHPID before the maximum values are exceeded.

  The new action “View unused resources (u)” is provided on the Channel Path List. It provides informational message for each selected CHPID; see Figure 19-16.

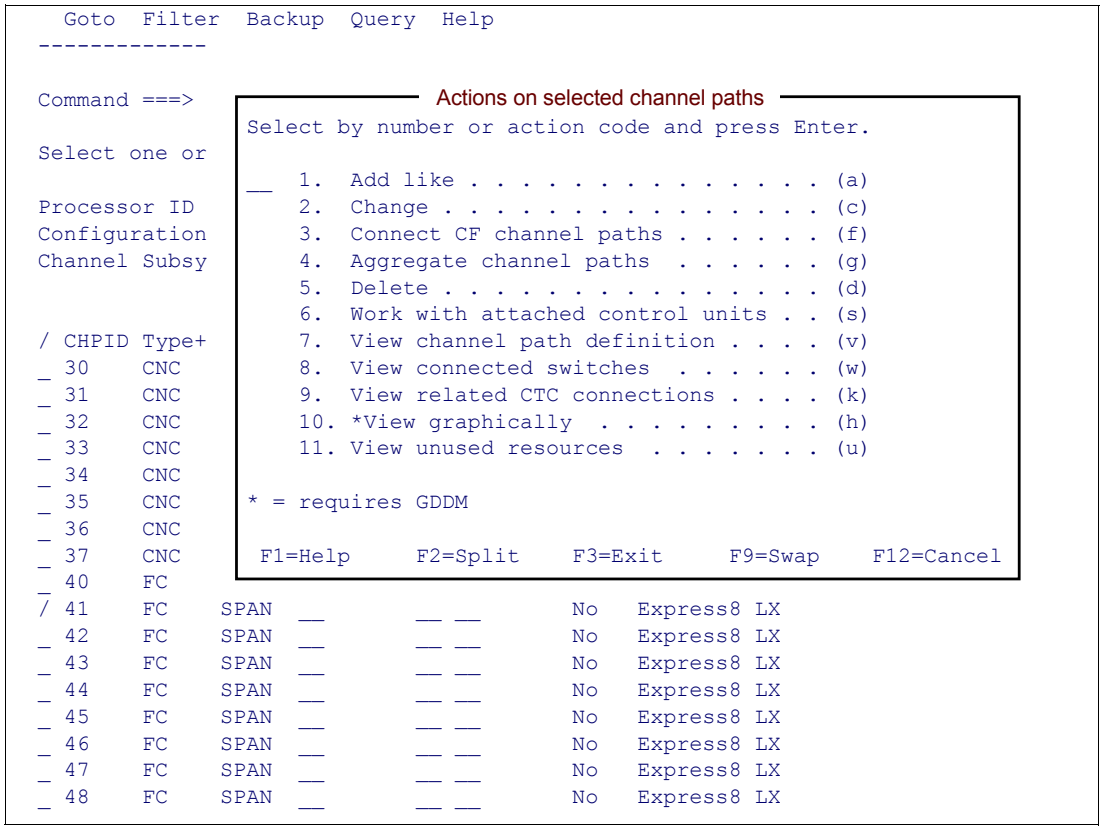


Figure 19-16 View unused resources

Users can thereby easily obtain the number of free resources on a channel path; see Figure 19-17.

```
CBDA377I 'For channel path chpid of processor proc, currently act_ua_num of
max_ua_num available unit addresses and act_cu_num of max_cu_num available
control units are used.'
```

Figure 19-17 New informational message

For several channel paths selected with the **u** action, the list is displayed as shown in Figure 19-18.

```
---------------------------------- Message List -----------------------------------
  Save  Query  Help
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
                                                                        Row 1 of 9
Command ===> _____________________________________________ Scroll ===> HALF

Messages are sorted by severity. Select one or more, then press Enter.

/ Sev Msg. ID  Message Text
_ I   CBDA377I For channel path 1.41 of processor SCZP301, currently 0
#              of 16384 available unit addresses and 0 of 256 available
#              control units are used.
_ I   CBDA377I For channel path 1.44 of processor SCZP301, currently 0
#              of 16384 available unit addresses and 0 of 256 available
#              control units are used.
_ I   CBDA377I For channel path 1.47 of processor SCZP301, currently 0
#              of 16384 available unit addresses and 0 of 256 available
#              control units are used.
***************************** Bottom of data ******************************
```

Figure 19-18 Message list

Migration and coexistence

Note the following migration and coexistence considerations:

- The new HCD action to View unused resources on a CHPID has been retrofitted to z/OS V1R12 with APAR OA29367 (PTF UA55564).
  - Emphasize CF partition in CF Channel Path Connectivity List and Report

    When a CF connection is established, the channel path need to have access to a CF partition. This new field in the CF connectivity dialog and report shows whether a CF partition is defined to the channel.

    The new columns named `CF` in both the source and in the destination channel path sections indicate whether at least one partition in the channel path's access or candidate list is of usage type CF or CF/OS.

    In the CF Channel Path Connectivity Report, the access and candidate lists are combined into a column called `ACCESS/CAND LIST` where all partitions from the candidate list are flagged with `(C)`. Additionally, all partitions of usage type CF or CF/OS are indicated by a preceding asterisk (*).
  - Introduce PPRC usage type NONSYSPLEX

    Because HCD acts on devices with PPRC usage type DUPLEX, a client is unable to document DUPLEX DASD devices that are used in a duplex connection but will not have its OFFLINE parameter reversed when building the operating system configuration of the disaster recovery site.

    In z/OS V1R13, HCD provides an additional value, NONSYSPLEX (N), for the PPRC usage type of a DASD device. This is for documentation purposes only. Thus, all DUPLEX devices can now be documented.

    The prompt for the PPRC usage type in the device dialog shows an additional selection value NONSYSPLEX (N) that is accepted by device validation.
- Additional checks to warn users of various definition errors
  - Check of PPRC secondary device

    If a PPRC secondary device D/T3390D is contained in the operating system configuration (in an alternate subchannel set), a corresponding base device D/T3390B must be defined in subchannel set 0.

    When a production IODF is built, HCD checks that each D/T3390D device that is defined to an operating system configuration has a D/T3390B base device defined with

the same device number. If this is not the case, warning message CBDA398I `PPRC secondary device xxxx in OS configuration xyz does not have a PPRC primary device defined in subchannel set 0.` is issued.

Users are thereby warned about incomplete definitions.

– Warning with device subchannel set mismatch

When defining a device, the subchannel set number in the device-to-processor and the device-to-operating system definitions must match. Otherwise, the device cannot be used.

If a device is defined both to a processor and to an operating system, HCD checks that the subchannel number is the same. If this is not the case, warning message CBDG534 `Device xxxx (range nnn) specifies different subchannel set numbers for its processor and operating system definitions.` is issued.

Users are thereby warned of mismatches at definition time.

– IODF consistency check when building a production IODF

A defect in an IODF can only be detected if an IODF check is performed by issuing the TRACE ID=IODF command, or by including statement CHECK_IODF = YES in the HCD profile.

Whenever the IODF is validated for a production IODF, the IODF checker will also be invoked. If a defect is detected in the IODF, message CBDA999I `Defect(s) detected in IODF dsn` is issued as a severe warning message.

Users are thereby warned of defects in the IODF that can impact the use of the production IODF.

– Various improved messages

The following messages now have additional information or improved readability:

- Message CBDA269I `No Unit Information Table found for dev_type` is extended by the operating system type.
- The explanation of message CBDG181I `Keyword keyword not recognized in profile dsn (line stmt_no)` is extended to include the case that the HCD profile editor will remove the unknown statement during an update. This message is now issued as a warning message.
- Message CBDG454I `Devices of the following control unit(s) can only be attached to one host but have access to more than one partition: cu_1 cu_2 ...` and message CBDG483I `The following CF channel paths of processor proc1 are not connected: chpid1 chpid2 ...` are shown in condensed form for better readability when building a production IODF.

– Export and import of unconnected control units and devices

The tasks 'Build I/O configuration statements' and 'Migrate I/O configuration statements' do not include unconnected control units and devices.

In z/OS V1R13, with HCD profile option SHOW_CONFIG_ALL = YES, the exported statements for switch configurations include unconnected control units and devices.

The 'Migrate I/O configuration statements' task now also imports unconnected control units and devices into the IODF.

The export and import function of an IODF, for example, for an IODF repair, is simplified.

In a first step, we now export configuration statements for control units and devices without a processor and operating system connection. Such unconnected objects are not written when configuration statements for processor or operating system

configurations are exported. They can now be written by exporting all switch configurations, with profile option SHOW_CONFIG_ALL set to YES.

Exported unconnected control units and devices can be imported into an IODF using the switch migration function of the HCD batch utility.

– Flag those tasks that require a prerequisite product to be available

Various HCD tasks HCD (for example, “View graphically” or “Migrate switch configuration data”) require the existence of another optional product like IBM GDDM® or Tivoli System Automation. The error message that is given, if the prerequisite product is not installed, might not indicate this case clearly.

HCD tasks that require another product to be installed are now flagged in the ISPF dialog; see Figure 19-19.

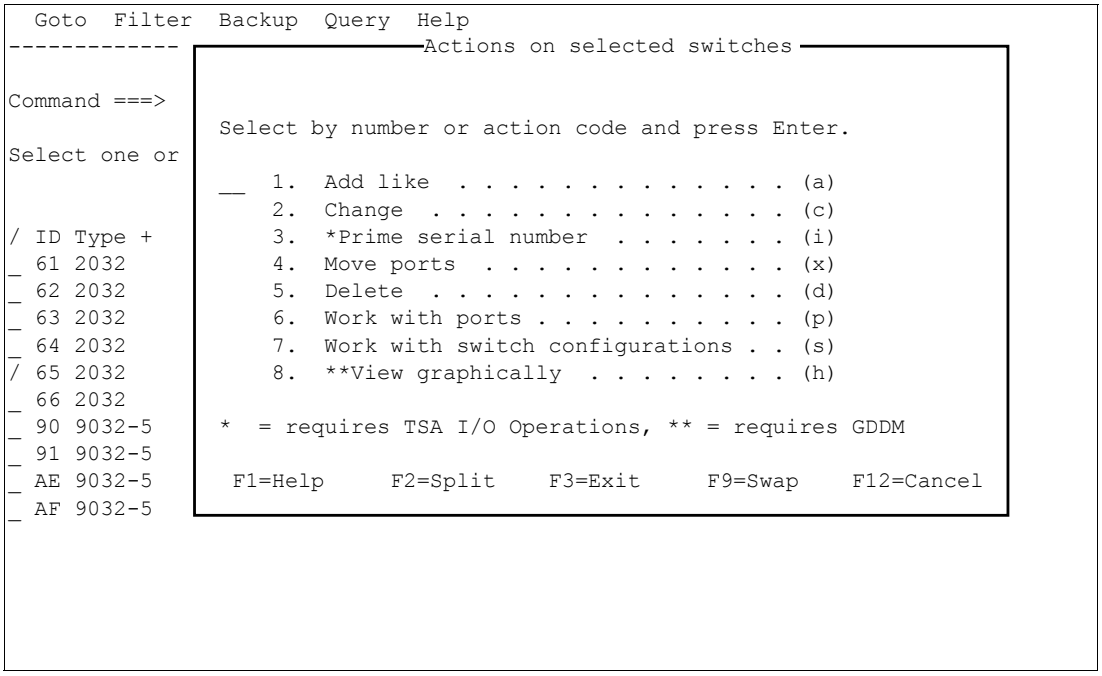


Figure 19-19 Tasks that require a prerequisite product

Users can thereby immediately recognize whether the task is available or not.

– Warning when changing local system name

A change of the local system name of a processor that has a CIB connection to another processor also changes the configuration of this other processor. An activation is required for both processor configurations. In the case of a stand-alone CF processor, a power-on reset (POR) is required.

When the local system name of a processor with CIB connections is changed, HCD warns the user of the consequences by issuing message CBDG400I `Change of local system name of processor proc1 causes a change of the I/O configurations for the following processor(s): proc2 ....`

Users are thereby warned of a potential outage of the coupling facility.

– Reject CF CIB connections between two z9 processors

CIB CF connections between two IBM z9 processors are not supported. However, HCD let them be defined.

Now HCD rejects the definition of a CIB connection between two z9 processors with error message CBDG405I `Channel path chpid1 of type CIB on processor proc1 cannot be connected to channel path chpid2 of type CIB on processor proc2.`

Clients are no longer able to define this unsupported I/O configuration.

- Enhanced performance and productivity of definitions in HCM
  - Performance improvement in HCM when working with CHPIDs

    In HCM, connecting CTC channel paths, adding or editing CHPIDs, or deleting unconnected CHPIDs take significant time compared to the instantaneous behavior in HCD.

    The performance in HCM when working with channel paths has been improved.

    Channel path related tasks in HCM can now be performed faster compared to releases previous to z/OS V1R13.

  - Reworked multiple control unit connect dialog in HCM

    When connecting multiple control units to multiple channel paths in HCM, users were unable to influence the logical control unit address (CUADD) assignment. This can lead to a validation error during the dialog.

    A new dialog step “Add Control Unit <-> Processor Definitions” is introduced with z/OS V1R13. The step allows the explicit setting of processor-related control unit attributes when connecting multiple control units to one or more channel paths or link addresses.

    The rework of the CUADD assignment also maintains the productivity gain of the I/O subsystem wizard in the multiple connect dialog.

- Improved detection and repair capability of corrupted IODFs

19.4.1 Miscellaneous enhancements

Note the following miscellaneous enhancements provided in z/OS V1R13 with HCD and HCM.

Support of Windows 7

Starting with z/OS V1R13, HCM can be installed and used on Windows 7.

Support of IPv6 in HCM for z/VM

The HCM logon dialog accepts, from z/OS V1R13 on, host addresses in IPv6 format.

Similarly, both HCM and the HCM dispatcher in z/VM HCD now support IPv6 connections.

20

C language

The XL C/C++ feature of the IBM z/OS licensed program provides support for C and C++ application development on the z/OS platform.

The XL C Metal compiler option, introduced in z/OS V1R9, generates code that does not require access to the Language Environment support at run time. Instead, the Metal option provides C-language extensions that allow you to specify assembly statements that call system services directly.

This chapter provides descriptions of the following XL C/C++ enhancements provided with z/OS V1R13:

- Performance improvement
- Feedback improvement
- Usability improvement
- Source and binary portability improvement
- Debugging support improvement
- New functions in Metal C

20.1 Improved Metal C optimization

Optimization with Metal C has always been one of the benefits of programming with Metal C in place of direct HLASM coding. By writing source code in Metal C, the program can be compiled with newer versions of the XL C/C++ compiler to benefit from newer hardware instructions (through the ARCHITECTURE option) and scheduling (through the TUNE option) without changing the source code, which is not usually possible with direct HLASM coding. An additional benefit of writing programs in Metal C is that when newer optimization opportunities are available with newer releases of the compiler, they can usually be used without changing the source code.

The z/OS V1R13 XL C/C++ compiler has made the optimization exploitation benefit even stronger for Metal C by bringing higher levels of optimization that were always available for non-Metal programming into the Metal C world. The high-order loop transformations optimization and interprocedural analysis optimizations now available for Metal C will allow further automatic optimization for Metal C programs.

Note: The continuing commitment to Metal by IBM in z/OS XL C/C++ V1R13 makes Metal programming easier and more efficient. It also produces programs that are faster than ever by using the strong performance enhancements for which the XL series of compilers is known.

Metal programming remains an excellent alternative to traditional HLASM programming through the virtues of automatic architecture targeting and performance tuning (through the ARCH, TUNE and optimization options), use of standard C syntax, and tie-ins to HLASM. New enhancements in the z/OS XL C/C++ V1R13 compiler for Metal, including standard C argument parsing, whole program optimizations, and additional debugging capabilities, continue to make Metal programming a better experience and provide even more benefits.

20.1.1 z/OS V1R13 enhancements

Starting with z/OS XL C compiler with z/OS V1R13, the IPA and HOT options are enabled for the Metal option. Both IPA and HOT are compiler optimization options that enable the compiler to find more optimization opportunities to improve your application performance.

Note: See *z/OS Metal C Programming Guide and Reference*, SA23-2225, for additional information when invoking IPA for Metal.

Although various techniques in this section are active at advanced optimization levels, certain types of applications can receive a performance benefit even when you apply only basic optimizations. Starting with z/OS V1R13, the HOT and IPA optimization technique will be available in the Metal C environment. The HOT transformations focus specifically on loops, which typically account for the majority of the execution time for most applications.

Table 20-1 lists and describes the available optimization techniques.

Table 20-1 Optimization techniques

Technique	Description
HOT	This technique minimizes loop execution time, which is beneficial to most applications that contain large loops or many small loops. HOT also improves memory access patterns in your application.
IPA	This technique performs whole program analysis, providing the optimization suite with a complete view of your entire application. This applies performance enhancements with more focus and robustness.
PDF	This technique targets the code paths that your application executes most frequently for optimization.

20.1.2 IPA compiler option

Interprocedural analysis (IPA) performs optimizations on the entire application, across all compilation units, in an attempt to reduce the total application execution time. This class of optimizations is possible by having the whole program available to analyze, which allows exploiting a number of optimizations that are not possible without this additional information.

IPA involves a two-step build process, namely the compile phase and the build phase. Because an additional assembly step is involved in building a Metal C program, be aware of the following adjustments when invoking IPA for Metal:

- During the IPA link phase, all external references must be resolved. For Metal C, IPA does not attempt to convert external object modules or load modules into object code for the inclusion in the IPA produced program. You need to provide the same set of library data sets to both IPA link and the binder for symbol resolution.
- If you supply your own prolog code and epilog code using the PROLOG and EPILOG compiler options, IPA will keep the relationship between the prolog code and epilog code and the designated functions at the CU level.
- If you have #pragma insert_asm in your source file, IPA will assume the strong connection between the string provided by the pragma and the functions in the source file. IPA will not move functions defined in that source file to anywhere else.

Note: For additional considerations, see *z/OS Metal C Programming Guide and Reference*, SA23-2225.

HOT compiler option

The HOT compiler option enables the optimizer to perform specialized loop transformation to generate more highly optimized code. This in turn speeds up the execution time of loops in a program. Because most programs spend significant time in loops, this optional optimization can provide significant benefits in execution time.

The goals of these optimizations include:

- Reducing memory access costs through effective cache use and translation look-aside buffers (TLBs). Increasing memory locality reduces cache and TLB misses.
- Overlapping computation and memory access through effective utilization of the hardware data prefetching capabilities.
- Improving processor resource utilization by reordering and balancing the use of instructions with complementary resource requirements. Loop computation balance

typically involves creating an equitable relationship between load/store operations and floating-point computations.

On a regular non-Metal complication, the high-order loop transformations can be enabled by specifying the HOT option along with any optimizing compilation when specifying the OPT(2) or higher level of optimization option.

Example 20-1 illustrates how to compile the source, loop.c, into loop.o with optimization level 2 and HOT.

Example 20-1 Compiling source with HOT

```
> xlc –O2 –qhot –c loop.c
```

Similarly, Example 20-2 illustrates how to compile a Metal C program with HOT. The command would be to produce loop.s.

Example 20-2 Compiling a Metal C program with HOT

```
> xlc –qmetal -S –O2 –qhot loop.c
```

Order levels

During compilation, you can specify various compiling optimization levels. Table 20-2 lists the optimization levels and their behavior.

Table 20-2 HOT optimization levels

Optimization level	Benefits
-o3	► In-depth aliasing analysis ► Better loop scheduling ► High-order loop analysis and transformations (-qhot=level=0) ► Inlining of small procedures within a compilation unit by default ► Eliminating implicit compile-time memory usage limits ► Widening, which merges adjacent load/stores and other operations ► Pointer aliasing improvements to enhance other optimizations
-o4	► Propagation of global and parameter values between compilation units ► Inlining code from one compilation unit to another ► Reorganization or elimination of global data structures ► An increase in the precision of aliasing analysis
-o5	► Most aggressive optimizations available ► Makes full use of loop optimizations and IPA

20.1.3 Interprocedural analysis

Interprocedural analysis (IPA) can analyze and optimize your application as a whole, rather than on a file-by-file basis. Run during the link step of an application build, the entire application, including linked libraries, is available for interprocedural analysis.

This whole program analysis opens your application to a powerful set of transformations available only when more than one file or compilation unit is accessible. IPA optimizations are also effective on mixed language applications.

Following are several of the link-time transformations that IPA can use to restructure and optimize your application:

- Inlining between compilation units
- Complex data flow analyses across subprogram calls to eliminate parameters or propagate constants directly into called subprograms
- Improving parameter usage analysis, or replacing external subprogram calls to system libraries with more efficient inline code
- Restructuring data structures to maximize access locality

To maximize IPA link-time optimization, you must use IPA at both the compile and link step. Objects you do not compile with IPA can only provide minimal information to the optimizer, and receive minimal benefit.

However, when IPA is active on the compile step, the resulting object file contains program information that IPA can read during the link step. The program information is invisible to the system linker, and you can still use the object file and link without invoking IPA.

The IPA optimizations use hidden information to reconstruct the original compilation and can completely analyze the subprograms the object contains in the context of their actual usage in your application.

Table 20-3 IPA levels

IPA level	Behaviors
qipa=level=0	► Automatically recognizes standard library functions ► Localizes statically bound variables and procedures ► Organizes and partitions your code according to call affinity, expanding the scope of the -O2 and -O3 low-level compilation unit optimizer ► Lowers compilation time in comparison to higher levels, though limits analysis
qipa=level=1	Level 0 optimizations Performs procedure inlining across compilation units Organizes and partitions static data according to reference affinity
qipa=level=2	► Level 0 and level 1 optimizations ► Performs whole program alias analysis which removes ambiguity between pointer references and calls, while refining call side-effect information ► Propagates interprocedural constants ► Eliminates dead code ► Performs pointer analysis ► Performs procedure cloning ► Optimizes intraprocedural operations, using specifically: – Value numbering – Code propagation and simplification – Code motion, into conditions and out of loops – Redundancy elimination techniques

IPA processing

During the link step, IPA restructures your application, partitioning it into distinct logical code units. After IPA optimizations are complete, IPA applies the same low-level compilation unit transformations as the -O2 and -O3 base optimization levels. Following those transformations, the compiler creates one or more object files and linking occurs with the necessary libraries through the system linker.

It is important to specify a set of compilation options as consistent as possible when compiling and linking your application. This includes all compiler options, not just `-qipa` suboptions.

When possible, specify identical options on all compilations and repeat the same options on the IPA link step. Incompatible or conflicting options that you specify to create object files, or link-time options in conflict with compile-time options, can reduce the effectiveness of IPA optimizations.

IPA option examples

Previously, without IPA support, the code needed to build this program as a Metal C application without the performance enhancements that IPA provides is shown in Example 20-3.

Example 20-3 Building a Metal C application without the performance enhancements IPA provides

```
> xlc –qmetal –S –o errorCounter.s errorCounter.c
> xlc –qmetal –S –o inventoryCheckMain.s inventoryCheckMain.c
> as -mgoff -o errorCounter.o errorCounter.s
> as -mgoff -o inventoryCheckMain.o inventoryCheckMain.s
> ld -o inventoryCheck inventoryCheckMain.o errorCounter.o
-l"//'CBC.SCCNOBJ'"
```

Building a Metal C application using the performance enhancements that IPA provides is shown in Figure 20-1.

Building a Metal C application with IPA involves a mixture of the methods shown in Example 20-3 for the IPA compile step that is similar to a non-Metal IPA compile, except for the extra `–qmetal` option:

```
> xlc –qmetal –qipa –c –o errorCounter.o errorCounter.c
> xlc –qmetal –qipa –c –o inventoryCheckMain.o
inventoryCheckMain.c
```

For the IPA link step that is like building a Metal C program without IPA:

```
> xlc –qmetal –S –qipa –o inventoryCheck.s errorCounter.o
inventoryCheckMain.o
```

To assemble, bind, run and see the results of the program:

```
> as –mgoff –o inventoryCheck.o inventoryCheck.s
> ld –o inventoryCheck inventoryCheck.o –l"//'CBC.SCCNOBJ'"
> inventoryCheck Test 0 5 211 -1 2 -3 94
> echo $?
2
```

Figure 20-1 Building a Metal C application with IPA

Note: The final assembly file can be compared with the non-IPA assembly files to get an idea of various optimizations done for this program. When building a Metal C application with IPA, keep in mind the same set of IPA considerations as for building a non-Metal application with IPA (for example, a main function is required).

20.2 New hardware built-ins

Interlocked-storage-access instructions, available on models where the interlocked-access-facility is installed, provide a means by which a load, update, and store

operation can be performed with interlocked update in a single instruction. Interlocked storage instructions are not directly available for user source. In z/OS V1R13, IBM adds support for built-in functions corresponding to the interlocked storage access instructions.

Supported interlocked-storage-access instructions are:

- Load and Add (LAA, LAAG)
- Load and Add Logical (LAAL, LAALG)
- Load and And (LAN, LANG)
- Load and Exclusive Or (LAX, LAXG)
- Load and Or (LAO, LAOG)
- Load Pair Disjoint (LPD, LPDG)

Source code can now use interlocked-storage-access instructions through built-in functions providing the benefit and speed of the hardware instructions. To make them usable for high level languages, IBM implemented an interface to the C/C++ language.

Important: This function works only on z196 machines. It is implemented under architecture option ARCH(9).

The instructions ending with G operate on 64-bit operands. The built-in functions will follow the same convention. Applications that make use of built-in functions that operate on 64-bit operands must be compiled and linked with the LP64 compiler option. Table 20-4 lists all new C/C++ functions to support the new hardware instructions supported by z196.

Table 20-4 New C functions

C function	Instruction
int __lad(int* op1, int op3, int* op2);	LAA
int __ladg(long* op1, long op3, long* op2);	LAAG
int __ladl(unsigned int* op1, unsigned int op3, unsigned int* op2);	LAAL
int __lan(unsigned int* op1, unsigned int op3, unsigned int* op2);	LAN
int __lang(unsigned long* op1, unsigned long op3, unsigned long* op2);	LANG
iint __lax(unsigned int* op1, unsigned int op3, unsigned int* op2);	LAX
int __laxg(unsigned long* op1, unsigned long op3, unsigned long* op2);	LAXG
iint __lao(unsigned int* op1, unsigned int op3, unsigned int* op2);	LAO
nt __laog(unsigned long* op1, unsigned long op3, unsigned long* op2);	LAOG
iint __lpd(unsigned int* op3, unsigned int* op4, unsigned int* op1, unsigned int* op2);	LPD
int __lpdg(unsigned long* op3, unsigned long* op4, unsigned long* op1, unsigned long* op2);	LPDG

20.3 Multiply and Add for hexadecimal types

Fused Multiply and Add instructions are not generated for hexadecimal floating point types. Due to performance and hardware reasons, it is not a good idea to emit these instructions in general cases.

Starting with z/OS V1R13, new support for zEnterprise makes this feasible to do, allowing clients to FLOAT(MAF) + FLOAT(HEX) for ARCH >= 9 only. Multiply and Add instructions can be generated, potentially improving the performance of the application. Example 20-4 shows a sample compiler call for using FLOAT support on zEnterprise.

Example 20-4 Sample call for using FLOAT

```
xlc -qfloat=hex -qfloat=maf -qarch=9 mysource.c
```

20.4 Informational messages in z/OS UNIX System Services

When using the INFO option to get diagnostic messages in UNIX System Services, users must remember to specify the FLAG(I) option to view informational severity messages.

Make FLAG(I) the default in UNIX System Services as it is in Batch compilation.

Users will no longer get the false impression that there are no potential errors in their code if they forget to specify FLAG(I).

20.5 Metal C: function property information

In prior releases, it can be difficult to find information about functions in Metal-generated code. z/OS V1R13 adds the Function Property Block (FPB), per-function property data that can be used to identify the C function and the associated properties by code scanning or dump reading. The FPB can be found through the new Function Entry Point Marker placed immediately before the entry point of each function.

The Function Property Block also contains an offset that can be used to find the Prefix Data generated for each compilation unit. Starting with z/OS V1R13, clients can enable Metal C users to find additional information about a function.

The Function Property Block is generated for all Metal C compiles. The FPB is composed of a fixed part (20 bytes in size) followed by a contiguous optional part, with the presence of optional fields indicated by flag bits. Optional fields, if present, are stored immediately following the fixed part of the FPB aligned on fullword boundaries in a given order. The detailed layout of the FPB is documented in *z/OS Metal C Programming Guide and Reference*.

Note: When the COMPRESS compiler option is in effect, the function name fields will not be present in the FPB.

20.6 Metal C DSA support

A common practice of pointing to the dynamic storage area in user source can be overwritten by compiler-generated code.

In z/OS V1R13, the new option DSAUSER is added for requesting a user field of the size of a pointer to be reserved on the stack. This user field can be utilized by the user-provided prolog/epilog code for the purpose for which it was intended.

The user field can be located through the new HLASM global set symbol &CCN_DSAUSER, which provides the offset to the user field. The compiler merely allocates the new field on the stack without any code to initialize it. Table 20-5 lists the characteristics of the new DSAUSER option.

Table 20-5 New DSAUSER option

Name	DSAUSER | NODSAUSER
Abbreviation	DSAU
Default	NODSAUSER
Category	Object control mode
#pragma option	None
Syntax	| - NODSAUSER - | >>---- DSAUSER ----------><
Description	When DSAUSER is specified with the Metal option, a field of the size of a pointer is reserved on the stack. The user field is a 4-byte field for AMODE 31, and an 8-byte field for AMODE 64. The user field is only allocated if the function has user-supplied prolog/epilog code. The user field can be addressed by using the global set symbol &CCN_DSAUSER, which is described in *z/OS Metal C Programming Guide and Reference*. IPA effects: If the DSAUSER option was specified during any of the IPA compile step, it will be applied to all partitions created by the IPA link step.

Table 20-6 describes the characteristics of the new global DSA symbol.

Table 20-6 New global DSA symbol

Global Set symbol	Type	Description
&CCN_DSAUSR	Character	The assembly time computed offset to the user field of the stack of the function.

Based on this support, Metal C users now have a reliable way to obtain a field of the size of a pointer (that is, 4 bytes for AMODE 31 and 8 bytes for AMODE 64) on the stack reserved for the user.

20.7 Metal C argument parsing

The common method of processing command line arguments (argc and argv) in C programs is not natively supported in Metal. The “argc” and “argv” format parsing capability is added to Metal C programs. If your main() function uses the standard argc and argv arguments, the Metal C initialization routine is called to parse the raw parameter data received from the hosting environment and to convert the parameter to the standard argc and argv format.

If your program is not invoked in the z/OS UNIX System Services environment, you can use the ARGPARSE or NOARGPARSE options to determine if the EXEC PARM needs to be further parsed into individual arguments. The EXEC PARM must be in this format: a halfword

length field followed by a maximum of 100 characters where the length field contains a binary count of the number of bytes in the PARM field.

For more information about the ARGPARSE option, see ARGPARSE | NOARGPARSE in *z/OS XL C/C++ User's Guide*, SC09-4767.

If your main() function uses argc and argv arguments and you do not want the parsing to be performed, you can set the new Global Set Symbol &CCN_APARSE to 0 in your prolog code to conditionally bypass the argument parsing. This allows Metal C programs that do need the standard argc and argv style of parameters to have the arguments automatically parsed. Table 20-7 describes the characteristics of the new &CCN_APARSE symbol.

Table 20-7 New &CCN_APARSE symbol

Global Set symbol	Type	Default	Description
&CCN_APARSE	Logical	1	Set to "1" to trigger parser call. Set to "0" to disable parser call.

20.8 C++: template depth

In prior releases of the C compiler, the immutable limit of 50 recursively instantiated template specializations are processed by the compiler before it halts compilation and emits an error.

In z/OS V1R13, the new compiler option `TEMPLATEDEPTH` with a single integer suboption allows users to specify their own value for how deep they want the compiler to instantiate recursive template specializations. The reason this limit exists is to prevent the compiler from entering infinite loops while instantiating improperly written user template code. This limit has been increased to 300 and is now controlled by the `TEMPLATEDEPTH` option.

Carefully crafted template code that needs more recursive instantiations is now able to compile, as illustrated in Example 20-5.

Example 20-5 New template usage

```
template <int n> void foo() {
  foo<n-1>();
}
template <> void foo<0>() {}
int main() {
  foo<400>();
}
```

20.9 Compatibility support

There are numerous compatibility enhancements provided in z/OS V1R13.

20.9.1 Text following #endif

The C99 C standard does not allow extraneous text following #else and #endif; the XL C/C++ compiler issues a warning on any extraneous text that is found after #else and #endif.

Now a new suboption of LANGLVL, textafterendif, has been implemented to instruct the compiler to suppress the warning for extraneous text following #else and #endif.

Note: The warning message is only suppressed if the new suboption of LANGLVL is explicitly specified.

This feature enables users to indicate that they want the XL C/C++ compiler to be silent about this deviation from the standard, thereby increasing portability. Example 20-6 shows a sample program with extra text following #else.

Example 20-6 Sample mysource.c

```
#ifdef MY_MACRO
#else MY_MACRO not defined
#endif MY_MACRO

int main(void) {
  return 55;
}
```

With the z/OS V1R13 compatibility enhancement, the Compilation command using the new option returns no errors:

```
xlc -qlanglvl=textafterendif mysource.c
```

20.9.2 Function attribute gnu_inline

GCC changed inline keyword behavior to conform to Standard C99, but various existing GCC programs still rely on the old behavior of GCC inline. A new function attribute, gnu_inline, was introduced by GCC to allow users to retain the old GCC inline behavior. This implementation now offers support for the gnu_inline function attribute to match the GCC inline behavior, as illustrated in Example 20-7.

Example 20-7 New attribute gnu_inline

```
extern inline __attribute__((gnu_inline)) foo() {…};
static inline __attribute__((gnu_inline)) bar() {…};
```

This feature makes it easier for users to port their programs to the XL C/C++ compiler.

Note: A key difference in all of the inline behaviors is under what conditions a definition of the function is kept and externalized.

20.9.3 Function attribute used

Prior to z/OSV1R13, the C compiler might remove a function that it does not see used in the program. The function is referenced only in inline assembly.

Starting with z/OS V1R13, the Function attribute instructs the compiler to emit the code for a function even if it appears that the function is not referenced; see Example 20-8.

Example 20-8 New attribute usage

```
__attribute__((used)) void foo() { }
   int main() { foo(); }
```

If the attribute is used in combination with attribute gnu_inline (discard the definition), GCC's behavior where gnu_inline wins is now replicated, and the function definition is discarded. This allows you to keep functions that would otherwise be removed.

Note: For static functions, a definition will be always emitted, if they are marked with attribute used.

20.9.4 Function attribute malloc

Certain functions have properties that can be exploited to increase performance, but there is no way for the compiler to know that. One such property is that a non-null pointer returned cannot alias any other pointer that is valid at the time of the function call.

Now, in z/OS V1R13, the function attribute malloc is used to instruct the compiler to treat a function as though any non-NULL pointer it returns cannot alias any other pointer that is valid when the function returns.

The optimization that this attribute enables at the moment will only occur at -O5; see Example 20-9.

Example 20-9 New malloc() attribute

```
void* foo()  __attribute__ ((__malloc__)) { ... }
```

Note: The malloc() function attribute cannot be used if the user cannot guarantee that the pointer returned by a function points to unique storage. Otherwise, the optimization performed might lead to problems at run time.

This feature might speed up the execution time of the program because it provides information that is helpful for extra optimization.

20.9.5 Temporary lifetime extension

Users who port applications from another compiler that implements late temporary destruction might want to extend the lifetime of such temporaries to replicate the previous nonstandard compliant behavior.

In z/OS V1R13, the compiler option `-qlanglvl=tempsaslocals` is used to extend the lifetime of C++ temporaries beyond that specified by the C++ Language Standard in 12.2.

When enabled, the lifetime of temporaries will be treated as that of local variables declared in the innermost containing lexical scope where possible. In various contexts, temporaries will be treated in the standard compliant way, even when this feature is enabled. When a program incorrectly depends on resources that might have been previously released, this feature might help. Example 20-10 shows a sample temporary lifetime extension.

Example 20-10 Temporary lifetime extension sample

```
#include<cstdio>

 struct S {
   S() { printf("S::S() ctor at 0x%lx.\n", this); }
   S(const S& from) { printf("S::S(const S&) copy ctor at 0x%lx.\n", this); }
```

```
  ~S() { printf("S::~S() dtor at 0x%lx.\n", this); }
} s1;

void foo(S s) { }

int main() {
  foo(s1);
  printf("hello world.\n");
  return 0;
}
```

With `-qlanglvl=tempsaslocals`, the temporary `s` created for the function argument is destroyed after the lexical block of `main`. By default, `s` is destroyed upon returning from `foo`.

20.9.6 Binding rvalue to non-const reference

Non-compliant compilers might allow a non-const reference to be bound to an rvalue.

z/OS V1R13 allows a non-const reference to bind to an rvalue only in the declaration of a function parameter or function return type where an initializer is not required, and only for user-defined types. This feature also permits an rvalue to bind to a const-volatile reference, and it only applies to top-level CV qualifiers on reference types. The following option might accept a non-compliant program:

```
-qlanglvl=compatRValueBinding
```

The option `-qinfo=por` will enable an informational message indicating that this binding has taken place despite being illegal. Example 20-11 shows a sample non-const reference.

Example 20-11 Non-const reference sample

```
struct hey{};
void func(hey& x){}
int main(void)
{
  func(hey());
  return 0;
}
```

By default, the sample program shown in Example 20-11 will be rejected with CCN5295 (S) `A parameter of type "hey &" cannot be initialized with an rvalue of type "hey".`

Using `-qlanglvl=compatRValueBinding`, however, it will compile cleanly.

20.9.7 Intrinsic complex types

Previously, intrinsic complex types were only implemented by the C compiler. The C++ compiler lacked this functionality.

In z/OS V1R13, the complex types of float _Complex, double _Complex and long double _Complex are provided by the C++ compiler as built-in types, according to ISO/IEC 9899:1999 Standard.

This allows clients to compile C programs with built-in complex types C++. Such programs do not need to convert intrinsic complex types to Complex class template implementation to compile with C++. The feature is normally enabled with the following option:

```
-qlanglvl=c99complex
```

As shown in Example 20-12, the complex types and both unary operators __real__ and __imag__ can be enabled with the following option:

```
-qlanglvl=gnu_complex
```

Example 20-12 Complex types sample

```
#include <stdio.h>
#include <complex.h>

int main() {
float _Complex a, b;
a= 2.0f + 3.0f * _Complex_I;
b = 4.0f - 2.0f * _Complex_I;
a = a + b;
printf("a = %f + %f * I . \n", __real__(a), __imag__(a));
```

Compiling and running the sample program shown in the example with `-qlanglvl=gnu_complex` produces the following output:

```
a = 6.000000 + 1.000000 * I
```

20.9.8 Addressable labels

Porting source code from other platforms or compilers might not work on the z/OS XL C/C++ compiler if addressable labels are used.

z/OS V1R13 adds support for the Labels-as-values and Computed-goto features that are implemented by other compilers, for example, GCC. This makes porting code to the z/OS XL C/C++ compiler easier; see Example 20-13.

Example 20-13 Addressable labels

```
int main(void) {
  void* la = &&label1;
  goto *la;
  return 66;
label1:
  return 55;
}
```

When you use the following steps to compile and run the sample program shown in the example, it returns with code 55.

- `xlc mysource.c`
- `./a.out`

20.9.9 Trailing return type

Previously, when given an expression such as `a*b`, where `a` and `b` are arbitrary types, we cannot say: `type of a*b`.

With z/OS V1R13, C++0x Trailing Return Type in conjunction with C++0x Decltype removes this limitation. Clients are now able to declare function templates with return types that depend on the types of the template arguments; see Example 20-14.

Example 20-14 Tailing return codes R12 sample

```
template <class A, class B>
decltype(*(A*)(0)**(B*)(0)) multiply (A a, B b)
{
  return a*b;
}
```

Although the sample in Example 20-14 compiles with -q`langlvl=decltype` available in R12, it introduces code clutter and is error-prone.

In contrast, the sample shown in Example 20-15 uses more elegant syntax that removes code clutter. It can be compiled with `-qlanglvl=autotypededuction:decltype` or with simply `-qlanglvl=extended0x`.

Example 20-15 Tailing return codes R12 sample

```
template <class A, class B>
auto multiply(A a, B b)->decltype(a*b)
{
  return a*b;
}
```

20.10 New debugging APIs

Prior releases did not allow you to find function entry points during debug.

In z/OS V1R13, CDA provides APIs for DBX to get the entry point of functions and the first statement address of each function. The newly added APIs can help debugger developers to access debug information in the `.mdbg` and `.dbg` files more directly.

ddpi_function_get_func_entrypt

The `ddpi_function_get_func_entrypt` operation shown in Example 20-16 returns the entry point of a function, that is, its low pc.

Example 20-16 New ddpi_function_get_func_entrypt API

```
int ddpi_function_get_func_entrypt(
          Ddpi_Function         function,
          Dwarf_Addr*           ret_func_entrypt,
          Ddpi_Error*           error);
```

The ddpi_function_get_first_stmt_addr operation shown in Example 20-17 returns the address of the first executable statement of a function, for example the machine instruction following the function prolog.

Example 20-17 New ddpi_function_get_first_stmt_addr API

```
int ddpi_function_get_first_stmt_addr(
          Ddpi_Function         function,
          Dwarf_Addr*           ret_first_stmt_addr,
          Ddpi_Error*           error);
```

20.11 Debugging inline procedures

In current releases of z/OS, clients cannot debug inline procedures. For example, although an entry breakpoint for a procedure can be set, the breakpoint might not be hit if the procedure is inlined. This is because the debugger can only set a breakpoint at the original procedure, not the inline instances.

Now, with the newly generated debug information, debugging developers can set entry breakpoints at all inline instances so that the user will not miss the breakpoint. The inline debug information will be generated with:

```
DEBUG(FORMAT(DWARF)) + OPT
```

For example,

```
xlC -qDEBUG -O2 -o a a.cpp
```

This generates debug information for both the original procedure and the inline instance; see Example 20-18.

Example 20-18 New generated debug information

```
<1><  308>      DW_TAG_subprogram
                DW_AT_type                  <146>
                DW_AT_name                  foo
...
<2><  379>      DW_TAG_inlined_subroutine
                DW_AT_abstract_origin       <308>
                DW_AT_low_pc                0x124
```

21

Storage management enhancements

The real storage manager (RSM) controls the allocation of main storage during initialization, and pages in user or system functions for execution.

Within an application server address space, many application programs run under a single server program. An error in one of these application programs can cause it to overwrite the code or data of the other application programs or of the server program itself. Subspaces provide a means of limiting the application server address space storage that an application program can reference, thus limiting the damage an application program error can do within the application server address space.

This chapter describes storage management enhancements introduced in z/OS V1R13. First, recent 64-bit enhancements provided both with hardware changes and with new z/OS releases are reviewed. Next, the latest enhancements provided with z/OS V1R13 are described, including the following:

- 64-bit subspace support
- Large page enhancements

21.1 64-bit storage overview

Before discussing the storage management enhancements that are introduced in z/OS V1R13, a review of 64-bit storage is useful. The two gigabyte address in the address space is marked by a virtual line called “the bar.” The bar separates storage below the two gigabyte address, known as “below the bar,” from storage above the two gigabyte address, called “above the bar.” The area above the bar is intended to be used for data only, not for executing programs.

Programs use the IARV64 macro to obtain storage above the bar in “chunks” of virtual storage known as *memory objects*. Your installation can set a limit on the use of the address space above the bar for a single address space. The limit is called the MEMLIMIT.

Figure 21-1 shows a z/OS V1R10 map of virtual storage areas illustrating these concepts.

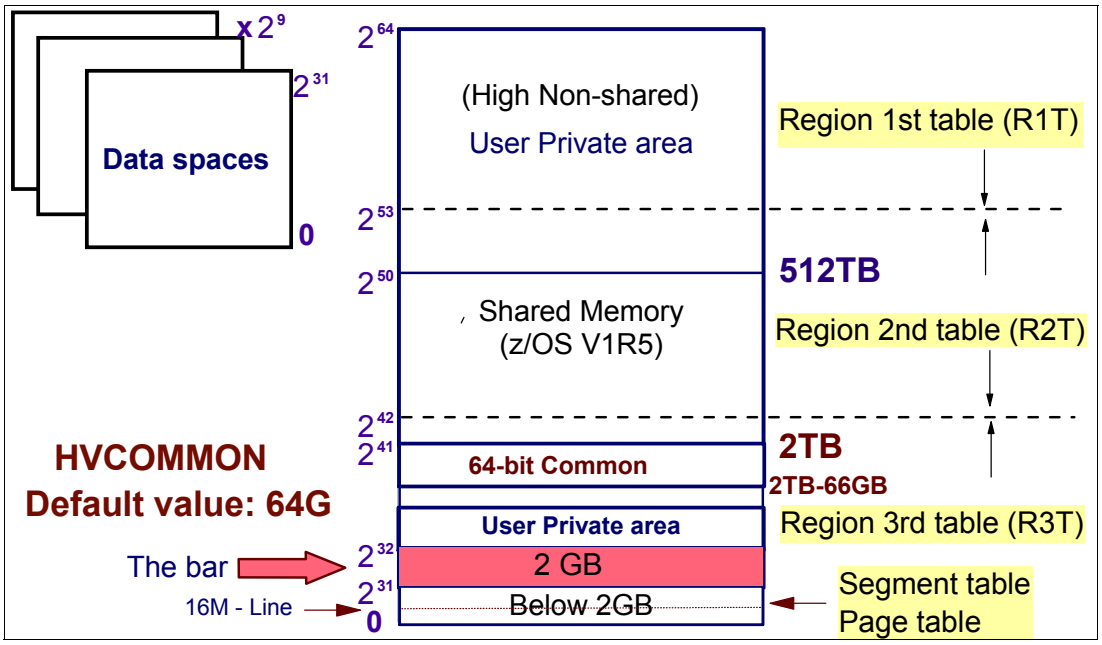


Figure 21-1 z/OS V1R10 map of virtual storage areas

21.1.1 64-bit common virtual storage

z/OS V1R10 introduced support for common storage above the 2 GB bar. A new virtual storage area known as the high common storage area (64-bit Common) is defined. Storage management services and RMF support for 64-bit Common are also planned. This support provides the infrastructure required for many users of CSA and ECSA storage to move data above the bar. This is expected to lead to virtual storage constraint relief (VSCR) over time.

Many system components currently get CSA or SQA storage using the GETMAIN or STORAGE macro. As we strive for VSCR, many of these components have been asked to move their blocks above the bar. In addition to all the effort to change to amode 64 to access storage above the bar, many of these components would need to create a 64-bit storage manager. This support allows components to make a simple substitution of macro IARST64 for either GETMAIN or STORAGE.

21.1.2 Local system area

A function introduced in z/OS V1R12 supports a system area in the storage above the 2 GB bar, as shown in Figure 21-2. This storage is designed to be used for system storage as an equivalent to LSQA below the 2 GB bar. Storage is this area will not be copied during the fork() process when RSM copies the parent storage to the child address space.

Note: A keyword has been added to the IARV64 REQUEST=GETSTOR, LOCALSYSAREA=NO|YES, to indicate that the memory object is to be allocated from the system area of the 64-bit address space map.

LOCALSYSAREA memory objects are allocated in the system area that starts at X'8_00000000' - 32 GB and ends at X'28_00000000' - 288 GB.

Only authorized users may use this keyword.

The system area is shown in Figure 21-2.

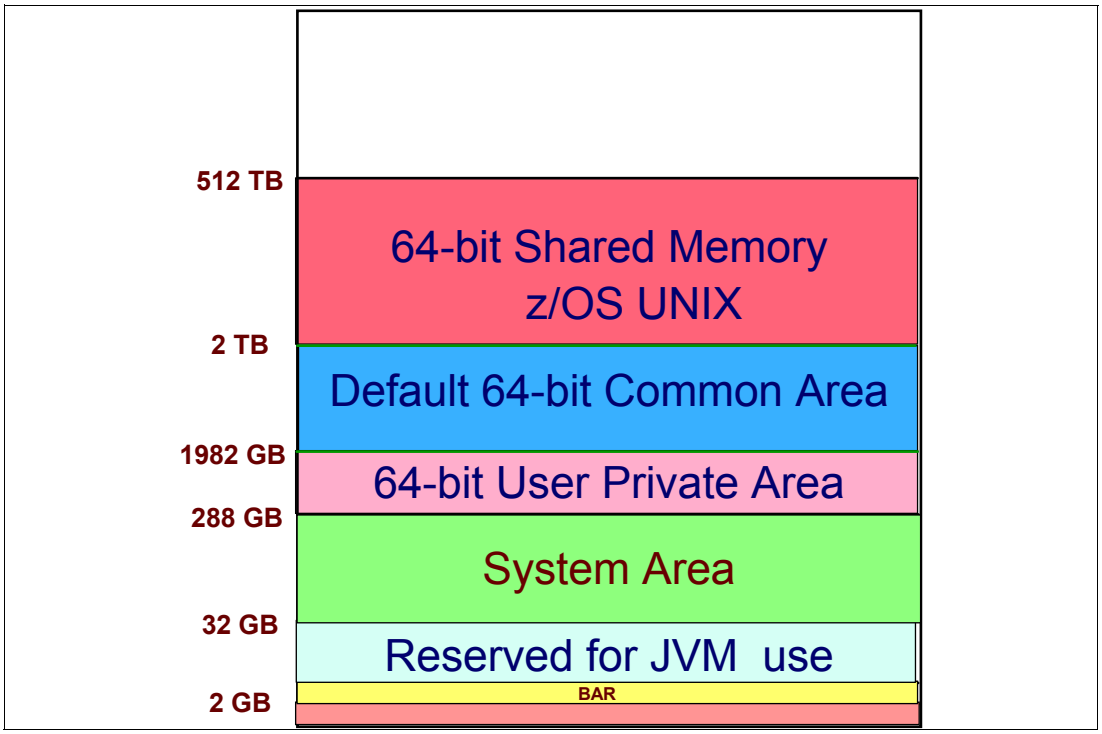


Figure 21-2 Private area storage map showing the system area

fork() processing starting with z/OS V1R12

When a REQUEST=GETSTOR is processed, a required parameter known as LOCALSYSAREA is specified, as follows:

- REQUEST=GETSTOR creates a private memory object if LOCALSYSAREA=YES is not specified.
- If LOCALSYSAREA=YES is specified, then a system memory object is returned. The storage obtained in the system area using the LOCALSYSAREA keyword will not be copied during fork() processing.

The use of local system area storage does not preclude checkpoint from succeeding. Upon completion, the memory object is created in the address space indicated.

LOCALSYSAREA=YES processing

When LOCALSYSAREA=YES is specified, MEMLIMIT is ignored.

During the 64-bit copy phase for fork() processing, memory objects that are allocated in the system area will not be copied to the child space. Fork processing will fail only if the child has 64-bit memory objects allocated in the User Private Area, shown in Figure 21-2. This function allows fork() processing to continue to use 64-bit virtual under any circumstance.

21.1.3 z/OS V1R13 virtual storage

The latest z/OS releases have been structuring virtual storage as reflected in Figure 21-3.

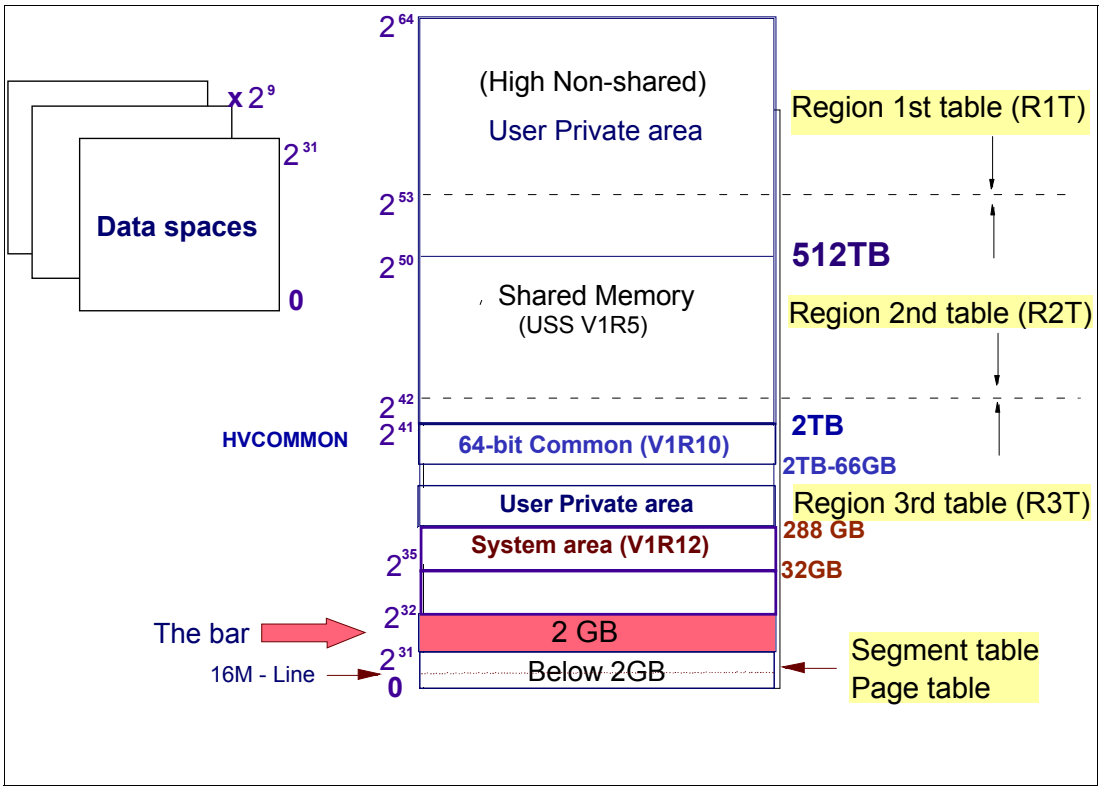


Figure 21-3 z/OS V1R13 map of virtual storage areas

21.2 64-bit subspace support

A *subspace* is a specific range of storage in the private area of an address space, and it is designed to limit the storage a program can reference. A program that is associated with a subspace cannot reference some of the private area storage outside of the subspace storage range; the storage is protected from the program. Whether a given range of private area storage is protected from a program associated with a subspace depends on whether the storage is:

- Eligible to be assigned to a subspace (or “subspace-eligible”)
- Assigned to a subspace
- Not eligible to be assigned to a subspace

You control these storage “states” through the IARSUBSP macro. Storage outside of the private area is not affected by subspaces.

A program that runs in an address space that owns subspaces also has full address space addressability, until it issues an instruction to limit the storage it can reference. In an address space that owns subspaces, issuing the BSG instruction controls whether a program runs with full or limited address space addressability. In this section, a program running with limited addressability is said to be running in a subspace.

A program running in a subspace can reference storage that is assigned to its own subspace and storage that is not eligible to be assigned to a subspace. It cannot reference storage that is eligible to be assigned to a subspace other than the one in which the program is running.

That is, a subspace allows a program running in it to reference all of the storage associated with the address space except the private area storage that is eligible to be assigned to a subspace or assigned to another subspace.

When storage is not eligible to be assigned to a subspace and not assigned to a subspace, it can be referenced by a program running in a subspace or a program running with full address space addressability. This storage can be referenced by all subspaces and by programs running with full address space addressability.

An address space that owns subspaces is also called a “base space.”

IARSUBSP macro is used to create and delete subspaces. A subspace is a section of address space private area storage that you have set up to contain and protect a program and its data.

Subspaces provide isolation between multiple programs running in a single address space by allowing a program that runs in the subspace to reference only certain storage in the address space private area.

A program that is associated with a subspace cannot reference certain private area storage outside of the subspace storage range; the storage is protected from the program. Whether a given range of private area storage is protected from a program associated with a subspace depends on whether the storage is:

- Eligible to be assigned to a subspace (known as “subspace-eligible”)
- Assigned to a subspace
- Not eligible to be assigned to a subspace

z/OS V1R13 and subspaces

In z/OS V1R13, storage management is changed to allow tasks using subspaces to access 64-bit private and 64-bit shared virtual storage. This is intended to help provide virtual storage constraint relief by making it easier for applications to exploit 64-bit storage and to use system services that use 64-bit storage.

A program that runs in an address space that owns subspaces also has full address space addressability. While running in a subspace, a program now has access to 64-bit private and shared storage. It can reference the 64-bit storage while in subspace mode and no longer needs to issue the Branch in Subspace Group (BSG) instruction to switch to the base mode to reference the 64-bit storage. As a result of this modification, note the following points:

- A subspace can directly access high virtual storage (without BSG).
- This change can provide a performance improvement.
- This makes for simpler application programming.
- Subspaces can directly access any valid high virtual storage.

21.2.1 Subspace mode

The base space and each subspace are separate virtual structures (address spaces) with their own segment-table designations. The using subsystem (for example, CICS) has code in each subspace, the equivalent of what can be called “application common” or “subsystem common.”

Basically, the way that a branch in a subspace group (BSG) instruction works is that the “branch” is done to the “space” (actually, simply a segment-table designation replacement) that is represented by the access-list-entry token (ALET) in the access register of the instruction. Also, that ALET must be on an access list. z/OS restricts the use of that ALET such that it can be added only to the dispatchable-unit access-list of the unit of work that created the subspace (that is, the owner). Thus, there can be no cross-space branching.

Access register translation

Ordinary access register translation is done for data space access. The special access register translation performed by BSG instruction is contrasted to ordinary access register translation as follows:

- The special access register translation is performed regardless of whether the CPU is in the access register mode.
- If the ALET being translated is 00000000 hex, known as ALET 0, then the base space is taken as the new address space from which instructions are fetched.
- If the ALET is 00000001 hex, known as ALET 1, then the last subspace entered by the dispatchable unit by means of a branch in the subspace group instruction, is taken as the new address space from which instructions are fetched.
- If the ALET is other than ALET 0 and ALET 1, the selected address space is located by obtaining its origin from an access list entry in a way similar to ordinary access register translation.
- When the ALET is other than ALET 0 and ALET 1, the special access register translation may be performed by using the access register translation lookaside buffer.

The branch in subspace group (BSG) instruction changes the primary space that instructions are fetched from. There is also a mechanism in the architecture to remember the base space (various fields in the dispatchable unit control table are used to keep track of things), and to restrict certain operations in subspace mode.

Using subspaces with CICS

CICS has been using subspaces run in key 9. When CICS is running in key 8, subspaces require a public storage key (protection storage key or subsystem storage protection) for proper operation. A public storage key protects CICS, and subspaces isolate the selected applications of CICS.

All applications are mapped into the base space but execute in the subspace where they are commonly mapped, both in the base and their own individual subspace. CICS can reference them in the base but they execute in the subspace.

For CICS it is not required to have an absolute separation, so the application runs with 8 and 9 in its PSW-key mask and can change to key 8. However, if it does, it opens up an isolation flaw.

This scheme was designed to separate CICS transactions. It has dramatically reduced CICS downtime.

21.2.2 64-bit subspace

As described in “Subspace mode” on page 462, CICS exploits subspaces to provide transaction isolation and data integrity. Its transactions run in subspace mode.

- Today, CICS transactions need access to 64-bit private or shared storage.
- Currently, subspaces cannot directly reference 64-bit private and shared virtual storage. They can result in 0C4 abends.
- A CICS transaction must switch using a branch in subspace group (BSG) instruction to switch to base space for accessing 64-bit private or shared storage.

BSG usage to switch in and out of subspace mode represents a performance degradation, because it causes a programming environment context change, which is now improved with z/OS V1R13.

Subspace structure support

Figure 21-4 shows the DAT structure for the base space and subspaces in the subspace group. An ASN-Second-Table-Entry (ASTE) contains information that describes an address space or subspace.

When the ASN-second-table entry is marked as valid, the ASTE contains the Address-Space-Control Element (ASCE) that defines the region or segment table origin of the top DAT table of the address space.

The ASCE also contains the designated table type (region 1st, region 2nd, region 3rd, or segment table) of the top table for the address space. With the introduction of 64-bit Common, every address space and subspace in z/OS has a region 3rd table or higher table as its top DAT table.

With this 64-bit subspace support, the base space and subspaces in the subspace group will have separate top DAT tables as designated by their respective ASCEs. These top DAT tables for both the base space and every subspace in the subspace group will point to shared lower level tables. The only exception will be the segment table that maps the 0-2G range.

This segment table will not be shared, and the base space and its subspaces will have their own segment table that maps the 31-bit virtual storage areas (the 0-2G virtual address range).

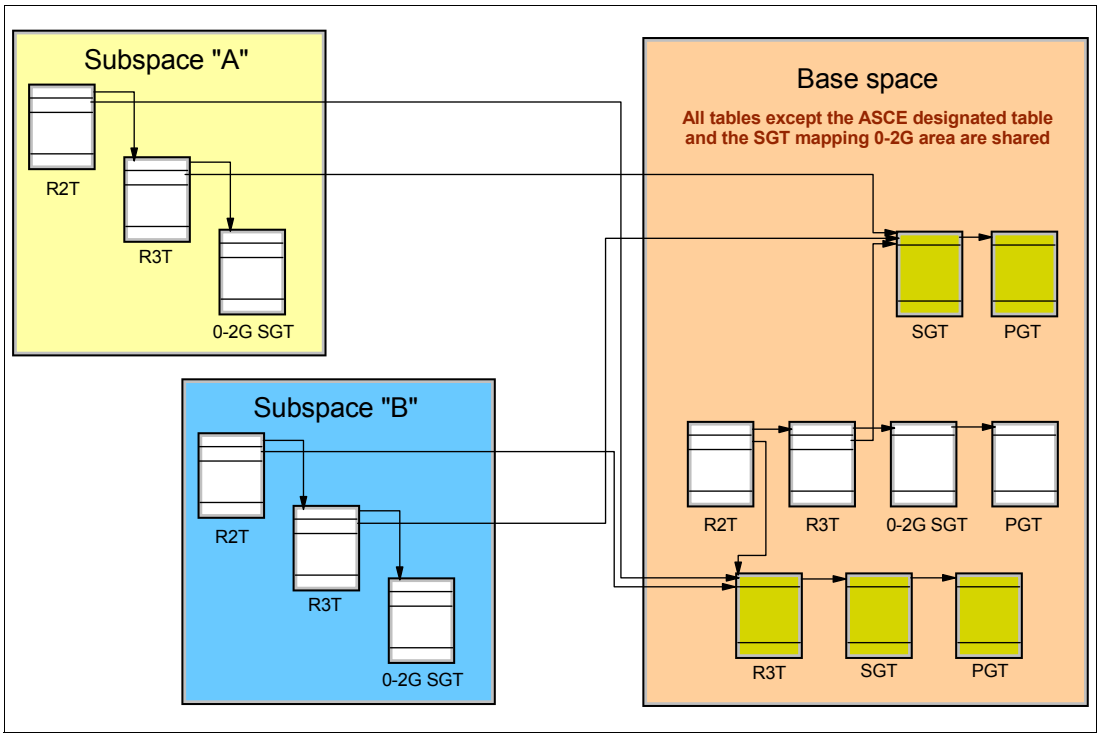


Figure 21-4 Subspace mode

64-bit subspace considerations

This new support allows a subspace to have direct access to 64-bit private, common or shared storage. Note the following points:

- Storage isolation is only for 31-bit private storage ranges; it is unchanged.
- No subspace storage isolation for high virtual storage is provided; that is, 64-bit virtual storage ranges cannot be assigned to a subspace.
- This support is rolled back to z/OS V1R12 (PTF UA58696, APAR OA34311).

This allows a subspace to directly access high virtual storage (without BSG). Consequently, it reduces context switching and thereby allows gains in performance.

Note: The implemented solution allows subspaces direct access to any 64-bit virtual storage. However, 64-bit storage ranges cannot be assigned to a subspace. Therefore, no storage isolation is supported for 64-bit storage.

Otherwise stated, with this 64-bit subspace support, application environments such as provided by CICS, are no longer strictly protected with data separation and integrity because there is no more control (beyond mere storage keys) over which application or transaction can or cannot access 64-bit shared or common storage.

Installation considerations

This new function cannot be turned off, disabled, or recycled by a command.

To that end, a new external flag in the RCE control block, RceSubspace64, is provided so that the application can test if 64-bit subspace is supported. If this bit has a value of '1'B, any application running in subspace mode can freely access any valid 64-bit virtual storage.

The new services and access calls are as follows:

- Currently, the IARV64 system service cannot be issued by subspace mode callers.
- A caller gets ABEND DC2 with reason xx0012xx. This restriction is now removed for all the IARV64 request types.
- An application in subspace mode can issue the IARV64 service to obtain a 64-bit memory object, read or write access to it, and release the memory objects.

IARV64 system services are now available to subspace mode. The following IARV64 request types can be issued in subspace mode:

GETSTOR	Create a memory object.
DETACH	Free one or more memory objects.
PAGEOUT/PAGEIN	PAGEOUT tells the system that data within physical pages of memory objects will not be used in the near future. PAGEIN is the reverse.
DISCARDDATA	Discard data within physical pages of memory objects.
CHANGEGUARD	Requests that part of a private memory object be changed from the guard area to the usable area, or vice versa.
GETSHARD	Create a memory object that can be shared across multiple address spaces.
PAGEFIX/PAGEUNFIX	Fix physical pages within nonshared memory objects.
PROTECT/UNPROTECT	Data within memory objects be made read-only. UNPROTECT makes data modifiable.
SHAREMEMOBJ	Requests that caller address space be given access to specified shared memory objects.
CHANGEACCESS	Requests the view type for segments within the specified 64-bit shared memory objects can be changed (readonly, sharedwrite, hidden).
GETCOMMON	Create a 64-bit common memory object.
LIST	Requests a list of objects be provided to the caller.

21.3 Large page enhancements

The large page support that increased the page size from 4 KB to 1 MB was introduced in z/OS V1R10 for the z10 EC models. When large pages are used in addition to the existing 4 KB page size, they are expected to reduce memory management overhead for exploiting applications.

Using large pages provided better performance by decreasing the number of translation lookaside buffer (TLB) misses an application can incur. The performance benefits turned out to be better than anticipated.

21.3.1 Large page support for the nucleus

z/OS V1R12 introduced large page support to back the nucleus. The nucleus is in 31-bit storage and so this support uses large pages in 31-bit storage. Backing the nucleus with large pages results in improved performance to the z/OS core itself.

It is expected that long-running, memory access-intensive applications will benefit from large page frames. Note that large pages are treated as fixed pages and are never paged out.

Translation lookaside buffer (TLB)

As hardware usage has shown in recent years, the translation lookaside buffer (TLB) coverage has been shrinking as a percentage of memory size, as explained here:

- Over the past few years application memory sizes have dramatically increased due to support for 64-bit addressing in both physical and virtual memory.
- TLB sizes have remained relatively small due to low access time requirements and hardware space limitations.
- TLB coverage today represents a much smaller fraction of an application's working set size, leading to a larger number of TLB misses.
- Applications can suffer a significant performance penalty resulting from an increased number of TLB misses and the increased cost of each TLB miss.

The solution introduced with z10 EC is to increase TLB coverage without proportionally enlarging the TLB size by using large pages, as follows:

- Large pages allow for a single TLB entry to fulfill many more address translations.
- Large pages will provide exploiters with better TLB coverage.

Large page performance considerations

More thorough examination is required to adequately exploit large pages, as follows:

- Large page is a special purpose performance improvement feature. It is not recommended for general use. Large page usage provides performance value to a select set of applications.

  These are primarily long-running memory access-intensive applications, for which locality of reference does not hold in the long term.
- Not all applications benefit from using large pages. Certain applications can be severely degraded by the use of large pages.

  Short-lived processes with small working sets are usually not good candidates for large pages.
- Factors to consider when trying to either estimate the potential benefit or understand measured performance differences of using larger pages instead of 4 K pages include:
  - Memory usage, and more precisely its pattern of references
  - A workload's page translation overhead

Large page potential exploiters

A future release of DB2 will support large pages for buffer pools, and Java 6.0 SR1 for z/OS is planned to support large pages. Large pages can be used to back the object heap.

Large page support

The large page support function is not enabled without the required software support. Without the large page support, page frames are allocated at the current 4 K size.

Memory reserved for large page support can be defined with a parameter in the IEASYSxx parmlib member:

```
LFAREA=xx% | xxxxxxM | xxxxxxG
```

This parameter specifies the amount of real storage to be made available for 1 MB pages, either in absolute value or as a percentage of real storage, as follows:

xx% This indicates that the amount of real storage to be used for large pages is specified as a percentage of all online real storage available at IPL time. The maximum amount of real storage that can be used to back large pages is 80% of the online storage available at IPL minus 2 GB.

If the amount specified exceeds this value, the specification is rejected and the system issues a prompt for a value to be specified on the LFAREA parameter.

The maximum amount of real storage that can be used to back large pages is 80% of the online storage available at IPL minus 2 GB. The formula to calculate the maximum amount of real storage that can be used to back large pages is listed in the following items.

xxxxxxG This indicates the amount of online real storage available at IPL time in gigabytes that is to be used for large pages. The maximum amount of real storage that can be used to back large pages is 80% of the online storage available at IPL minus 2 GB. If the amount specified exceeds this value, the specification is rejected and the system issues a prompt for a value to be specified on the LFAREA parameter.

For calculations in gigabytes, xxxxxx = online real storage at IPL in GB.
MAX_LFAREA = (x - 2G) * .8

xxxxxxM This indicates the amount of online real storage at IPL time in megabytes that is to be used to back large pages. The maximum amount of real storage that can be used to back large pages is 80% of the online storage available at IPL minus 2 GB. If the amount specified exceeds this value, the specification is rejected and the system issues a prompt for a value to be specified on the LFAREA parameter.

For calculations in megabytes, x = online real storage at IPL in MB.
MAX_LFAREA = (y - 2048M) * .8

Important: Default Value is none (No LFAREA is defined). This parameter cannot be changed dynamically during an IPL.

z/OS V1R13 IAR021I message

When the following message is issued, it means that the LFAREA parameter was specified but sufficient storage is not available:

```
IAR021I THE LFAREA WAS SPECIFIED BUT SUFFICIENT STORAGE IS NOT AVAILABLE
```

Note: If the use of large pages is required, ensure that enough storage is installed for the z/OS image. Large pages cannot be backed below 2 GB, so you need more than 2 GB of real storage to use large page support. Also, if 80% of the online storage at IPL minus 2 GB is less than 1 MB, the LFAREA specification is ignored.

21.3.2 Large page coalesce support starting with z/OS V1R12

With the large page support, installations defined how much main storage was to be used for large pages through the LFAREA parameter in the IEASYSxx parmlib member. The disadvantage of main storage backing unused large pages is that it cannot be used during page shortage situations.

The function provides large page coalesce support. During page shortage situations, RSM will convert available large pages into 256 usable 4 K pages. When the shortage condition is over, large page coalesce will occur and the 256 4 K pages will revert back to form one large page. This reduces the chance that applications will not be able to get 4 K pages during page shortage situations.

APAR OA31116

APAR OA31116 fixes large page-related problems:

- Installations can use the LFAREA parameter to define the size of their large page area. After it is defined, this area can only be used for large page requests. The only exception to this rule was when the system was storage constrained. When storage is constrained, RSM breaks up a large page into 256 $K pages to satisfy 4 K page requests. When storage is no longer constrained, RSM will perform a large page coalesce (which reforms the 256 pages that were broken up into 4 K pages back into a large page). This APAR fixes various large page coalesce problems.
- RSM recovery did not properly clean up large pages during a task or address space termination if large pages or a single large page are allocated. This can result in a permanent loss of those large pages and a single large page.

DISPLAY VIRTSTOR,LFAREA

With z/OS V1R13, a new command has been introduced that allows you to display what is the LFAREA current usage, as follows:

```
D VIRTSTOR,LFAREA
 IAR019I  11.34.10 DISPLAY VIRTSTOR 846
  SOURCE =  OO
  TOTAL LFAREA = 100M
  LFAREA AVAILABLE = 100M
  LFAREA ALLOCATED (1M) = 0M
  LFAREA ALLOCATED (4K) = 0M
  MAX LFAREA ALLOCATED (1M) = 12M
  MAX LFAREA ALLOCATED (4K) = 0M
```

The source of the LFAREA specification is as follows:

- TOTAL LFAREA is the total size of the LFAREA in megabytes. The amount displayed is the amount specified by the installation (or defaulted to) using the LFAREA keyword in the IEASYSxx parmlib member.
- LFAREA AVAILABLE is the amount, in megabytes, of available 1 M pages.
- LFAREA ALLOCATED (1 M) is the amount, in megabytes, of allocated 1 M pages on behalf of 1 M page requests.
- LFAREA ALLOCATED (4 K) is the amount, in megabytes, of allocated 1 M pages on behalf of 4 K page requests.
- MAX LFAREA ALLOCATED (1 M) is the high water mark, in megabytes, of allocated 1 M pages on behalf of 1 M page requests.
- MAX LFAREA ALLOCATED (4 K) is the high water mark, in megabytes, of allocated 1 M pages on behalf of 4 K page requests.

21.4 RSM component trace

Message IAR007I now provides information about the data space names for RSM trace buffers, as follows:

```
IAR007I RSM COMPONENT TRACE DATA SPACE IS NAMED "SYSIARnn"
```

This message is issued when the operator starts the real storage manager (RSM) component trace using the **TRACE CT** operator command and requests that RSM use a data space for its trace buffers.

Use the data space name of this message to obtain the RSM trace buffers in an SVC dump requested by a DUMP command or in a stand-alone dump. However, to obtain the dump when the RSM trace is turned off or through the use of the DMPREC option, the data space name is unnecessary.

22

Common Information Model

The Common Information Model (CIM) is a standard data model developed by a consortium of major hardware and software vendors including IBM. The consortium is known as the Distributed Management Task Force (DMTF) and is part of the Web Based Enterprise Management (WBEM) initiative.

WBEM includes a set of standards and technologies that provide management solutions for a distributed network environment. Interoperability is a major focus of WBEM, and using WBEM technologies can help you to develop a single set of management applications for a diverse set of resources.

This chapter describes the latest enhancements to Common Information Model (CIM) provided by z/OS V1R13.

After providing an introduction to CIM in a brief overview, the following details regarding upgrades or enhancements specific to z/OS V1R13 are discussed:

- Comply with OpenPegasus 2.11
- Support for the WS-Management protocol
- Support CIM Schema version 2.25
- Enhance the Java client part to Java Version 2.1.5
- Indication support for
  - FICON channel port controllers through CIM_FCPortController
  - FICON channel paths through CIM_InitiatorTargetLogicalUnitPath

22.1 Introduction to Common Information Model

As mentioned, the Common Information Model is a standard data model developed by a consortium of major hardware and software vendors including IBM. CIM was introduced in z/OS V1R7.

The idea behind CIM is to allow management applications from different vendors to manage a heterogeneous environment of systems through the same technology. All applications operate on the same set of common data (the standard CIM Schema), using the same access protocol (CIM-XML over HTTP).

There is no need for vendors of management applications to either ship their own set of instrumentation or to install their own agent technology on the systems to be managed.

Specific attributes of the various platforms are still available through CIM and can be either ignored or dynamically discovered by management applications. It is also still possible to create management applications for a specific platform only, by exploiting the extended CIM classes created for that platform. In the future we will be able to avoid the installation of multiple management agents on the managed systems. The infrastructure for all types of management will be the generic CIM Server.

22.1.1 CIM cross-platform management

Figure 22-1 illustrates the concept of management applications from different vendors managing a heterogeneous environment of systems using the same technology. All of the applications are operating on the same set of common data, such as the standard CIM Schema, using the same CIM-XML over the HTTP access protocol.

As shown in the figure, a CIM client application requests the CIM Server to return information about z/OS resources, which in this case is about basic z/OS data and RMF metrics. The CIM Server invokes the appropriate CIM providers, which retrieve the requested data associated to z/OS system resources. The z/OS RMF monitoring provider invokes the RMF Distributed Data Server (DDS), which in turn collects RMF Monitor III performance data. The CIM Server consolidates the data from the providers and returns them back to the calling client through the CIM/XML over HTTP access protocol.

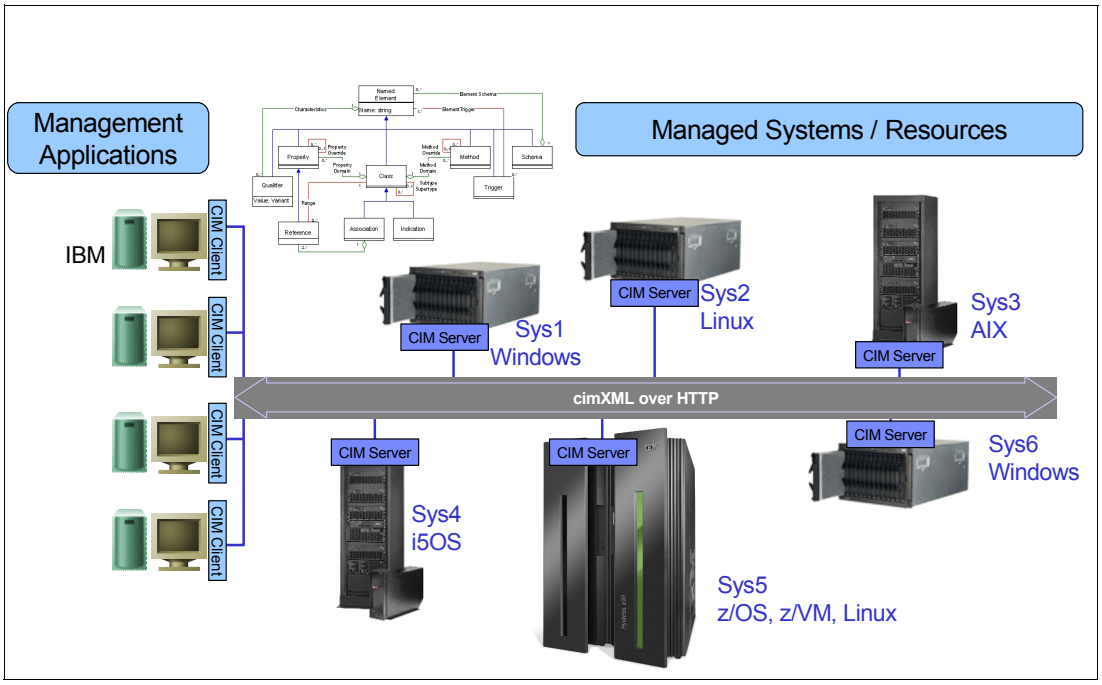


Figure 22-1 Cross-platform management with CIM

22.1.2 CIM components and dependencies

The overall goal of this open architecture is to provide simplified systems management functionality through a common, easy-to-use interface. However, you cannot simply put a GUI onto the operating system and expect something worthwhile to happen. The ability to interact with z/OS resources requires a concerted effort from the user interface (UI), to a remote interface, to a set of abstractions that organize the management operations functions provided by the operating system into understandable resource models, down to the low-level functions that carry out these operations on z/OS itself.

With support for the CIM Server on systems running z/OS, users have the ability to access z/OS resources through an extendible industry standard model. The CIM Server, shown in Figure 22-2, is used to receive client requests, collect the requested metrics and data from the managed system, and return the results to the client.

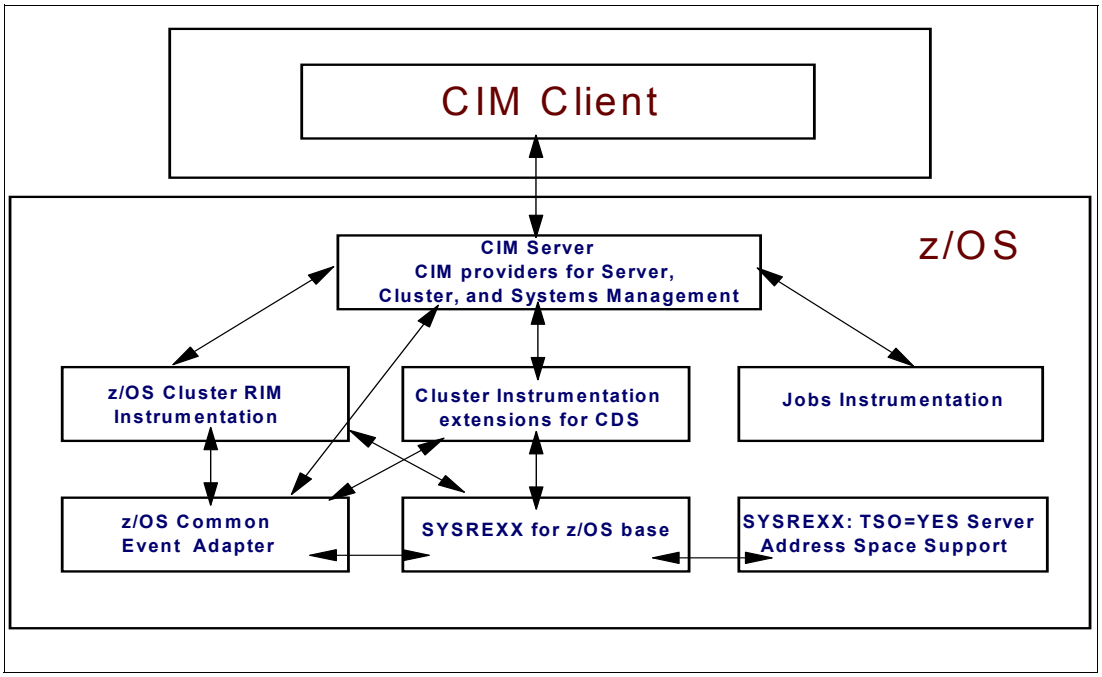


Figure 22-2 CIM components and dependencies

CIM Server in z/OS environment

The CIM Server is benefiting from the functionality of the common event adapter (CEA). CEA is a z/OS component that provides the ability to deliver z/OS events to C language clients. A CEA address space is started automatically during initialization of every z/OS system. For the address space to start successfully, you must configure CEA to work with z/OS. Failure to do so will cause CEA to run in a minimum function mode.

Jobs instrumentation

The jobs instrumentation enables the jobs and process CIM provider to invoke the proper system interfaces to obtain and change the status of batch jobs and z/OS UNIX processes; see Figure 22-3 on page 475.

The jobs instrumentation extends the reach of JES and z/OS UNIX System Services management up to the CIM Server.

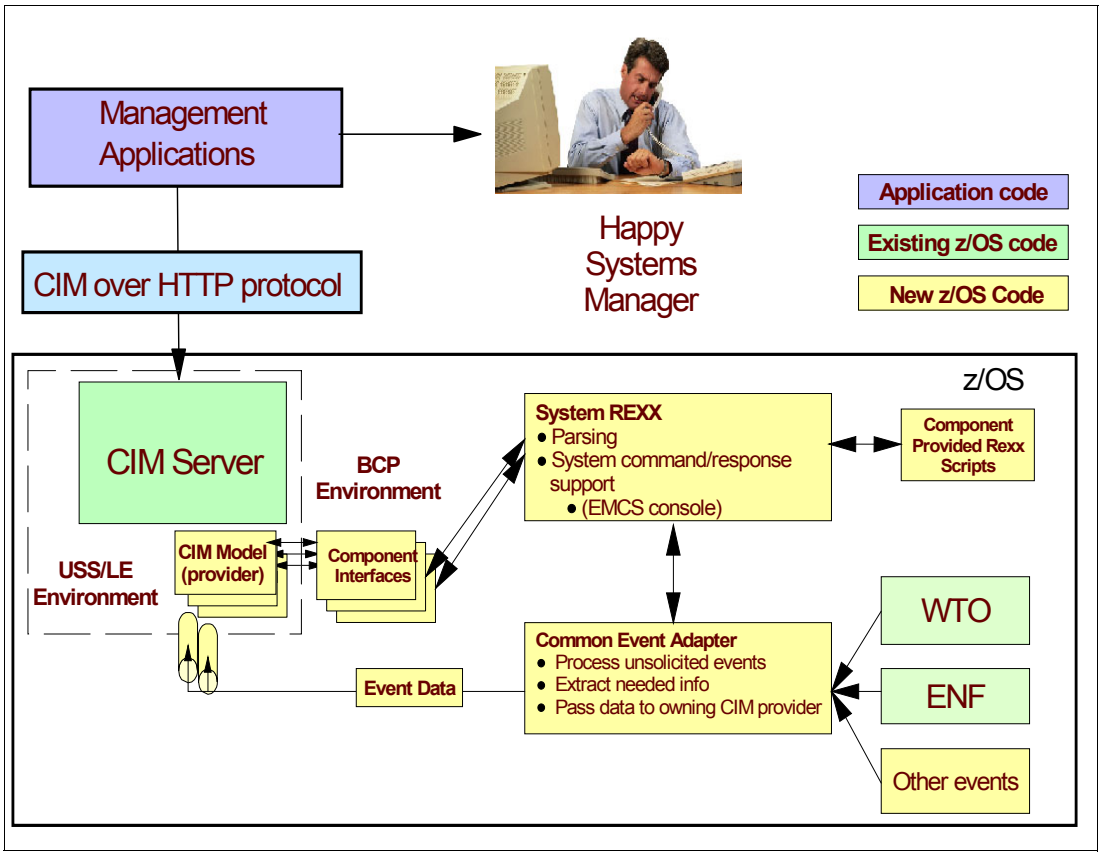


Figure 22-3 Cluster and jobs instrumentation

22.1.3 CIM server and z/OSMF

CIM is a z/OSMF prerequisite because it is needed for Incident log and WLM z/OSMF plug-in.

z/OSMF and zSystem accelerators

As of z/OS V1R11, z/OS CIM server processing is eligible to run on the System (zIIP). This includes the CIM server and CIM provider workloads, as shown in “zIIP support” on page 483).

Other CIM-related workloads (such as CIM client and CIM-enabled resource systems processing) are not eligible for zIIP. Parts of z/OSMF V1R13 use the z/OS CIM Server, so this workload is eligible for the zIIP. The z/OSMF application is written in Java and is eligible for zAAP.

Security considerations

A user will need to authenticate to z/OSMF. Additional system requirements exist such as:

- To obtain contents of a spool file for a job, the user must have access to the JESSPOOL profile associated with the spool data set.
- To submit a job, the user must be authorized to run jobs on the system and be able to access any protected resources that the job might require.
- To cancel a job, change a job's class, or purge a job, the user must be authorized to use the CIM server and be permitted to the JES3 or JES2Jobs CIM provider.
- To cancel a job or purge a job, the user must be authorized to cancel the job.

z/OSMF setup

Because CIM is a prerequisite of z/OSMF, in our case we have installed and set up CIM as part of the global z/OSMF setup. The reader can find details of these set-ups in Appendix A, “Setting up WebSphere OEM, z/OSMF, CIM, and Capacity Provisioning” on page 765.

Note: Although z/OSMF documentation refers to using the “CIM server quick setup and verification” process, we found that further customization was required in our environment. Specifically this was the must-stay-clean feature that was enabled in our environment (by the existence of RACF FACILITY class BPX.DAEMON profile). RACF program control profiles were required for z/OSMF non-core functions, as described in *z/OS Common Information Model User's Guide*, SC33-7998.

22.1.4 CIM Server overview

Since z/OS V1R9, the CIM Server has been composed of several components, as shown in Figure 22-4.

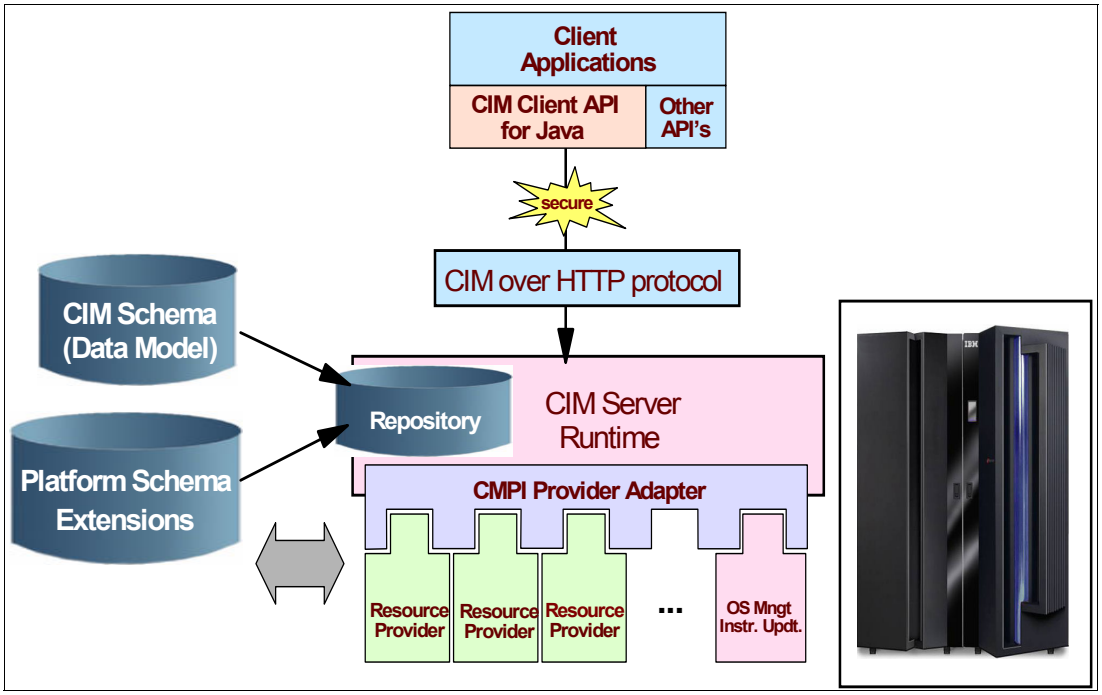


Figure 22-4 CIM Server overview

CIM components

The CIM components are described here:

- CIM client API for Java

  The CIM client for the Java library from the SBLIM project is included with z/OS CIM. The CIM Client for Java is a programming API that enables z/OS applications written in Java for local and remote access of CIM instrumentation through the CIM-XML over HTTP access protocol. It consists of a Java library and associated online Java documentation.

- CIM Server runtime

  The CIM Server runtime environment security is split into authentication and authorization. Authentication is always enabled for the CIM Server. The CIM Server supports authorization through the RACF class WBEM, in which currently the single profile

CIMSERV restricts access to the CIM Server. Since z/OS V1R9, this environment has been enhanced with the following function support:

– General updates and improvements

– Automatic Restart Manager (ARM) support

  To extend the server availability, the CIM Server is enabled to exploit the features of the Automatic Restart Manager. If an ARM policy has been defined for CIM and the CIM Server is authorized to register with ARM, then the CIM Server will be automatically restarted by ARM.

  The ARM element name CFZ_SRV_<SYSNAME> must be defined as shown in Figure 22-5.

```
DEFINE POLICY NAME(CFZARMPO) REPLACE(YES)

   RESTART_GROUP(CFZCIMRESGRP)
   /* List all systems where the CIM Server can be started  */
   TARGET_SYSTEM(SC63,SC70)
   /* Wait 10 sec before restarting to free resources       */
   RESTART_PACING(10)

   ELEMENT(CFZ_SRV_*)
     RESTART_ATTEMPTS(3,300)
     RESTART_TIMEOUT(300)
     READY_TIMEOUT(300)
     /* cross-system restart is not allowed.                */
     /* No restart after system failure                     */

     TERMTYPE(ELEMTERM)
     RESTART_METHOD(ELEMTERM,STC,'S CFZCIM')
```

Figure 22-5 Sample ARM policy

  In a sysplex, no cross-system restarts are allowed, because there is always one CIM Server per MVS image.

– SSL certificate-based authentication

  The CIM Server is enabled to exploit the AT-TLS facilities to authenticate CIM clients. AT-TLS provides a feature to enable and use SSL/TLS connections and encryption for communication with the CIM clients. Since z/OS V1R11, the CIM Server is aware of AT-TLS, and CIM clients can be authenticated through certificates as shown in Figure 22-6.

– The logging facility is changed to use the syslog daemon.

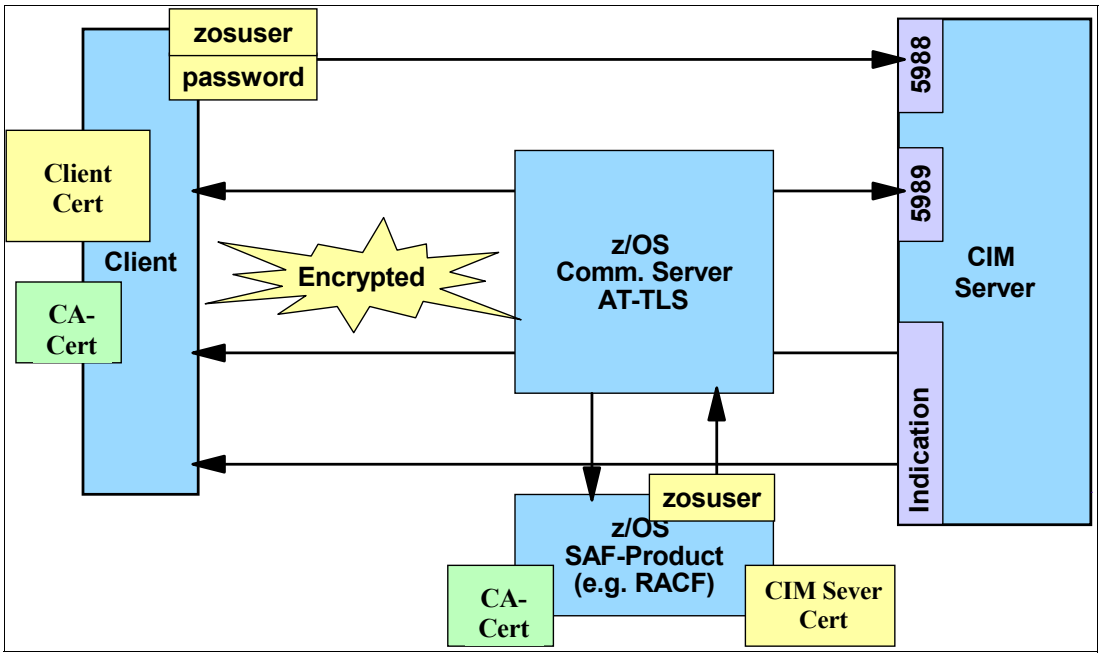


Figure 22-6 SSL certificate-based authentication

- Operating system management instrumentation

  The CIM base classes have been extended with several new classes such as IBMzOS_LogicalDisk, which provides support for logical disk volumes. New instrumentation for job and sysplex management has been added for the management of jobs, sysplex, and Coupling Facility resources through CIM.

CIM security

Because this support allows for the modification of key system resources, maintaining appropriate authorization is key, as explained here:

1. The CIM Server authenticates the user.
2. The CIM Server checks the authority of the user based on provider-level checks.
3. The credentials of the user are propagated from the CIM Server, through the provider, down to the CEA, SYSREXX and other component interfaces, where authorization checks are made.

The secure check, shown in Figure 22-7, requires the following security customization definitions when CEA is to be started in full function mode:

- Configure CEA to work with z/OS by updating the RACF database to permit CEA to use the Automatic Restart Manager. Use this command:

```
ADDUSER CEA DFLTGRP(SYS1) OMVS(UID(0) HOME('/') FILEPROCMAX(1024)) SPECIAL
RDEFINE STARTED CEA.** STDATA(USER(CEA) GROUP(SYS1) TRACE)
```

- Define the OMVS segment that allows CEA to work in the UNIX environment. Use this command:

```
ADDUSER userid DFLTGRP(SYS1) OMVS(UID(0) HOME('/') FILEPROCMAX(1024))
SPECIAL
SPECIAL
```

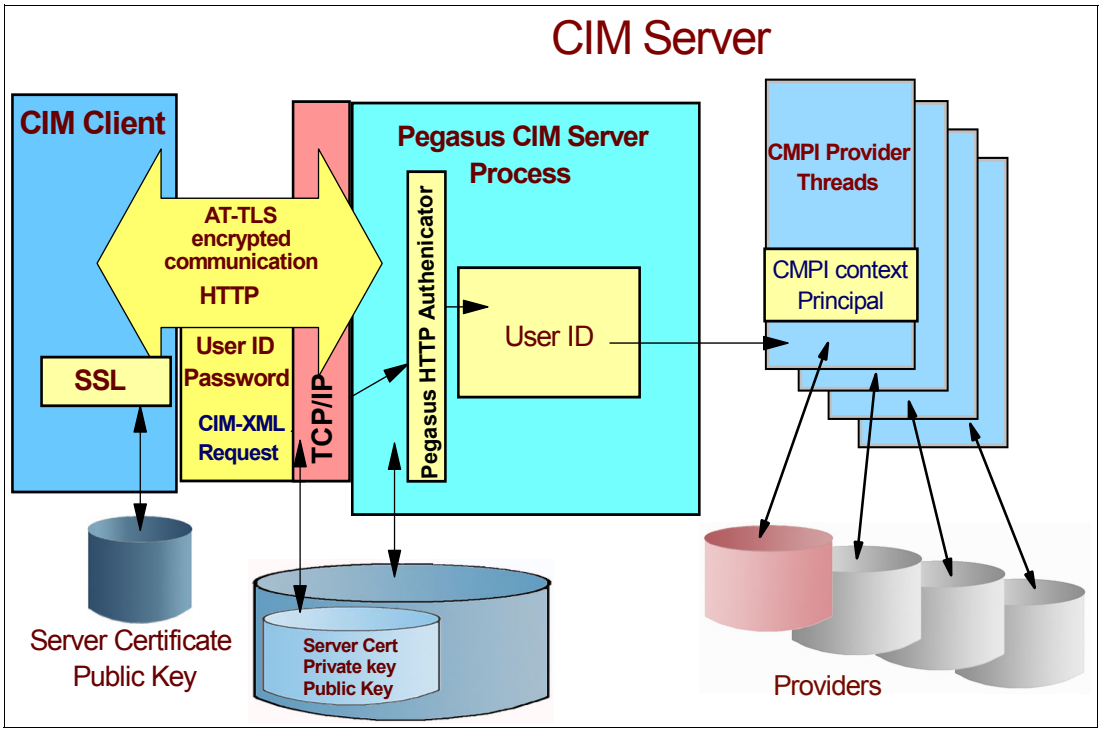


Figure 22-7 CIM client logging on to a CIM Server

The RACF user ID may be CEA, but because CEA is a started task, it is not required to be. If the STARTED class or started procedures table (ICHRIN03) contains another user ID, CEA will have that user ID assigned to it.

For example, a generic entry might specify that a user ID such as STCUSER or IBMUSER is to be assigned to any started task that is not defined with its own entry. If the user ID CEA was not set up and assigned to the started task, then the generic entry will be used and an IEF695I message will indicate that START CEA was assigned to the generic user ID.

If the default user ID that is assigned does not have an OMVS segment, then a default OMVS segment is sought through the FACILITY class profile BPX.DEFAULT.USER. RACF does not use the default OMVS segment unless the task is running with a RACF-defined user ID.

22.1.5 CIM client-to-CIM Server access

Communication between the CIM Server and a CIM client using AT-TLS and RACF uses a client's user ID and a password. SSL certificates and encryption can be used for communication with the CIM clients, because AT-TLS is enabled to authenticate CIM clients by SSL certificates in cooperation with RACF, as shown in Figure 22-7. The CIM client sends an SSL certificate to AT-TLS. Then AT-TLS sends the certificate to RACF, and RACF associates the certificate to the appropriate user ID, which then can access the CIM Server. Vice versa, the CIM Server returns its responses to the client's requests using SSL certificates.

CIM Server

For the CIM Server for z/OS, users log in over HTTP or HTTPS using basic authentication. When logging in, users are authenticated using their z/OS user ID and password as defined, for example, in the Resource Access Control Facility (RACF).

A CIM user must have at least READ access to the CIMSERV RACF profile in the WBEM class to access the CIM Server. To use any of the administrative command line tools of the CIM Server, a user instead requires CONTROL access to the CIMSERV profile.

The CIM Server authenticates users with the z/OS Security Server RACF to determine which users can log into it. Authentication is performed for every new connection (local or remote) before a user is granted access to the CIM Server, as shown in Figure 22-7.

The CIM Server offers an optional authorization check. This check is optionally performed on a per provider basis, meaning that a RACF profile in the WBEM class can be related to a single provider library. Correlation between a provider and a RACF profile occurs during provider registration by the addition of a property in the PG_Provider class.

The provider-based authorization is defined by the vendor of a provider rather than by the CIM Server administrator. Therefore, specific RACF requirements will need to be documented on a per provider basis.

22.1.6 CIM enablement

z/OS CIM comes with the z/OS package and is installed through SMP/E. After a successful SMP/E installation, the components of z/OS CIM are located in the `/usr/lpp/wbem/` hierarchical file system directory. The configuration files and repository are located in the `/var/wbem/` directory.

The CIM Server runs as a daemon in z/OS UNIX. The first-time installation of CIM is straightforward. It requires a RACF security setup, a start procedure for the PROCLIB and several environment settings in z/OS UNIX. For more information, see *z/OS V1R13 Common Information Model User's Guide*.

During startup, the z/OS CIM Server automatically corrects missing file tags. In addition, it detects whether an existing repository is current. If back-level, the CIM Server automatically upgrades the repository. If needed, the CIM Server migrates the previous repository content to the current repository.

If there are any quotes in the cimserver.env file in `/etc/wbem`, the CIM Server removes the quotes, stops the startup procedure, and requires you to restart the CIM Server.

```
S CFZCIM
 $HASP100 CFZCIM  ON STCINRDR
  IEF695I START CFZCIM   WITH JOBNAME CFZCIM   IS ASSIGNED TO USER
 CFZSRV  , GROUP CFZSRVGP
  $HASP373 CFZCIM   STARTED
  IOEZ00397I Recovery statistics for CFZSRV.ZFS:
    elapsed time 20 ms 1 log pages
    61 log records, 5 data blocks modified
    54 redo-data, 0 redo-fill records
    0 undo records, 0 unwritten blocks
  IRR812I PROFILE ** (G) IN THE STARTED CLASS WAS USED 547
          TO START BPXAS WITH JOBNAME BPXAS.
 $HASP100 BPXAS   ON STCINRDR
  $HASP373 BPXAS    STARTED
BPXP024I BPXAS INITIATOR STARTED ON BEHALF OF JOB CFZCIM 1 RUNNING IN ASID 0025
 CFZ12580I: CIM server running eligible for zIIP.
 CFZ10025I: The CIM server is listening on HTTP port 5988.
 CFZ10028I: The CIM server is listening on the local connection socket.
 CFZ10030I: Started CIM Server version 2.11.0 Development.
 CFZ12532I: The CIM server successfully registered to ARM using
 element name CFZ_SRV_SC74
CFZ00001I: PGS18204: SLP Registration Initiated
```

Figure 22-8 CIM Server startup

22.1.7 CIM client API for Java

A CIM client API for Java is provided to address the requirement of a Java-based client API to exploit the CIM Server on z/OS; see Figure 22-9. This is expected to enable vendors and other exploiters to write CIM client applications in Java on z/OS.

It supports the CIM-XML over HTTP(S) protocol. The interface, which is not yet standardized, uses JSR. The code is located at `/usr/lpp/wbem/jclient`. Version 1.3 is shipped under the form of a client library named sblimCIMClient.jar, with a configuration file located in cim.defaults. API Java documentation is available in `sblim-cim-client-doc.zip`. Further information can be found at:

`http://sourceforge.net/projects/sblim/`

The SBLIM CIM client is a pure Java-based implementation of the following:

- The WBEM operations API
- The CIM metamodel representation
- An indication listener

  In CIM terminology, an *indication* is the representation of the occurrence of an event. For example, an event can be the unexpected termination of a program, or the modification of a property value of a CIM instance. For example, an event can be the unexpected termination of a program, or the modification of a property value of a CIM instance. There is not necessarily a one-to-one correspondence between events and indications. In particular, multiple indications can be generated for the same underlying event if multiple CIM client applications had subscribed for the event.

  An event can also occur without causing a related indication to be raised (for example, if no subscription was made for the event).

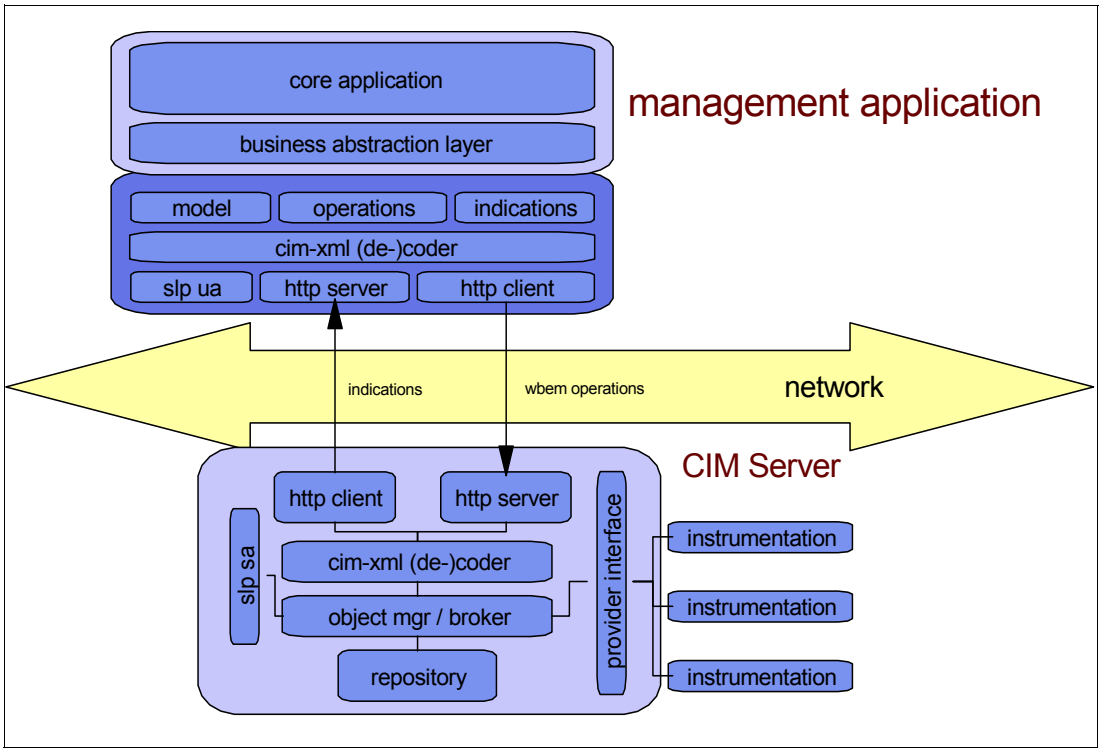


Figure 22-9 CIM client API for Java

22.1.8 CIM client/server implementation

The z/OS base element Common Information Model (z/OS CIM) implements the CIM Server, based on the OpenPegasus open source project. A CIM monitoring client invokes the CIM Server, which in turn collects z/OS metrics from the system and returns it to the calling client.

Figure 22-10 illustrates how the CIM Server works in the z/OS environment, as described here:

1. A CIM client application requests the CIM Server to return information about z/OS resources, in this case about basic operating system data and RMF metrics.
2. The CIM Server invokes the appropriate CIM providers, which retrieve the requested data associated to z/OS system resources.
3. The z/OS RMF monitoring provider invokes the RMF Distributed Data Server (DDS), which in turn collects RMF Monitor III performance data.
4. The CIM Server consolidates the data from the providers and returns it back to the calling client through the CIM-XML over HTTP protocol.

In addition, Figure 22-10 shows two types of CIM providers:

- RMF monitoring providers that use the RMF DDS to access the z/OS system data

  A CIM monitoring client invokes the CIM Server which, in turn, collects z/OS metrics from the system and returns it to the calling client. To obtain the z/OS metrics, the CIM Server invokes the z/OS RMF monitoring provider, which retrieves the metrics associated with z/OS system resources. The z/OS RMF monitoring provider uses existing and extended RMF Monitor III performance data. The metrics obtained by this new API are common across eServer platforms, so you can use it to create end-to-end monitoring applications.

- z/OS operating system management providers that access the z/OS system data directly

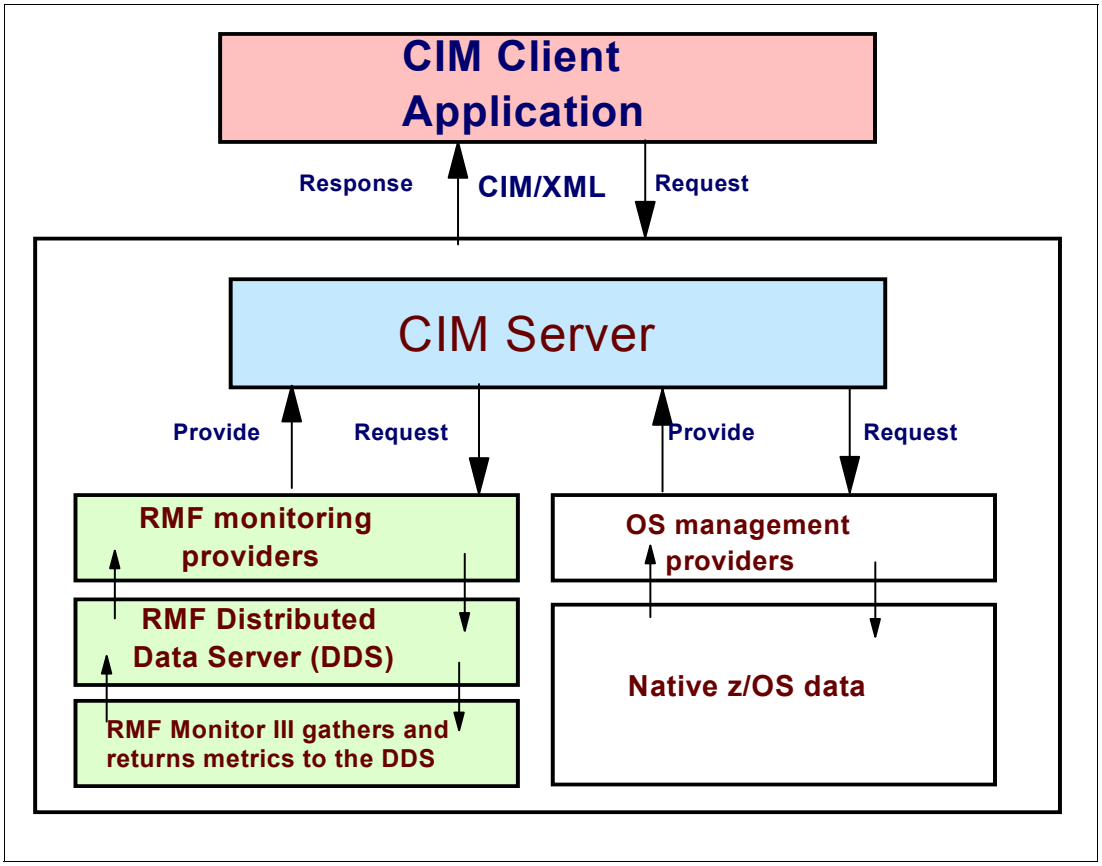


Figure 22-10 Client/server requests for data to z/OS using CIM

22.1.9 zIIP support

With z/OS V1R11, the CIM server and its data providers have been enabled to run zIIP. This alleviates costs caused by non-business-related workload (that is, systems management) from consuming valuable central processor resources, specially with z/OSMF V1R13 for which CIM is a prerequisite. If a zIIP is available in the system, the CIM server will automatically switch to zIIP.

Function overview

The support for CIM server to “switch” to zIIP is introduced with z/OS V1R11.

The CIM server and its data providers are eligible for zIIP. Applications that access CIM-enabled resources and providers can benefit, as described here:

- Information providers such as RMF, WLM, DFSMS, and BCP
- Systems management exploiters such as System z Capacity Provisioning Manager and z/OS Management Facility

22.2 Enhancements with z/OS V1R13

z/OS V1R13 CIM keeps up with the latest CIM / WBEM systems management standard, as follows:

- CIM server is updated to OpenPegasus 2.11, which is the latest version available in OpenSource as of 2011 and includes support for the Web Services-Management (WS-Management) SOAP-based protocol.
- CIM server in z/OS V1R13 is also updated to support CIM Schema version 2.25.
- CIM Client is updated from Java Version 2 to Version 2.1.5

There is no change in the way the CIM Server and command line tools are invoked.

22.2.1 SMI-S support enhancement

The SMI-S specifications define the various domains of storage management in the form of CIM profiles and sub-profiles. For z/OS CIM V1R12, the Storage HBA and Host Discovered Resources (HDR) profiles were implemented. For z/OS CIM V1R13, the SMI-S instrumentation is enhanced for the Storage HBA and SB Multipath Management (which is a sub-profile of HDR) profiles by adding support for CIM Indications.

The support of CIM Indications avoids the periodic polling of management applications such as TPC for monitoring resource or resource state changes.

Background information

The Storage Management Initiative Specification (SMI-S) was developed by members of the Storage Networking Industry Association (SNIA) and defines “an interface for the secure, extensible, and interoperable management of a distributed and heterogeneous storage system. This interface uses an object-oriented, XML-based, messaging-based protocol designed to support the specific requirements of managing devices and subsystems in this storage environment. Using this protocol, this Technical Specification describes the information available to a WBEM Client from an SMI-S compliant WBEM (CIM) Server.”

The “WBEM Client” mentioned is a storage and/or SAN management application such as the IBM TotalStorage Productivity Center (TPC), IBM Tivoli, and other vendor products. The “SMI-S compliant WBEM Server” is a CIM Server (like OpenPegasus on z/OS or alternative CIMOMs on other platforms) that supports the set of profiles and classes defined by SMI-S and/or DMTF as required or useful in one or more management domains.

New z/OS V1R13 support

SNIA storage HBA (Host Bus Adapter) profile enhancements are implemented for FICON channel ports on z/OS since V1R12.

z/OS V1R13 is now supporting events for HBA inserts and removals through CIM indications.

SNIA SB Multipath Management enhancements, which describes the paths between ports and I/O devices, are implemented for FICON ports and storage devices on z/OS since V1R12.

z/OS V1R13 now supports events for path changes, such as:

- Path creation
- Path state changes (online/offline)
- Path removal

z/OS V1R13 avoids polling CIM objects for FICON channel ports and channel paths to disk devices. To achieve this objective, z/OS V1R13 CIM supports CIM indications (events) for:

- CIM_PortController (representing FICON channel ports) for:

- HBA inserts and removals

- CIM_InitiatorTargetLogicalUnitPath (representing channel paths to disk devices), at:
  - Path creation
  - Path state changes (online/offline)
  - Path removal

Thus, z/OS V1R13 facilitates asynchronous notification through CIM about such changes.

CIM indications

In the overview provided in *z/OS Common Information Model User's Guide*, SC33-7998, and shown in Figure 22-11, to bet notified about state changes of resources that are managed through CIM, a CIM Client subscribes to the CIM Server for one or more CIM indications.

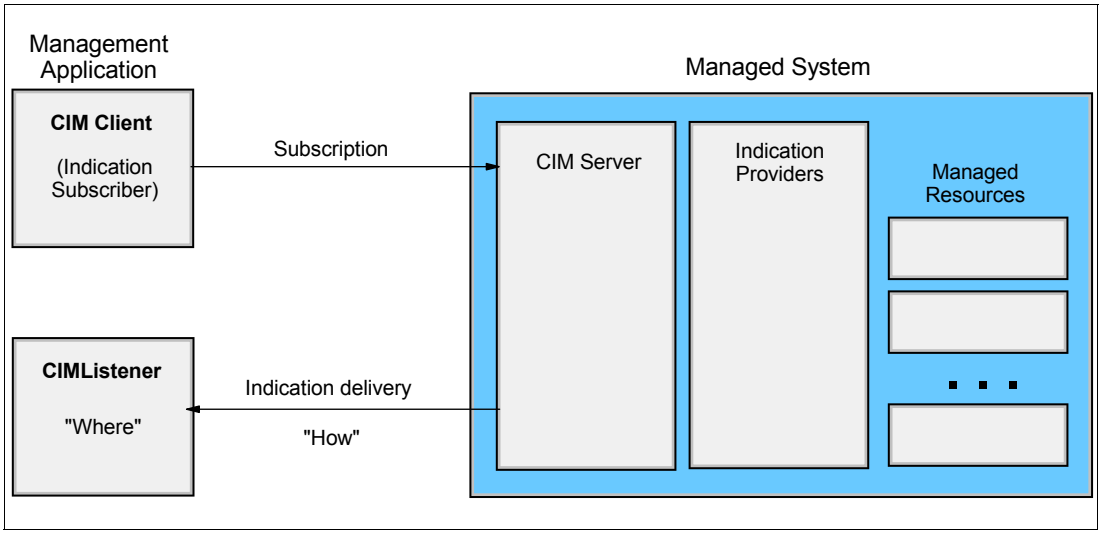


Figure 22-11 CIM indications overview

The CIM Server routes this request to the CIM providers that are responsible for monitoring the according resources.

If the state of resource changes for which a client is subscribed, the CIM provider notifies the CIM Server and the CIM Server sends an indication to the Indication Listener, which was defined in the Indication subscription.

CIM indication usage

The SMI-S specification defines that an implementation may optionally support asynchronous notification of HBA inserts and removals using the InstCreation and InstDeletion indications.

For this the following CIM life cycle indications are added in z/OS CIM V1R13 for class IBMzOS_PortController:

- SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_PortController, it indicates a PortController (HBA) creation.
- SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_PortController, it indicates a PortController (HBA) removal.

To allow Storage Management Applications to subscribe for changes on the relationship of target and initiator ports, the following CIM life cycle indications are added in z/OS CIM V1R13 for class IBMzOS_SBInitiatorTargetLogicalUnitPath:

- SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA CIM_InitiatorTargetLogicalUnitPath indicates a path creation.
- SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA CIM_InitiatorTargetLogicalUnitPath indicates a path deletion.
- SELECT * FROM CIM_InstModification WHERE SourceInstance ISA CIM_InitiatorTargetLogicalUnitPath AND SourceInstance.State <> PreviousInstance.State indicates a path state change.

Subscribing to a CIM Indication is a three-step process:

- Creating a Filter (CIM_IndicationFilter)

  Defines the criteria about which change the subscriber wants to be notified. A filter is defined in the form of a CQL query.
- Creating a Handler (CIM_ListenerDestination)

  Defines where to send the indication by IP address/host name and port.
- Creating a Subscription (CIM_IndicationSubscription)

  Connects a Handler with a Filter through an association.

22.3 CIM customization and setup

CIM customization was done following the instructions in *z/OS Common Information Model User's Guide*, SC33-7998. Default directory locations, along with default RACF user ID (CFZSRV for the server) and groups (CFZADMGP and CFZUSRGP) were used, meaning that supplied jobs CFZSEC and CFZRCUST can run with minor modification.

If you choose to customize CIM using this method, reply N (the default) to the prompt “Do you need assistance in setting up security for the Common Information Model (CIM) server?” when running the z/OSMF configuration script (`izusetup.sh`).

Jobs CFZSEC and CFZRCUST might have already been customized and run as part of base z/OS installation.

Note: Although the z/OSMF documentation refers to using the “CIM server quick setup and verification” process, we found that further customization was required in our environment. Specifically this was the must-stay-clean feature which was enabled in our environment (by the existence of RACF FACILITY class BPX.DAEMON profile). RACF program control profiles were required (for z/OSMF non-core functions), as described in section Setting up Program Control in *z/OS Common Information Model User's Guide*, SC33-7998.

We advise reviewing the topic “First-time CIM Server setup” if you plan to run the CIM server in a production environment.

22.3.1 Required parmlib updates

The following parmlib parameters have to be defined to enable the job and cluster providers:

- MAXCAD limit

  This parameter defaults to 50. If the installation sets a lower limit, it can be necessary to increase this setting to accommodate the Common Event Adapter (CEA) common area data space (CADS).

- APF authorize SYS1.MIGLIB

  The following line must be added to the installation's PROGxx parmlib member to enable the CFRM-related CIM providers to function:

  ```
  APF ADD DSNAME(SYS1.MIGLIB) VOLUME(******)
  ```

- REXX alternate library

  The couple data set providers require the use of compiled REXX execs provided as part of SYSREXX. These execs require the use of the REXX alternate library. The following addition to the installation's PROGxx parmlib member is one way to accomplish this:

  ```
  LNKLST ADD,NAME(LNKLST00),DSN(REXX.V1R3M0.SEAGALT),ATTOP
  ```

Note: A sample TSO CLIST is provided that performs the necessary RACF setup to permit CEA to use Automatic Restart Manager, and to permit CEA to operate in UNIX System Services with the cluster, couple data set, and JES2/JES3 jobs CIM providers.

23

Predictive Failure Analysis

Software failures are abnormal yet allowable behaviors that can slowly lead to the degradation of the operating system. To help eliminate soft failures, z/OS has developed Predictive Failure Analysis (PFA). PFA is designed to predict if a soft failure will occur sometime in the future and to identify the cause while keeping the base operating system components stateless. PFA is intended to detect abnormal behavior early enough to allow you to correct the problem before it affects your business. PFA uses remote checks from IBM Health Checker for z/OS to collect data about your installation. Next, PFA uses machine learning to analyze this historical data to identify abnormal behavior. It warns you by issuing an exception message when a system trend might cause a problem.

This chapter describes Predictive Failure Analysis, which is a component of z/OS that was made available as an SPE in March 2009 for z/OS V1R10. PFA provides support using remote checks from IBM Health Checker for z/OS to collect data about your installation.

The first two checks became available with z/OS V1R10. The second two checks became available with z/OS V1R11 and more with further releases. The chapter discusses the highlights of PFA and explains these remote checks in more details. The following topics are covered:

- Predictive Failure Analysis overview
- New PFA checks in z/OS V1R13:
  - Enqueue request rate check
  - JES spool usage check for JES2
- New in z/OS V1R13, PFA integration with Runtime Diagnostics
- PFA infrastructure
- PFA installation

23.1 Predictive Failure Analysis overview

Predictive Failure Analysis (PFA) is designed to predict potential problems with your systems. PFA extends availability by going beyond failure detection to predict problems before they occur. z/OS has implemented PFA to help eliminate soft failures such as exhaustion of common storage usage where a low priority authorized task obtains common storage, but obtains significantly more common storage than usual. This can cause a critical authorized system component to fail when attempting to obtain a normal amount of common storage. Soft failures usually occur in four generic areas:

- Exhaustion of shared resources
- Recurring or recursive failures often caused by damage to critical control structures
- Serialization problems such as classic deadlocks and priority inversions
- Unexpected state transition

Health checks

As mentioned, PFA uses remote checks from IBM Health Checker for z/OS to collect data about your installation. Using this data, PFA constructs a model of the expected (future) behavior of the z/OS images and then compares the actual behavior with the expected behavior. If the behavior is abnormal, PFA issues a health check exception.

z/OS UNIX and Java

PFA uses a z/OS UNIX System Services (z/OS UNIX) file system to manage the historical and problem data that it collects. Java 1.4 or later is required.

23.1.1 PFA-detected system failures

Detected system failures, which are caused by abnormal behavior, often generate “sympathy sickness.” With sympathy sickness, a minor problem escalates over time to the point where the service stops working. These failures are difficult to detect, which makes failure isolation difficult. Sympathy sickness has been observed when either hard failures or abnormal behavior generates a system failure that cannot be isolated to a failing component or subcomponent.

There are three general categories of software-detected system failures:

Masked failure This is a system failure that is detected by the software and corrected by the software.

Hard failure This failure occurs when the software fails completely, quickly, and cleanly, for example when an operating system kills a process.

Soft failure This failure is caused by abnormal behavior. A system failure caused by abnormal behavior is defined as unexpected, unusual, or abnormal behavior which causes the software solution to not provide the service requested. This abnormal behavior of the software, combined with events that usually do not generate failures, produce secondary effects that might eventually result in a system failure.

Abnormal system failures

A failure caused by abnormal behavior is difficult to recognize within the component, and the system can be damaged.

Systems in this state can be referred to as “sick but not dead.” There are several categories of problems that can generate “sick but not dead” systems:

Damaged systems These are systems suffering recurring or recursive errors caused by software defects.

Serialization problems These problems can be caused by incorrect priority assignments, by classic deadlocks, or by the resource owner terminating.

Resource exhaustion These problems can be caused by physical resources such as main storage or disk storage, or by software resources such as address space control blocks.

Indeterminate states The problems caused by these states are not possible to determine.

Predictive Failure Analysis uses historical data combined with machine learning and mathematical modelling to detect abnormal behavior and its potential causes. The objective is to detect events that lead to a “sick but not dead” system and allow corrective action to be taken. In general, errors fall into categories like recovery, logical errors, over-consumption, and sympathy sickness.

23.2 How PFA chooses address spaces to track

From all the information related to all the address spaces, PFA chooses address spaces to track based on the following metrics:

- Certain metrics require data for the entire system to be tracked.
  - Exhaustion of common storage for entire system.
  - LOGREC arrivals for entire system grouped by key.
- Certain metrics call for tracking only persistent address spaces.
  - Persistent address spaces are those that start within the first hour after IPL.

    For example, track frames and slots usage to detect potential virtual storage leaks in persistent address spaces.
- Certain metrics are most accurate when using several categories.
  - Track “chatty” persistent address spaces
    - Those that start within the first hour after IPL that have the highest rates after a warm-up period
    - First hour after IPL is ignored.
    - The same address spaces will be tracked after an IPL or PFA restart if they are still running after PFA restart.
    - Duplicates with the same name are not tracked, but address spaces that were tracked and then restarted will still be tracked after restart.
- Other persistent address spaces as a group.
- Non-persistent address spaces as a group.
- Total system rate (“chatty” + other persistent + non-persistent).

23.3 Efficient PFA usage

For the best use of Predictive Failure Analysis (PFA), perform these actions:

- Reduce the number of false positives.

  To avoid false positives, PFA can perform “supervised” learning, which excludes certain data that PFA uses when making predictions of future behavior. For example, when you suspect certain jobs or address spaces are inconsistent and have the potential of being restarted often, you can increase the accuracy of PFA by excluding the job or address spaces from analysis. Supervised learning applies to the following checks:

  - “LOGREC arrival rate check” on page 501
  - “Frames and slots usage check” on page 501
  - “Message arrival rate check” on page 501
  - “SMF arrival rate check” on page 502

- Eliminate the jobs causing false positives.

  The supervised learning service can help you avoid false positives by excluding certain data that PFA uses when making predictions of future behavior. To minimize the impact to check performance, only use EXCLUDED_JOBS for the conditions that cause you the most inconvenience. Instead, use other tuning parameters for the check such as STDDEV.

  A sample EXCLUDED_JOBS file ships in the `/usr/lpp/bcp/samples/PFA` directory. It is named EXCLUDED_JOBS and includes an example comment line. You can modify the file using the **`OEDIT`** command, and then use the **`F PFA,UPDATE`** command to have PFA read in the contents of the file and start to use it in during processing.

  See 23.3.2, “Eliminate jobs causing false positives” on page 493, for more information about this topic.

- Automate the PFA IBM Health Checker for z/OS exceptions.

  Beginning with z/OS V1R12, PFA uses the single `ini` file for all checks in the `/etc/PFA` directory. This means you only have to update and maintain one `ini` file.

  If you want a specific check to use a level of Java other than the one that is specified in the `/etc/PFA/ini` directory, then provide an `ini` file in the check directory for the check. For example, create an `ini` file in the `pfa_directory/PFA_MESSAGE_ARRIVAL_RATE/` directory if you want to use another level of Java for the PFA_MESSAGE_ARRIVAL_RATE check.

- Use the check’s report in SDSF to aid your investigation.

  PFA creates report output using the IBM Health Checker for z/OS. It displays reports with the z/OS System Display and Search Facility (SDSF) and the message buffer.

- Use Runtime Diagnostics to analyze PFA results.

  A new MVS subsystem (component name HZR) is designed to help you analyze a system that has potential soft failures.

The following sections explain these actions in greater detail.

23.3.1 Reduce the number of false positives

Follow these steps to reduce the number of false positives:

- Use default configuration parameters, which have been carefully constructed to minimize configuration parameters.

- THRESHOLD, STDDEV, TRACKEDMIN, and EXCEPTIONMIN are the parameters, depending on the check.
  - A higher THRESHOLD requires a larger current usage of common storage before an exception is issued.
  - A higher TRACKEDMIN requires a persistent job to have a rate higher than this value during the warm-up period before it will be considered “chatty” enough to be tracked individually.
  - Testing has shown that most problems that can cause a damaged or hung system likely show erratic behavior such that the problem can be many standard deviations higher than normal.
    - A lower STDDEV value allows an exception to be issued if the actual rate is closer to the expected rate and the predictions across the time ranges are consistent.
    - A higher STDDEV allows an exception to be issued if the actual rate is significantly greater than the expected rate even if the predictions across the various time ranges are inconsistent.
  - Testing has shown that occasionally an exception will be issued for a small number of events. To increase the number of events needed before an exception is issued, increase the EXCEPTIONMIN parameter, if available.
    - A higher EXCEPTIONMIN value requires the prediction and the current value to be higher than this value before an exception will be issued.

23.3.2 Eliminate jobs causing false positives

False positives must be quickly eliminated during the investigation. Address spaces that are inherently unstable for a given metric or are restarted often can produce false positive exceptions; for example:

- Exclude test programs that issue many LOGRECs and cause exceptions
- Exclude address spaces that issue many WTOs, but are inconsistent or “spiky” in their behavior and cause message arrival rate exceptions

Support has been implemented to eliminate these false positives by using the supervised learning support:

- The `EXCLUDED_JOBS` file must exist in the check’s configuration directory.
  - It is read at PFA start and when the **`F PFA,UPDATE`** command occurs for check.
  - It supports wildcards.
  - The job name and system must match. The remaining data is informational.
- The format of the `EXCLUDED_JOBS` file can be found in *z/OS Problem Management*, G325-2564. For example:

```
JLA*,*,03/15/2010 12:08, Exclude all JLA* jobs on all systems
```

23.3.3 Automate the PFA IBM Health Checker for z/OS exceptions

PFA is designed to predict potential problems with your systems. It extends availability by going beyond simply failure detection to predict problems before they occur. PFA provides this support using remote checks from IBM Health Checker for z/OS to collect data about your installation.

As mentioned, by using this data, PFA constructs a model of the expected future behavior of the z/OS images and compares the actual behavior with the expected behavior. If the behavior is abnormal, PFA issues a health check exception.

PFA uses a z/OS UNIX System Services (z/OS UNIX) file system to manage the historical and problem data that it collects. It uses remote checks from IBM Health Checker for z/OS to collect data about your installation.

To automate the PFA IBM Health Checker for z/OS exceptions:

- You can add exception messages to the existing message automation product.
- You can use exception messages and other information to tailor alerts.

23.3.4 PFA detection of damaged systems

Figure 23-1 depicts how PFA can intercept a failure in the software stack and issue a PFA exception before the failure causes a “sick, but not dead” condition indicative of a damaged system. Various metrics that PFA uses in its analysis are closer to the hardware. Other metrics are closer to the application.

When the metric is closer to the hardware, it is easier for PFA to detect an error. When the metric is closer to the application, it is harder for PFA to detect an error. As illustrated in Figure 23-1:

- The LOGREC arrival rate check is closer to the hardware and detects LOGRECs grouped by program key across multiple time ranges.
- The message arrival rate check is detecting when WTO and WTOR messages are issued at an abnormal rate, which can indicate a damaged system.
- Moving closer to the application layer, the SMF arrival rate check detects an abnormal generation rate of SMF records.

All of these metrics can detect a damaged system, but the operator is alerted by PFA prior to the soft failure occurring.

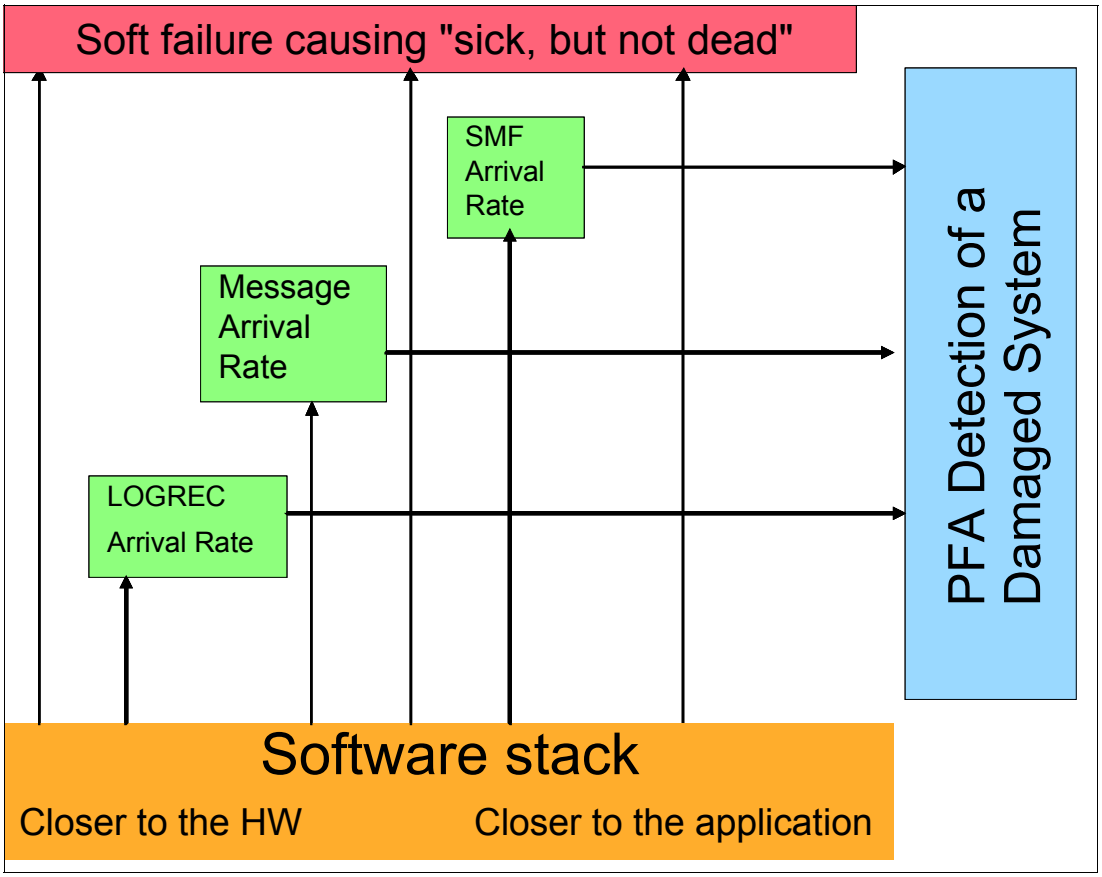


Figure 23-1 How PFA detects soft failures: a layered approach

23.3.5 Supervised learning support

PFA provides the ability to do supervised and unsupervised learning to determine what is abnormal behavior. In general, PFA is designed to require minimum client configuration. Therefore, it has always used unsupervised learning.

However, there are address spaces with behavior that is not predictable for certain checks. There are also scenarios such as combined test and productions systems where PFA is unable to accurately determine abnormal behavior.

In such cases, PFA can be configured to ignore certain jobs or sets of jobs that are making the predictions too “noisy” or are generating false positive exceptions. PFA now supports both supervised learning and unsupervised learning:

- Unsupervised learning is the machine learning that PFA does automatically.
- Supervised learning allows clients to exclude jobs that are known to cause false positives; for example:
  - Exclude test programs that issue many LOGRECs and cause exceptions
  - Exclude address spaces that issue many WTOs, but are inconsistent or spiky in their behavior and cause message arrival rate exceptions

As of /OS V1R12, a new configuration option has been made available for all checks (except the PFA_COMMON_STORAGE_USAGE check). PFA now allows an `EXCLUDED_JOBS` file. This file must exist in the `/config` subdirectory of the check’s directory.

All address spaces that are listed in this file will be excluded from processing by this check. Wildcard names are allowed for both the job name and the system name.

A sample `EXCLUDED_JOBS` file is provided in `/usr/lpp/bcp/samples/PFA`. It can, for instance, be placed as shown in Figure 23-2.

```
/u/pfauser/PFA_MESSAGE_ARRIVAL_RATE/config/EXCLUDED_JOBS
```

Figure 23-2 EXCLUDED_JOBS file placement

23.4 PFA infrastructure

The PFA infrastructure manages the PFA address space, connects to IBM Health Checker for z/OS (referred to as “Health Checker” in this chapter), displays the status of PFA, and launches the JVM to model the data to create an estimation.

There are no hardware dependencies for PFA, and exploiters of this function are all system operators. PFA simply has to be started and then monitored for the exceptions it produces; you can automate the monitoring. There are no APIs to exploit.

The software dependencies for PFA are shown in Figure 23-3. They are IBM Health Checker for z/OS, z/OS UNIX file system, and Java 1.4 or later.

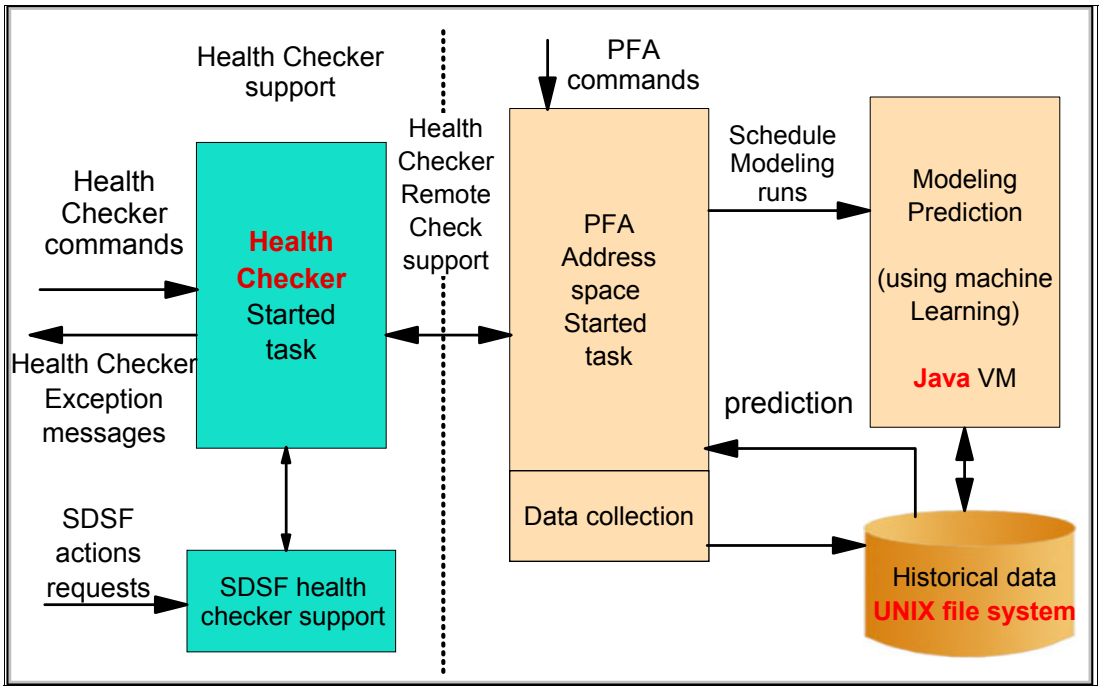


Figure 23-3 PFA infrastructure

23.4.1 PFA parameters for health checks

PFA is analyzed using remote health checks. Therefore, the health check commands, the interface through SDSF, and the reporting mechanism available through Health Checker are fully usable for the PFA checks.

PFA also contains check-specific code that collects the data for an individual check, models the data to generate a estimation, and compares the actual values to the estimations to issue an exception or an informational message.

PFA checks have three basic internal functions, as explained here:

- PFA checks collect data.

  Data collection for a check is specific to that check; the check code collects data from the system that is pertinent to the check. For example, if the check needs to calculate a type of storage usage, it interrogates the system control blocks to accumulate the storage used. If it is counting message arrivals of a certain kind, it uses the appropriate system interface to collect that data.

  Data collection occurs asynchronously on an interval that can be configured by the user. For example, the default value for the data collection function for checks might be to collect data every 15 minutes. This parameter is called COLLECTINT.

- PFA checks model the data to generate an estimation based on the data collected.

  This modeling function takes the collected data and estimates the value that it expects to see at the end of the model interval. The model interval can also be configured by the user. For example, the default value for the modeling interval for checks might be to model data every 6 hours. This parameter is called MODELINT. Modeling also runs asynchronously when it is determined that it is time to model.

- PFA checks perform the comparisons needed to issue an exception or an informational message.

  The checks compare what is occurring on the system to what was estimated and issues the appropriate message and report. This function is typically initiated by Health Checker when the time in the INTERVAL parameter for the check is reached. It can also be performed for most checks by a user running the check using Health Checker commands.

PFA and z/OS UNIX

PFA also manages its data store, which is in the UNIX file system. The collected data is stored for use by the modeling code to produce an estimation. The estimations are also stored in the file system for use by the code that performs the comparisons and produces the reports.

All PFA checks have two additional check-specific parameters: COLLECTINACTIVE and DEBUG, as explained here:

- The COLLECTINACTIVE parameter is set to yes by default. It collects and models data for the check even if the check is not active (enabled) in Health Checker.
- The DEBUG parameter is used to collect additional debug information to help analyze a PFA problem.

PFA checks can also have check-specific parameters. At this time, each check has a parameter to assist PFA in reducing false positive numbers. These parameters can also be configured by the user to allow for greater flexibility on a per-system basis.

23.4.2 Differences between PFA checks and other remote health checks

As mentioned, PFA checks have several parameters. Health Checker requires all parameters to be specified on a modify even if only one parameter is being changed. And, if using the Health Checker **`modify`** command on the command line, only 126 characters are allowed and not all PFA check parameters can be specified. In addition, it was not considered useful to need to input all parameters simply to change the value of one parameter.

Therefore, PFA made a change so that not all parameters need to be specified when modifying parameters using Health Checker. PFA internally tracks the values of each parameter so that if not all parameters are specified, the previous value for the parameters not specified are retained.

However, Health Checker is not aware of the internal storage of the PFA parameters. Therefore, it only has the capability of displaying the last modify operation performed. If you use multiple modify operations or do not specify all of the parameters, Health Checker can only display the last parameter modified. Therefore, to see all parameters in use by any PFA check, use the **`modify PFA display`** command.

When the debug parameter in Health Checker is set, PFA is not notified until the next run of the check. However, debug data needs to be generated for the collect and model phases of the checks as well. Therefore, every PFA check has a debug parameter that is a check-specific parameter applying to all phases of PFA checks. The debug parameter in Health Checker is ignored by PFA.

After an exception is issued, the exception is issued every time the check is run even though new data has not yet been collected and no new estimations have been modeled. This problem was especially annoying for system operators who carry a pager. Therefore, PFA has been adapted so that when an exception is issued, the check is deactivated in Health Checker so that the check is not run again.

- If COLLECTINACTIVE is on, then after new data is collected and new estimations are modeled, the check is activated again which immediately runs the check.
- If the exception needs to be issued again, it is issued with new data on the report and the check is deactivated again.
- If everything is now OK, the informational message is issued.
- If COLLECTINACTIVE is off, PFA will not collect or model data and the check will remain deactivated until manually activated by the user or COLLECTINACTIVE is turned on.

Figure 23-4 illustrates the overall PFA processes.

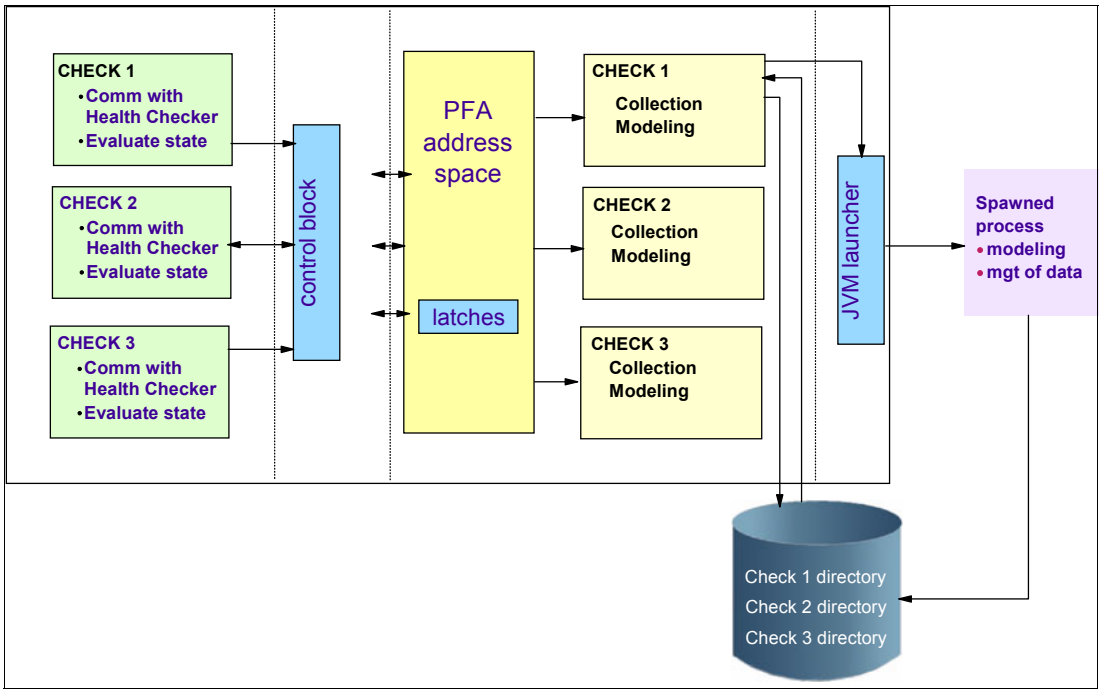


Figure 23-4 Overall PFA process

23.5 PFA and IBM Health Checker for z/OS

The PFA component was introduced in z/OS V1R10 as an SPE with two checks. The Common Storage Usage check was looking for resource exhaustion and the LOGREC arrival rate check was looking for a damaged system.

Predictive Failure Analysis (PFA) provides the following remote checks:

- PFA_COMMON_STORAGE_USAGE
- PFA_LOGREC_ARRIVAL_RATE
- PFA_FRAMES_AND_SLOTS_USAGE
- PFA_MESSAGE_ARRIVAL_RATE
- PFA_SMF_ARRIVAL_RATE

New in z/OS V1R13:

- PFA_ENQUEUE_REQUEST_RATE
- PFA_JES_SPOOL_USAGE check for JES2

Old in z/OS V1R13:

- JES3 spool usage command

When PFA issues an exception, the PFA check's WTOTYPE parameter changes to NONE in IBM Health Checker for z/OS so that the check does not continue to issue exceptions to the console until more data is collected and new predictions are made. The check continues to run at the defined interval so that the latest exception report data is available using the CK panel in SDSF.

PFA provides this support using remote checks from IBM Health Checker for z/OS to collect data about your installation. Using this data, PFA constructs a model of the expected or future behavior of the z/OS images, compares the actual behavior with the expected behavior, and if the behavior is abnormal, PFA issues a health check exception. PFA uses a z/OS UNIX System Services (z/OS UNIX) file system to manage the historical and problem data that it collects.

23.5.1 PFA_COMMON_STORAGE_USAGE health check

Many VSCR problems led to the introduction of 64-bit common support in z/OS V1R10. This trend, which started with z/OS V1R10, has opened a huge piece of common storage. However, the use of that storage must be monitored in a predictable manner, which is why Predictive Failure Analysis was developed in z/OS V1R10.

The common storage usage PFA check in z/OS V1R10 is an “exhaustion of common resources” check by which PFA determines when common storage usage will be exhausted. PFA uses machine learning and historical data from this system to predict the future (within a bubble of predictability) level of common storage usage and determine whether the current trend is going to exceed the available common storage. The check combines observations at CSA and SQA for below-the-line common storage and ESQA and ECSA for above-the-line common storage.

If PFA detects that there is a potential for the exhaustion of common storage, it issues exception message AIRH101E and provides a list of suspect tasks in the report. During the analysis, this check writes the common storage usage data at intervals to a z/OS UNIX

System services file in comma-separated value (`.csv`) format. The check identifies a list of users of common storage that might contribute to exhausting common storage. If deeper analysis is necessary, PFA also provides files that contain additional diagnostic information that you can examine.

The PFA_COMMON_STORAGE_USAGE check generates reports for all PFA checks when everything is normal and when an exception occurs.

Note: The PFA_COMMON_STORAGE_USAGE check shown in Figure 23-5 is looking to see if there is a potential for storage to be exhausted in the upcoming PFA model interval. PFA analyzes the following storage locations:

- Common storage area (CSA)
- System queue area (SQA)
- Extended common storage area (ECSA)
- Extended system queue area (ESQA)
- CSA + SQA
- ECSA + ESQA

The PFA_COMMON_STORAGE_USAGE check detects three classes of common storage exhaustion:

- Spike
- Leak
- Creep

```
CHECK(IBMPFA,PFA_COMMON_STORAGE_USAGE)
START TIME: 08/17/2011 09:36:20.929877
CHECK DATE: 20071101  CHECK SEVERITY: MEDIUM
CHECK PARM: DEBUG(0) THRESHOLD(2) COLLECTINT(15) MODELINT(720)
COLLECTINACTIVE(1)

                Common Storage Usage Prediction Report

Last successful model time     :  08/17/2011 01:59:10
Next model time                :  08/17/2011 13:59:10
Model interval                 :  720
Last successful collection time:  08/17/2011 09:30:57
Next collection time           :  08/17/2011 09:45:57
Collection interval            :  15

                                                Capacity When  Percentage
Storage     Current Usage  Prediction           Predicted      of Current
Location    in Kilobytes   in Kilobytes         in Kilobytes   to Capacity
__________  _____________  _____________        _____________  ___________
CSA                  484            485                 2388          20%
SQA                 1463           1465                 3864          38%
CSA+SQA             1947           1945                 6252          31%
ECSA               44192          44336               131148          34%
ESQA               22241          22496                78880          28%
ECSA+ESQA          66433          66423               210028          32%
```

Figure 23-5 Common storage usage prediction report

It was also reported that in z/OS V1R10 and z/OS V1R11, the CSA check can have high CPU utilization while the modeling phase was occurring, and sometimes when data collection was occurring. The check has been refactored so that these periods of high CPU utilization do not occur during collection or modeling. The only time that the extra processing is done that caused the high CPU utilization is when an exception occurs or at “run check” time when the debug parameter is on and the check has recently modeled.

23.5.2 LOGREC arrival rate check

LOGREC arrival rate check detects damaged address spaces or a damaged LPAR by tracking LOGREC arrivals accumulated by program key groups across time ranges:

- Key 0
- Keys 1 to 7
- Keys 8 to 15

The following enhancements were done for the check to improve the modeling and comparison algorithms:

- To allow additional tuning in your environments to reduce false positives in the LOGREC arrival rate check, the **`EXCEPTIONMIN`** parameter has been added. The predicted arrival rate and the current arrival rate must be greater than this value for an exception to occur. The default value is 25. Therefore, installations with low arrival rates with small spikes will no longer see exceptions until both the prediction and the current rate are greater than this value.
- The LOGREC arrival rate has been enhanced so that it can use data prior to the IPL in the same manner that the message arrival rate and the SMF arrival rate checks do. Thus, it no longer needs to wait 24 hours before making predictions and issuing exceptions.

  The last hour prior to shutdown and the first hour after IPL are excluded from modeling to avoid skewing the predictions for LOGRECs generated during shutdown and IPL.

23.5.3 Frames and slots usage check

In z/OSV1R11, PFA added two more checks. The frames and slots usage check detects a damaged system by predicting resource exhaustion by detecting abnormal increased usage of frames and slots by persistent address spaces.

23.5.4 Message arrival rate check

The message arrival rate check detects damaged address spaces or a damaged LPAR by tracking WTO and WTORs normalized by CPU across time ranges

- Chatty, persistent address spaces
- Non-chatty, persistent address spaces
- Non-persistent address spaces
- Total system rate (“chatty” + other persistent + non-persistent)

Predictive Failure Analysis shipped with z/OS V1R11 was the first pass of z/OS capability to detect a damaged z/OS image or address space due to the exhaustion of common resources or recurring errors.

23.5.5 SMF arrival rate check

In z/OS V1R12, PFA added an additional check (the SMF arrival rate check) to detect a damaged system. Thus, there are now two checks that detect resource exhaustion and three checks that detect a damaged system. These checks detect errors at various layers of the software stack and complement each other, so that many soft failures that used to be undetected are now correctable incidents.

The SMF arrival rate check, supervised learning support, and the many improvements to performance, modeling, serviceability, and usability all lead to an increased ability to detect soft failures and correct them before they cause a system outage.

The SMF arrival rate check is similar in several ways to the message arrival rate check. The SMF arrival rate check accumulates the SMF arrivals in a collection interval and normalizes it by the CPU seconds used in the collection. If the arrival rate found at the last collection is excessively high when compared to the prediction, an exception message is issued. The exception message can be issued by comparing the collected and predicted rates for the entire system, for each individual persistent job being tracked, for the other persistent jobs as a group, or for the non-persistent jobs as a group.

The definition of persistent jobs is the same as for the frames and slots usage check. That is, the job is considered persistent if it starts within one hour after IPL. All SMF arrival rates collected in the first hour after IPL are discarded to allow the system time to stabilize after the IPL.

The SMF arrival rate check collects, models, and compares four categories:

- Chatty, persistent address spaces
- Non-chatty, persistent address spaces
- Non-persistent address spaces
- Total system rate (“chatty” + other persistent + non-persistent)

SMF configuration considerations

If the SMF configuration changes, it is advisable to stop PFA, delete the `/data` directory, and restart PFA so that the previously collected data is not used with the new configuration. If SMF is stopped and restarted across a collection interval, PFA will detect this change. It will automatically delete the previously collected data and reenter the warmup phase to detect which jobs to track.

Note: The SMF arrival rate check is not designed to detect abnormal SMF patterns or individual types of SMF records.

Tracking persistent jobs

The persistent jobs that are tracked individually are determined either by the jobs that were tracked prior to IPL or by the jobs that had the highest arrivals after a 6-hour warmup phase that beings an hour after IPL or when PFA starts, whichever is later.

If PFA had not previously been running, or if data had not been collected prior to the IPL, PFA chooses the individual persistent jobs to track based on the arrival rates in the 6 hours from hour 1 to hour 7 after IPL. The persistent jobs with the highest arrival rates are tracked individually. All other persistent jobs are put in the “other persistent jobs” category.

If PFA had been running prior to IPL and had been collecting data, the jobs that were previously tracked are tracked again if that entire list of jobs is persistent again after the IPL.

Also, if PFA had collected data prior to IPL, the last hour's worth of data collected that exists in the files when PFA starts again is discarded so that the arrivals collected during shutdown do not skew the predictions.

When an exception occurs, a report pertaining to the category is produced that lists the address spaces that had the highest rates in the last collection interval. This list helps determine the address space causing the problem.

PFA ships with default parameters for each check that have been tuned for most installations. If you need to tune parameters due to false positive exceptions or missed exceptions, the SMF arrival rate check provides two parameters, namely an STDDEV parameter and an EXCEPTIONMIN parameter.

If exceptions are being produced for low rates on your system and these exceptions are not needed, set the EXCEPTIONMIN parameter higher.

If exceptions are produced that are not needed and the difference between the expected value predicted and the current usage is not drastic enough, set the STDDEV (for standard deviation) higher.

23.5.6 Miscellaneous improvements

The following enhancements have been done for the checks to improve the modeling and comparison algorithms:

- All checks have been enhanced to have the default model interval be 12 hours to reduce modeling on stable environments. The checks will model more frequently in less stable environments when a check is close to issuing an exception.
  - Every 720 minutes (12 hours) by default instead of every 6 hours in previous releases
  - PFA dynamically determines when to model more frequently based on system behavior
- Checks that perform comparisons using data for time ranges have been enhanced to require enough data even when there are gaps in the data caused by PFA being stopped or by an IPL, and to use an enhanced comparison algorithm when data across the time ranges is deemed to be inconsistent to reduce false positives for spikes caused by processing done in certain time ranges only.
- Improve performance by reducing the number of model requests when the system is stable.

23.6 PFA enhancements with z/OS V1R13

PFA enhancements for z/OS V1R13 improve availability and resiliency and further detect damaged or hung address spaces or systems. PFA can also return Runtime Diagnostics report information when activity is absent or unusually low for the PFA message arrival rate check and the PFA SMF arrival rate check.

The following new PFA checks are provided in z/OS V1R13:

- Enqueue request rate check

  This check detects a damaged address space or damaged system by comparing the number of enqueue requests per CPU second to the rate expected.

- JES spool usage check for JES2

  This check detects abnormal behaviors in persistent jobs by modeling the change in their JES2 spool usage, for example, a change in the number of track groups used from one collection to the next.

- Integration of PFA with Runtime Diagnostics

  This detects whether certain metrics are too low by using predictive technology along with the results of Runtime Diagnostics. In certain instances, when Runtime Diagnostics finds a critical message, it performs additional analysis based on the job name in the message or other information in the message text. For example, if Runtime Diagnostics identifies an XCF stalled connector message, it performs additional analysis of the identified address space to help isolate the problem.

  A key feature of Runtime Diagnostics is its ability to summarize internal processing errors and return the results to you in a message response. With z/OS V1R13, PFA can also return Runtime Diagnostics report information when activity is absent or unusually low for the PFA message arrival rate check and the PFA SMF arrival rate check.

These enhancements increase availability and resiliency by detecting abnormal behavior and giving alerts before the situation becomes an outage.

23.6.1 PFA ENQUEUE REQUEST RATE check

The PFA_ENQUEUE_REQUEST_RATE check detects damage to an address space by using the number of enqueue requests per CPU second used as the tracked metric. If PFA detects that the enqueue request rate is lower than expected, PFA calls Runtime Diagnostics to detect if an address space is hung. If PFA detects that the enqueue request rate is higher than expected, PFA calls Runtime Diagnostics to detect if there is a damaged address space.

By detecting these conditions early, you can correct the problem before it causes the system to hang or crash. The enqueue request rate check issues an exception using the following two types of comparisons:

- Tracked jobs
- Total system

After the PFA_ENQUEUE_REQUEST_RATE check issues an exception, it does not perform the next comparison type.

To avoid skewing the enqueue request rate, PFA ignores the first hour of enqueue data after IPL and the last hour of enqueue data prior to shutdown. In addition, PFA attempts to track the same persistent ASIDs that it tracked prior to IPL or PFA restart if the same persistent ASIDs are still active.

The objective of this check is to determine if there is potential of damage to an LPAR by using the number of enqueues per CPU second used as the tracked metric.

The enqueue request rate is calculated as:

```
Enqueue request rate = Number of enqueues requested / CPU Utilization
```

Two categories are compared across three time ranges, as follows:

- “Chatty” persistent address spaces are tracked individually.
- Persistent is defined as an address space that must have started within an hour after IPL.
- “Chatty” means 20 address spaces having the highest rates in a warmup period.
- The total system rate is compared every 1 hour, 24 hours, and 7 days.
- The first hour after IPL and the last hour prior to shutdown are ignored.

PFA_ENQUEUE_REQUEST_RATE Prediction Report

An exception message is sent as a WTO when everything is OK; see Figure 23-6.

```
            Enqueue Request Rate Prediction Report

Last successful model time     :  01/27/2011 11:08:01
Next model time                :  01/27/2011 23:08:01
Model interval                 :  720
Last successful collection time:  01/27/2011 17:41:38
Next collection time           :  01/27/2011 17:56:38
Collection interval            :  15

Enqueue request rate
   at last collection interval      :       83.52
Prediction based on 1 hour of data  :       98.27
Prediction based on 24 hours of data:       85.98
Prediction based on 7 days of data  :      100.22

Top persistent users:

                                          Predicted Enqueue
                      Enqueue                Request Rate
  Job                 Request
  Name     ASID          Rate        1 Hour       24 Hour         7 Day

  TRACKED1 001D         58.00         23.88         22.82         35.82
  TRACKED2 0028         11.00         10.34         11.11         12.11
  TRACKED3 0029         11.00         12.43         12.36          8.36
```

Figure 23-6 Enqueue request rate prediction report

The Prediction report and the resulting messages are also available in SDSF. They can help to determine which category is causing the exception; for example:

- All tracked jobs shown for everything OK.
- Only tracked jobs causing exceptions are shown for the tracked job exception.

23.6.2 PFA_JES_SPOOL_USAGE check for JES2

This new PFA check in z/OS V1R13 detects a damaged address space or system based on persistent job usage of the number of track groups in JES2 spool. Persistent job tracking is performed as described here:

- A persistent job is one that must have started within an hour after IPL.
- Persistent jobs are tracked by name.
- If abnormal behavior is detected based on the expected values for a persistent job that has been modeled, PFA issues a health check exception message.
- When an exception occurs, the check reports the persistent jobs with JES spool usage that was abnormal (those jobs that caused the exception).
- When no problem exists, the check reports the persistent jobs that were modeled (for informational purposes).
- The check detects when a persistent job has restarted and considers it persistent even if the restart occurred more than one hour after IPL.
- If there are duplicate jobs that are persistent, neither job is tracked.

23.6.3 Checking track group usage

Starting in z/OS V1R13, PFA models 15 persistent jobs with the highest increase in their track group usage from one collection to the next, as shown in Figure 23-7:

- The number of actual track groups used is irrelevant because we are looking for a damaged address space or system rather than exhaustion of track groups.
- Dynamic modeling occurs when “top jobs” changes significantly to model new top jobs.

```
                    JES Spool Usage Prediction Report

Last successful model time     :  01/28/2011 06:10:15
Next model time                :  01/28/2011 18:11:57
Model interval                 :  720
Last successful collection time:  01/28/2011 18:05:57
Next collection time           :  01/28/2011 18:10:57
Collection interval            :  5

Address spaces causing exception:

                Current Change in   Expected Change in          Current
 Job              Number of Track      Number of Track   Number of Track
 Name     ASID        Groups Used          Groups Used       Groups Used
 _______ _____   _______________      _______________   _______________
 JOB1     0019               252                  10               892
 JOB55    000E               129                   3               400
```

Figure 23-7 JES2 spool usage prediction report

An exception is caused by an abnormal increase in the number of track groups used, not the number of track groups used.

Note: This PFA check is only valid for JES2 environments. Systems running JES3 have a command to check the equivalent spool usage.

JES3 spool usage command

JES3 has long provided the **`*I Q SP=ALL U N=10`** command to display spool usage; see Figure 23-8.

```
*I Q SP=ALL U N=10
 IEA631I  OPERATOR LAFITTE  NOW INACTIVE, SYSTEM=SC75    , LU=SC38TC4C
IAT8583 JES3PART  TOTAL IN USE            155,413 TRKGPS
 IAT8583 JES3PART  TOTAL IN USE BY JES3        601 TRKGPS
 IAT8583 JES3PART  TOTAL IN USE BY JOBS    154,812 TRKGPS
 IAT8587 JES3PART  USERS FOUND=      7 JES3   1,490 JOBS;  10 DISPLAYED
 IAT8527 JES3PART: JOB GPMDDSPP (JOB21456) 61,537 TRKGPS,   3%
 IAT8527 JES3PART: JOB GPMDDSPP (JOB21448) 61,537 TRKGPS,   3%
 IAT8527 JES3PART: JOB HZSPROC  (JOB19453)  1,100 TRKGPS,  <1%
 IAT8527 JES3PART: JOB GPMDDSPP (JOB21451)    833 TRKGPS,  <1%
 IAT8527 JES3PART: JOB GPMDDSPP (JOB21454)    833 TRKGPS,  <1%
 IAT8527 JES3PART: JOB GPMDDSPP (JOB21450)    833 TRKGPS,  <1%
 IAT8527 JES3PART: JOB GPMDDSPP (JOB21437)    824 TRKGPS,  <1%
 IAT8527 JES3PART: JOB HZSPROC  (JOB19777)    792 TRKGPS,  <1%
 IAT8591 INQUIRY ON SPOOL SPACE USAGE COMPLETE
.............
```

Figure 23-8 JES3 command to display spool usage

23.6.4 Define DASD storage for Predictive Failure Analysis

Prior to z/OS V1R13, Predictive Failure Analysis did not document the requirement for additional DASD storage to accommodate check output. Starting with z/OS V1R13, z/OS Problem Management contains DASD requirements to ensure PFA has enough space to update and create files in the z/OS UNIX file system to store check output.

In addition, because zFS no longer stores data in 1 K fragments, zFS for z/OS V1R13 might need more DASD storage to store the same amount of data than was required in previous releases.

Defining DASD storage

Defining additional DASD storage for PFA might be needed. The total space for the PFA file system for each LPAR depends on the release of z/OS you are running, as listed here:

z/OS V1R11 200 cylinders primary; 50 cylinders secondary on a 3390 device.

z/OS V1R12 200 cylinders primary; 50 cylinders secondary on a 3390 device.

z/OS V1R13 300 cylinders primary; 50 cylinders secondary on a 3390 device.

23.7 Runtime Diagnostics and z/OS V1R13

In z/OS V1R12, Runtime Diagnostics was introduced as a new MVS subsystem (component name HZR) designed to help you analyze a system that has potential soft failures. In contrast to catastrophic failures, a “soft” failure is defined as the combination of typical and abnormal behavior that causes the software to withhold a requested service. Soft failures are often difficult or impossible to detect and can slowly lead to the degradation of the solution that is using z/OS.

Runtime Diagnostics performs many of the same tasks you might typically perform when looking for a failure, such as:

- Reviewing critical messages in the log.
- Analyzing contention.
- Examining address spaces with high central processing unit (CPU) usage.
- Looking for an address space that might be in a loop.
- Evaluating local lock conditions.
- New with z/OS V1R13, Runtime Diagnostics analyzes various types of contention including ENQs, GRS latch contention, and z/OS UNIX file system latch contention

In many cases, when Runtime Diagnostics finds a critical message, it performs additional analysis based on the job name in the message or other information in the message text. For example, if Runtime Diagnostics identifies an XCF stalled connector message, it performs additional analysis of the identified address space to help narrow down the problem. Runtime Diagnostics and Omegamon XE then provide additional lower-level details.

Note: To resolve contention, determine whether the blocking job is running properly or if you should cancel it. To understand ENQ contention and additional analysis steps, see the topic on contention management in *z/OS MVS Planning: Global Resource Serialization*, SA22-7600.

23.7.1 Using Runtime Diagnostics

Runtime Diagnostics allows systems programmers to quickly analyze a system experiencing “sick but not dead” symptoms. Runtime Diagnostics is enhanced for z/OS V1R13. There are minimal migration issues.

Attention: To enable Runtime Diagnostics to run on z/OS V1R12 and below, set up a separate PROC statement for each image running the lower release.

For example, if you have z/OS V1R13 installed and Runtime Diagnostics, but you also have a V1R12 system that you need Runtime Diagnostics to analyze, enter the following statement on the V1R12 system:

```
S procname,JOBNAME=HZR
```

OPERLOG considerations

Runtime Diagnostics searches for certain messages and message combinations in the operations log (OPERLOG) stream and attempts to identify other system symptoms with minimal dependencies on other system services.

By default Runtime Diagnostics performs analysis of the home system, but you can use the **SYSNAME=xxxxx** parameter to direct HZR to perform a subset of its functions analyzing a system in the sysplex other than where the HZR function is running.

For example, if you are using Runtime Diagnostics on a current system such as SYSA, then all analysis is for the current system. If you want to analyze a system other than SYSA, such as SYSB, specify a **F HZR,ANALYZE,SYSNAME=SYSB** command.

When you use this method, Runtime Diagnostics only performs a limited analysis of the requested system based information available on the invoking system, like critical messages in OPERLOG and ENQ information.

Starting Runtime Diagnostics

Runtime Diagnostics can be started at IPL through an entry in the COMMNDxx parmlib member or by the installation's automation. Use the **Modify** (or **F**) command to tell Runtime Diagnostics to do a run.

Note: You must use the following instructions and be running z/OS V1R13 or above for Runtime Diagnostics to be operational at IPL. If you are analyzing a separate system in the sysplex, that system can operate using z/OS V1R10 and above.

1. Update the COMMNDxx parmlib member to configure Runtime Diagnostic as the HZR address space:

   ```
   COM='START hzrproc'
   ```

   This member enables the HZR address space to be automatically started during the IPL of the system.

2. To enable Runtime Diagnostics to run under the master subsystem, update the IEFSSNxx parmlib member with the following statement:

   ```
   SUBSYS SUBNAME(HZR)
   ```

3. If you previously used Runtime Diagnostics in V1R12, you must ensure the `hzrproc` points to PGM=HZRINIT, and not to PGM=HZRIMAIN.

Update any system automation your installation uses to start and restart major system address spaces. In z/OS V1R13, member HZR ships in the SYS1.PROCLIB data set with HZROUT specified as DD DUMMY.

To analyze a system, enter the following **`MODIFY`** (or **`F`**) command:

```
F HZR,ANALYZE
```

The output of Runtime Diagnostics is a WTO to the operator console (actually, a multiline WTO) that is issued to the console that originally issued the **`F HZR,ANALYZE`** command. If the MCS console that issued the **`F HZR`** command has an out-of-line display area setup (through a `K A,xx`) then the output will be displayed in the display area. The output of Runtime Diagnostics can also be directed to a sequential data set.

Runtime Diagnostics processing

Runtime Diagnostics attempts to analyze a sick but not dead system in a minute or so and provides suggested next steps to take. It looks for evidence of soft failures such as component issues and global resource contention for the important address space execution issues.

Runtime Diagnostics reduces the skill level needed by systems programmers when examining z/OS for “unknown” problems. Using Runtime Diagnostics, systems programmers can quickly analyze a sick system for the following classes of problems:

- Component problems that emitted critical messages in OPERLOG
- ENQ contention for system address spaces
- Address spaces with a high local lock suspension rate
- Address spaces using high CPU
- Address spaces that appear to be in a TCB-enabled loop
- GRS latch contention
- z/OS File System latch contention

Using Runtime Diagnostics, systems programmers can quickly be told what actions to perform next or potentially what jobs might need to be cancelled. Perform further investigation on a class of resources or a single address space using a monitor like RMF or Tivoli Omegamon.

23.7.2 Runtime Diagnostics command examples and messages

If a HZR0201I message returns when you enter the **F HZR** command, there are no error events present on the system the command was run against. For example:

```
HZR0201I SUCCESS. TIME (2009/06/09 - 10:25:01). NO RUNTIME DIAGNOSTICS EVENTS
WERE FOUND FOR SYSTEM: SY1
```

Note: When Runtime Diagnostics analyzes a target system other than the home system (a target system is a system not equal to the home system), it can find critical messages and perform ENQ analysis, but it cannot perform any processing that requires control block analysis.

Consequently, when analyzing a target system, Runtime Diagnostics cannot perform z/OS UNIX file system latch contention, GRS latch contention, loop detection, CPU analysis, or local lock suspension. Various command examples are given showing various Runtime Diagnostics usages.

Figure 23-9 shows an example of what you might receive when analyzing a target system.

```
HZR0200I RUNTIME DIAGNOSTICS RESULT 593
SUMMARY: SUCCESS - NO EVENTS FOUND
REQ: 001 TARGET SYSTEM: SYS3 HOME: SY1 2010/12/21 - 14:52:50
INTERVAL: 60 MINUTES
EVENTS:
FOUND: 00 - PRIORITIES: HIGH:00 MED:00 LOW:00
PROCESSING BYPASSED:
OMVS.......SPECIFIED TARGET SYSTEM IS NOT THE HOME SYSTEM.
LATCHES....SPECIFIED TARGET SYSTEM IS NOT THE HOME SYSTEM.
LOOP.......SPECIFIED TARGET SYSTEM IS NOT THE HOME SYSTEM.
HIGHCPU....SPECIFIED TARGET SYSTEM IS NOT THE HOME SYSTEM.
LOCK.......SPECIFIED TARGET SYSTEM IS NOT THE HOME SYSTEM.
------------------------------------------------------------------------
```

Figure 23-9 Message output for target system

Figure 23-10 shows a Runtime Diagnostics status image.

```
f hzr,analyze
HZR0200I RUNTIME DIAGNOSTICS RESULT 974
SUMMARY: SUCCESS
REQ: 001 TARGET SYSTEM: SY1 HOME: SY1 2010/12/21 - 11:30:57
INTERVAL: 60 MINUTES
EVENTS:
FOUND: 05 - PRIORITIES: HIGH:05 MED:00 LOW:00
TYPES: CF:04
TYPES: HIGHCPU:01
```

Figure 23-10 Status message reporting Runtime Diagnostics success

Runtime Diagnostic success and failure messages

Runtime Diagnostics also reports when part of its processing fails (that is, it is unable to complete processing for one or more events) as `QUALIFIED SUCCESS` in the `SUMMARY:` portion of the report, as shown in Figure 23-11.

The message explains which part of the processing was unsuccessful in `PROCESSING FAILURES:`.

In this case, although Runtime Diagnostics was unable to connect to OPERLOG to examine messages, it continues to find other soft failures (LOOP and HIGHCPU).

```
f hzr,analyze
HZR0200I RUNTIME DIAGNOSTICS RESULT 751
SUMMARY: QUALIFIED SUCCESS - SOME PROCESSING FAILED
REQ: 001 TARGET SYSTEM: SY1 HOME: SY1 2010/12/21 - 11:25:55
INTERVAL: 60 MINUTES
EVENTS:
FOUND: 02 - PRIORITIES: HIGH:02 MED:00 LOW:00
TYPES: HIGHCPU:01
TYPES: LOOP:01
PROCESSING FAILURES:
OPERLOG....IXGCONN REQ=CONNECT ERROR.......RC=00000008 RS=0000080B
---------------------------------------------------------------------
EVENT 01: HIGH - HIGHCPU - SYSTEM: SY1 2010/12/21 - 11:25:56
ASID CPU RATE:99% ASID:002E JOBNAME:IBMUSERX
STEPNAME:STEP1 PROCSTEP: JOBID:JOB00045 USERID:IBMUSER
JOBSTART:2010/12/21 - 11:22:51
ERROR: ADDRESS SPACE USING EXCESSIVE CPU TIME. IT MIGHT BE LOOPING.
ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
---------------------------------------------------------------------
EVENT 02: HIGH - LOOP - SYSTEM: SY1 2010/12/21 - 11:25:35
ASID:002E JOBNAME:IBMUSERX TCB:004FF1C0
STEPNAME:STEP1 PROCSTEP: JOBID:JOB00045 USERID:IBMUSER
JOBSTART:2010/12/21 - 11:22:51
ERROR: ADDRESS SPACE MIGHT BE IN A LOOP.
ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
---------------------------------------------------------------------
```

Figure 23-11 Status message reporting Runtime Diagnostics QUALIFIED SUCCESS

Note: If you do not connect to the OPERLOG, a system logger message and possibly a RACF (or equivalent security product) message will display as shown in Figure 23-12.

For message analysis, ensure the system is running OPERLOG and you have read permissions set for the OPERLOG.

For details about how to define a log stream using system logger services, see *z/OS Problem Management*, G325-2564.

```
f hzr,analyze
HZR0200I RUNTIME DIAGNOSTICS RESULT 751
SUMMARY: QUALIFIED SUCCESS - SOME PROCESSING FAILED
REQ: 001 TARGET SYSTEM: SY1 HOME: SY1 2010/12/21 - 11:25:55
INTERVAL: 60 MINUTES
EVENTS:
FOUND: 01 - PRIORITIES: HIGH:01 MED:00 LOW:00
TYPES: HIGHCPU:01
TYPES: LOOP:01
PROCESSING FAILURES:
OPERLOG....IXGCONN REQ=CONNECT ERROR.......RC=00000008 RS=0000080B
-------------------------------------------------------------------------
ICH408I USER(IBMUSR2) GROUP(SYS1) NAME(####################) 996
SYSPLEX.OPERLOG CL(LOGSTRM)
INSUFFICIENT ACCESS AUTHORITY
ACCESS INTENT(READ) ACCESS ALLOWED(NONE)
IXG231I IXGCONN REQUEST=CONNECT TO LOG STREAM SYSPLEX.OPERLOG DID NOT 997
SUCCEED FOR JOB IBMUSR2A. RETURN CODE: 00000008 REASON CODE:
0000080D DIAG1: 00000008 DIAG2: 00000000 DIAG3: 03010000 DIAG4:
00000000
SY1 HZR0200I RUNTIME DIAGNOSTICS RESULT 805
SUMMARY: QUALIFIED SUCCESS - SOME PROCESSING FAILED
------------------
```

Figure 23-12 RACF message reporting log stream connection failure for Runtime Diagnostics

CPU analysis

Runtime Diagnostics provides a point-in-time check of any address space that is using more than 95% of the capacity of a single CPU. This might indicate the address space is in a loop, as shown in Figure 23-13.

The analysis is a one-second sample interval based on the capacity of a single CPU within the LPAR. Be aware that it is possible for the usage to be reported greater than 100% if the address space has multiple TCBs and several of the TCBs are individually using a high percentage of the capacity of a CPU.

```
f hzr,analyze
HZR0200I RUNTIME DIAGNOSTICS RESULT 568
SUMMARY: SUCCESS
REQ: 003 TARGET SYSTEM: SY1 HOME: SY1 2010/12/21 - 13:45:49
INTERVAL: 60 MINUTES
EVENTS:
FOUND: 02 - PRIORITIES: HIGH:02 MED:00 LOW:00
TYPES: HIGHCPU:01
TYPES: LOCK:01
------------------------------------------------------------------------
EVENT 01: HIGH - HIGHCPU - SYSTEM: SY1 2010/12/21 - 13:45:50
ASID CPU RATE:99% ASID:002E JOBNAME:IBMUSERX
STEPNAME:STEP1 PROCSTEP: JOBID:JOB00045 USERID:IBMUSER
JOBSTART:2010/12/21 - 11:22:51
ERROR: ADDRESS SPACE USING EXCESSIVE CPU TIME. IT MIGHT BE LOOPING.
ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
------------------------------------------------------------------------
EVENT 02: HIGH - LOCK - SYSTEM: SY1 2010/12/21 - 13:45:50
HIGH LOCAL LOCK SUSPENSION RATE - ASID:000A JOBNAME:WLM
STEPNAME:WLM PROCSTEP:IEFPROC JOBID:++++++++ USERID:++++++++
JOBSTART:2010/12/21 - 11:15:08
ERROR: ADDRESS SPACE HAS HIGH LOCAL LOCK SUSPENSION RATE.
ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
------------------------------------------------------------------------
```

Figure 23-13 Runtime Diagnostics HIGHCPU and LOCK report

Local lock suspension and loop detection

Runtime Diagnostics provides a point-in-time check of local lock suspension for any address space. For the local lock suspension, Runtime Diagnostics calculates the amount of time an address space is suspended waiting for the local lock. If an address is suspended more than 50% of the time waiting for a local lock, Runtime Diagnostics reports it as an event. For an example, see Figure 23-14.

Runtime Diagnostics looks through all tasks in all address spaces to determine whether a task might be looping. Runtime Diagnostics does this by examining various system information for indicators of consistent repetitive activity that are typical when a task is in a loop. When both a HIGHCPU event and a LOOP event (shown in Figure 6) list the job name, there is a high probability that a task in the job is in a loop. The normal corrective action is to cancel the job name listed.

```
f hzr,analyze
HZR0200I RUNTIME DIAGNOSTICS RESULT 581
SUMMARY: SUCCESS
REQ: 004 TARGET SYSTEM: SY1 HOME: SY1 2010/12/21 - 13:51:32
INTERVAL: 60 MINUTES
EVENTS:
FOUND: 04 - PRIORITIES: HIGH:04 MED:00 LOW:00
TYPES: HIGHCPU:01
TYPES: LOOP:01 ENQ:01 LOCK:01
------------------------------------------------------------------------
EVENT 01: HIGH - ENQ - SYSTEM: SY1 2010/12/21 - 13:51:32
ENQ WAITER - ASID:0038 - JOBNAME:IBMUSER2 - SYSTEM:SY1
ENQ BLOCKER - ASID:002F - JOBNAME:IBMUSER1 - SYSTEM:SY1
QNAME: TESTENQ
RNAME: TESTOFAVERYVERYVERYVERYLOOOOOOOOOOOOOOOOOOOOOONGRNAME1234567...
ERROR: ADDRESS SPACES MIGHT BE IN ENQ CONTENTION.
ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE BLOCKING JOBS AND
ACTION: ASIDS.
------------------------------------------------------------------------
EVENT 02: HIGH - HIGHCPU - SYSTEM: SY1 2010/12/21 - 13:51:33
ASID CPU RATE:99% ASID:002E JOBNAME:IBMUSERX
STEPNAME:STEP1 PROCSTEP: JOBID:JOB00045 USERID:IBMUSER
JOBSTART:2010/12/21 - 11:22:51
ERROR: ADDRESS SPACE USING EXCESSIVE CPU TIME. IT MIGHT BE LOOPING.
ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
------------------------------------------------------------------------
EVENT 03: HIGH - LOOP - SYSTEM: SY1 2010/12/21 - 13:51:14
ASID:002E JOBNAME:IBMUSERX TCB:004FF1C0
STEPNAME:STEP1 PROCSTEP: JOBID:JOB00045 USERID:IBMUSER
JOBSTART:2010/12/21 - 11:22:51
ERROR: ADDRESS SPACE MIGHT BE IN A LOOP.
ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
------------------------------------------------------------------------
EVENT 04: HIGH - LOCK - SYSTEM: SY1 2010/12/21 - 13:51:33
HIGH LOCAL LOCK SUSPENSION RATE - ASID:000A JOBNAME:WLM
STEPNAME:WLM PROCSTEP:IEFPROC JOBID:++++++++ USERID:++++++++
JOBSTART:2010/12/21 - 11:15:08
ERROR: ADDRESS SPACE HAS HIGH LOCAL LOCK SUSPENSION RATE.
ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
------------------------------------------------------------------------
```

Figure 23-14 Runtime Diagnostics LOOP and HIGHCPU report

z/OS UNIX latch contention

Runtime Diagnostics gathers information about z/OS UNIX file system contention. When z/OS UNIX latch contention or waiting threads exist for greater than five minutes in z/OS UNIX, the z/OS UNIX file system latch contention and waiting threads record, as shown in Figure 23-15, displays at the console.

As shown in the figure, if all the counts are zero 90), the record does not display. Follow the instructions listed in the ACTION statement in the event record by issuing **D OMVS,W,A** command, which returns the ASID and job names of any latch waiters.

```
f hzr,analyze
HZR0200I RUNTIME DIAGNOSTICS RESULT 588
SUMMARY: SUCCESS
REQ: 002 TARGET SYSTEM: SY1 HOME: SY1 2010/11/15 - 12:29:08
INTERVAL: 60 MINUTES
EVENTS:
FOUND: 03 - PRIORITIES: HIGH=03 MED=00 LOW=00
TYPES: OMVS=01 CF=02
---------------------------------------------------------------------
EVENT 01: HIGH - OMVS - SYSTEM: SY1 2010/07/07 - 13:07:32
ASID:000E - JOBNAME:OMVS
MOUNT LATCH WAITERS: 2
FILE SYSTEM LATCH WAITERS: 0
XSYS AND OTHER THREADS WAITING FOR z/OS UNIX: 3
ERROR: z/OS UNIX MIGHT HAVE FILE SYSTEM LATCH CONTENTION.
ACTION: ISSUE D OMVS,W,A TO INVESTIGATE z/OS UNIX FILE SYSTEM
ACTION: LATCH CONTENTION, ACTIVITY AND WAITING THREADS.
---------------------------------------------------------------------
```

Figure 23-15 Displaying the z/OS UNIX file system latch contention and waiting threads record

Note: For additional information about diagnosing and resolving latch contention, refer to those topics in *z/OS UNIX System Services Planning*, GA22-7800 and in *z/OS MVS Diagnosis: Reference*, GA22-7588.

For zFS file system contention, see *z/OS Distributed File Service zSeries File System Administration*, SC24-5989.

GRS latch contention

Figure 23-16 shows the Runtime Diagnostics event record summarizing latch contention. When any of the counts are greater than zero (0), Runtime Diagnostics displays a summary record. Follow the ACTION statement in the summary to determine the source of the contention and further actions.

```
HZR0200I RUNTIME DIAGNOSTICS RESULT 928
SUMMARY: SUCCESS
REQ: 002 TARGET SYSTEM: SY1 HOME: SY1 2010/12/21 - 14:32:01
INTERVAL: 60 MINUTES
EVENTS:
FOUND: 02 - PRIORITIES: HIGH:02 MED:00 LOW:00
TYPES: LATCH:02
------------------------------------------------------------------------
EVENT 01: HIGH - LATCH - SYSTEM: SY1 2010/12/21 - 14:32:01
LATCH SET NAME: SYSTEST.LATCH_TESTSET
LATCH NUMBER:3 CASID:0039 CJOBNAME:TSTLATCH
TOP WAITER - ASID:0039 - JOBNAME:TSTLATCH - TCB/WEB:004E2A70
TOP BLOCKER- ASID:0039 - JOBNAME:TSTLATCH - TCB/WEB:004FF028
ERROR: ADDRESS SPACES MIGHT BE IN LATCH CONTENTION.
ACTION: D GRS,AN,LATCH,DEP,CASID=0039,LAT=(SYSTEST.L*,3),DET
ACTION: TO ANALYZE THE LATCH DEPENDENCIES. USE YOUR SOFTWARE
ACTION: MONITORS TO INVESTIGATE BLOCKING JOBS AND ASIDS.
------------------------------------------------------------------------
EVENT 02: HIGH - LATCH - SYSTEM: SY1 2010/12/21 - 14:32:01
LATCH SET NAME: SYSTEST.LATCH_TESTSET
LATCH NUMBER:3 CASID:003B CJOBNAME:TSTLATC2
TOP WAITER - ASID:003B - JOBNAME:TSTLATC2 - TCB/WEB:004E2A70
TOP BLOCKER- ASID:003B - JOBNAME:TSTLATC2 - TCB/WEB:004FF028
ERROR: ADDRESS SPACES MIGHT BE IN LATCH CONTENTION.
ACTION: D GRS,AN,LATCH,DEP,CASID=003B,LAT=(SYSTEST.L*,3),DET
ACTION: TO ANALYZE THE LATCH DEPENDENCIES. USE YOUR SOFTWARE
ACTION: MONITORS TO INVESTIGATE BLOCKING JOBS AND ASIDS.
------------------------------------------------------------------------
```

Figure 23-16 HZR event displaying the GRS latch contention event record

Blocking jobs

The types of problems that can be detected by Runtime Diagnostics are listed here:

Critical message analysis Runtime Diagnostics reads through the last hour of OPERLOG looking for critical messages. If any are found, Runtime Diagnostics lists the critical message as an error event. For a subset of critical messages, Runtime Diagnostics performs additional analysis based on the message identifier and the content of the message. If less than one hour of message content is available in OPERLOG, message analysis is done using the messages available.

ENQ contention checking With z/OS V1R13, Runtime Diagnostics provides a point-in-time check of ENQ contention equivalent to issuing the **D GRS,AN,WAITER** command. It compares the list of job names that are waiters with the list of system address spaces that are started at IPL to determine if any system address spaces are waiters. If ENQ contention is found, Runtime Diagnostics issues an error event within message HZR0200I stating the job name of the waiter for ENQ resource, as shown in Figure 23-17.

To resolve contention, you need to decide whether the blocking job is running properly, as shown in the figure, or to cancel it.

```
f hzr,analyze
HZR0200I RUNTIME DIAGNOSTICS RESULT 568
---------------------------------------------------------------------
EVENT 01: HIGH - ENQ - SYSTEM: SY1 2010/12/21 - 11:43:33
ENQ WAITER - ASID:0038 - JOBNAME:IBMUSER2 - SYSTEM:SY1
ENQ BLOCKER - ASID:002F - JOBNAME:IBMUSER1 - SYSTEM:SY1
QNAME: TESTENQ
RNAME: TESTOFAVERYVERYVERYVERYLOOOOOOOOOOOOOOOOOOOOOONGRNAME1234567...
ERROR: ADDRESS SPACES MIGHT BE IN ENQ CONTENTION.
ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE BLOCKING JOBS AND
ACTION: ASIDS.
---------------------------------------------------------------------
```

Figure 23-17 Runtime Diagnostics ENQ contention report

CPU analysis

Runtime Diagnostics provides a point-in-time check of any address space that is using more than 95% of the capacity of a single CPU. This might indicate the address space is in a loop, as shown in Figure 23-18. The analysis is a one-second sample interval based on the capacity of a single CPU within the LPAR.

Be aware that it is possible for the usage to be reported greater than 100% if the address space has multiple TCBs and several of the TCBs are individually using a high percentage of the capacity of a CPU.

```
f hzr,analyze
HZR0200I RUNTIME DIAGNOSTICS RESULT 568
SUMMARY: SUCCESS
REQ: 003 TARGET SYSTEM: SY1 HOME: SY1 2010/12/21 - 13:45:49
INTERVAL: 60 MINUTES
EVENTS:
FOUND: 02 - PRIORITIES: HIGH:02 MED:00 LOW:00
TYPES: HIGHCPU:01
TYPES: LOCK:01
---------------------------------------------------------------------
EVENT 01: HIGH - HIGHCPU - SYSTEM: SY1 2010/12/21 - 13:45:50
ASID CPU RATE:99% ASID:002E JOBNAME:IBMUSERX
STEPNAME:STEP1 PROCSTEP: JOBID:JOB00045 USERID:IBMUSER
JOBSTART:2010/12/21 - 11:22:51
ERROR: ADDRESS SPACE USING EXCESSIVE CPU TIME. IT MIGHT BE LOOPING.
ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
---------------------------------------------------------------------
EVENT 02: HIGH - LOCK - SYSTEM: SY1 2010/12/21 - 13:45:50
HIGH LOCAL LOCK SUSPENSION RATE - ASID:000A JOBNAME:WLM
STEPNAME:WLM PROCSTEP:IEFPROC JOBID:++++++++ USERID:++++++++
JOBSTART:2010/12/21 - 11:15:08
ERROR: ADDRESS SPACE HAS HIGH LOCAL LOCK SUSPENSION RATE.
ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
---------------------------------------------------------------------
```

Figure 23-18 Runtime Diagnostics HIGHCPU and LOCK report

Additional Runtime Diagnostics analysis

Runtime Diagnostics compares messages IXC105I and IXC418I to determine whether they were issued for the same system as the IXC101I. If so, Runtime Diagnostics lists the activity in the IXC101I event; see Figure 23-19.

```
HZR0200I RUNTIME DIAGNOSTICS RESULT
SUMMARY: SUCCESS
REQ: 001 TARGET SYSTEM: SY1 HOME: SY1 2009/06/09 - 10:34:28
INTERVAL: 60 MINUTES
EVENTS:
FOUND: 02 - PRIORITIES: HIGH=02 MED=00 LOW=00
TYPES: XCF=02
---------------------------------------------------------------------
EVENT 01: HIGH - XCF - SYSTEM: SY1 2009/06/09 - 10:34:10
IXC101I SYSPLEX PARTITIONING IN PROGRESS FOR SY3
ERROR: MESSAGE IXC105I WAS ALSO ISSUED FOR SAME SYSNAME
ACTION: LOOK FOR AND CORRECT ANY PROBLEMS WITH THE ETR CLOCK,
ACTION: SIGNALING PATHS, OR COUPLE DATA SET.
---------------------------------------------------------------------
EVENT 02: HIGH - XCF - SYSTEM: SY1 2009/06/09 - 10:34:10
IXC105I SYSPLEX PARTITIONING HAS COMPLETED FOR SY3
ERROR: XCF REMOVED A SYSTEM FROM THE SYSPLEX.
ACTION: LOOK FOR AND CORRECT ANY PROBLEMS WITH THE ETR CLOCK,
ACTION: SIGNALING PATHS, OR COUPLE DATA SET.
---------------------------------------------------------------------
```

Figure 23-19 Runtime Diagnostics critical message analysis

Following are new messages with z/OS V1R13:

IXC431I Runtime Diagnostics compares the stalled identifiers for multiple IXC431I messages listing the last IXC431I message that contains the stalled identifier and analyzes the job name to determine if it is the waiter for any ENQ contention. The stalled identifiers appear as `ID: s#.r#` in the message description; for example: `STALLED AT sdate stime ID: s#.r#`

IXC246E Runtime Diagnostics examines IOS messages, occurring one minute before and one minute after the system issues IXC246E, looking for IOS messages that contain the same devnum (DASD device number where the data set resides). If Runtime Diagnostics finds additional IOS messages with the same devnum, it lists these with the IXC246E events.

23.7.3 PFA integration with Runtime Diagnostics

Starting with z/OS V1R13, PFA detects a damaged or hung address space or system based on rates as being too low.

Runtime Diagnostics is an MVS utility (component HZR) that can perform many of the same tasks you might manually perform when looking for a the cause of a hung address space and other tasks.

PFA health checks

PFA checks work differently than traditional checks. Although you can modify PFA checks using the IBM Health Checker for z/OS commands and policies, modifications to the PFA checks are unique in the following ways.

Quiescing a PFA check

Use the **MODIFY** command to quiesce a PFA check. If the check is to permanently quiesce until PFA restarts, use the **MODIFY** command to delete the check. Because PFA checks are remote health checks, you must restart PFA to add a previously deleted check. If the check is to be quiesced now, but restarted later while PFA is still running, use the **MODIFY** command to deactivate the check. Deactivating a check stops the comparisons from occurring, but to stop collection and modeling you must set the COLLECTINACTIVE parameter to zero (0).

To stop collections, modeling, and comparisons for an individual check so that the check can be reactivated without restarting PFA, follow these steps:

1. Stop PFA from collecting and modeling data by setting the COLLECTINACTIVE parameter to zero (0).

   For example:

   ```
   f hzsproc,update,check(ibmpfa,pfa_logrec_arrival_rate),
   parm('collectinactive(0)')
   ```

2. Ddeactivate the check in IBM Health Checker for z/OS.

   For example:

   ```
   f hzsproc,deactivate,check(ibmpfa,pfa_logrec_arrival_rate)
   ```

z/OS V1R13 PFA and Runtime Diagnostics

Beginning with z/OS V1R13, PFA can invoke Runtime Diagnostics to analyze and report insufficient metric activity from the following checks:

- PFA_ENQUEUE_REQUEST_RATE
- PFA_MESSAGE_ARRIVAL_RATE
- PFA_SMF_ARRIVAL_RATE

When PFA detects an abnormally low condition, Runtime Diagnostics is executed. If the results of Runtime Diagnostics indicates a problem, the exception is issued. The PFA prediction report includes the Runtime Diagnostics output.

As illustrated in Figure 23-20, data spaces have brought a containing structure defined within the architecture and based on a capability (EAX). This is represented in the figure by the blue shaded area on the left side.

In contrast, 64-bit Common has been introduced as a large space of common storage beyond “the bar”, without any containing structure (apart from storage keys).

Note: An EAX is the way MVS uses them to control access to address spaces through ARs in a way similar to the way it uses AXs to check if a program has the authority to issue the SSAR instruction with an address space as the target of the SSAR instruction.

To be EAX-authorized to the target address space, a program's EAX, when used as an index into the address space's authority table, must point to an entry that indicates SSAR authority.

An AX value is related to an address space; all programs running in an address space have the same AX value at any given time. An EAX value is related to a PC routine. The caller has that EAX value only while the PC routine runs. When the PC returns control, the EAX value returns to what it was before the call.

To deal with this situation, PFA was introduced in z/OS V1R10 and has been extended since then to check for other behavioral resources consumption. This is represented by the PFA Infrastructure shaded area on the right side of Figure 23-20.

With z/OS V1R13, checking is added for low activity (instead of only high rates). This is handled by integrating Runtime Diagnostics. This is represented in the CPU Runtime Diagnostics area in the middle of Figure 23-20.

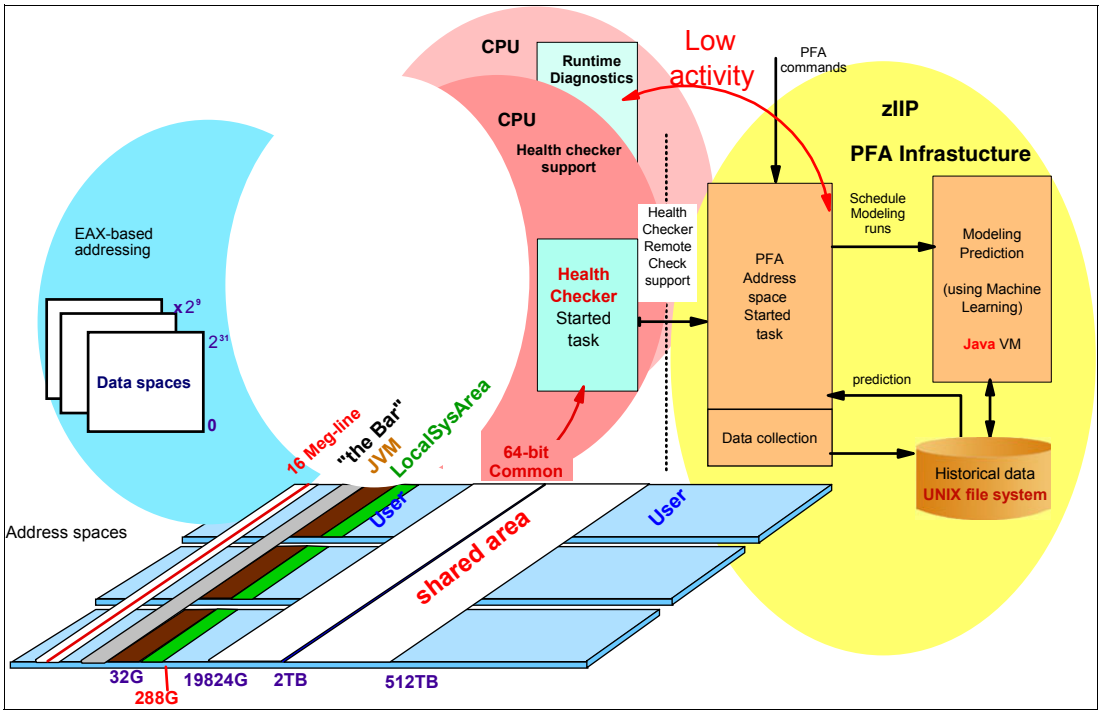


Figure 23-20 PFA and Runtime Diagnostics

When PFA issues a check exception because metric activity is unusually low, the IBM Health Checker for z/OS report will include information from Runtime Diagnostics. The Runtime Diagnostics information in the report points to the specific job or address space and provides the next action you can take. The additional Runtime Diagnostic output can help you quickly determine your next course of action and possibly help you avoid additional problems.

PFA_MESSAGE_ARRIVAL_RATE check

Figure 23-21 is an example of the Runtime Diagnostics output that might appear in the message arrival rate check when PFA determines the tracked jobs had a lower-than-expected AIH206E message arrival rate.

As shown in the figure:

- A “too low” exception message sent as a WTO.
- Runtime Diagnostics output was included in the PFA report.
- The Prediction report and the result message are available in SDSF.
- The Prediction report relevant to the comparison category is causing the exception.

This integration with Runtime Diagnostics for the “too low” condition is supported by three checks:

- Message arrival rate
- SMF arrival rate
- Enqueue requester

It is supported by three categories (if supported by the check):

- Tracked jobs with Runtime Diagnostics is executed for tracked jobs that PFA indicated were too low.
- Other persistent jobs with Runtime Diagnostics is executed for this system.
- Total system with Runtime Diagnostics is executed for this system.

```
...
Persistent address spaces with low rates:
Predicted Message
Message Arrival Rate
Job Arrival
Name ASID Rate 1 Hour 24 Hour 7 Day
________ ____ ____________ _____________ _____________ ____________
JOBS4 001F 1.17 23.88 22.82 15.82
JOBS5 002D 0.30 8.34 11.11 12.11
Runtime Diagnostics Output:
Runtime Diagnostics detects a problem in job: JOBS4
EVENT 06: HIGH - HIGHCPU - SYSTEM: SY1 2009/06/12 - 13:28:46
ASID CPU RATE: 96% ASID: 0027 JOBNAME: JOBS4
STEPNAME: STEPA PROCSTEP: STEPA JOBID: STC00042 USERID: ++++++++
JOBSTART: 2009/06/12 - 13:28:35
ERROR: ADDRESS SPACE USING EXCESSIVE CPU TIME. IT MAY BE LOOPING.
ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
------------------------------------------------------------------------
EVENT 07: HIGH - LOOP - SYSTEM: SY1 2009/06/12 - 13:28:46
ASID: 0027 JOBNAME: JOBS4 TCB: 004E6850
STEPNAME: STEPA PROCSTEP: STEPA JOBID: STC00042 USERID: ++++++++
JOBSTART: 2009/06/12 - 13:28:35
ERROR: ADDRESS SPACE APPEARS TO BE IN A LOOP.
ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
```

Figure 23-21 Runtime Diagnostics report within the PFA message arrival rate

SMF arrival rate check

Figure 23-22 shows an example of the Runtime Diagnostics output that might appear in the SMF arrival rate check when PFA determines the tracked jobs exception report for jobs that had a lower-than-expected SMF arrival rate (for AIH208E).

```
...
Persistent address spaces with low rates:
Predicted SMF
SMF Arrival Rate
Job Arrival
Name ASID Rate 1 Hour 24 Hour 7 Day
________ ____ ____________ ____________ _____________ _____________
TRACKED4 005D 0.20 23.88 22.82 15.82
TRACKED5 0034 0.01 12.43 11.11 8.36
Runtime Diagnostics Output:
Runtime Diagnostics detected a problem in job: TRACKED4
EVENT 06: HIGH - HIGHCPU - SYSTEM: SY1 2009/06/12 - 13:28:46
ASID CPU RATE: 96% ASID: 0027 JOBNAME: TRACKED4
STEPNAME: STEPA PROCSTEP: STEPA JOBID: STC00042 USERID: ++++++++
JOBSTART: 2009/06/12 - 13:28:35
ERROR: ADDRESS SPACE USING EXCESSIVE CPU TIME. IT MAY BE LOOPING.
ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
------------------------------------------------------------------------
EVENT 07: HIGH - LOOP - SYSTEM: SY1 2009/06/12 - 13:28:46
ASID: 0027 JOBNAME: TRACKED4 TCB: 004E6850
STEPNAME: STEPA PROCSTEP: STEPA JOBID: STC00042 USERID: ++++++++
JOBSTART: 2009/06/12 - 13:28:35
ERROR: ADDRESS SPACE APPEARS TO BE IN A LOOP.
ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
```

Figure 23-22 Runtime Diagnostics report within the SMF arrival rate check

23.8 Using PFA with commands and SDSF

PFA activity and results can be viewed either from the console or from SDSF. Summary information for the checks show the check name, whether it is active in Health Checker, the last collection time, and the last model time. Either all checks can be shown or individual checks can be shown by specifying the name of a check or a wildcard that matches more than one check. Wildcards can be specified as the last character of the check name.

PFA modify command

This section provides examples of the PFA **MODIFY** command. It can be used to display summary or detailed information for the PFA checks and to display status information for the PFA infrastructure. Figure 23-23 shows a PFA check summary.

The syntax of the PFA **MODIFY** command is similar to the Health Checker **MODIFY** command and is documented in the PFA documentation.

```
F PFA,DISPLAY,CHECKS
AIR013I 13:47:15 PFA CHECK SUMMARY
                                      LAST SUCCESSFUL LAST SUCCESSFUL
CHECK NAME                         ACTIVE  COLLECT TIME     MODEL TIME
PFA_COMMON_STORAGE_USAGE             YES
PFA_LOGREC_ARRIVAL_RATE              YES
PFA_FRAMES_AND_SLOTS_USAGE           YES
PFA_MESSAGE_ARRIVAL_RATE             YES
```

Figure 23-23 PFA check summary

PFA command to display check details

Detailed information for a check shows counts and times for collection and modeling. It also shows the parameters specific to the check. In fact, to display the cumulative set of parameters for this check, you must use the PFA **MODIFY** command. PFA allows the user to modify parameters individually and accumulate the changes, rather than requiring all parameters to be specified when modifying. Be aware that displaying the check's parameters using Health Checker commands will not show the cumulative list of parameters if the parameters were changed using more than one **MODIFY** command.

The **F PFA,DISPLAY,CHECKS,DETAIL** command output displayed in the following five figures show a single command response, with each figure displaying one of the five checks available for PFA.

Note: The single command displays all five checks.

Figure 23-24 shows the PFA_COMMON_STORAGE_USAGE check output.

```
AIR018I 14:25:17 PFA CHECK DETAIL
CHECK NAME: PFA_COMMON_STORAGE_USAGE
    ACTIVE                          : YES
    TOTAL COLLECTION COUNT          : 663
    SUCCESSFUL COLLECTION COUNT     : 663
    LAST COLLECTION TIME            : 08/02/2010 14:24:38
    LAST SUCCESSFUL COLLECTION TIME: 08/02/2010 14:24:38
    NEXT COLLECTION TIME            : 08/02/2010 14:39:38
    TOTAL MODEL COUNT               : 52
    SUCCESSFUL MODEL COUNT          : 0
    LAST MODEL TIME                 : 08/02/2010 08:24:37
    LAST SUCCESSFUL MODEL TIME      :
    NEXT MODEL TIME                 : 08/02/2010 20:24:37
    CHECK SPECIFIC PARAMETERS:
      COLLECTINT                    : 15
      MODELINT                      : 720
      COLLECTINACTIVE               : 1=ON
      DEBUG                         : 0=OFF
      THRESHOLD                     : 2
```

Figure 23-24 PFA_COMMON_STORAGE_USAGE check output

Figure 23-25 on page 524 shows the PFA_LOGREC_ARRIVAL_RATE check output.

```
CHECK NAME: PFA_LOGREC_ARRIVAL_RATE
    ACTIVE                          : YES
    TOTAL COLLECTION COUNT          : 164
    SUCCESSFUL COLLECTION COUNT     : 164
    LAST COLLECTION TIME            : 08/02/2010 13:40:38
    LAST SUCCESSFUL COLLECTION TIME: 08/02/2010 13:40:38
    NEXT COLLECTION TIME            : 08/02/2010 14:40:38
    TOTAL MODEL COUNT               : 44
    SUCCESSFUL MODEL COUNT          : 0
    LAST MODEL TIME                 : 08/02/2010 08:40:38
    LAST SUCCESSFUL MODEL TIME      :
    NEXT MODEL TIME                 : 08/02/2010 20:40:38
    CHECK SPECIFIC PARAMETERS:
      COLLECTINT                    : 60
      MODELINT                      : 720
      COLLECTINACTIVE               : 1=ON
      DEBUG                         : 0=OFF
      STDDEV                        : 2
      EXCEPTIONMIN                  : 25
```

Figure 23-25 PFA_LOGREC_ARRIVAL_RATE check output

Figure 23-26 shows the PFA_FRAMES_AND_SLOTS_USAGE check output.

```
CHECK NAME: PFA_FRAMES_AND_SLOTS_USAGE
    ACTIVE                          : YES
    TOTAL COLLECTION COUNT          : 663
    SUCCESSFUL COLLECTION COUNT     : 663
    LAST COLLECTION TIME            : 08/02/2010 14:24:38
    LAST SUCCESSFUL COLLECTION TIME: 08/02/2010 14:24:38
    NEXT COLLECTION TIME            : 08/02/2010 14:39:38
    TOTAL MODEL COUNT               : 52
    SUCCESSFUL MODEL COUNT          : 0
    LAST MODEL TIME                 : 08/02/2010 08:24:38
    LAST SUCCESSFUL MODEL TIME      :
    NEXT MODEL TIME                 : 08/02/2010 20:24:38
    CHECK SPECIFIC PARAMETERS:
      COLLECTINT                    : 15
      MODELINT                      : 720
      COLLECTINACTIVE               : 1=ON
      DEBUG                         : 0=OFF
      STDDEV                        : 3
```

Figure 23-26 PFA_FRAMES_AND_SLOTS_USAGE check output

Figure 23-27 on page 525 shows the PFA_MESSAGE_ARRIVAL_RATE check output.

```
CHECK NAME: PFA_MESSAGE_ARRIVAL_RATE
    ACTIVE                          : YES
    TOTAL COLLECTION COUNT          : 634
    SUCCESSFUL COLLECTION COUNT     : 634
    LAST COLLECTION TIME            : 08/02/2010 14:10:38
    LAST SUCCESSFUL COLLECTION TIME: 08/02/2010 14:10:38
    NEXT COLLECTION TIME            : 08/02/2010 14:25:38
    TOTAL MODEL COUNT               : 13
    SUCCESSFUL MODEL COUNT          : 13
    LAST MODEL TIME                 : 08/02/2010 05:40:36
    LAST SUCCESSFUL MODEL TIME      : 08/02/2010 05:40:36
    NEXT MODEL TIME                 : 08/02/2010 17:40:36
    CHECK SPECIFIC PARAMETERS:
      COLLECTINT                    : 15
      MODELINT                      : 720
      COLLECTINACTIVE               : 1=ON
      DEBUG                         : 0=OFF
      STDDEV                        : 10
      TRACKEDMIN                    : 0
      EXCEPTIONMIN                  : 1
    EXCLUDED JOBS:
      NAME     SYSTEM   DATE ADDED       REASON ADDED
      JES*     *        2010/03/31 00:00 Exclude JES* jobs on all sys
```

Figure 23-27 PFA_MESSAGE_ARRIVAL_RATE check output

Figure 23-28 shows the PFA_SMF_ARRIVAL_RATE check detail.

```
CHECK NAME: PFA_SMF_ARRIVAL_RATE
    ACTIVE                          : YES
    TOTAL COLLECTION COUNT          : 634
    SUCCESSFUL COLLECTION COUNT     : 634
    LAST COLLECTION TIME            : 08/02/2010 14:10:38
    LAST SUCCESSFUL COLLECTION TIME: 08/02/2010 14:10:38
    NEXT COLLECTION TIME            : 08/02/2010 14:25:38
    TOTAL MODEL COUNT               : 13
    SUCCESSFUL MODEL COUNT          : 13
    LAST MODEL TIME                 : 08/02/2010 05:40:36
    LAST SUCCESSFUL MODEL TIME      : 08/02/2010 05:40:36
    NEXT MODEL TIME                 : 08/02/2010 17:40:36
    CHECK SPECIFIC PARAMETERS:
      COLLECTINT                    : 15
      MODELINT                      : 720
      COLLECTINACTIVE               : 1=ON
      DEBUG                         : 0=OFF
      STDDEV                        : 3
      TRACKEDMIN                    : 0
      EXCEPTIONMIN                  : 1
```

Figure 23-28 PFA_SMF_ARRIVAL_RATE check detail

Using the PFA operator commands

The EXCLUDED_JOBS file is always read and processed when PFA starts. To support changes to this file without needing to stop the PFA address space, the PFA **MODIFY** command has been enhanced to allow updates to this file. The **MODIFY** command's display option has also been updated to display the contents of the file.

To exclude a job that causes false positives from future processing:

- Create or edit the EXCLUDED_JOBS file in the check's /config directory.
  - ABC*,LPAR1,03/03/2010,Exclude all ABC* jobs on LPAR1
  - Allows a wildcard in both the job name and system name
- If PFA is running, issue the **MODIFY PFA update** command for the check; for example:

```
f pfa,update,check(PFA_MESSAGE_ARRIVAL_RATE)
```

To display which jobs are being excluded by the check, display the details of the check using the PFA **MODIFY** command's display option; for example:

```
f pfa,display,check(PFA_MESSAGE_ARRIVAL_RATE),detail
```

If PFA is running and updates have been made to an EXCLUDED_JOBS file and you want the changes to take effect immediately, you must issue the **MODIFY PFA update** command for the file to be reread.

The display option of the **MODIFY PFA** command has also been enhanced to display the list of jobs being excluded for each check

Display PFA infrastructure status

The PFA infrastructure status can also be displayed using the PFA **MODIFY** command. The number of checks registered is the number of checks that exist in the PFA infrastructure. The number of checks active are the number of PFA checks that are ACTIVE(ENABLED) in Health Checker as shown in Figure 23-29.

```
F PFA,DISPLAY
AIR017I 15:29:35 PFA STATUS
NUMBER OF CHECKS REGISTERED    : 5
NUMBER OF CHECKS ACTIVE        : 5
COUNT OF COLLECT QUEUE ELEMENTS: 0
COUNT OF MODEL QUEUE ELEMENTS  : 0
COUNT OF JVM TERMINATIONS      : 0
```

Figure 23-29 PFA checks status

23.8.1 SDSF support for PFA

After PFA has been started, the SDSF Health Checker panel (CK) displays the PFA checks, as shown in Figure 23-30.

```
Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
SDSF HEALTH CHECKER DISPLAY  SC74                      LINE 55-77 (158)
COMMAND INPUT ===>                                        SCROLL ===> HALF
PREFIX=SYSLOG  DEST=(ALL)  OWNER=*  SYSNAME=SC74
NP   NAME                            CheckOwner       State              Status
S    PFA_COMMON_STORAGE_USAGE        IBMPFA           ACTIVE(ENABLED)    SUCCES
     PFA_FRAMES_AND_SLOTS_USAGE      IBMPFA           ACTIVE(ENABLED)    SUCCES
     PFA_LOGREC_ARRIVAL_RATE         IBMPFA           ACTIVE(ENABLED)    SUCCES
     PFA_MESSAGE_ARRIVAL_RATE        IBMPFA           ACTIVE(ENABLED)    SUCCES
     PFA_SMF_ARRIVAL_RATE            IBMPFA           ACTIVE(ENABLED)    SUCCES
     RACF_FACILITY_ACTIVE            IBMRACF          ACTIVE(ENABLED)    SUCCES
     RACF_GRS_RNL                    IBMRACF          ACTIVE(ENABLED)    SUCCES
     RACF_IBMUSER_REVOKED            IBMRACF          ACTIVE(ENABLED)    EXCEPT
     RACF_ICHAUTAB_NONLPA            IBMRACF          ACTIVE(ENABLED)    SUCCES
     RACF_OPERCMDS_ACTIVE            IBMRACF          ACTIVE(ENABLED)    SUCCES
     RACF_SENSITIVE_RESOURCES        IBMRACF          ACTIVE(ENABLED)    EXCEPT
     RACF_TAPEVOL_ACTIVE             IBMRACF          ACTIVE(ENABLED)    EXCEPT
     RACF_TEMPDSN_ACTIVE             IBMRACF          ACTIVE(ENABLED)    EXCEPT
     RACF_TSOAUTH_ACTIVE             IBMRACF          ACTIVE(ENABLED)    SUCCES
     RACF_UNIXPRIV_ACTIVE            IBMRACF          ACTIVE(ENABLED)    SUCCES
     RCF_PCCA_ABOVE_16M              IBMRCF           ACTIVE(ENABLED)    SUCCES
     RRS_ARCHIVECFSTRUCTURE          IBMRRS           ACTIVE(ENABLED)    SUCCES
     RRS_DUROFFLOADSIZE              IBMRRS           ACTIVE(ENABLED)    SUCCES
     RRS_MUROFFLOADSIZE              IBMRRS           ACTIVE(ENABLED)    SUCCES
     RRS_RMDATALOGDUPLEXMODE         IBMRRS           ACTIVE(ENABLED)    EXCEPT
     RRS_RMDOFFLOADSIZE              IBMRRS           ACTIVE(ENABLED)    SUCCES
     RRS_RSTOFFLOADSIZE              IBMRRS           ACTIVE(ENABLED)    SUCCES
     RRS_STORAGE_NUMLARGELOGBLKS     IBMRRS           ACTIVE(ENABLED)    SUCCES
```

Figure 23-30 SDSF CK panel showing the PFA checks

Browsing check output

To browse the output of a check, enter `S` to display the common storage usage check; see Figure 23-30.

The check output is displayed in Figure 23-31.

```
 Display  Filter  View  Print  Options  Search  Help
 -------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY PFA_COMMON_STORAGE_USAGE      LINE 0       COLUMNS 02- 81
 COMMAND INPUT ===>                                           SCROLL ===> HALF
******************************** TOP OF DATA *********************************
CHECK(IBMPFA,PFA_COMMON_STORAGE_USAGE)
START TIME: 08/02/2010 15:38:52.435417
CHECK DATE: 20071101  CHECK SEVERITY: MEDIUM
CHECK PARM: DEBUG(0) THRESHOLD(2) COLLECTINT(15) MODELINT(720)
COLLECTINACTIVE(1)


AIRH103I Comparisons of predictions and arrivals will occur when the
check is run after modeling has run and succeeded.  Modeling is
scheduled for 08/02/2010 20:24:37.

END TIME: 08/02/2010 15:38:52.438177  STATUS: SUCCESSFUL
******************************* BOTTOM OF DATA ********************************
```

Figure 23-31 PFA Common Storage Usage check output

Note: It is desirable to predict common storage problems before they occur, determine the cause of the problem, and take the appropriate action. When IBM Health Checker for z/OS issues exception message AIRH101E, PFA has predicted that the amount of storage allocated to the Common storage area is in jeopardy of being exhausted.

Possible exception messages

If the check has an exception condition, follow these to analyze the problem.

1. Examine the Common Storage Usage Prediction Report issued with the exception message.

   This report contains the total current usage and predictions for each of the six storage locations: CSA, SQA, ECSA, ESQA, CSA+SQA, and ECSA+ESQA. It also contains up to ten “users” each of CSA, SQA, ECSA, and ESQA whose usage has changed the most in the last model interval. The cause of the problem is most likely within this list of users.

2. If the cause of the problem is not obvious from the common storage usage report, you can obtain additional information in the `csadata` and the `csaAlldata` files, or from other checks that list the top users of storage such as the checks owned by IBMVSM (VSM_CSA_THRESHOLD and VSM_SQA_THRESHOLD).

   These are text files in comma-separated value (`.csv`) format. They contain the historical data on the usage for each interval. You can export the files into any spreadsheet-type program.

Exception report example

An example of the Common Storage Usage Prediction Report is shown in Figure 23-32.

```
Common Storage Usage Prediction Report
Last successful model time : 07/09/2009 11:08:44
Next model time : 07/09/2009 23:12:44
Model interval : 720
Last successful collection time: 07/09/2009 11:10:52
Next collection time : 07/09/2009 11:25:52
Collection interval : 15
Capacity When Percentage
Storage Current Usage Prediction Predicted of Current
Location in Kilobytes in Kilobytes in Kilobytes to Capacity
_________ _____________ _____________ _____________ ___________
*CSA 2796 3152 2956 95%
SQA 455 455 2460 18%
CSA+SQA 3251 3771 5116 64%
ECSA 114922 637703 512700 22%
ESQA 8414 9319 13184 64%
ECSA+ESQA 123336 646007 525884 23%
Address spaces with the highest increased usage:
Job Storage Current Usage Predicted Usage
Name Location in Kilobytes in Kilobytes
_________ ________ _________________ _________________
JOB3 *CSA 1235 1523
JOB1 *CSA 752 935
JOB5 *CSA 354 420
JOB8 *CSA 152 267
JOB2 *CSA 75 80
JOB6 *CSA 66 78
JOB15 *CSA 53 55
JOB18 *CSA 42 63
JOB7 *CSA 36 35
JOB9 *CSA 31 34
* = Storage locations that caused the exception.
```

Figure 23-32 Common Storage Usage Prediction report

23.9 Installing PFA

PFA is shipped with z/OS, starting in V1R11. It contains everything available in the original SPE for V1R10, and two additional checks. For both releases, follow the install instructions carefully. Installation and configuration takes approximately 30 minutes. You are required to create a user and to run a provided installation script.

PTFs required for z/OS V1R10

The PTFs required for z/OS V1R10 are:

- APAR OA27165 (PTFs UA46241 and UA46243)
- A prerequisite MAPMVS APAR OA25773 (PTF UA43943)

PFA documentation is provided in *z/OS Problem Management*, G325-2564. For z/OS V1R10, documentation has been available online since March 2009.

Creating a user ID

Create a user ID (for instance, pfauser) to define the location in the z/OS UNIX file system that stores the PFA data and connects the PFA user ID to an existing or new RACF group. The home directory of the user ID that owns the PFA started task must match where the install script is run.

If you are using PFA in a sysplex that shares file systems for z/OS UNIX, use a unique directory for each LPAR so that the event data that PFA writes to the file system is stored separately for each system.

Note: Do not use the same user ID that is assigned to the IBM Health Checker for z/OS.

Defining the PFA started task

Define the PFA started task by creating a RACF profile for the pfauser with the following characteristics:

- OMVS segment with a UID parameter (for example, omvs(uid(7)))
- Home directory (for example, home(/pfa))
- PROGRAM path name of `/bin/sh` (for example, `program(/bin/sh)`)

Copying the sample PFA procedure

Copy the sample PFA procedure AIRPROC from SYS1.SAMPLIB to the PFA member of the SYS1.PROCLIB data set. If SMP/E does not write the executable code in the z/OS UNIX file system to `PARM='path=(/usr/lpp/bcp)'`, then change the PARM value in AIRPROC to the path in which you store the executable code.

Starting PFA in z/OS V1R13

PFA is an address space started by **`start pfa`**. In z/OS V1R13, PFA is dependent on Runtime Diagnostics running, which is started by **`start hzr`**. It is advisable to start both at IPL

Automation is recommended for the following new PFA exception messages with z/OS V1R13:

- PFA_ENQUEUE_REQUEST_RATE check
  - AIRH192E – Tracked job enqueue request rate higher than expected
  - AIRH210E – Total system enqueue request rate higher than expected
- PFA_JES_SPOOL_USAGE check
  - AIRH198E – JES2 spool usage higher than expected
  - LI1971 – PFA Integration with Runtime Diagnostics
  - AIRH190E – Tracked job enqueue request rate lower than expected
  - AIRH211E – Total system enqueue request rate lower than expected
  - AIRH206E – Tracked job message arrival rate lower than expected
  - AIRH207E – Other persistent job message arrival rate lower than expected
  - AIRH153E – Total system message arrival rate lower than expected
  - AIRH208E – Tracked job SMF arrival rate lower than expected
  - AIRH209E – Other persistent job SMF arrival rate lower than expected
  - AIRH175E – Total system message SMF arrival rate lower than expected

23.9.1 Migration considerations for PFA with z/OS V1R12

There is a new parameter provided when running the PFA install script AIRSHREP.sh. The new AIRSHREP.sh parameter is migrate. It is used to install only new checks that are added by z/OS V1R12. Figure 23-33 shows an example of its usage.

When your installation is migrating from z/OS V1R10 or V1R11 to further releases, you must provide one of the following parameters when running the install script AIRSHREP.sh.

Important: If you do not append the migrate parameter or the new parameter, or if you specify the parameter incorrectly, the script fails.

If the system does not find the `/etc/PFA` directory when running the install script AIRSHREP, the `ini` file is copied to each check directory (for both new and the migrate option).

Beginning with z/OS V1R12, PFA can use the single `ini` file for all checks in the `/etc/PFA` directory. This means you only have to update and maintain one `ini` file.

If you want a specific check to use a level of Java other than the one that is specified in the `/etc/PFA/ini` directory, then provide an `ini` file in the check directory for the check. For example, create an ini file in the `pfa_directory/ PFA_MESSAGE_ARRIVAL_RATE/` directory if you want to use another level of Java for the PFA_MESSAGE_ARRIVAL_RATE check.

If the path to the JDK for your installation is not the same as the path in the ini file in `/etc/PFA/` and in the checks' directories (if they exist), or if you installed the PFA Java code in a location other than the default path, you must update each `ini` file after running the install script for PFA.

New parameter starting with z/OS V1R12

This parameter has to be used either from the home directory that PFA is using, or when using the sample JCL provided in SYS1.SAMPLIB.

migrate Use the migrate parameter to preserve PFA history data from the prior release. The migrate option is recommended for all installations that previously used PFA.

Note: The script creates the directory structures for all checks that have not been previously installed on your system.

The script copies the first `ini` file found from one of the existing check directories to the `/etc/PFA/` directory starting with the `pfa_directory/PFA_COMMON_STORAGE_USAGE/` check.

By copying an existing `ini` file, the Java configuration from previous installations of PFA for an existing check is automatically applied.

If you do not want this Java configuration for all the checks, you can create the `ini` files on a per-check basis.

new Use the new parameter if you are installing PFA for the first time, or if you want to delete everything from prior releases and start PFA with empty directories.

Note: If you specify the new parameter when running the install script, the script deletes the existing check directories and creates a new directory structure for all the checks.

The script copies the `ini` file from the /usr/lpp/bcp/samples/PFA/ directory to the `/etc/PFA/` directory.

Figure 23-33 shows the usage of the migrate option.

```
ROGERS @ SC74:/u/rogers>/usr/lpp/bcp/AIRSHREP.sh migrate
Parameter specified on invocation: migrate
 Successfully created the Frame and Slots Usage check directory structure.
 Successfully created the SMF Arrival Rate check directory structure.
 Successfully created the EXCLUDED_JOBS file for the Message Arrival Rate check
 Successfully copied the sample ini file to the /etc/PFA directory
 >>>> Update the /etc/PFA/ini file to the level of Java
 >>>> that your installation uses.
```

Figure 23-33 PFA installation for z/OS V1R12

Figure 23-34 shows the list of PFA directories.

```
ROGERS @ SC74:/u/rogers>ls -al
total 200
drwxr-x---  13 SYSPROG  SYS1         928 Aug  2 16:33 .
dr-xr-xr-x   8 SYSPROG  TTY            0 Aug  2 16:16 ..
-rw-------   1 SYSPROG  SYS1        1707 Aug  2 16:36 .sh_history
drwxr-xr-x   2 SYSPROG  SYS1         256 Sep 13  2006 00000100
drwxr-xr-x   5 SYSPROG  OMVSGRP     1120 Jul 26  2006 ITSO-link
drwxrwxrwx   4 SYSPROG  SYS1         352 Aug  2 16:33 PFA_COMMON_STORAGE_USAGE
drwxr-xr-x   4 SYSPROG  SYS1         320 Aug  2 16:33 PFA_FRAMES_AND_SLOTS_USAGE
drwxrwxrwx   4 SYSPROG  SYS1         352 Aug  2 16:33 PFA_LOGREC_ARRIVAL_RATE
drwxrwxrwx   4 SYSPROG  SYS1         352 Aug  2 16:33 PFA_MESSAGE_ARRIVAL_RATE
drwxr-xr-x   4 SYSPROG  SYS1         320 Aug  2 16:33 PFA_SMF_ARRIVAL_RATE
drwxrwxrwx   3 SYSPROG  SYS1         320 Aug 12  2009 PFA_VIRTUAL_STORAGE_USAGE
drwxr-xr-x   2 SYSPROG  SYS1         256 Sep 13  2006 db2
drwxr-xr-x   2 SYSPROG  SYS1         256 Sep 13  2006 echo
drwxr-xr-x   2 SYSPROG  SYS1         256 Sep 13  2006 ims
```

Figure 23-34 PFA directory structure

Directory structure with z/OS V1R12

Under UNIX System Services (for example, OMVS as shown in Figure 23-35), the files needed for PFA are located in the `/usr/lpp/bcp` directory.

```
ROGERS @ SC74:/u/rogers>ls -al /usr/lpp/bcp
total 1888
drwxr-xr-x   5 SYSPROG  OMVSGRP      544 Jun 16 16:32 .
drwxr-xr-x  59 SYSPROG  OMVSGRP     2112 Jun 29 08:17 ..
-rwxr-xr-x   2 SYSPROG  OMVSGRP    73011 May 18 04:05 AIRJCART.jar
-rwxr-xr-x   2 SYSPROG  OMVSGRP    49602 May 18 04:05 AIRJCHK.jar
-rwxr-xr-x   2 SYSPROG  OMVSGRP   249856 May 18 04:05 AIRLCSAC
-rwxr-xr-x   2 SYSPROG  OMVSGRP   278528 May 18 04:05 AIRLMARC
-rwxr-xr-x   2 SYSPROG  OMVSGRP   245760 May 18 04:05 AIRLVSUC
-rwxr-xr-x   2 SYSPROG  OMVSGRP    14230 Jun 16 16:32 AIRSHREP.sh
drwxr-xr-x   2 SYSPROG  OMVSGRP     1504 May 18 04:05 IBM
drwxr-xr-x   7 SYSPROG  OMVSGRP      416 May 18 02:45 mca
drwxr-xr-x   3 SYSPROG  OMVSGRP      480 May 18 04:05 samples
ROGERS @ SC74:/u/rogers> ===>
```

Figure 23-35 Directory structure with z/OS V1R12

Installing PFA in a z/OS UNIX shared file system environment

If you have a zFS shared file system environment, a z/OS UNIX file system can be created that is shared among members of the sysplex with directories that are local to the LPAR.

To install PFA in such an environment, follow these steps:

1. Define the file systems as one for each LPAR.
2. After defining the file systems for each LPAR, define a symbolic link (that is, a symlink) to the sysplex root. From the `root` directory, enter the command shown in Figure 23-36, using the UID you assigned to pfauser.

```
cd
ln -s \$SYSNAME/pfa pfa
```

Figure 23-36 Define symlink

   This results in a home directory of `/systemname/pfa`.
3. Create the PFA directory in each of the system directories by entering the following commands for each of your system names. For example, the command for system Z1 is **`mkdir /Z1/pfa:`** and so on for each system.
4. Create the new file systems (one for each system) and mount them at the appropriate system mount point. For example, for OMVSSPT.Z1.PFA.ZFS, the mount point is `z1/pfa/etc` for each system.
5. Place an entry in the SYS1.PARMLIB(BPXPRMxx) member to mount the new file systems during IPL. (If you do not want to wait until the next IPL, you can manually mount these file systems.) Use the UNMOUNT attribute on the BPXPRMxx parmlib member to unmount the file system when OMVS or the LPAR is taken down. The file system mount point is `/sysname/pfa` as shown here:

```
SYS1.PARMLIB(BPXPRM00)
MOUNT FILESYSTEM('OMVSSPT.&SYSNAME..PFA.ZFS') TYPE(ZFS)
MODE(RDWR) MOUNTPOINT('/&SYSNAME./pfa') UNMOUNT
```

23.9.2 PFA supports one or multiple ini files

In z/OS V1R10, there was one `ini` file for each check when PFA was first introduced with only two checks. However, after z/OS V1R11 when there were four checks, having one `ini` file per check became cumbersome. Therefore, in z/OS V1R12, PFA was enhanced to support having one `ini` file for all checks or having multiple `ini` files so individual checks can have their own `ini` file and other checks can continue to use the default.

The only reason to have multiple `ini` files (one per check) is to use different Java configurations for each check.

The default `ini` file is located in the `/etc/PFA` directory. If an `ini` file exists in a check's directory, that file will be used. Otherwise, the `ini` file in `/etc/PFA` will be used.

The `/etc/PFA` directory is automatically created when z/OS V1R12 is installed. However, if your installation uses steps that can cause the `/etc/PFA` directory to no longer exist, the new `ini` file processing cannot be used.

If you do not have the `/etc/PFA` directory after installing, you can create it manually and assign it the proper authorities as described in z/OS problem management documentation.

23.10 Serviceability information

Originally, PFA was designed to be a "black box" that you started and then forgot about. However, it was later decided through client interaction that more information was needed to ensure that PFA runs correctly. Therefore, a comprehensive **`modify PFA`** command was created to display status information for the PFA infrastructure and detailed status for each individual check.

When a check exception is issued, the reports previously described greatly assist in analyzing the problem. In addition, the PFA documentation outlines best practices for each check to provide advice on how to analyze the problem.

Each check has a `/data` directory in the pfauser's home directory. For example, if the PFA user is "pfauser," then the data directory for the CSA usage check is:

```
/u/pfauser/PFA_COMMON_STORAGE_USAGE/data
```

The files needed by PFA to collect data, model data, and perform the check are found in the `/data` directory. These files are documented in the PFA documentation and can be used to help you analyze the exception data.

If a PFA problem is suspected, keep in mind that the `/data` directory also contains log files that might contain additional debug information. If the PFA debug parameter is turned on, additional information will be found in the log files. If a PFA problem is suspected, IBM service will likely request the last exception report and the `/data` directory for the check. Preferably, recreate the problem with debug turned on.

For a "debug" parameter that is check-specific, note the following points:

- Additional diagnostic information is generated in the log files when debug is turned on.
- This parameter is not the debug parameter available through Health Checker because the Health Checker parameter did not apply to all three major functions. Instead, it is a parameter listed in the check-specific parameters for each check.

If a problem is suspected in PFA itself, PFA can simply be stopped. In addition, if a problem is suspected in one of the PFA checks, that check can be deleted from Health Checker.

Deleting only the check that is failing will allow the other checks to continue processing. Deletion can be performed by a command as shown in Figure 23-37.

```
f hzsproc,delete,check(ibmpfa,PFA_COMMON_STORAGE_USAGE)
```

Figure 23-37 Deleting a check from Health Checker

23.10.1 Serviceability improvements starting with z/OS V1R12

Prior to z/OS V1R12, there were several log files for each check, but much of the processing wrote to one log file and overlaid the logging of the previous step.

Starting in z/OS V1R12, each check has several log files that are consistently named. There is a log file for configuration changes (CONFIG.LOG); a log file for collection processing (COLLECT.LOG); a log file for modeling (MODEL.LOG); and a log file for when the check is run (RUN.LOG).

There are other log files for each check, depending on the type of processing the check performs.

In addition, if PFA issues an exception for a potential problem, the log files and the data files needed to investigate the exception are copied to a new directory that is created in the check's directory.

For example, if an exception is issued for the PFA_COMMON_STORAGE_USAGE check, the new directory with the data and log files are created in the `/u/pfauser/PFA_COMMON_STORAGE_USAGE` directory. The name of the new directory will start with EXC_ and the time stamp of the exception will be concatenated to it.

After 30 directories have been created, the oldest directory and its contents will be deleted prior to creating a new directory.

Software dependencies

PFA requires Java version 5 or greater.

New with z/OS V1R13:

With the integration of Runtime Diagnostics with PFA, using operlog is suggested for complete analysis.

Interactions

PFA is built into the operating system. It looks for a small number of generic events that can cause a soft failure. It does not look for events or soft failures in specific address spaces unless they can cause a system crash or hang.

PFA is operating system-centric in that it works on z/OS. It needs minimal installation and configuration. It learns the behavior of the individual behavior and creates predictions for that behavior. It detects soft failures by using complex algorithms imbedded in the component to compare the model behavior for that particular system to the current behavior.

PFA is built using remote health check support. It provides the information about soft failures through IBM Health Checker for z/OS, which issues the exception to the console (if so configured) and the exception and report data to the health check output in SDSF.

From the messages provided by PFA through the health checker support to the console, other products can be used to further analyze the situation. Figure 23-38 illustrates, for example, how PFA and Tivoli products work together.

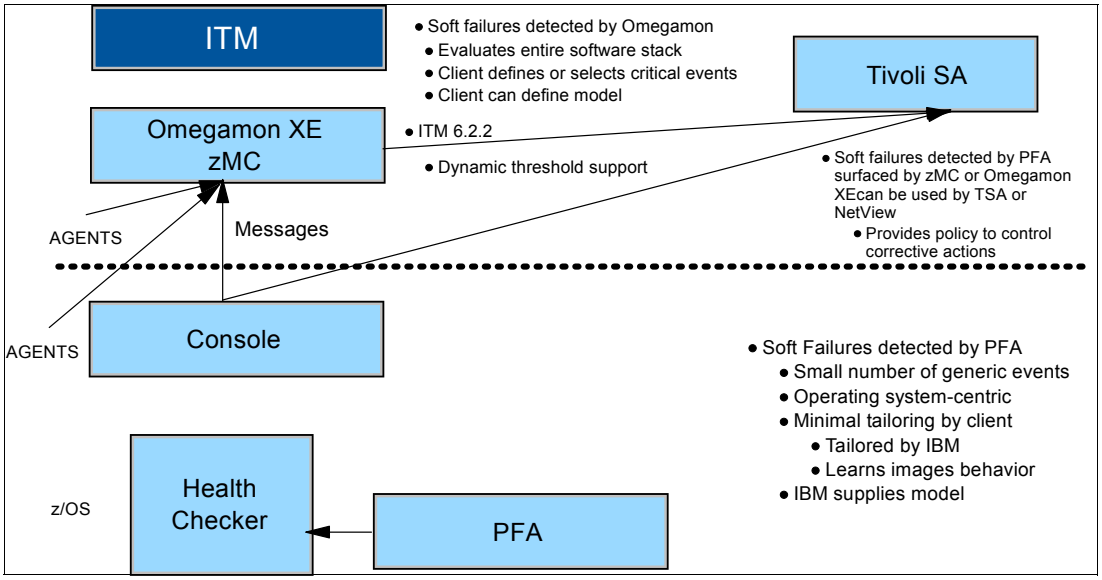


Figure 23-38 How PFA and Tivoli products work together

The OMEGAMON XE for Management Console will see all health check alerts, including the PFA alerts. You can build a solution that will alert you if a PFA check is raised and forward that event to other Tivoli event management products such as OMNIBUS or Tivoli Event Console.

23.10.2 Migration actions to z/OS V1R12 and beyond

If your installation saves the `/etc` directory and then restores it after a release upgrade, you must create the `/etc/PFA` directory manually after release upgrade because it will have been overlaid. If you omit this step and the `/etc/PFA` directory does not exist on your system, you cannot benefit from this new feature.

Consider the following installation and customization items:

- PFA provides sample JCL in z/OS V1R12 so that your PFA installation can run in batch.
- Run AIRSHREP.sh directly or use the JCL file provided in SYS1.SAMPLIB(AIRINJCL)
- If using AIRSHREP.sh directly, you must run it from the PFA user's home directory.
- If using AIRINJCL, you must update it to specify your PFA user's home directory.
- If you are running AIRSHREP.sh directly, you must execute it from the pfauser's home directory so that files are created in proper locations and so that permissions are correct.
- If you are using AIRINJCL to run AIRSHREP.sh, you must update the AIRINJCL file to specify your PFA user's home directory.
- If this is a new install request, any previous directories are deleted, all directory paths are created, the sample `ini` file in `/usr/lpp/bcp/samples/PFA` is copied to `/etc/PFA`, and the

EXCLUDED_JOBS file is created for the message arrival rate check. You must specify the required parameter to tell the script whether to create a new installation or to migrate previous data.

Installation options

PFA allows you to either create a totally new installation or to use data from a previous release by specifying either the “new” or “migrate” options when running the install script. If you have used PFA in previous releases, use the “migrate” option.

If this is a request to migrate, it is assumed that you want to use a default ini file in `/etc/PFA` instead of maintaining an `ini` file for each check. When migrate is requested, all directory paths that do not exist are created, the `ini` file from an existing check starting with the common usage check is copied to the /etc/PFA directory, all `ini` files in the check’s directories are deleted, and the EXCLUDED_JOBS file is created for the message arrival rate check.

The `ini` file is copied from an existing check, so you will not need to update the new default file in `/etc/PFA` if your Java configuration has not changed. The script will attempt to copy an existing `ini` file. If it cannot locate one, it will copy the one from `/usr/lpp/bcp/samples/PFA`.

After the installation is complete, update the Java configuration in `/etc/PFA/ini` if it is not correct for your installation. If you still want to use multiple `ini` files, create them in the checks’ directories and update those as well.

23.10.3 Runtime Diagnostics setup

Note the following points when using the Runtime Diagnostics integrated with PFA:

- Use the new HZR proc shipped with z/OS V1R13.
- HZROUT now must go to a normal PS data set with a DISP=SHR, a LRECL=121, and a BLKSIZE=0.
- Runtime Diagnostics uses a QSAM option to always extend (that is, append new data to the end of the data set) the data set regardless of the DISP keyword used.
- GDGs are not recommended.
- Update COMMNDxx to start HZR during the IPL process.
  - As as alternative, you can use the installation's automation to start HZR.
  - Start the HZR address space under the master subsystem, as shown in Figure 23-39.

```
S HZR,SUB=MSTR
IRR812I PROFILE ** (G) IN THE STARTED CLASS WAS USED 065
    TO START HZR WITH JOBNAME HZR.
IEF695I START HZR      WITH JOBNAME HZR      IS ASSIGNED TO USER
IBMUSER , GROUP SYS1
IEF196I IEF695I START HZR      WITH JOBNAME HZR      IS ASSIGNED TO
IEF196I USER IBMUSER , GROUP SYS1
IEF196I IEF236I ALLOC. FOR HZR HZR
IEF196I IEF237I DMY  ALLOCATED TO HZROUT
HZR0112I RUNTIME DIAGNOSTICS INITIALIZATION COMPLETE
```

Figure 23-39 Starting HZR under the master subsystem

Otherwise, it cannot be started; see Figure 23-40.

```
S HZR
 IRR812I PROFILE ** (G) IN THE STARTED CLASS WAS USED 045
         TO START HZR WITH JOBNAME HZR.
 $HASP100 HZR      ON STCINRDR
 IEF695I START HZR      WITH JOBNAME HZR      IS ASSIGNED TO USER
 IBMUSER , GROUP SYS1
 $HASP373 HZR      STARTED
 HZR0107I RUNTIME DIAGNOSTICS WAS NOT STARTED UNDER THE MASTER
 SUBSYSTEM
 $HASP395 HZR      ENDED
```

Figure 23-40 Starting HZR under JES2

24

Extended address volume

z/OS V1R10 introduced the extended address volume (EAV), which allowed DASD storage volumes to be larger than 65,520 cylinders. The space above the first 65,520 cylinders is referred to as cylinder-managed space. Tracks in cylinder-managed space use extended addressing space (EAS) techniques to access these tracks. Data sets that are able to use cylinder-managed space are referred to as being EAS-eligible. EAV support has been enhanced for both z/OS V1R11, z/OS V1R12, and z/OS V1R13.

This chapter describes the following features, which are available with each of the releases. It also explains the changes that are specifically for z/OS V1R13:

- Basic EAV functionality
- Enhancements to EAV in z/OS V1R13
- New functions in DFSMSdfp to better cope with the huge spaces opened up by EAV capabilities, including:
  - Catalog PARMLIB member
  - Alias number constraint relief
  - Catalog VVDS expansion
- Enhanced FTP support for EAVs

24.1 Extended address volume overview

The extended address volume (EAV) is the next step in providing larger volumes for z/OS. z/OS provided this support for the first time in z/OS V1R10 of the operating system. Over the years, volumes have grown by increasing the number of cylinders and thus GB capacity. However, the existing track addressing architecture has limited the required growth to relatively small GB capacity volumes. This has put pressure on the 4-digit device number limit. (The largest available volume is one with 65,520 cylinders or approximately 54 GB).

With EAV volumes, an architecture is implemented that provides capacities of hundreds of terabytes for a single volume. However, the first releases are limited to a volume with 223 GB or 262,668 cylinders.

24.1.1 3390 Model A

A volume of this size must be configured in the IBM DS8000® as a 3390 Model A. However, a 3390 Model A is not always an EAV. A 3390 Model A is any device configured in the DS8000 to have from 1 to 268,434,453 cylinders (z/OS V1R10 to z/OS V1R12). Figure 24-1 illustrates the 3390 device types.

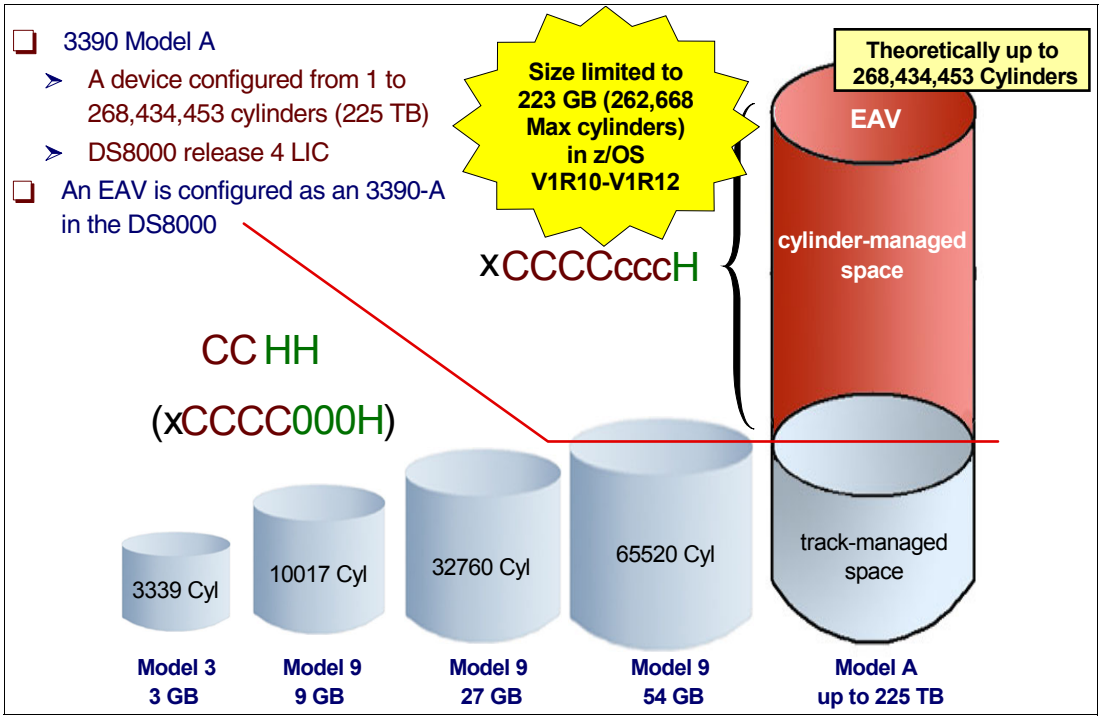


Figure 24-1 DS8000 support for device type 3390

Be aware of the following points:

- 3390 Model A support is provided in DS8000 3.1 versions.
- The EAV support is provided in the DS8000 4.0 version and above.

Note: With the 3390 Model A, the model A refers to the model configured in the DS8000. It has no association with the 3390A notation in HCD that indicates a PAV-alias UCB in the z/OS operating system.

The model A was chosen so that it did not imply a particular device size, as previous models 3390-3 and 3390-9 did.

How an EAV is managed by the system allows it to be a general purpose volume. However, EAVs can work especially well for applications with large files. PAV and HyperPAV technologies help in both these regards by allowing I/O rates to scale as a volume gets larger.

24.1.2 EAV basic definitions

Note the following points regarding extended address volumes.

- Only 3390 Model A devices can be EAV.
- EAV is supported starting in z/OS V1R10 and further releases.
- The size is limited to 223 GB (262,668 cylinders) in z/OS V1R10-V1R12.

Important: The 3390 Model A as a device can be configured to have from 1 to 268,434,453 cylinders on an IBM DS8000. It becomes an EAV if it has more than 65520 cylinders defined. With current z/OS releases, this maximum size is currently not supported.

24.1.3 EAV terminology

Note: Two sets of terms are used to reference an EAV. One set is used to describe how space is managed. The other set is used to describe how the disk is addressed. The context of what is being described dictates which terminology to use.

extended address volume (EAV)	This term refers to a volume with more than 65520 cylinders. Only 3390 Model A devices can be an EAV.
track address	This term refers to a 32-bit number that identifies each track within a volume. It is in the format hexadecimal CCCCcccH, where CCCC is the low order 16 bits of the cylinder number, ccc is the high order 12 bits of the cylinder number, and H is the four-bit track number. For compatibility with older programs, the ccc portion is hexadecimal 000 for tracks in the base addressing space.
extended addressing space (EAS)	On an extended address volume, this term refers to cylinders with addresses that are equal to or greater than 65536. These cylinder addresses are represented by 28-bit cylinder numbers.
base addressing space	On an extended address volume, this term refers to cylinders with addresses below 65536. These cylinder addresses are represented by 16-bit

cylinder numbers or by 28-bit cylinder numbers with high order 12 bits of zero (0).

multicylinder unit — This term refers to a fixed unit of disk space that is larger than a cylinder. Currently, on an EAV, a multicylinder unit is 21 cylinders and the number of the first cylinder in each multicylinder unit is a multiple of 21.

cylinder-managed space — This term refers to the space on the volume that is managed only in multicylinder units. Cylinder-managed space begins at cylinder address 65520. Each data set occupies an integral multiple of multicylinder units. Space requests targeted for the cylinder-managed space will be rounded up to the next multicylinder unit. The cylinder-managed space exists only on EAVs.

track-managed space — This term refers to the space on a volume that is managed in tracks and cylinders. Track-managed space ends at cylinder address 65519. Each data set occupies an integral multiple of tracks. Track-managed space also exists on all non-EAVs.

breakpoint value (BPV) — When a disk space request is this size or more, the system prefers to use the cylinder-managed space for that extent. This applies to each request for primary or secondary space for data sets that are eligible for the cylinder-managed space. If not enough cylinder-managed space is available, then the system will use the track-managed space or will use both areas. The breakpoint value is expressed in cylinders.

When the size of a disk space request is less than the breakpoint value, the system prefers to use the track-managed area. If not enough space is available there, then the system will use the cylinder-managed space, or will use both areas.

24.2 EAV key design points

An important EAV design point is that IBM maintains its commitment to clients that the 3390 track format and image size, and tracks per cylinders, will remain the same as previous 3390 model devices. An application using data sets on an EAV will be comparable to how it runs today on 3390 “numerics” kind of models. The extended address volume has two managed spaces: the track-managed space and the cylinder-managed space, as shown in Figure 24-2.

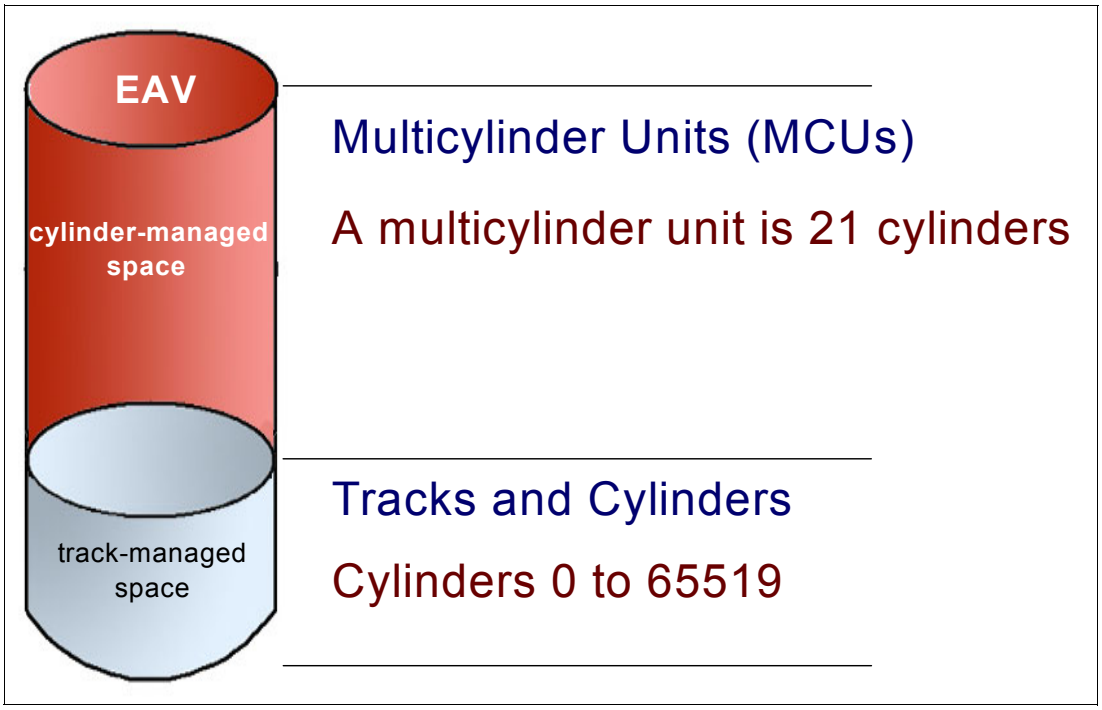


Figure 24-2 EAV and multicylinder units

24.2.1 Track-managed space

The track-managed space on a volume is managed in track and cylinder increments. All volumes today have track-managed space. The track-managed space ends at cylinder address 65519. Each data set occupies an integral multiple of tracks. The track-managed space allows existing programs and physical migration products to continue to work. Physical copies can be performed from a non-EAV to an EAV and have those data sets accessible.

24.2.2 Cylinder-managed space

The cylinder-managed space on a volume is managed only in multicylinder units). Cylinder-managed space begins at cylinder address 65520. Each data set occupies an integral multiple of multicylinder units. Space requests targeted for the cylinder-managed space is rounded up to the next multicylinder unit. The cylinder-managed space exists only on EAVs. A data set allocated in cylinder-managed space may have its requested space quantity rounded up to the next MCU.

Data sets allocated in cylinder-managed space are described with a new type of data set control blocks (DSCB) in the VTOC. Tracks allocated in this space will also be addressed using the new track address. Existing programs that are not changed will not recognize these new DSCBs and therefore will be protected from seeing how the tracks in cylinder-managed space are addressed.

24.2.3 Multicylinder unit

A multicylinder unit (MCU) is a fixed unit of disk space that is larger than a cylinder. Currently, on an EAV, a multicylinder unit is 21 cylinders and the number of the first cylinder in each multicylinder unit is a multiple of 21; see Figure 24-2.

The 21-cylinder value for the MCU is derived from being the smallest unit that can map out the largest possible EAV and stay within the index architecture (with a block size of 8192 bytes), as follows:

- It is also a value that divides evenly into the 1 GB storage segments of an IBM DS8000.
- These 1 GB segments are the allocation unit in the IBM DS8000 and are equivalent to 1113 cylinders.

These segments are allocated in multiples of 1113 cylinders starting at cylinder 65536.

24.2.4 DASD track address format

As explained, an EAV is defined to be a volume with more than 65520 cylinders. A volume of this size has to be configured in the DS8000 as a 3390 Model A. However, a 3390 Model A is not always an EAV.

A 3390 Model A is any device configured in the DS8000 to have from 1 to 268,434,453 cylinders. 3390 Model A support is provided in DS8000 3.1 versions. EAV support is provided in Version 4.0.

“EAV terminology” on page 541 discusses cylinder-managed space and track-managed space and describes how space is managed. The following sections describe how the disk is addressed using its new track address format.

Addressing extended address volumes

The extended address volume has two addressing spaces. To distinguish them from virtual storage, the term “address space” is not used.

- The base addressing space

  The base addressing space is the area on an EAV located within the first 65,536 cylinders, as shown in Figure 24-3. Tracks are addressed in this area with 16-bit cylinder numbers, described with the CCHH notation. The CC represents 16 bits for a cylinder address. The HH represents 16 bits for a track address, of which only the low order 4 bits are used. This is how all disks are addressed today.

- The extended addressing space (EAS)

  The extended addressing space (EAS), also shown in the figure, is the area on a EAV located above the first 65,536 cylinders. Tracks are addressed in this area with 28-bit cylinder numbers, described with the CCCCcccH notation; see “New track address for extended address volume” on page 546.

  This addressing is comparable to all 16-bit cylinder addressing. This area is similar to the cylinder-managed spaces, but is a subset of it. We often interchange the terminology of EAS and cylinder-managed space. This area is similar to the track-managed space but has a larger set of cylinders.

28-bit cylinder addressing

The 28-bit cylinder addressing architecture allows existing programs to address tracks in the base addressing space. The extended addressing space provides with a method to protect existing programs from accessing tracks in the EAS. This is done with the new data set control blocks (DSCBs) in the VTOC, which are discussed in “New format DSCBs for EAVs” on page 550.

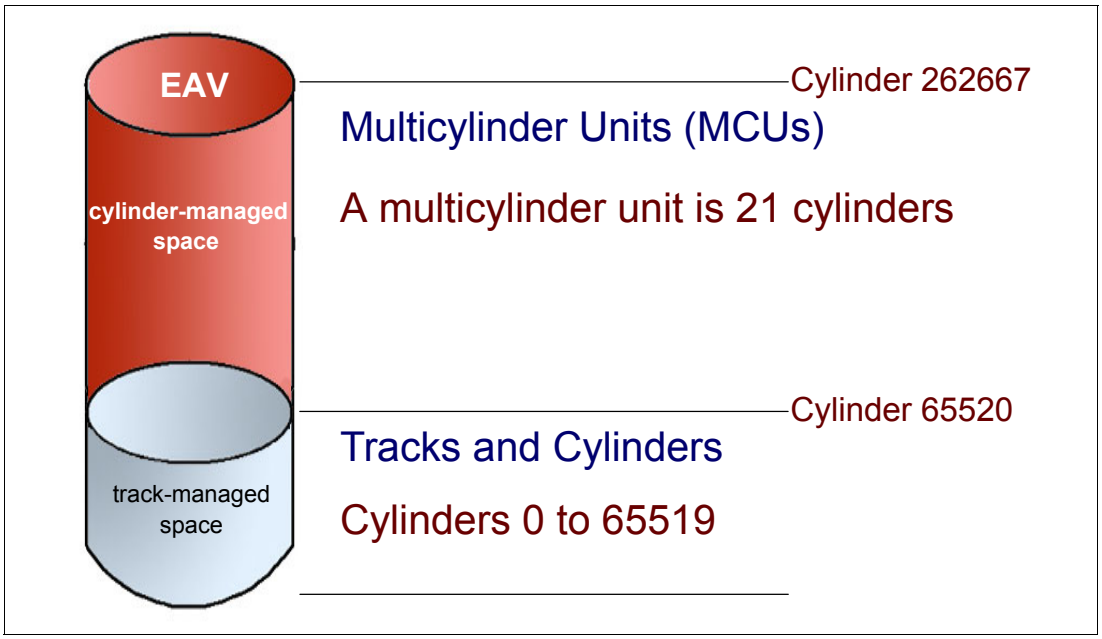


Figure 24-3 New track addressing for EAV volumes

Old track address

The base addressing space is the area on an EAV located within the first 65536 cylinders. As shown in Figure 24-4, tracks are addressed in this area with 16-bit cylinder numbers described with the CCHH notation, as follows:

- CC represents 16 bits for a cylinder address.
- HH represents 16 bits for a track address, of which only the low order 4 bits are used.

This is how all disks are addressed today. We generally refer to a track address using the CCHH notation. However, now a track address can be shown using the CCCCHHHH notation. This track address is a 32-bit number that addresses each track within a volume. Each cylinder and track number uses a 16-bit number. For the track number, only the low order 4 bits are used. The high order 12 bits of the track number are not used. Thus, to handle cylinder numbers greater than 65,520, a new format for the track address is required.

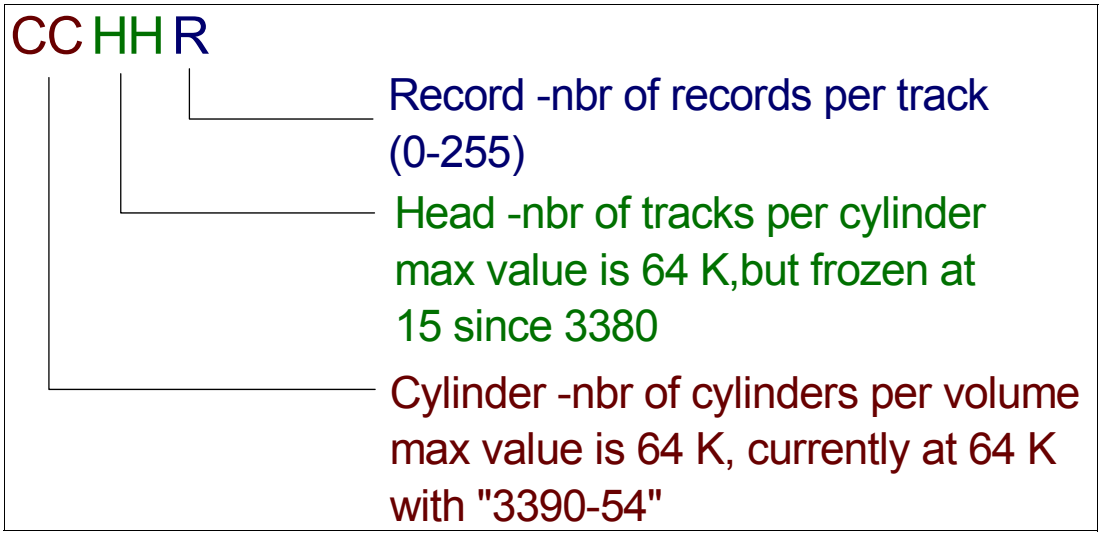


Figure 24-4 Old track address

New track address for extended address volume

The extended addressing space (EAS) is the area on an EAV located above the first 65536 cylinders. Tracks are addressed in this area with 28-bit cylinder numbers, described with the CCCCcccH (showing hex digits) notation, as follows:

- CCCC represents the low order 16 bits of a 28-bit number.
- ccc represents the high order 12 bits of a 28-bit number.
- H represents a 4-bit track number.

This addressing is comparable to all 16-bit cylinder addressing. This area is similar to the cylinder-managed spaces, but is a subset of it. We often interchange the terminology of EAS and cylinder-managed space. This area is similar to the track-managed space but has a larger set of cylinders.

The 28-bit cylinder addressing architecture allows existing programs to address tracks in the base addressing space. The extended addressing space provides a method to protect existing programs from accessing tracks in the EAS. This is done with the new data set control blocks (DSCBs) in the VTOC discussed in 24.2.8, “New format DSCBs for EAVs” on page 550.

For compatibility with older programs, the ccc portion is hexadecimal 000 for tracks in the base addressing space. This track address method is referred to as a 28-bit cylinder number. This format preserves the 3390 track geometry. Track addresses for space in track-managed space will be comparable to today’s track addresses. However, track addresses for space in cylinder-managed space will *not* be comparable to previous track addresses.

Note: For compatibility reasons, the 32 bits in each track address on an EAV is in the following format: CCCCcccH. The 12 high order bits of the cylinder number are in the high order 12 bits of the two old HH bytes. This format might be written as CCCCcccH, as shown in Figure 24-6.

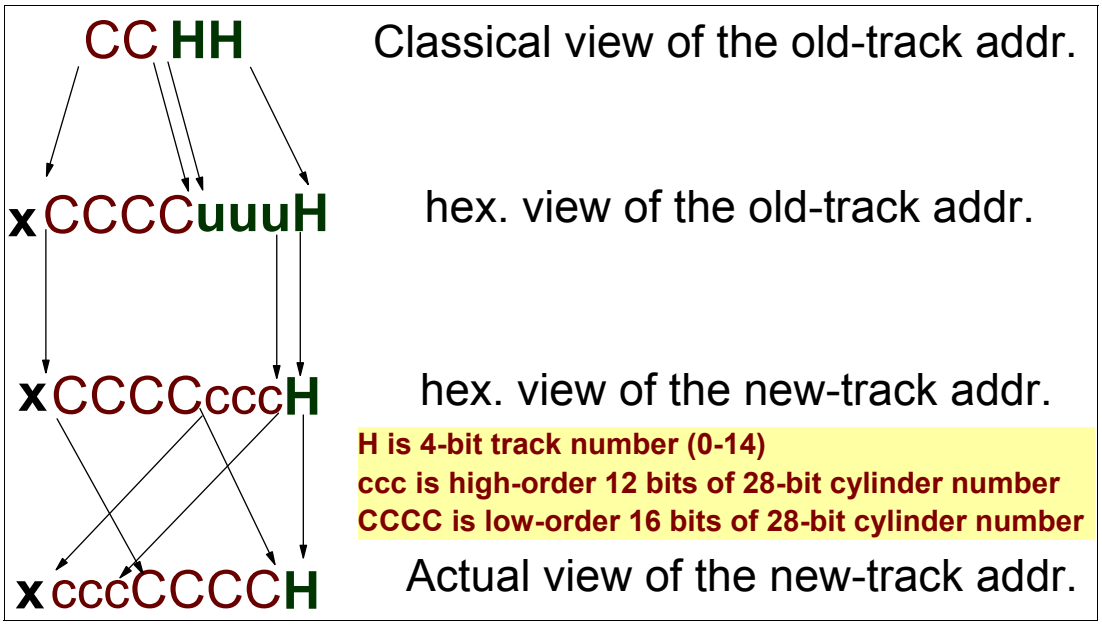


Figure 24-5 From old track address to new track address

Key points

The cylinder number is in a non-contiguous form. Reading this new track address, the hex digits must be rearranged as shown in Figure 24-6. This format preserves the 3390 track geometry.

Track addresses in existing channel programs and extent descriptors in DSCBs and elsewhere are in the form of CCHH, where CC is the 16-bit cylinder number and HH is the 16-bit track number in that cylinder. If the volume is an EAV, the cylinder number in these four CCHH bytes is 28 bits and the track number is four bits.

For compatibility reasons, the 32 bits in each track address on an EAV are in this format: CCCCcccH.

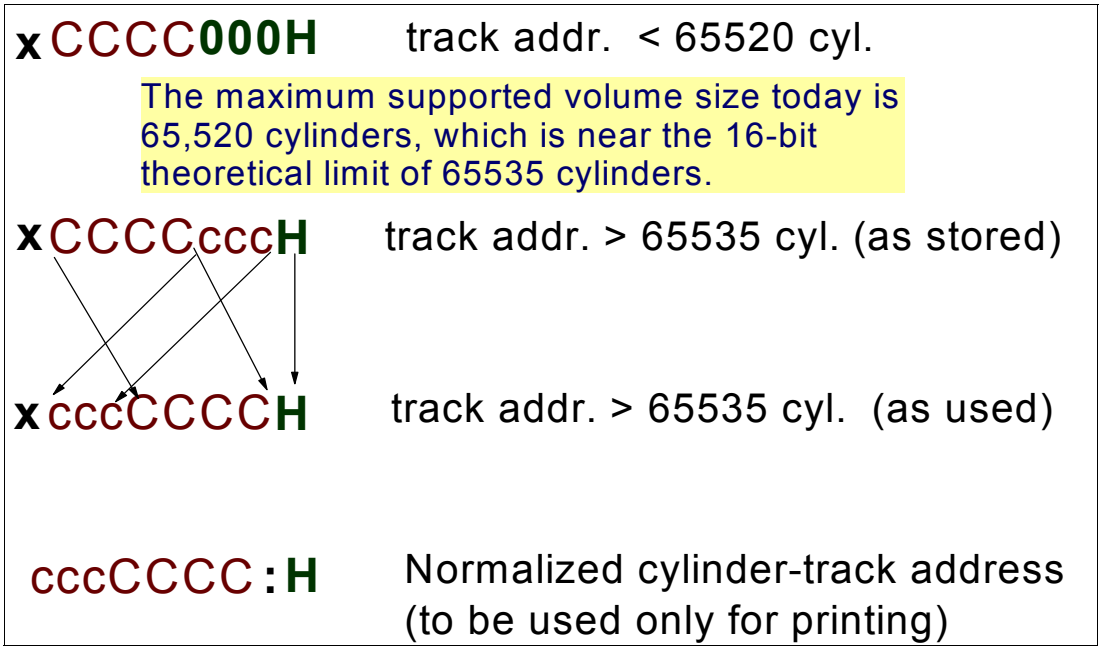


Figure 24-6 EAV: the two types of track address

24.2.5 Extended address volume attributes

In z/OS V1R10, each extended address volume, as shown in Figure 24-7, has either all or none of the following attributes. However, in a later release, they might be independent of each other. The new volume attributes are provided in the VTOC format-4 DSCB and the UCB device class extension. Each of these attributes is to be tested independently.

An exception is that a volume with more than 65520 cylinders requires format-8 and format-9 DSCBs. A non-extended address volume has none of these attributes.

As of today, an extended address volume has all of the following attributes:

- The volume supports an extended addressing space.
- The volume supports cylinder-managed space.
- The volumes supports extended attribute DSCBs.

Note: It is possible that in a future release a volume might simply have a lower level attribute. For example, we might have a volume that supports format-8 and format-9 DSCBs (extended attribute DSCBs) while the volume is smaller than an extended address volume.

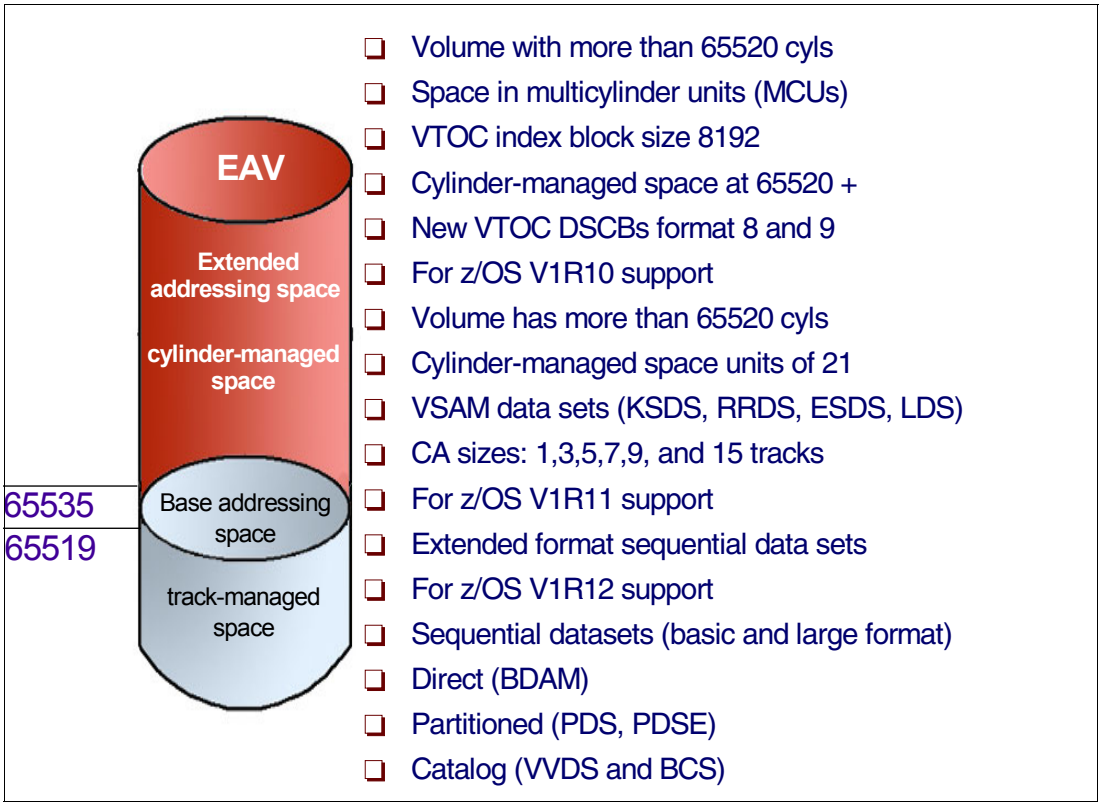


Figure 24-7 EAV attribute summary

24.2.6 EAS-eligible data sets

EAS-eligible data sets are those that can be allocated in the extended addressing space, which is the area on an EAV located above the first 65536 cylinders. This is sometimes referred to as cylinder-managed space. All of the following data sets can be allocated in the base addressing space of an EAV:

- SMS-managed VSAM (all types)
- Non-SMS VSAM (all types)
- zFS data sets (which are VSAM LS)
  - zFS aggregates are supported in an EAV environment
  - zFS aggregates/file systems can reside in track-managed space or cylinder-managed space (subject to any limitations that DFSMS might have)
  - zFS still has an architected limit of 4 TB for the maximum size of a zFS aggregate
- Database (DB2, IMS) use of VSAM
- VSAM data sets inherited from prior physical migrations or copies
- With z/OS V1R11: Extended-format sequential data sets that are SMS-managed can be allocated in the extended addressing space of an EAV.

- With z/OS V1R12**:** the following types of data sets can also be EAS eligible:
  - Sequential (Basic and Large Format)
  - Direct (BDAM)
  - Partitioned (PDS and PDSE), with some exceptions such as SYS1.NUCLEUS or SYS1.PARMLIB
  - Catalogs: VSAM volume data set (VVDS) basic catalog structure (BCS)
  - JES2 spool and checkpoint data sets
  - JES3 spool and checkpoint data sets

Note: Only VSAM data sets that are allocated with compatible control areas (CAs) for non-striped VSAM, and minimum allocation units (MAUs), for striped VSAM, can reside or be extended in cylinder-managed space. A compatible CA or MAU size is one that divides evenly into the multicylinder unit of value of cylinder-managed space.

VSAM CA sizes

The following CA sizes and MAUs are compatible because they divide evenly into the multicylinder unit of 21 cylinders (315 tracks):

1, 3, 5, 7, 9, 15 tracks

The system ensures that, for all new allocations on all volume types, a compatible CA or MAU is selected.

24.2.7 EAS non-eligible data sets

An EAS-ineligible data set may exist on an EAV, but it is not eligible to have extents (through Create or Extend) in the cylinder-managed space.

The following types of data sets are not supported and are exceptions to EAS eligibility:

- VTOC (continues to be restricted to within the first 64 K-1 tracks)
- VTOC index
- Page data sets
- VSAM data sets with imbed or keyrange attributes
- VSAM data sets with incompatible CA sizes
- HFS file systems
- XRC Control, master or cluster non-VSAM data sets
  - EXCP processing in XRC for the control, master or cluster non-VSAM data sets is not changed to support extended attribute DSCBs and 28-bit cylinder numbers. Therefore, attempts to access these data sets when allocated with extended attribute DSCBs (format 8 and 9 DSCBs) will be prevented.
  - State data sets are EAS eligible starting in z/OS V1R12
  - Journal data sets are EAS eligible starting in z/OS V1R11
- Certain system data sets such as SYS1.NUCLEUS, SYS1.PARMLIB

24.2.8 New format DSCBs for EAVs

Data set control blocks (DSCBs) are volume table of contents (VTOC) entries that describe data set attributes and allocated extent information. This extent information describes allocated space using beginning and ending track addresses. These are called *extent descriptors*. These extent descriptors may contain 28-bit cylinder numbers for their track addresses.

DSCBs also contain metadata, in the format-1 DSCB, that are the characteristics or attributes of the allocated data set. There is no more space available in the format-1 DSCB to add additional attributes.

Extended attribute DSCBs

There are new DSCB types that provide a method of protecting existing programs from seeing unexpected track addresses (28-bit cylinder numbers) and new format DSCBs, as follows:

Format-8 DSCB This DSCB is equivalent to a format-1 DSCB. It contains a chain pointer to a format-9 DSCB.

Format-9 DSCB This DSCB provides attribute data and a list of pointers to each possible format-3 DSCB. It contains a chain pointer to the possible next format-9 or format-3 DSCB. These attributes are maintained only for the first volume.

There is only one format-9 DSCB in z/OS V1R10.

Format-9 DSCB

The format-9 DSCB is new as of V1R10. It has all the information z/OS needs to record the attributes of a data set in the EAS of an EAV. The format-9 DSCB can point to one or more format-3 DSCBs, as depicted in Figure 24-8.

Thus, we see that format-8 DSCB exists strictly to highlight the fact that a format-9 DSCB with all the EAS information has been inserted between the format-1 (also known as format-8) DSCB and the format-3 DSCB chain.

Notes: The logical DSCB chain for a data set today is classically a format-1 and up to 10 possible format-3 DSCBs.

The logical DSCB chain for an EAS-eligible data set on an EAV is a format-8 and one or more format-9s, and up to 10 possible format-3 DSCBs.

In both cases the chain pointer in each DSCB points to the next, if one exists.

The format-9 DSCB is a place for additional attribute information. It contains direct pointers to each possible format-3 DSCB. With this new service in the system, (OBTAIN, CVAFDIR) can read the entire logical DSCB chain for a data set in one call. There are no more loops to read DSCBs until the chain pointer is zero (0).

Starting with z/OS V1R11, data set attributes are recorded in the format-9 DSCB. These fields indicate the job and step name and time since midnight that the data set was created. In future releases, additional format-9 DSCBs might be chained between the subtype 1 and any format-3 DSCBs.

The format 9 DSCB exists only for all EAS-eligible data sets. It contains the following EAV information:

- The format identifier is x'F9'.
- A subtype field.
- In the first EAV release, the subtype is 1.
- In future releases, additional subtypes might be added.
- Track addresses which point directly to up to 10 format-3 DSCBs.
- All the format-3 DSCBs can be read with one channel program.
- A 20-byte field, DS9ATRV1, that IBM is reserving for vendors. IBM will not specify or monitor its content.
- A "next DSCB" address points to a possible format-3 DSCB. The format-3 DSCBs continue to be chained.

Format-4 DSCB

The format-4 DSCB contains the following new information for EAVs:

- The number of cylinders on a volume.
- The existing 2-byte field DS4DSCYL in the format-4 DSCB contains the value of x'FFFE' (65 534). This identifies the volume as an EAV.
- DS4EAV is defined as a constant value of X'FFFE'.
- A new four-byte field DS4DCYL contains the number of cylinders on the volume.
- A new allocation unit for cylinders above 65,520.
- A new 2-byte field DS4LCYL contains a code value of x'0010' to indicate that the cylinder-managed space after the first 65,520 cylinders must be allocated in units that are larger than one cylinder.
- This value represents 65,520 cylinders divided by 4095. For a non-EAV, this will be zero (0).
- The new field DS4MCU (minimum allocation unit) contains the number of cylinders that each extent in the cylinder-managed area must be a multiple of. For an EAV, this is 21. For a non-EAV, this will be zero (0). It is valid only when the value in DS4DSCYL is DS4EAV.

This DSCB is identical to the format-1 DSCB with the following exceptions:

- The format identifier (DS1FMTID) is x'F8' instead of x'F1'. New symbols defined:
  - DS1IDC constant value of X'F1' in DS1FMTID.
  - DS8IDC constant value of X'F8' in DS1FMTID.
- Track addresses in the extent descriptors starting in DS1EXT1 use a new track address format (that is, they may contain 28-bit cylinder numbers).
- The "next DSCB" address (DS1PTRDS) always points to a format-9 DSCB (a new type of DSCB), instead of to a possible first format-3 DSCB.

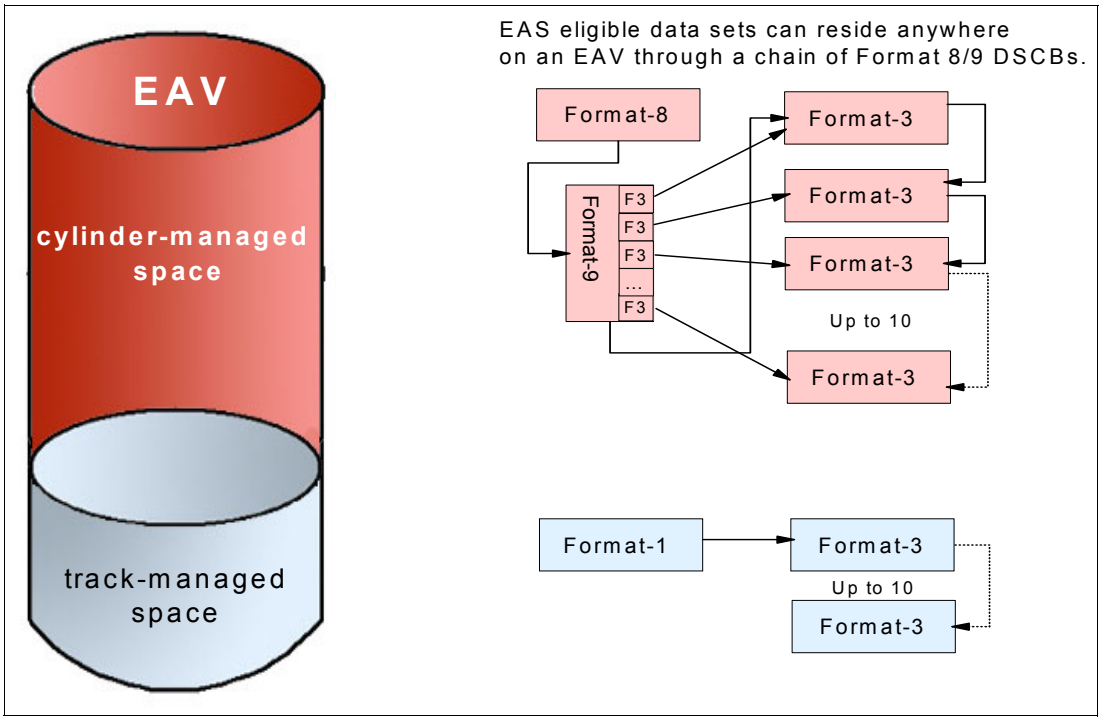


Figure 24-8 Access to EAS-eligible data sets on EAV

EADSCB=OK keyword

To access extended attribute DSCBs, the system requires the specification of a new permission keyword on system services that read DSCBs. By specifying the EADSCB=OK keyword, the invoking program is indicating to the system service that it understands extended attribute DSCBs and the 28-bit cylinder numbers that can be present in the data set's extent descriptors.

EADSCB=OK keyword specifies whether this program supports data sets with format-8 and format-9 DSCBs. Such data sets can appear on extended address volumes.

Attention: Code EADSCB=NOTOK when your program does not support data sets that have format-8 and format-9 DSCBs. The extent descriptors in DSCBs for a data set described with these formats may have track addresses that contain cylinder addresses 65,520 or larger.

EADSCB=OK is accepted for data sets described by all DSCB types, including format-1 DSCBs, regardless of the volume size where the data set resides. Your program can also run on an older level of the system that does not support this keyword.

In these cases, EADSCB=OK is ignored. EADSCB=OK sets byte 2 bit 4 in the OBTAIN parameter list to ON.

The EADSCB=OK keyword has been added to the following services:

- OBTAIN (CAMLST macro)
- CVAFDIR
- CVAFFILT
- CVAFDSM
- CVAFSEQ
- OPEN (DCBE macro) - opening VTOC or VSAM data set with EXCP access.

Attention: Not specifying the EADSCB=OK keyword causes these services to fail if issued to a data set that supports extended attribute DSCBs or a volume that supports cylinder-managed space (CVAFDSM and OPEN).

Code this keyword when an application supports EADSCB=OK. Specify it on all invocations of each service, regardless of whether the application runs on pre-z/OS V1R10 systems or accesses volumes that do not support extended attribute DSCBs. Macros on earlier releases do not recognize EADSCB=OK but can be assembled with EADSCB=OK on z/OS V1R10 and above systems and run on downlevel releases, where EADSCB=OK has no effect.

Specifying EADSCB=OK indicates the program understands 28-bit cylinder numbers and format-8 and format-9 DSCBs.

For information about VTOC usage and the VTOC DSCBs, see *z/OS DFSMSdfp Advanced Services*, SC26-7400.

24.2.9 EATTR data set attribute

EAS-eligible data sets are defined to be those that can be allocated in the extended addressing space and have extended attributes. This is sometimes referred to as *cylinder-managed space*.

For all EAS-eligible data sets a new data set attribute, EATTR, was added in z/OS V1R11 to allow a user to control whether a data set can have extended attribute DSCBs and thus control whether it can be allocated in EAS space on a EAV volume.

DFSMShsm checks the data set level attribute EATTR when performing non-SMS volume selection. The EATTR data set level attribute specifies whether a data set can have extended attributes (Format 8 and 9 DSCBs) and optionally reside in EAS space on an extended address volume (EAV). Valid values for the EATTR are NO and OPT.

EATTR(NO) NO indicates that the data set cannot have extended attributes or reside in EAS. This is the default for non-VSAM data sets. In z/OS V1R10 in the absence of EATTR, this is equivalent to what the system uses for non-VSAM.

EATTR(OPT) OPT indicates that the data set can have extended attributes and can optionally reside in EAS. This is the default for VSAM data sets. In z/OS V1R10 in the absence of EATTR, this is equivalent to what the system uses for VSAM.

Using the EATTR parameter

EATTR is specifiable for all data set types at the level of the following interfaces:

- JCL
- ALLOCATE
- AMS DEFINE CLUSTER
- Dynamic allocation
- SMS data class
- ISPF

The EATTR value is encoded to a value and written in the format 1 or 8 DSCB for all data set types and in the VVDS for VSAM clusters. The EATTR value is recorded for a data set type that is not supported as being EAS-eligible. It will have no effect until a future time when the system might begin supporting that data set type for EAS.

EATTR value is listed by IEHLIST ISPF, ISMF, LISTCAT, Catalog Search Interface (CSI), DCOLLECT.

Volume selection

Volume selection uses the EATTR values for SMS and HSM non-SMS volumes processing. EATTR is determined in the following order:

- In the JCL interface by merging EATTR from the JCL
- Coding the LIKE= parameter on the DD statement to refer to an existing extended format data set.
- Data class

Programs can read DSCBs by issuing OBTAIN, CVAFDIR, or CVAFFILT macros or by reading a VTOC. In a format 1 or 8 DSCB, the EATTR value is recorded in two bits at offset 61, as shown in Figure 24-9.

```
DS1FLAG1
.... ..00 - EATTR not specified.
.... ..01 - DS1EATTR_NOEATTR=NO.
.... ..10 - DS1EATTR_OPTEATTR=OPT.
```

Figure 24-9 Reading EATTR value

24.2.10 Other miscellaneous support

With z/OS V1R12, the following support is provided:

- Binder supports data sets in EAS.
- DFSMShsm supports ML1, ML2 and backup EAV.
- Catalog data sets can be allocated in EAS.
- DFSORT data sets can be allocated in EAS.
- JES3 spool, checkpoint and JCT data sets are supported in EAV.
- JES2 spool and checkpoint data sets are supported in EAV.
- Stand-alone dump and superzap have been updated to provide support for EAVs.

24.3 z/OS V1R12 enhancements in DFSMSdfp related to EAV

Though not related to EAV support, the following sections are a direct consequence of the large spaces opened up by EAV in which you can place huge data sets or many middle-size data sets. This enlarged environment triggers new requirements on some DFSMSdfp functionalities, as detailed here.

24.3.1 IGGCATxx parmlib member

Previously, the only way to customize the catalog environment was through the SYS1.NUCLEUS(SYSCATxx) and SYS1.PARMLIB(LOADxx). Only 1 line (80 characters) was available, which prevented any new parameters from being added. This did not allow changing any parameters after the system had been IPLed.

z/OS V1R13 allows for catalog to provide its own PARMLIB member, and any number of parameters can be added and changed with and without a system IPL. Systems programmers can now create their own catalog parmlib member or members to customize the catalog environment. The parameters can be changed by doing an IPL or a simple restart of the catalog address space (CAS).

The new IGGCATxx parmlib member allows you to define catalog system parameters and make permanent changes between IPLs. You must specify the IGGCATxx parmlib member or members in the CATALOG=xx parameter in IEASYSxx parmlib member. This specifies which IGGCATxx parmlib members to use during the current IPL.You can specify other catalog parameters in the LOADxx parmlib member and the SYSCATxx parmlib member of SYS1.NUCLEUS.

Note: The IGGCATxx parmlib member or members are optional. However, if specified, the parameters specified within take precedence over the parameters specified in the LOADxx and SYSCATxx members.

Displaying catalog system parameters

Use the **`D IPLINFO,CATALOG`** command to display the catalog system parameters currently in effect at a given time. The IGGCATxx parmlib member parameters are processed both during the IPL and when the catalog address space is restarted.

24.3.2 IEASYSxx parmlib member

You must specify the current IGGCATxx parmlib member or members in the CATALOG=xx parameter in IEASYSxx parmlib member. The xx suffix can be any 2 alphanumeric characters or national characters (@,#,$). The default is 00 (zeros). If the system finds no CATALOG parameter, it uses default IGGCAT00. If the system finds no IGGCAT00 default member, default values will be used.

If you specify multiple IGGCATxx parmlib members in the CATALOG=xx parameter, the system processes them in the order specified. When the same parameter appears in multiple members, the value specified in the last member processed becomes the value in effect.

If the system does not find a IGGCATxx parmlib member matching the CATALOG=xx specification, the system issues a message for each missing IGGCATxx member.

Figure 24-10 illustrates IEASYSxx parmlib member specification of IGGCATxx members.

```
CATALOG={aa }
{(aa,bb,...)}
   This parameter identifies the IGGCATxx members to use during the current
   IPL.
   The two alphanumeric characters, represented by aa (or bb, and so forth),
   are
   appended to IGGCAT to form the names of the IGGCATxx members.
   The IGGCATxx parmlib members specify catalog parameters for the initializing
   system.
   Value Range: Any two alphanumeric characters.
   Default Value: CATALOG=00
   Associated Parmlib Member: IGGCATxx
```

Figure 24-10 IEASYSxx parmlib member specification of IGGCATxx members

24.3.3 IGGCATxx new statements and parameters

The following parameters are supported to provide this new function:

VVDSSPACE VVDSSPACE(primary,secondary) – This specifies the number of tracks for primary and secondary allocations that the Catalog Address Space (CAS) is to use for an implicitly defined VVDS. The specified values are preserved across a CAS restart and are applied at IPL. The default is 10 tracks for both parameters.

TASKMAX TASKMAX(n) – This specifies the catalog address space user service task upper limit, which is the maximum number of catalog service tasks that can run at any given time. After this limit has been reached, further requests for CAS services are delayed until a task control block becomes available. The default is 180. The minimum is 24. The maximum is 360.

NOTIFYEXTEBT NOTIFYEXTENT(n) – This specifies the desired percentage threshold of the extents allocated for a catalog before the system issues message IEC361I to warn that the catalog is becoming full. The specified value is preserved across a CAS restart and is applied at IPL. Note that this value can be changed with a CAS restart. The default is 80%. A percentage value of zero indicates that the monitoring for extent usage is suppressed. If a catalog exceeds 90% utilization of the maximum extents, the system will issue message IEC361I even if the threshold has been set to zero (0).

IGGCATxx statement example

To use the new IGGCATxx parmlib member support, you must create one or more members and populate it with the desired settings such as shown in Figure 24-11.

```
VVDSSPACE(40,50)
TASKMAX(75)
NOTIFYEXTENT(19)
```

Figure 24-11 IGGCATxx examples of new statements

Note: Multiple declarations of any IGGCATxx parameters are allowed. The last valid value will be used for the parameter. For example, if the following are specified:

```
VVDSSPACE(10,10)
VVDSSPACE(14,14)
```

then VVDSSPACE(14,14) will be used as the final value for VVDSSPACE in this case.

Using the IGGCATxx parmlib member specifications

The IGGCATxx parmlib member parameters are processed both at IPL and when CAS is restarted. The IGGCATxx parmlib member or members are optional. However, if specified, the parameters specified within take precedence over the parameters specified in the LOADxx and SYSCATxx members.

The suffixes are to be specified in the new IEASYSxx parmlib member, CATALOG=(list of suffixes separated by commas). When multiple members are specified, the members are processed in the order specified. For example:

```
CATALOG=AA - (specifies one member. No parens needed if only one member)
CATALOG=(AA,BB,05) - (specifies multiple members. Values in member IGGCAT05
will override those in members IGGCATBB and IGGCATAA.)
```

- If CATALOG= is specified, and a particular IGGCATxx member within it is not found, it is skipped. If none of the members are found, the default PARMLIB member (IGGCAT00) is searched.
- If CATALOG= is not specified, the default member (IGGCAT00) is searched.
- If IGGCAT00 does not exist, then default values are used for the parameters.

24.3.4 MODIFY CATALOG command

The **MODIFY CATALOG** command has the following new parameters for z/OS V1R13 that include enable or disable parameters options.

The ENABLE(feature) enables a particular optional feature, where feature can be any one of the following.

DELFORCEWNG This parameter specifies that message IDC1999I is not to be issued if a DELETE UCAT RECOVERY command is attempted. DELRECOVWNG is disabled, by default.

EXTENDEDALIAS EXTENDEDALIAS enables the ability to create extension records for user catalog aliases on the current system. Only enable this feature when all systems in the sysplex are z/OS V1R13 or greater.

The DISABLE(feature) disables a particular optional feature, where feature can be any one of the following.

EXTENDEDALIAS This parameter disables the ability to create extension records for user catalog aliases on the current system. EXTENDEDALIAS is disabled, by default.

DELRECOVWNG DELRECOVWNG specifies that message IDC1999I be issued if a DELETE UCAT RECOVERY command is attempted.

24.3.5 Catalog address space considerations

When you IPL a system, the maximum number of catalogs that can be open in the catalog address space is set at 9999. The maximum number of CAS service tasks available for user requests is set to either 180 (the default) or 90% of the value optionally specified in the SYSCATxx parmlib member of SYS1.NUCLEUS. You can specify the number of catalogs and tasks as follows:

- Specify the service task lower limit in SYSCATxx parmlib member (SYS1.NUCLEUS), or in the LOADxx parmlib member.
- As new support with z/OS V1R13, when you specify the maximum number of concurrent user service tasks in the TASKMAX parameter of the IGGCATxx parmlib member. The value of TASKMAX in IGGCATxx is to d be no more than 90% of the maximum number of concurrent catalog requests specified in the SYSCATxx or LOADxx parmlib members.

You can also change these values temporarily using the CATMAX or TASKMAX parameters on the **MODIFY CATALOG** command. Changing these values can help you manage or limit the amount of storage used by CAS to perform catalog functions.

CATMAX Use this parameter to close all open catalogs and set a maximum to the number of catalogs that might be open in CAS. When the maximum is reached, the least recently used catalog is closed. This conserves storage.

> **Note:** When you use CATMAX to change the maximum number of catalogs that can be open in CAS and the new limit is lower than the previous limit, all opened catalogs are closed. This does not unallocate catalogs. Catalogs remain allocated to CAS, but in restart status. All the storage associated with the catalogs that were closed is freed.
>
> If a request for a closed catalog must be processed after the limit for open catalogs is reached, the least recently used catalog is closed and the required catalog is opened.
>
> Limiting the number of open catalogs affects catalog performance. However, if space is a primary consideration, you might need to set a maximum.

TASKMAX Use this parameter to specify an upper limit to the number of service tasks to process catalog requests. After the limit is reached, new requests must wait. Setting an upper limit reduces the storage used by CAS.

Note: When you use TASKMAX to change the maximum number of CAS service tasks, first determine the current number of CAS tasks. The REPORT parameter shows this value in the "# ATTACHED SERVICE TASKS" field.

When the upper limit for tasks is reached, any new requests that require catalog resources wait until another task is finished. No user jobs fail because the limit is reached, but if the limit is set too low, it might cause a significant performance degradation for jobs on the system.

Catalog management reserves 10% of the total number of tasks for its exclusive use, with the remaining tasks being available for both catalog management use and user requests. The reserved tasks are only allocated if all the other tasks are in use, and catalog management requires the use of a CAS service task. Thus, the highest number of service tasks listed for the catalog address space might be as high as 180 (if the default value is used) or 360 (if the largest possible value is specified in the SYSCATxx parmlib member).

24.3.6 The intersection of SYSCATxx, LOADxx, and IGGCATxx

There are several places to define catalog configuration values: SYSCATxx, LOADxx, and IGGCATxx parmlib members. Table 24-1 lists the attributes you can specify in each.

Table 24-1 Attributes that can be specified in parmlib members

Attributes of parmlib members	Parmlib members
► Master catalog type ► SYS% facility on or off ► Master catalog name ► Maximum number of concurrent catalog requests	SYSCATxx only
► Master catalog's volume ► Tape volume catalog high level qualifier ► AUTOADD indicator ► CAS service task lower limit (maximum number of CAS service tasks or requests)	SYSCATxx LOADxx, SYSCAT statement

SYSCATxx parmlib member

After you have defined your master catalog and have cataloged the necessary system data sets in it, you must identify the master catalog to the system to use it as a master catalog. The SYSCATxx member of SYS1.NUCLEUS contains the information for identifying the master catalog to MVS. If you use the LOADxx member to identify the master catalog, the system uses the master catalog specified in the member, and the master catalog message is not issued. You can have multiple SYSCATxx members. The default name is SYSCATLG. Each copy of the SYSCATxx member can identify separate master catalogs. Figure 24-12 shows the format of the SYSCATxx parmlib member of SYS1.NUCLEUS (the member can contain only one record).

```
         10        20        30        40        50        60        70
----+----+----+----+----+----+----+----+----+----+----+----+----+----+--
MXACAT123CCATALOG.MXAICFM.VMXACAT                    SMS       999
```

Figure 24-12 Format of a SYSCATxx parmlib member

LOADxx parmlib member

The LOADxx parmlib member provides sysplex-related definitions early in system initialization so other parmlib members can use those definitions.

24.3.7 Installation considerations

Although there is currently no overlap between IGGCATxx, SYSCATxx and LOADxx parameters, the system gives IGGCATxx the highest priority. The system applies the parameters in SYSCATxx, LOADxx, and IGGCATxx in the following order:

1. Parmlib member IGGCATxx, if specified, takes the highest priority, followed by...
2. Parmlib member LOADxx followed by...
3. SYSCATxx member of SYS1.NUCLEUS followed by...
4. System-defined defaults

The IGGCATxx parameters are processed both at IPL and when CAS is restarted through an **F CATALOG,RESTART** command. The IGGCATxx member or members are optional. Howerver, if specified, the parameters specified within take precedence over the parameters specified in the LOADxx and SYSCATxx members.

New messages

New messages are provided in z/OS V1R13 along with the new parameters. When an internal error is detected during a service that was called by catalog during parsing of the following catalog PARMLIB member, then message IEC385W is issued:

```
IEC385W IGGCATxx BYPASSED DUE TO xxxxxxxx ERROR. RETURN CODE IS rc
```

When a syntax error is detected on a particular line during the parsing of IGGCATxx, then message IEC386W is issued:

```
IEC386W INVALID KEYWORD DETECTED IN aaaaaaaa AT LINE: text
```

When an internal error i detected while releasing storage obtained during IGGCATxx processing, message IEC387W is issued:

```
IEC387W ERROR RELEASING IGGCATxx STORAGE. ERROR CODE IS rc
```

24.3.8 Alias number constraint relief

To use a catalog, the system must be able to determine which data sets should be defined in that catalog. The simplest way to accomplish this is to define aliases for the catalog. Before defining an alias, though, carefully consider the effect the new alias will have on old data sets. A poorly chosen alias can make some data sets inaccessible.

Catalog aliases are defined in the master catalog, which contains an entry for the user catalog. The number of aliases a catalog can have is limited by the maximum record size for the master catalog. If the master catalog is defined with the default record sizes, there is a practical maximum of 3500 aliases per catalog, assuming the aliases are only for high-level qualifiers. If you use multilevel aliases, fewer aliases per catalog can be defined.

Catalog aliases are defined in the master catalog, as shown in Figure 24-13, which contains an entry for the user catalog.

By default, the number of aliases a catalog can have was limited by the maximum record size for the master catalog. If the master catalog is defined with the default record sizes, there is a practical maximum of 3000 aliases per catalog assuming the aliases are only for high-level qualifiers. If you use multilevel aliases, fewer aliases per catalog can be defined.

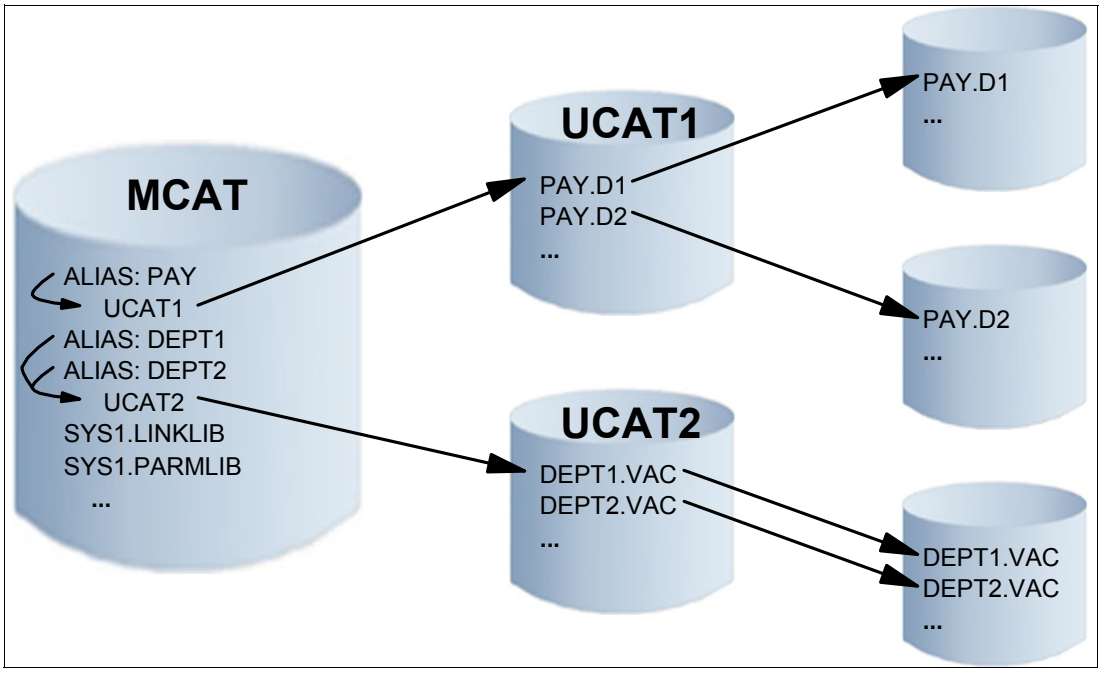


Figure 24-13 Using aliases

The default of the maximum record size is 32768 bytes (32 KB). Catalog records are limited to 32768 byte (32 KB) record size. User catalog connectors are catalog records.

New support in z/OS V1R13

z/OS V1R13 introduces a new extension record type V for user catalog connectors. The maximum number of user catalog connector extension records is 255. This allows users to define more aliases per user catalog. From now on, theoretically the limit is over 500000 aliases per user catalog.

After a user catalog connector extension record is created, it will exist for the rest of the life of the user catalog connector. For example, when all the association entries in the user catalog connector extension record are deleted, the user catalog connector extension record will still exist with an empty association cell.

Also, there will be no associated order for the association entries of the user catalog connector to be returned.

Installation considerations

z/OS V1R13 comes with a switch to help in migrating to this enhancement. The new function is disabled during the install. You can exploit this increased number of aliases by specifying the EXTENDEDALIAS enable feature on the **F CATALOG** command. This function can be turned ON or OFF via the following catalog modify commands:

```
F CATALOG,ENABLE(EXTENDEDALIAS)
or
F CATALOG,DISABLE(EXTENDEDALIAS)
```

Note: By default, EXTENDEDALIAS is disabled. Only enable EXTENDEDALIAS to increase the number of possible catalog aliases when all systems in the sysplex are z/OS V1R13 or greater.

In addition, the function can also be activated through the catalog IGGCATxx parmlib member by specifying the following command:

```
EXTENDEDALIAS(YES/NO)
```

Coexistence support

Toleration and coexistence APARs are provided for prior releases:

- For catalog: OA33517
- For IDCAMS: OA34486

These APARs allow access to aliases in user catalog extension records:

- Deletion of aliases in user catalog extension records
- Import of z/OS V1R13 portable data sets with the current limit alias number

However, this coexistence PTF will not allow deletion of user catalog connectors if user connectors have extension records by:

- Export disconnect
- Delete UserCatalog Recovery
- Delete UserCatalog Force

It will not allow users to define more aliases if a user catalog connector has extension records.

24.3.9 Catalog VVDS expansion

This implementation describes what in the VSAM volume data set (VVDS) is changing in z/OS V1R13, and both the direct and indirect effects of VVDS expansion. A VSAM volume data set (VVDS) is one which describes the characteristics of VSAM and system-managed data sets that reside on a given DASD volume; part of a catalog.

z/OS V1R12 AMS was changed to support EATTR as a keyword on a DEFINE CLUSTER where the object is a VVDS data set. Currently the catalog VVDS is limited in size to X'FFFF' (65535 decimal) VSAM control intervals (CI). Each 4 K CI can hold one to several VSAM volume records (VVR) or non-VSAM volume records (NVR) for data sets that reside on the VVDS of the volume. The number of data sets that may reside on a volume is therefore limited thenby the size of the VVDS.

With z/OS V1R13, the size of the VVDS increases from X'FFFF' to X'FFFFF' (1048575) CIs, which is a sixteen-fold increase, allowing for larger VVDSs supporting more data sets per volume.

Basic catalog structure (BCS) in z/OS V1R12

Starting with z/OS V1R12, you can explicitly create a VVDS that can optionally be placed in Extended Addressable Storage (EAS), if available. Allocation amounts whichthat are intended for EAS space may be rounded up to a multicylinder unit (MCU). The potential rounding up of EATTR(OPT) allocations has the effect of lowering the VVDS maximum size to CI values that are consistent with the MCU boundary.

Note: Track allocations may be rounded up to the next cylinder value. For example, a define request for 5455 tracks will likely allocate 5460 tracks, which is evenly divisible by 15 and is the next cylinder boundary.

Catalog structure

Every catalog consists of one BCS and one or more VVDSs. A BCS does not “own” a VVDS. Instead, more than one BCS can have entries for a single VVDS. Every VVDS that is connected to a BCS has an entry in the BCS.

A catalog consists of two separate kinds of data sets:

- A basic catalog structure (BCS)

  The BCS can be considered the catalog, whereas the VVDS can be considered an extension of the volume table of contents (VTOC). The basic catalog structure is a VSAM key-sequenced data set. It uses the data set name of entries to store and retrieve data set information. For VSAM data sets, the BCS contains volume, security, ownership, and association information. For non-VSAM data sets, the BCS contains volume, ownership, and association information.

- A VSAM volume data set (VVDS)

  The VSAM volume data set is a VSAM entry-sequenced data set. A VVDS resides on every volume that contains a catalog or an SMS-managed data set that is cataloged. It contains the data set characteristics, extent information, and the volume-related information of the VSAM data sets cataloged in the BCS. If you are using the Storage Management Subsystem (SMS), the VVDS also contains data set characteristics and volume-related information for the non-VSAM, SMS-managed data sets on the volume.

  A VVDS can be defined either explicitly or implicitly, as explained here:

  – Explicitly, using DEFINE CLUSTER or

  – Implicitly, when the first catalog or SMS-managed data set is defined on the volume

  A VVDS is defined with the name SYS1.VVDS.Vvolser, where volser is the volume serial number of the volume containing the VVDS. SYS1.VVDS.Vvolser does not have to be cataloged in the master catalog.

  An explicitly defined VVDS is not related to any BCS until a data set or catalog object is defined on the volume. As data sets are allocated on the VVDS volume, each BCS with catalog or SMS-managed data sets residing on that volume is related to the VVDS.

Relationship of the BCS and VVDS

Figure 24-14 illustrates how data set entries are contained in both the VVDS and the BCS. Information about a data set is also contained in the VTOC of the volume on which the data set resides, even if the data set is cataloged.

To successfully perform all possible operations on a cataloged data set using the catalog, all three elements, the VVDS, BCS, and VTOC, must be synchronized. That is, any equivalent information contained in the BCS and VVDS entries for the data set, and the VTOC DSCB for the data set, must be the same. This is normally done automatically.

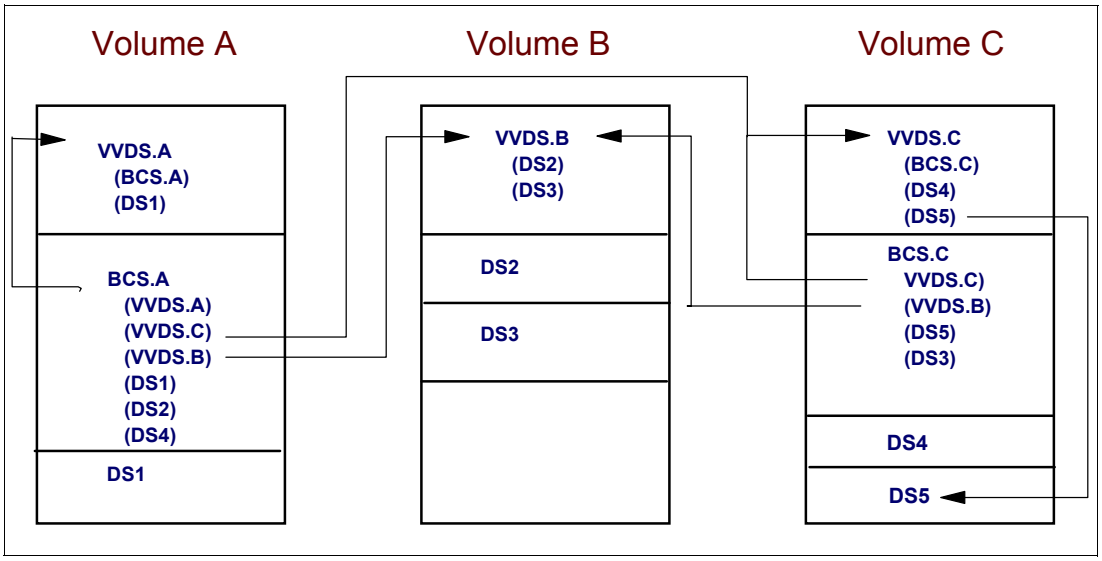


Figure 24-14 Relationship of the BCS and the VVDS

Extended format BCS

When estimating space for an extended format BCS, the BCS is limited to 4 GB unless you define it as an extended format BCS, which means it can use extended addressability. Using extended addressability, the size limit for a BCS is determined by the control interval size multiplied by 4 GB.

For example, a control interval size of 4 KB yields a maximum data set size of 16 TB. A control interval size of 32 KB yields a maximum data set size of 128 TB. No increase in processing time is expected for extended format data sets that grow beyond 4 GB. To use extended addressability, the BCS must be SMS managed and defined as extended format.

You can specify extended format for a BCS using SMS data class DSNTYPE=EXT parameter and subparameters R (meaning required) or P (meaning preferred) on the ISMF DATA CLASS DEFINE/ALTER panel. Use R to ensure the BCS is extended. The Extended Addressability value must be set to Y (Yes).

The only extended format option available for a BCS is extended addressable. This means that BCSs cannot be compressed or striped.

24.3.10 VVDS changes in z/OS V1R13

Each VVDS is a portion of the data set catalog. A VVDS exists on each SMS volume that contains data sets, and on each non-SMS volume that contains a VSAM data set. The VVDS contains a small number of internal control records (VVCR, VVCM, VVCN, and so on), VSAM Volume Records (VVRs), and Non-VSAM Volume Records (NVRs). The VVDS resembles a VSAM entry-sequenced data set that has 4 KB control interval (CI) size, which holds the record type or types described here.

Note: The *control interval* (CI) is a concept that is unique to VSAM. A CI is formed by one or several physical records (usually just one). It is the fundamental building block of every VSAM file. A CI is a contiguous area of direct access storage that VSAM uses to store data records and control information that describes the records. A CI is the unit of information that VSAM transfers between the storage device and main storage during one I/O operation. Whenever a logical record is requested by an application program, the entire CI containing the logical record is read into a VSAM I/O buffer in virtual storage. The desired logical record is then transferred from the VSAM buffer to a user-defined buffer or work area (if in move mode).

Size increase for the VVDS

The VVDS is currently limited to roughly 65,000 CIs, or 'FFFF' in hexadecimal. This new support provided with z/OS V1R13 will increase the maximum number of CIs to approximately 1,000,000 CIs, or 'FFFFF' in hexadecimal. This enhancement in z/OS V1R13 provides a greater number of CIs in the VVDS. That is, this larger VVDS can hold more VVRs and NVRs, thus allowing a greater number of data sets on SMS volumes, and allowing a greater number of VSAM data sets on non-SMS volumes. With the trend of increasingly larger volumes (especially opened up with EAVs), this enhancement addresses the need for VVDSs to support larger numbers of data sets per volume and their VVR and NVR records.

Estimating maximum number of data sets per volume

The VVDS can hold a maximum of 1048575 control intervals. This limits defining primary and secondary allocation sizes to 87375 tracks or 5825 cylinders or fewer. This limit is also in effect when extending the VVDS. Thus, if extending the VVDS by the secondary allocation amount causes it to exceed the CI limit, the extend will fail.

If you know the average number of VVRs per CI in the VVDS, you can estimate the maximum number of data sets that the VVDS can support on the volume (IDCAMS PRINT of the VVDS can assist in coming to an estimate of average VVRs per CI).

To estimate the maximum number of data sets per volume, you must consider the following:

- VVDS control records like VVCR, VVCMs, and other records fully occupy a VVDS CI, and typically make up less than 0.2% of the VVDS's CIs.
- Data set VVRs and NVRs vary in size by type and within type categories.
  - Catalog data set VVRs are very large and can take up to an entire CI.
  - VSAM data sets vary greatly in VVR size. VSAM data sets with indexes have two VVRs. Each VVR tends to get bigger if SMS, RLS or long data set name, and grows with the number of extents in the data set; an estimate would be 6 to 12 VVRs per CI.
  - NVRs are the smallest record, and the NVR record seems to grow if the data set name is long; an estimate would be 12 to 20 NVRs per CI in the VVDS.

As an example, assume that an SMS volume contains catalog, VSAM, and non-VSAM data sets. Also assume that VVDS CI, on average, holds 8 records (NVR/VVR). In this case the maximum data sets per volume by release, which is about a 16-fold improvement with z/OS V1R13, is as follows:

- Systems prior to z/OS V1R13

  8 ds/ci x 65k ci = ~520,000 data sets

- With the z/OS V1R13 enhancement

  8 ds/ci x 1 million = ~ 8,000,000 data sets

Example of a VVDS definition

The default size for implicit VVDS creation remains TRACKS(10 10). z/OS V1R13 VVDSs can be defined with either the primary or secondary allocation values as high as 5825 cylinders or 87375 tracks as the new limit. The limit for previous releases is 364 cylinders or 5460 tracks. If either the primary or secondary amount exceeds the release limit, message IDC3009I with return code 50 reason code 32 (50-32) is issued.

An IDC3009I 50-32 message will also be issued if the VVDS attempts to extend beyond the release maximum CI limit.

For example, in Figure 24-15 the DEFINE CLUSTER will fail in releases prior to z/OS V1R13 with an IDC3009I 50-32 message because this only succeeds in z/OS V1R13.

```
DEFINE CLUSTER(NAME(SYS1.VVDS.VEXMP01) -
       NONINDEXED                      -
       CYLINDERS(365  2)               -
       VOLUMES(EXMP01))
```

Figure 24-15 IDCAMS JCL to define a VVDS

Coexistence considerations

Coexistence of z/OS V1R13 VVDSs with lower z/OS releases has no impact when the z/OS V1R13 VVDS is smaller than the current VVDS CI limit.

Note: z/OS V1R13 VVDSs that are greater than 364 cylinders or 5460 tracks will require coexistence APAR OA34940.

The coexistence PTF allows lower levels to add, delete, read, search, and modify all VVRs/NVRs in the larger VVDS. These release levels will not be able to create or extend an existing or V1R13 VVDS beyond the previous limit.

The coexistence PTF is needed on lower than V1R13 *only* when the volume being shared has a z/OS V1R13 VVDS greater than the previous release limit.

Summary

The VVDS limit is expanding with z/OS V1R13 from approximately 65,000 CIs per VVDS to approximately 1,000,000 CIs per VVDS. The increase in the total number of CIs that a VVDS can contain affects the maximum number of VSAM data sets a volume can hold, and effects the maximum number of non-VSAM and VSAM data sets an SMS volume can manage. For this reason, it is more suitable to EAV environments.

24.3.11 EAS-eligible data sets

Table 24-2 summarizes DFSMS support for data sets eligible to reside in the EAS of an EAV.

The first column indicates that in z/OS V1R10, DFSMS enabled VSAM data sets to be EAS eligible. z/OS Communications Server FTP does not support VSAM data sets. In z/OS V1R11, DFSMS enabled physical sequential extended format data sets to be EAS eligible. In z/OS V1R12, most types of data sets can be EAS eligible, and they do not need to be SMS managed.

The types of data sets most interesting to FTP are shown in bold in the table, namely physical sequential data sets, PDS data sets, and libraries (PDSE data sets).

Keep in mind that DFSMS support for a type of data set does not automatically grant other elements of z/OS or application programs the ability to access these types of data sets when they are EAS eligible. Other elements of z/OS and application programs can require upgrading to support all types of EAS-eligible data sets.

Table 24-2 Data sets that can be EAS eligible

z/OS V1R10	z/OS V1R11	z/OS V1R12
VSAM data sets	Physical sequential extended format data sets	Physical sequential data sets: basic and large format PDS and library data sets Catalogs, basic data sets Data sets can be SMS-managed or not

24.3.12 Review of FTP existing support (z/OS V1R11)

In z/OS V1R11, Communications Server gave access to non-VSAM EAS-eligible FTP configuration data sets. In z/OS V1R11, that meant extended format physical sequential data sets only.

Figure 24-16 shows configuration data sets that z/OS Communications Server FTP supports as EAS-eligible data sets. The data sets on the left side are z/OS FTP client configuration data sets. The data sets on the right side are z/OS FTP server configuration data sets. FTP.DATA is common to the FTP client and server.

The arrow shows that when FTP.DATA is an EAS-eligible data set, the TSO **HOMETEST** command cannot successfully be issued.

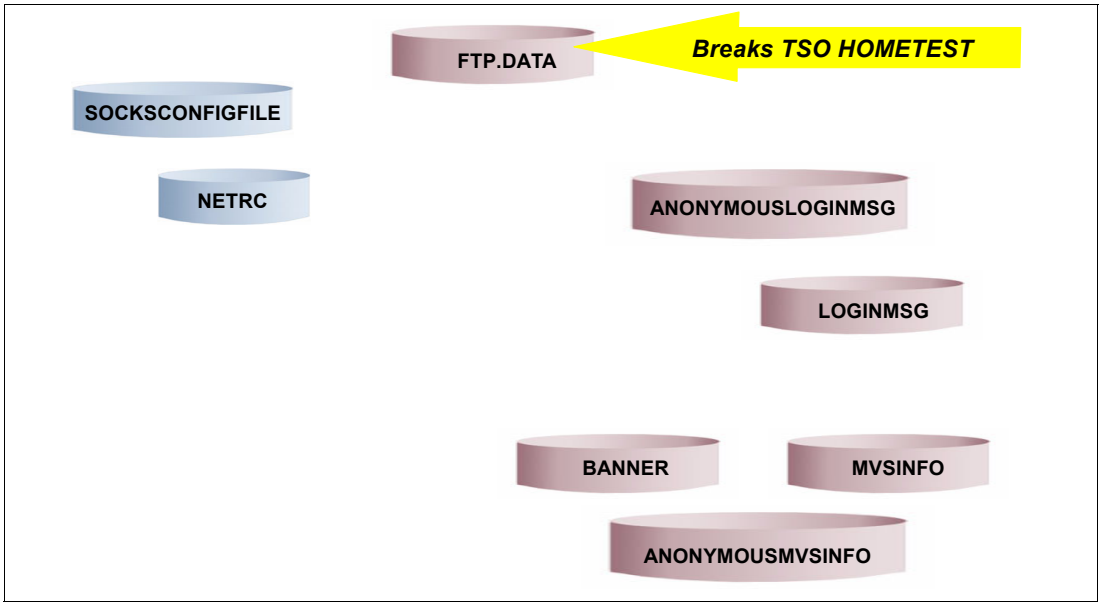


Figure 24-16 z/OS V1R11 FTP configuration data sets in EAS

File transfer (FTP) and EAS-eligible data sets

z/OS V1R11 Communications Server FTP supports file transfer to and from EAS-eligible data sets. In z/OS V1R11, this means file transfer to and from physical sequential extended format data sets.

When the destination must be an EAS-eligible data set, you must allocate the data set *before* the file transfer to ensure it is allocated as EAS eligible. That is because z/OS V1R11 Communications Server FTP does not provide any configuration option to explicitly allocate a data set as EAS eligible. An alternative is to set up an SMS data class that supplies default attributes of EATTR=OPT when the target volume is the EAV. z/OS Communications Server FTP allows you to specify a data class with the DATACLASS configuration option.

24.4 z/OS V1R13 enhancements to EAV support

z/OS V1R13 provides additional disk addressable storage, giving users the option to use larger extended address volumes (EAVs). Support will also be provided as a Small Programming Enhancement (SPE) for z/OS V1R12. Table 24-3 shows the new EAV sizes available with z/OS V1R13.

Note: Systems prior to z/OS V1R13 (or z/OS V1R12 with the SPE) cannot use the larger EAV size.

Table 24-3 Compairing EAV sizes for z/OS V1R13 and previous releases

EAV size for z/OS V1R13 (or z/OS V1R12 with the SPE)	EAV size for previous releases (z/OS V1R10, 11, and 12)
1 TB	223 GB
1,182,006 cylinders	262,668 cylinders
17,730,090 tracks	3,940,020 tracks

Important: Large format sequential data sets (DSNTYPE=LARGE) are limited to no more than 16,777,215 tracks by the access method. Prior to the 1 TB volume, the size of the volume kept large format sequential data sets within this limit. With this support, these data sets will be limited in size by the system so as to not exceed 16,777,215 tracks per volume. This is more than 4 times larger than the first EAV maximum size of 223 GB.

24.4.1 Enhanced FTP support for EAV with z/OS V1R13

With z/OS V1R13, the z/OS FTP client and server support the allocation of data sets that are eligible for extended addressing space (EAS). You can transfer data to and from the following types of EAS-eligible data sets:

- Sequential data sets (basic, extended, and large formats)
- Partitioned data sets and extended partitioned data sets

You can configure FTP to allocate new MVS data sets as eligible for EAS.

Restriction: You cannot allocate z/OS UNIX files as EAS-eligible files.

Dependency: DS8000 Licensed Internal Code 4.0 or higher is required to support extended address volumes (EAVs).

Using the enhanced FTP support for extended address volumes

With z/OS V1R13, if you want to use the enhanced FTP support for extended address volumes, perform the appropriate tasks:

- Configure FTP to allocate new MVS data sets with extended attributes.
- Determine whether new data sets will be allocated with extended attributes.

24.4.2 FTP support for large-format data sets

z/OS V1R13 Communications Server for z/OS FTP supports transfer to and from physical sequential large format data sets. You can configure FTP to allocate new physical sequential data sets as physical sequential basic format data sets or as physical sequential large-format data sets.

Restriction: The **LIST** command reply cannot always display accurate size information when it reports information about large data sets. When FTP cannot provide accurate size information, affected fields display a plus sign (+) to indicate that the actual size is larger than the **LIST** command reply can represent.

z/OS V1R13 Communications Server for z/OS FTP also supports transfer to and from z/OS UNIX files that are as large as or larger than two gigabytes.

Using the FTP support for large-format data sets

To use FTP transfer to and from physical sequential large format data, perform the following appropriate tasks to enable the support for the transfer of z/OS UNIX files that are as large as or larger than two gigabytes. That support is automatically enabled.

- Configure the FTP client or server to allocate new physical sequential data sets as physical sequential large-format data sets:
  - **LOCSIte** subcommand; see “LOCSite subcommand” on page 571
  - **SIte** subcommand; see “SIte subcommand” on page 572
- Determine whether the FTP client allocates new physical sequential data sets as physical sequential large-format data sets or as physical sequential basic format data sets:
  - **LOCSTat** subcommand; see “LOCSTat subcommand” on page 573
- Determine whether the FTP server allocates new physical sequential data sets as physical sequential large-format data sets or as physical sequential basic format data sets:
  - **STAtus** subcommand; see“STAtus subcommand” on page 574

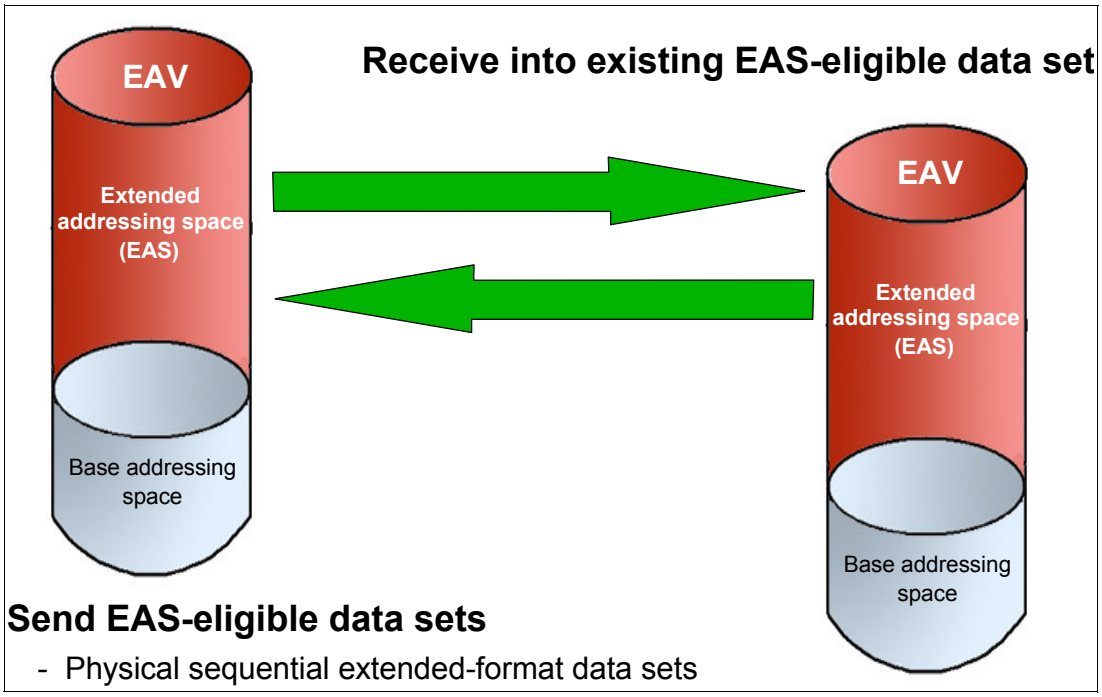


Figure 24-17 Send or receive of EAS eligible data sets

To finish the recall of existing support for EAVs, in z/OS V1R11 Communications Server allowed renaming, deleting, and listing EAS-eligible data sets. That is, z/OS V1R11 Communications Server allowed renaming, deleting, and listing physical sequential extended format data sets. Furthermore, Communications Server for z/OS FTP was enhanced to report space statistics for EAVs. Space statistics are part of the report generated by the **QDISK** parameter of the **LOCSIte** subcommand, and the server **SITE** command.

24.4.3 FTP z/OS V1R13

With z/OS V1R13, Communications Server FTP provides support for the types of EAS-eligible data sets that DFSMS enabled in z/OS V1R12, as follows:

- Transfer to and from EAS-eligible data sets
  - PDS and library allocations
  - Physical sequential basic and large extended formats
  - GDG data sets
  - SMS managed or not

z/OS V1R13 Communications Server FTP also supports block mode restart of interrupted file transfer to or from an EAS-eligible data set. This enhancement provides a configuration option to explicitly allocate a new data set as EAS eligible. This applies to PDS allocations, library allocations, and physical sequential allocations. The new configuration option is named EATTR after the JCL parameter and the dynalloc() text unit used to allocate EAS-eligible data sets.

Note: This support adds FTP support for the types of EAS-eligible data sets that DFSMS enabled in z/OS V1R12.

FTP.DATA data set for FTP client parameter

In FTP.DATA, there is a new field for new data set allocation in the EATTR parameter. It specifies whether new data sets can have extended attributes and whether the data sets can reside in the EAS, where the new EATTR option for FTP.DATA in z/OS V1R13 is, as follows:

```
EATTR(NO | OPT | SYSTEM)
```

Where:

NO The data set cannot have extended attributes and cannot reside in the EAS, and its VTOC entry cannot have extended attributes.

OPT Extended attributes are optional. The data set can have extended attributes if it resides on an extended address volume. The new data set can reside in the EAS, and its VTOC entry can have extended attributes if the volume supports them.

SYSTEM Use the EATTR value assigned to the SMS data class. If no data class is configured or no EATTR value is assigned, the data set is allocated with the system EATTR value. This is the default.

This EATTR configuration option specifies whether a new data set allocated by FTP can have extended attributes and reside in the EAS of an EAV. A data set must possess extended attributes to be EAS eligible.

Note: The SYSTEM value is provided for migration purposes; it means FTP allocates the data set the way it did in prior releases.

As in prior releases, if you configure an SMS data class that specifies the EATTR value, the new data set is allocated with the EATTR value from the data class. If you have not configured a data class that specifies the EATTR value, then FTP will allow the data set to be allocated with the system value. The default value for the EATTR configuration option is SYSTEM.

Specifying EATTR(OPT) does not guarantee that new data sets will possess extended attributes. The new data set will possess extended attributes only if the target volume supports extended attributes. In z/OS V1R12 and V1R13, a volume with extended attributes is always an extended address volume.

Data sets that do not possess extended attributes are allowed to reside in the base addressing space of an EAV. A data set with extended attributes can reside anywhere on an EAV. However, a data set must possess extended attributes to reside in the EAS of an EAV.

24.4.4 LOCSite subcommand

Use the **LOCSIte** subcommand to specify information that is used by the local host to provide services specific to that host system. The **LOCSIte** subcommand can be used after starting the FTP client to override the EATTR value coded in the client's FTP.DATA.

The **LOCSite** subcommand uses values **SYSTEM**, **NO**, and **OPT**, which have the same meaning here as for the FTP.DATA statement. The DSNTYPE parameters are shown here:

```
LOCSite EATTR=(NO | OPT | SYSTEM) DSNTYPE=(SYSTEM | BASIC | LARGE)
```

Where the new options are:

DSNTYPE This specifies the data set name types for new physical sequential data sets, as follows:

SYSTEM Physical sequential data sets are allocated with the SMS data class value. If no data class is defined, or if the DSNTYPE attribute is not defined, new physical sequential data sets will be allocated with the system default value.

BASIC This allocates physical sequential data sets as physical sequential basic format data sets.

LARGE This allocates physical sequential data sets as physical sequential large format data sets.

EATTR This specifies whether newly allocated data sets can have extended attributes, and whether new data sets can reside in the EAS of an EAV.

SYSTEM The data set uses the SMS data class EATTR value. If no SMS data class is defined, or if the data class contains no EATTR specification, the data set is allocated with the system default.

NO The data set cannot reside in the EAS, and its VTOC entry cannot contain extended attributes.

OPT The data set can reside in the EAS, and its VTOC entry can have extended attributes if the volume supports them.

Note: You can specify more than one parameter with the **LOCSIte** subcommand. Delimit each parameter with a blank space. Issue the **HELP LOCSIte** subcommand to display a list of configuration options available on the local host.

LOCSite subcommand example

In Figure 24-18, as an example, you enter the **LOCSIte** subcommand to configure the client to allocate new physical sequential data sets as large format. Then you create a new physical sequential data set in the FTP client's file system by retrieving a data set from the server with the **Get** subcommand. Because `user1.new.large` did not already exist in the client's file system, FTP allocated a new physical set.

```
locsite dsntype=large
Command:
get 'user1.old.large' 'user1.new.large'
>>> EPSV
229 Entering Extended Passive Mode (|||5000|)
>>> RETR 'user1.old.large'
125 Sending data set USER1.OLD.LARGE
226 Transfer completed successfully.
41 bytes transferred in 0.030 seconds. Transfer rate 1.37 Kbytes/sec.
Command:
```

Figure 24-18 LOCSite subcommand example

24.4.5 SIte subcommand

The EATTR option for the **SIte** subcommand has the same syntax as the one for **LOCSite**, and the values have the same semantics, as described in "LOCSite subcommand" on page 571.

Note: You can use the **SIte** subcommand after logging into the FTP server to override the EATTR value coded in the server's FTP.DATA.

24.4.6 LOCSTat subcommand

The **LOCStat** subcommand displays the state of the local (FTP client) configuration. This includes the **EATTR** setting. When the **EATTR** parameter is entered, status displays only the **EATTR** configured value as shown in Figure 24-19.

```
EZA1460I Command:
LOCSTat EATTR
 EZA2916I local site variable EATTR is set to SYSTEM
 EZA1460I Command:
```

Figure 24-19 LOCSTat EATTR subcommand example

LOCSTat EATTR display

Figure 24-20 shows an example of the **LOCSTat** subcommand using the **EATTR** parameter. In this example, **EATTR** is set to OPT, indicating that MVS data sets are allocated with extended attributes if the volume the data set resides on supports extended attributes.

```
Command:
locstat EATTR
local site variable EATTR is set to OPT
Command:
```

Figure 24-20 LOCSTat EATTR display

LOCSTat display

When the **LOCSTat** subcommand is entered, as shown in Figure 24-21, all client configuration status is displayed.

```
Command: locstat
Trace: FALSE, Send Port: TRUE
Send Site with Put command: TRUE
Connected to:9.42.104.38, Port: FTP control (21), logged in
local site variable DSWAITTIME is set to 0
VCOUNT is 59
Prompting: ON, Globbing: ON
ASA control characters transferred as ASA control characters
New data sets catalogued if a store operation terminates abnormally
Single quotes will override the current working directory
UMASK value is 027
Data connections for the client are not firewall friendly.
local site variable EPSV4 is set to TRUE
local site variable SECUREIMPLICITZOS is set to TRUE
local site variable PASSIVEIGNOREADDR is set to FALSE
local site variable TLSRFCLEVEL is set to RFC4217
local site variable READVB is set to LE
local site variable EXTDBSCHINESE is set to TRUE
local site variable LISTSUBdir is set to TRUE
local site variable PROGRESS is set to 10
local site variable SEQNUMSUPPORT is set to FALSE
local site variable UNIXFILETYPE is set to FILE
local site variable FIFOIOTIME is set to 20
local site variable FIFOOPENTIME is set to 60
local site variable EATTR is set to SYSTEM
local site variable DSNTYPE is set to BASIC
Authentication mechanism: None Local Port: 1118
Tape write is not allowed to use BSAM I/O
Using /etc/ftp.data for local site configuration parameters.
Command:
```

Figure 24-21 Partial output of the LOCSTat subcommand example

24.4.7 STAtus subcommand

Use the **STAtus** subcommand to retrieve current configuration information from the FTP server. This information includes the current settings of the configuration variables, which can be initialized in the FTP.DATA data set or changed using various FTP subcommands.

Figure 24-22 shows an example of using this command. Here, the **STAtus** subcommand is entered from the z/OS FTP client. The client sent a **STAT** command to the server to obtain the information, and displayed the **STAT** command reply. (Due to space constraints here, various lines have been omitted from the reply.) The server STAT reply currently contains over sixty lines of information. The EATTR value is highlighted in contrasting font.

In this case, the server EATTR configuration option is set to SYSTEM. New data sets that the FTP server allocates will use the value from the SMS data class if an SMS data class that specifies the EATTR value has been configured for FTP. Otherwise, new data sets that the FTP server allocates will use the system default value.

```
Command:
stat
>>> STAT
211-Server FTP talking to host ::1, port 1026
<lines omitted>
211-Server site variable EATTR is set to SYSTEM
211-Server site variable DSNTYPE is set to SYSTEM
<lines omitted>
Command:
```

Figure 24-22 STATus subcommand

When you enter the **STAtus** subcommand with no parameter, all server configuration status is displayed. This includes the EATTR configuration option setting. You can also enter the **EATTR** parameter to display only the EATTR configured value.

The server site can be queried as shown in Figure 24-23.

```
EZA1460I Command:
STAT (EATTR
 EZA1701I >>> XSTA (EATTR
 211-Server site variable EATTR is set to SYSTEM
 211 *** end of status ***
EZA1460I Command:
```

Figure 24-23 STATus subcommand

24.4.8 Creating a directory on the remote host

Use the **MKdir** subcommand to create a PDS, PDSE, or z/OS UNIX directory on the remote host, as shown in Figure 24-24.

Figure 24-24 Format of MKdir subcommand

Where:

directory This specifies the name of the remote directory to be created.

local_directory This specifies the name of a local directory that is to be a model for the remote directory.

Attention: The local_directory and directory must be MVS directories. The server must be a Communication Server for z/OS FTP server.

Use this parameter *only* when you have called the FTP client interactively. You can use this parameter to specify a local MVS directory that has characteristics you want for the remote directory. For example, the **(like** parameter when allocating the target of a load module transfer ensures that the source and target directories are compatible for load module transfer.

However, there are restrictions regarding this usage, as explained here.

Restrictions: Only the 3390 device architecture is supported. Unpredictable results will occur if the source or the target directory is on devices that use another architecture.

FTP can only approximate the following characteristics of local_directory: SPACETYPE, DIRECTORY, PRIMARY, and SECONDARY.

Thus, the corresponding characteristics of directory might not match the original allocation of local_directory. For complete control over these characteristics, avoid using the **(like** parameter.

Results of the command

The requirements for using the command are listed here:

- FTP must read local_directory to determine its characteristics. Do not use the **(like** parameter if this is not acceptable.
- FTP sends a **SIte** command to the server for you to configure the FTP server to allocate a directory with the same characteristics as local_directory.

  Be aware that a **SIte** command changes the server configuration for the rest of the session. Consequently, the server configuration changes for the rest of the session when you specify the **(like** parameter. Do not use the **(like** parameter if you do not want the server configuration to change.
- If local_directory is a migrated data set, FTP checks the local AUTORECALL setting to determine whether to recall the data set or fail the request.
  - If AUTORECALL is true, FTP tries to recall the migrated data set.
  - If AUTORECALL is false, FTP fails the MKdir request.

  You can change the local AUTORECALL setting with the **LOCSIte** subcommand. Choosing AUTORECALL might result in a long delay when the FTP client waits for the data set to become available.
- If local_directory is a data set that is not mounted, FTP checks the local AUTOMOUNT setting to determine whether to mount the data set or fail the request.
  - If AUTOMOUNT is true, FTP tries to mount the data set.
  - If AUTOMOUNT is false, FTP fails the MKdir request.

  You can change the local AUTOMOUNT setting with the **LOCSIte** subcommand. Choosing AUTOMOUNT might result in a long delay when the client waits for the data set to become available.

24.4.9 Allocating an EAS-eligible PDS

The example shown in Figure 24-25 demonstrates using FTP to allocate an EAS-eligible partitioned data set (PDS). The z/OS Communications Server FTP server allocates a PDS or library when the client sends it an **MKD** command specifying a name suitable for an MVS PDS or library.

In the example, the **SIte** subcommand is used to set the server volume to an EAV volume, to set PDSTYPE to PDS, and to set EATTR to OPT. EATTR OPT allows the system to allocate an EAS-eligible data set if the volume the data set resides on is an EAV. An EAV volume was configured with the **volume** parameter.

Then a **MKdir** command was issued to create a new partitioned data set in the server file system. The target data set specified a fully qualified MVS data set name. The server created a PDS called ‘user2.eas.pds’, as shown.

```
Command:
site eattr=opt volume=EAVVOL pdstype=pds
>>> SITE eattr=opt volume=EAVVOL pdstype=pds
200 SITE command was accepted
Command:
mkdir ‘user2.eas.pds’ >>> MKD 'user2.eas.pds'
257 "'USER2.EAS.PDS'" created.
Command:
```

Figure 24-25 MKdir allocates a directory on a server host

Note: If you set PDSTYPE to PDSE, the server would have created a partitioned data set extended (PDSE) or library called ‘user2.eas.pds’.

The new PDS will have extended attributes if the volume that the new data set resides on supports extended attributes. In z/OS V1R13, that is equivalent to saying the new PDS is EAS eligible if the specified volume, EAVVOL, is an EAV volume. If the new PDS is EAS eligible, it can reside in the BAS or in the EAS of the EAV.

If you had specified a z/OS UNIX file as the **MKdir** subcommand target, the server would have created a z/OS UNIX directory. In that case, the server would ignore the EATTR, VOLUME, and PDSTYPE configured values because these attributes have no meaning to the z/OS UNIX file system.

24.4.10 Allocating an EAS-eligible PDSE

The example shown in Figure 24-26 demonstrates using FTP to allocate an EAS-eligible partitioned data set extended (PDSE) or library. The z/OS Communications Server FTP client allocates a PDS or PDSE when you issue the **LMkdir** subcommand specifying the name of an MVS PDS.

The **LOCSIte** subcommand is issued to set the client volume to an EAV, to set PDSTYPE to PDSE, and to set EATTR to OPT. EATTR OPT allows the system to allocate an EAS-eligible data set if the volume the data set resides on is an EAV. An EAV was configured with the **volume** parameter.

The **LMkdir** subcommand was issued to create a new library in the client file system. The target data set specified a name that might be a partially qualified MVS data set name, or a z/OS UNIX relative pathname. Because the local working directory is an MVS high level qualifier, the client resolves the name to an MVS data set name. As shown, the client created a PDSE or library called ‘user3.eas.pdse’.

```
Command:
lcd 'user3'
Local directory name set to USER3.
Command:
locsite eattr=opt volume=EAVVOL pdstype=pdse
Command:
lmkdir eas.pdse
USER3.EAS.PDSE created.
Command:
```

Figure 24-26 LMKdir allocates a directory on a local server

The new library will have extended attributes if the volume the new data set resides on supports extended attributes. In z/OS V1R13, that is equivalent to saying the new library is EAS eligible if EAVVOL is an EAV. If the new PDS is EAS eligible, it can reside in the BAS or the EAS of the EAV.

The allocation will succeed if the volume does not support extended attributes, but the new library will not be EAS eligible. If a z/OS UNIX file had been specified as the **LMkdir** subcommand target, then the client will create a z/OS UNIX directory. In that case, the client ignores the EATTR, VOLUME, and PDSTYPE configured values because these attributes have no meaning to the z/OS UNIX file system.

24.4.11 Using the HELP subcommand

Use the z/OS Communications Server FTP client **HElp** subcommand to obtain information about the server **SIte** command. In the example shown in Figure 24-27, the **Help** subcommand was entered with the argument server site. The argument server directs the client to forward the remaining arguments to the server as arguments of the **HELP** command. Thus, you see the client sent the command **HELP SITE** to the server.

The z/OS FTP server replies with many lines of help for the **SITE** command. Due to space constraints, most lines have been omitted from the reply shown here. Only the help added for the **DSNTYPE** parameter is displayed in the example.

You can enter the subcommand **HElp locsite** to obtain nearly identical help for the **LOCSIte** subcommand **DSNTYPE** parameter.

```
help server site
 EZA1701I >>> HELP SITE
 EZA1582I The foreign server has this help:
 214-The SITE command sub parameters are:
.................
214-EATTR=ÝNO|OPT|SYSTEM¨ specifies whether new data sets can have extended
 214-                attributes and whether the data sets can reside in the
 214-                EAS.  NO indicates that the data set cannot reside in the
 214-                EAS and its VTOC entry cannot have extended attributes.
 214-                OPT indicates that the data set can reside in the EAS and
 214-                its VTOC entry can have extended attributes if the volume
 214-                supports them.  SYSTEM indicates the data set will use the
 214-                SMS data class EATTR value, or system default EATTR value
 214-                if no SMS data class is defined or if the data class
 214-                contains no EATTR specification.
.........................................
```

Figure 24-27 HELP for EATTR

24.4.12 Diagnosis and messages

When trying to allocate an EAS-eligible data set on an EAV, remember to specify SETSMS USEEAV(YES) when you configure z/OS. You can code USEEAV(YES) in the IGDSMSxx parmlib member, or you can set it with the operator command **SETSMS USEEAV(YES)**. Check the console log for messages if you have trouble accessing EAS-eligible data sets. Issue the MVS operator command **SETSMS USEEAV(YES)** if the console log indicates you have not configured USEEAV(YES).

If this does not rectify your problem, obtain FTP traces as documented in *IP Diagnosis Guide*, which explain how to enable and capture the FTP trace.

The extended trace option DUMP 21 provides details of the parameters passed to dynamic allocation services when FTP attempts to create a new data set.

24.4.13 FTP client installation considerations

FTP clients other than the z/OS Communications Server V1R13 FTP client will not drive these server commands in the same way. However, you can configure the FTP server to allocate EAS-eligible data sets from any FTP client that supports the **QUOTE** subcommand or its equivalent. The **QUOTE** subcommand forwards its arguments to the server without processing them locally in any way.

Figure 24-28 shows how to enter the **QUOTE** subcommand arguments to configure the FTP server, and obtain configuration information from the server.

```
QUOTE STAT
QUOTE XSTA (EATTR
QUOTE SITE EATTR=OPT
QUOTE SITE EATTR=NO
QUOTE SITE EATTR=SYSTEM
QUOTE HELP SITE
```

Figure 24-28 QUOTE subcommand

Other considerations

FTP.DATA can be allocated as an EAS-eligible data set, but TSO HOMETEST does not support FTP.DATA when it is EAS eligible.

Although z/OS Communications Server FTP now supports EAV volumes, your other applications might not support EAV volumes. Consult DFSMS documentation and vendor documentation to determine and evaluate the capabilities of your other applications regarding this support.

25

Base Control Program internal interface

This chapter discusses the functions of the Base Control Program internal interface (BCPii).

IBM provides support within z/OS that allows authorized applications to query, change, and perform operational procedures against the installed System z hardware base through a set of application program interfaces. These applications can access the System z hardware that the application is running on and extend their reach to other System z processors within the attached process control (Hardware Management Console) network.

Using the Base Control Program internal interface (BCPii), an authorized z/OS application can perform the following actions:

- Obtain the System z topology of the current interconnected Central Processor Complexes (CPCs) and the images, capacity records, and activation profiles on a particular CPC
- Query various CPC, image (LPAR), capacity record, and activation profile information
- Set various configuration values related to CPC, image, and activation profiles
- Issue commands against both the CPC and image (LPAR) to perform minor or significant hardware- and software-related functions
- Listen for various hardware and software events that might take place on various CPCs and images throughout the HMC-connected network

Communication with the Support Element (SE) and Hardware Management Console (HMC) using BCPii is performed completely within the base operating system. It therefore does not require communication on an IP network (intranet) for connectivity, and this provides for the complete isolation of System z hardware communication from other network traffic within the intranet or Internet.

Additionally, starting with z/OS V1R12, the Common Information Model (CIM) function, which previously used SNMP, exploits BCPii instead.

25.1 Overview of BCPii

BCPii uses a low-level operating system connection to establish communication between an authorized application running on a z/OS image (LPAR) and the Support Element (SE) associated with the Central Processor Complex (CPC) that contains this z/OS image. You must configure the SE to permit these BCPii communications if BCPii services are required to be available by your installation.

The HWIBCPII address space, which supports the issuing of BCPii APIs from a z/OS image, runs on any level of hardware that supports the level of the z/OS operating system in which BCPii is included; see Figure 25-1. The address space provides a new API that allows authorized programs to communicate with the HMC.

BCPii is an authorized z/OS application that provides the following functionality:

- Monitoring status or capacity changes
- Obtaining configuration data related to the CPC or image
- Re-IPLing or resetting images

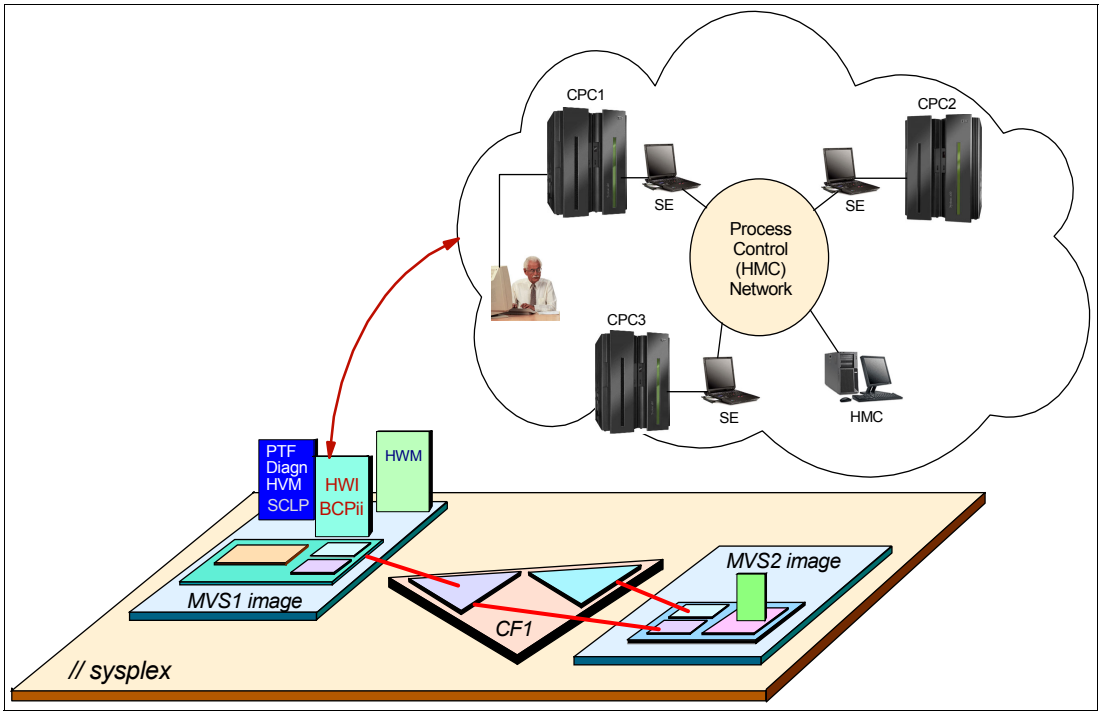


Figure 25-1 BCP internal interface (BCPii)

Current IBM BCPii exploiters

The Capacity Provisioning Manager (CPM) provides the capability of adding or removing temporary capacity based on goals defined in the WLM policy.

Sysplex Failure Manager (SFM) is used to expedite partitioning when a system fails. It can be used to determine whether a system has been reset. SFM can detect more quickly that an image has failed, compared to relying on a current timeout interval. This can minimize “sympathy sickness” by more aggressively removing a system from the sysplex. SFM minimizes the “Down error message” on the console when a missing XCF heartbeat occurs.

BCPii allows installations to query and possibly manipulate System z resources in a more automated, programmatic approach, in contrast to using a hands-on HMC environment that uses an intranet/Internet/internal network connected to the HMC.

25.1.1 Using BCPii

As an interface to authorized z/OS applications, through the process control (HMC) network, BCPii allows you to implement:

- Monitor status or capacity changes.
- Obtain configuration data related to CPC (not necessarily the zSeries system) or image.
- Re-IPL an image.
- Allow authorized z/OS applications to have HMC-like control over systems in the process control network: see Figure 25-2.
- Complete communication isolation of existing networks (intranet/Internet) from the process control network, because communication to the SE is completely within the base z/OS.
- A new z/OS address space; the BCPii address space is the bridge between a z/OS application and the SE.
- A set of authorized APIs; the BCPii address space is mandatory for any BCPii API request. The system attempts to start the HWIBCPii address space during IPL.
- Communication transport between a z/OS application and the local SE. BCPii also provides communication transport between other SEs connected to other CPCs routed by the HMC. BCPii uses a low-level operating system connection to establish communication between an authorized application running on a z/OS image (LPAR) and the SE associated with the CPC that contains this z/OS image. You must configure the SE to permit these BCPii communications if BCPii services are required to be available by your installation.

  z/OS programing interface is provided to perform SE and HMC functions in a secure way that does not need to connect the HMC network to the company intranet or Internet.

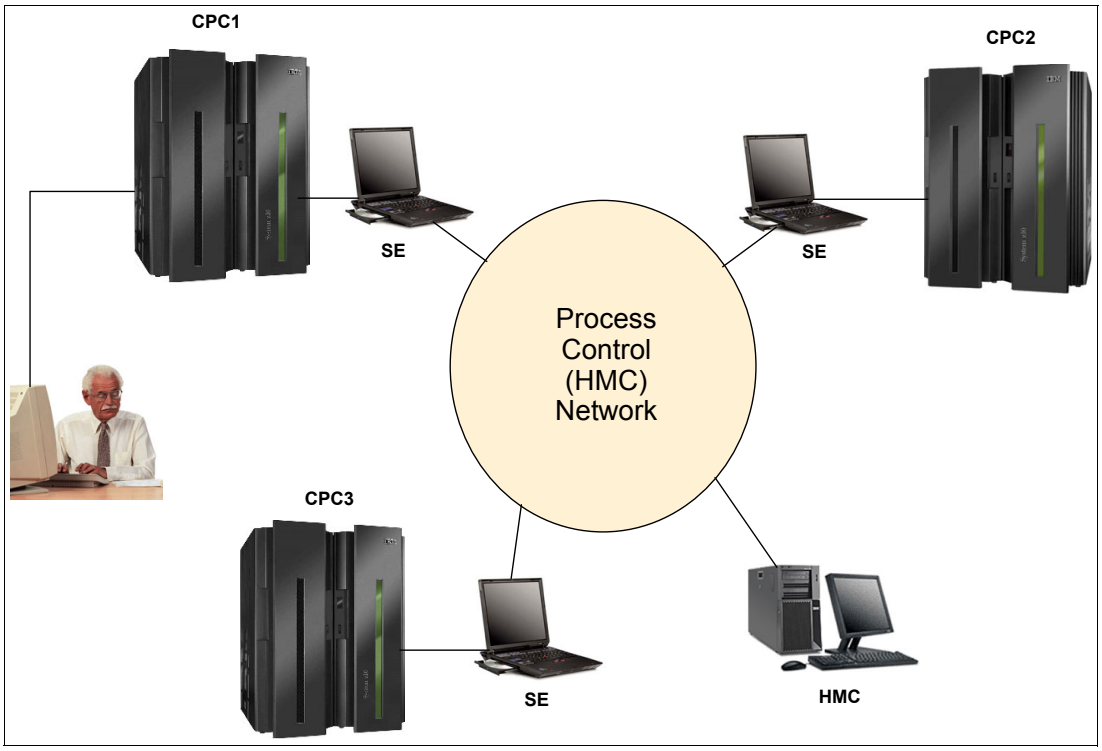


Figure 25-2 HMC network

25.1.2 BCPii address space

This section discusses the functioning of the BCPii address space.

Starting the BCPii address space

The BCPii address space, shown in Figure 25-3, starts automatically at IPL. It requires no initial setup before becoming active. The BCPii address space normally does not need to be started or shut down. If the configuration is correct, no further action is required. The address space remains active and ready to handle BCPii requests.

The address space can be stopped if necessary and be restarted by using a HWISTART procedure.

The BCPii address space can perform the following tasks:

- Manage all application connections.
- Build and receive all internal communication requests to the SE.
- Provide an infrastructure for storage required by callers and by the transport communicating with the SE.
- Provide diagnostic capabilities to help with BCPii problem determination.
- Provide security authentication of requests.

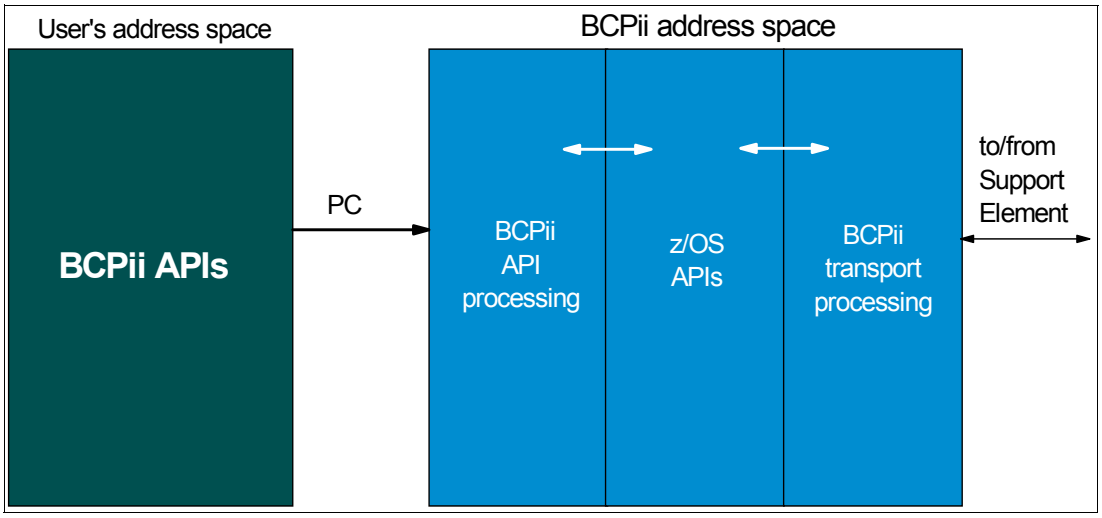


Figure 25-3 BCPii address space

BCPii address space not starting at IPL

If the HWIBCPii address space is not active after an IPL, look for HWI* messages in the system log. Generally, these messages pinpoint the reason for the failure of BCPii to become active.

In most cases, the address space did not start for one of two main reasons:

- The SE controlling the CPC that contains the image of z/OS on which BCPii is being started has the improper configuration.

  In this case, make sure all the steps have been followed in “Configuring the local Support Element to support BCPii” on page 593.

- The community name of this local CPC is either not defined in the security product, or it contains an incorrect value. This is accompanied by message HWI014I. See “Community name defined in the security product for each CPC” on page 602 for details about how to correct this problem.

After these problems have been corrected, restart the BCPii address space by issuing the following command:

```
S HWISTART
```

IBM ships the HWISTART procedure in SYS1.IBM.PROCLIB.

Ending the HWIBCPii address space

The application of certain kinds of code maintenance or other unusual circumstances might require that you stop the BCPii address space. To stop the BCPii address space, issue the **STOP** command for the BCPii address space:

```
P HWIBCPII
```

If it does not stop, you can use the commands **CANCEL** and then **FORCE**.

BCPii issues an ENF 68 broadcast to notify interested ENF listeners that BCPii services are no longer available.

Restarting the HWIBCPII address space

After the BCPii address space has ended, it can be restarted. A procedure supplied by IBM in SYS1.PROCLIB allows the BCPii address space to be restarted. Issue the **S HWISTART** command to restart the HWIBCPII address space. When message HWI001I appears, confirming that BCPii is active, then BCPii requests continue.

BCPii issues an ENF 68 broadcast when the address space has completely initialized to notify interested ENF listeners that BCPii services are now available.

25.1.3 BCPii diagnostics

Starting in z/OS V1R12, the following diagnostics tools are provided

- New API return codes
- CTRACE
  - BCPii cuts CTRACE records using the SYSBCPII CTRACE component
  - Default CTRACE CTIHWI00 parmlib member is shipped
  - Two CTRACE options are available:
    - Min
    - All
  - Dump occurs whenever CTRACE is turned off
- Symptom records
  - Limited first failure data capture for select problems
- Support Element tracing

25.2 New in z/OS V1R13

With z/OS V1R13, the z/OS programmatic BCPii interface supports more SE controls. In fact, it now supports almost the complete set of System z API interfaces, which increases z/OS programmatic control capability of the System z enterprise.

Now BCPii can be used to work with a new, user-defined image group such as “z/OS plex 1”; as shown in Figure 25-4. z/OS plex 1 is a group of images composed of LPARs LP1, LP2, LP5, and LP8. BCPii allows:

- Listing the image names in the group.
- Connecting to and disconnecting from the group.
- Querying values associated with the group.
- Issuing commands against all members in the group.

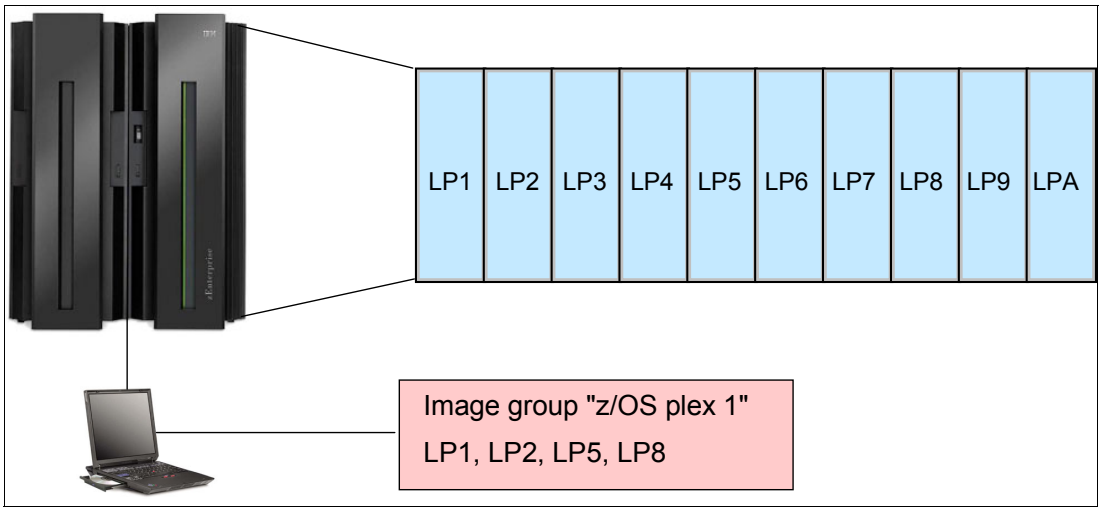


Figure 25-4 User-defined image group “z/OS plex 1”

25.2.1 BCPii services

BCPii services are available in any address space, from a program which is program-authorized, and SAF-authorized even in problem-state and storage-key 8. The program may be written in C and assembler programming languages. z/OS UNIX callers can receive event notifications through z/OS UNIX-only services utilizing Common Event Adapter (CEA).

The following functions can be performed using BCPii APIs:

- Obtaining the System z topology of the current interconnected CPCs, images (LPARs) and capacity records
- Querying various CPC, image (LPAR), and capacity record information
- Issuing commands against both the CPC and the image to perform hardware and software-related functions
- Listening for various hardware and software events that can take place on CPC and images

BCPii service types

BCPii offers various types of services, as listed:

HWICMD Issue a BCPii HMC console command to perform a command against an HMC-managed object that is associated with central processor complexes (CPCs), CPC images (LPARs) capacity records, or (new in z/OS V1R13) image groups.

This includes CPC commands such as:

- Activate, deactivate an entire CPC
- CBU request (activate or undo)
- On/off capacity on demand request (activate or undo)

Image commands, including:

- Sysreset, load, start, stop, add, or remove temporary capacity, issue operating system command. And (new in z/OS V1R13), for image

commands targeted to an image group, one image event is returned for each image in the user-defined image group.

HWICONN Establish a BCPii logical connection between the application and a central processor complex (CPC), a CPC image (LPAR), a capacity record, or other types of activation profiles (available in V1R11 with APAR OA29638), or (new in z/OS V1R13) user-defined image groups on a particular CPC.

This facilitates subsequent services to perform operations related to that CPC, image, capacity record, or activation profile.

Note: A connection remains active until a Disconnect service call (HWIDISC) is invoked; a loss of connectivity to the associated CPC has been detected by BCPii; or the address space of the caller is terminated.

When the address space of the caller is terminated, the connection is disconnected implicitly by BCPii.

HWIDISC Release a BCPii logical connection between the application and the identified CPC, image, capacity record, or activation profile. If the connect token represents a CPC, then any subordinate image, capacity record, or activation profile connection associated with the same CPC connection is also released.

HWIEVENT Register for BCPii events for the following purposes:

- Register an application and its connection to be notified for one or more hardware or software events occurring on the connected CPC or image, such as command completions, status changes, capacity changes, disabled waits, or BCPii status changes.
- Delete the registration for notification of one or more previously registered events.

HWILIST Retrieve HMC and BCPii configuration-related information.

- List CPCs to list the CPCs interconnected with the local CPC.
- List images to list the images (LPARs) contained on an individual CPC or (new in z/OS V1R13), an image group.
- List capacity records to list the capacity records contained on an individual CPC.
- List events to list the events already registered on a particular BCPii connection.

Depending on what information is requested, the data returned by this service can be used on subsequent BCPii service calls to take the following actions:

- Connect to a central processor complex (CPC), image (LPAR), capacity record (CAPREC), reset activation profile, image activation profile, or load activation profile using the HWICONN API.
- Register for the proper events (HWIEVENT) using the HWIEVENT API.
- Connect to the local CPC or Image.
- List Local CPC, List Local Image (available in V1R11) to obtain the name of the CPC name or image (LPAR) name that the BCPii application is currently running on.

- List Reset Activation Profiles, List Image A.P., and List Load A.P. (available in V1R11 with APAR OA29638) to list the currently defined activation profiles contained on an individual CPC.
- (New in z/OS V1R13) List User-defined Image Group Names - list the currently defined image group names contained on an individual CPC.

HWIQUERY BCPii retrieval of SE/HMC-managed objects managed by the SE or HMC related with central processor complexes (CPCs), CPC images (LPARs), capacity records, or other types of activation profiles (available in V1R11 with APAR OA29638), or (new in z/OS V1R13), user-defined image group on a particular CPC.

New in z/OS V1R13: New CPC attributes are introduced:

- HWI_VERSION: Version of the Support Element console application associated with this CPC.
- HWI_EC_MCL_INFO: Engineer code and microcode levels installed on this CPC.
- HWI_LIST_IP_ADDRESSES: IP addresses associated with this CPC
- HWI_AUTO_SWITCH_ENABLED: Flag indicating whether auto switch between the primary and the alternate support elements is enabled on this CPC

New image attribute:

- HWI_GROUP_PROFILE_CAPACITY: Workload unit capacity for the group profile associated with an image.

HWISET BCPii change or set SE/HMC-managed objects data for Hardware Management Console (HMC)-managed objects associated with Central Processor Complexes (CPCs), CPC images (LPARs), or activation profiles (available in V1R11 with APAR OA29638) and for a C application running in the z/OS UNIX System Services environment, with the principle that anything which can be queried, can be set.

New in z/OS V1R13: A new image attribute is introduced:

- HWI_GROUP_PROFILE_CAPACITY: Workload unit capacity for the group profile associated with an image.

25.2.2 BCPii callable services

You can use BCPii services to connect an authorized z/OS application to System z configuration resources (such as CPC, image, capacity record, or activation profile data), and allow that application to potentially modify these resources.

To use BCPii services, issue calls from high level language programs. Each service requires a set of parameters coded in a specific order on the CALL statement.

The BCPii callable services are as follows:

- HWICMD - Issue a BCPii Hardware Management Command
- HWICONN - Establish a BCPii Connection
- HWIDISC - Release a BCPii Connection
- HWIEVENT - Register for BCPii Events

- HWILIST - Retrieve HMC and BCPii Configuration-Related Information
- HWIQUERY -- BCPii retrieval of SE/HMC-managed objects data
- HWISET - BCPii set SE/HMC-managed objects data
- HWIBeginEventDelivery - Begin delivery of BCPii event notifications
- HWIEndEventDelivery - End delivery of BCPii event notifications
- HWIManageEvents - Manage the list of BCPii events
- HWIGetEvent - Retrieve outstanding BCPii event notifications

BCPii event notifications

The event notification services are listed here:

HWIBeginEventDelivery Begin delivery of BCPii event notifications to allow a C application running in the z/OS UNIX System Services environment to begin delivery of event notifications. This service must be issued before the HWIManageEvents service.

HWIEndEventDelivery End delivery of BCPii event notifications to allow a C application running in the z/OS UNIX System Services environment to end delivery of event notifications. This service unregisters the registration made by the HWIBeginEventDelivery service.

HWIManageEvents Manage the list of BCPii events to allow a C application running in the z/OS UNIX System Services environment to manage the list of events for which the application is to be notified. The HWIBeginEventDelivery service must have been called before the HWIManageEvents service being called, because the appropriate delivery token returned from the HWIBeginEventDelivery service is required as input.

HWIGetEvent Retrieve outstanding BCPii event notifications to allow a C application running in the z/OS UNIX System Services environment to retrieve outstanding BCPii event notifications.

New in z/OS V1R13: Note the following points regarding BCPii APIs enhancements for image group support:

- Only z10 and higher hardware is supported.
- With the operating system, Activate with activation profile and System reset with IPLToken commands cannot be targeted to an image group.
- If the image group contains the local image the application is running on, only “non-suicide” commands will be permitted.
- Security is strictly enforced.
- An application must have at least READ access to the CPC where the image groups reside to obtain a list of image group names, to list the images residing in an image group, or to query attributes associated with the image group.
- An application must have at least CONTROL access to each of the images contained in the image group for all commands targeted to an image group. Access validation is performed at the time of the HWICMD invocation.

Security authorization

Proper authority is to be previously established in RACF, to allow callers to access those services or objects.

To request BCPii services for programs from any address space, the environment must be as follows:

- Program-authorized
- SAF-authorized
- C and Assembler programming languages
- z/OS UNIX System Services callers can receive event notifications through z/OS UNIX System Services-only services using the Common Event Adapter (CEA).

Important: Granting the proper authority to a program in an address space to request BCPii services raises the possibility that this program might be able to shut off the entire CPC, even though the given program might be authorized within only one minor test LPAR of the given CPC.

As with z/OS V1R13 and current LIC levels, there is no known record, after a potential incident like this, that you can use to determine which program shut off the CPC.

A tentative solution to this situation is proposed in 25.6.3, “Potential benefits of a new command set” on page 609.

25.2.3 BCPii internal interfaces

IBM provides support within z/OS that allows authorized applications to query, change, and perform operational procedures against the installed System z hardware base through a set of application program interfaces. These applications can access the System z hardware that the application is running on and extend their reach to other System z processors within the attached process control (Hardware Management Console) network.

Using BCPii, an authorized z/OS application can perform the following actions:

- Obtain the System z topology of the current interconnected Central Processor Complexes (CPCs) and the images, capacity records and activation profiles on a particular CPC.
- Query various CPC, image (LPAR), capacity record, and activation profile information.
- Set various configuration values related to CPC, image, and activation profiles.
- Issue commands against both the CPC and image (LPAR) to perform minor or even significant hardware- and software-related functions.
- Listen for various hardware and software events that might take place on various CPCs and images throughout the HMC-connected network.

Communication to the Support Element (SE) / Hardware Management Console (HMC) using BCPii is done completely within the base operating system and therefore does not require communication on an IP network (intranet) for connectivity, providing complete isolation of your System z hardware communication from any other network traffic within the intranet/internet.

Information for query examples

The kinds of information that can be queried through BCPii are:

- CPC information

- General information, such as name, serial, machine type, ID, operating system information and networking information
- Status information
- Operating status and other status values
- Capacity information such as various CBU info, capacity on demand information, processor configuration, including IFA, IFL, ICF, and IIP
- Image information
- Capacity information such as defined capacity and processor weights
- Capacity record information, such as name, activation and expiration dates, activation days, record status, and the entire capacity record

New BCPii services with z/OS V1R12

New services or attributes of services provided with z/OS V1R12 include:

- STP_CONFIG (Query)
- HWI_NUMPGPP (Query)
- HWI_NUMPSAP (Query)
- HWI_NUMPAAP (Query)
- HWI_NUMPIFLP (Query)
- HWI_NUMPICFP (Query)
- HWI_NUMPIIPP (Query)
- HWI_BASIC_CPU_AUTH_COUNT_CNTL (Query and Set)
- HWI_PROBSTATE_CPU_AUTH_COUNT_CNTL (Query and Set)
- HWI_CRYPTOACTIVITY_CPU_AUTH_COUNT_CNTL (Query and Set)
- HWI_EXTENDED_CPU_AUTH_COUNT_CNTL (Query and Set)
- HWI_COPROCESSOR_CPU_AUTH_COUNT_CNTL (Query and Set)
- HWI_BASIC_CPU_SAMPLING_AUTH_CNTL (Query and Set)
- Hwi_Aprof_Store_Status (Query and Set)
- Hwi_Aprof_Loadtype (Query and Set)

Static power saving mode

The IBM zEnterprise 196 (z196) server has a mechanism to vary frequency and voltage, originally developed to help avoid interruptions due to cooling failures.

BCPii in z/OS V1R12 exploits this functionality by providing support in its APIs that allows users to query or register to receive notification of changes.

25.3 Installing BCPii

BCPii is installed by following these steps:

- Configuring the local Support Element to support BCPii
- Authorizing an application to use BCPii
- Configuring the BCPii address space
- Setting up the event notification mechanism for z/OS UNIX callers (if required).

25.3.1 Configuring the local Support Element to support BCPii

BCPii uses a low-level operating system connection to establish communication between an authorized application running on a z/OS image (LPAR) and the SE associated with the CPC that contains this z/OS image. You must configure the SE to permit these BCPii communications if BCPii services are required to be available by your installation.

Note: Configure the local SE to be unique for both z/OS BCPii and TSA BCPii. Define it only one time.

You must enable cross-partition authority on the SE to allow it to accept the BCPii APIs flowing from the user application through the HWIBCPII address space. This setting controls whether a logical partition can issue a subset of control program instructions to other logical partitions activated on the same CPC.

Note: This setting must be selected on the local SE associated with the CPC of the image that the z/OS BCPii application is running on. It must also be selected for any other system for which BCPii communication is required.

Steps on the HMC

To change this setting, perform the following steps on the HMC:

1. Select the CPC that is required.
2. Open Single Object Operations.
3. Highlight the CPC icon.
4. Open the CPC Operational Customization task list.
5. Open the Change LPAR Security task, and the Change Logical Partition Security window displays.
6. Check the cross-partition authority check box for each image (LPAR) that you want to grant BCPii access. Figure 25-5 shows partition A01, which is circled. At a minimum, the image (LPAR) where the BCPii address space is running must have this authority activated.

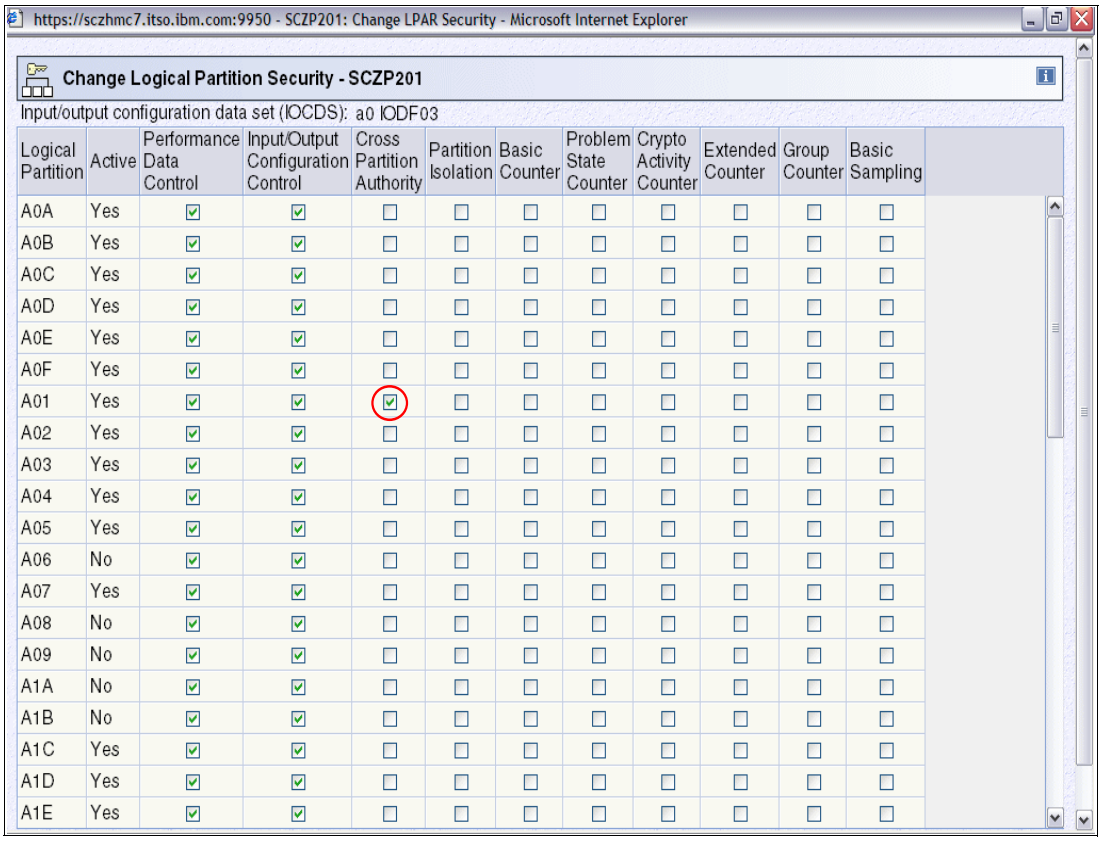


Figure 25-5 Change Logical Partition Security panel

7. When Save and Change is clicked, it might take time to update the various profiles. During this time the panel shown in Figure 25-6 is displayed.

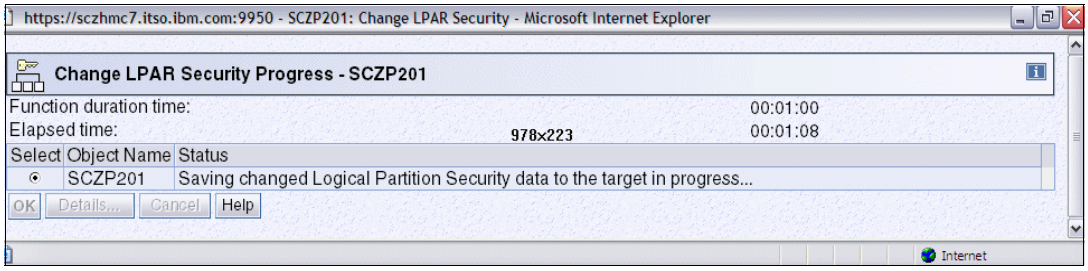


Figure 25-6 Change LPAR Security - save in progress

Failure to set this properly on the local SE associated with the image of z/OS that is running BCPii results in a severe BCPii address space initialization failure. You cannot start the address space, and you will receive communications error X'101' with a reason code of X'D4'. Failure to set this up properly on remote SEs to which you want to connect results in the same return code and reason code on the HWICONN service call.

Note: Make the same updates to all CPCs that you want BCPii to communicate with, and not simply to the CPC from which the BCPii application is going to run on.

Defining the BCPii community name on the Support Element

BCPii uses an SNMP community name to provide a level of security between the z/OS image that is executing the BCPii service and the SE itself. An “SNMP community” is a logical relationship between an SNMP agent and an SNMP manager. The community has a name, and all members of a community have the same access privileges: they are either read-only (members can view configuration and performance information) or read-write (members can view configuration and performance information, and also change the configuration).

To add the BCPii community name definition to the SE configuration, perform the following steps on the HMC:

1. Select the CPC that is required.
2. Open Single Object Operations.
3. Select the **Console Actions** view.
4. Select **Support Element Settings**.
5. Open the Customize API Settings; see Figure 25-7.

Note: The Customize API Settings button appears in the Support Element Settings panel only if you are logged on the ACSADMIN HMC user ID.

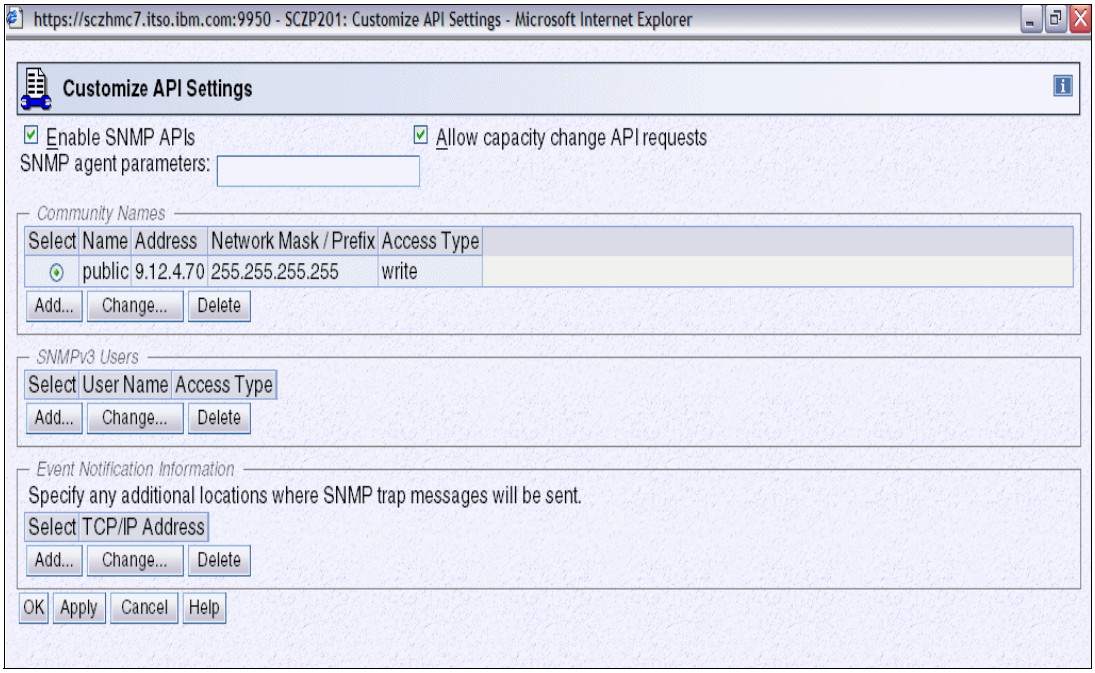


Figure 25-7 Customize API Settings

6. Check the **Enable SNMP APIs** check box; see Figure 25-7.
7. Consider checking the **Allow capacity change API requests** check box on a z10 or higher system (Figure 25-7) if the installation allows a BCPii application to perform temporary capacity upgrades.

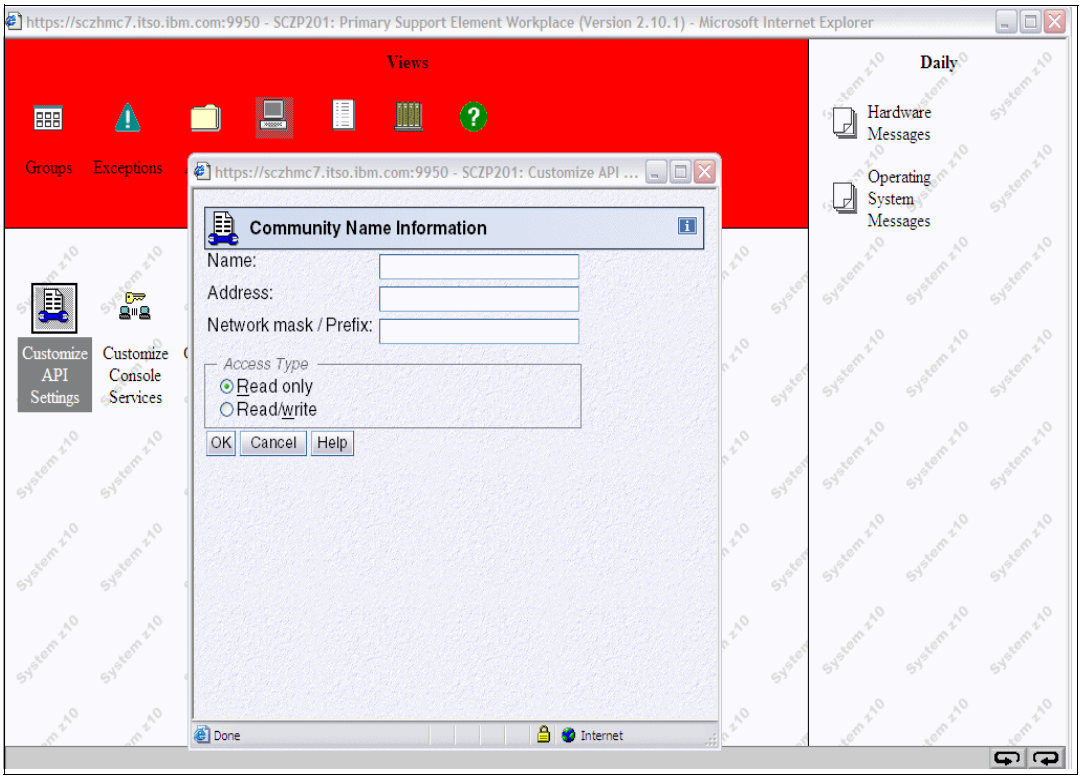


Figure 25-8 Community name information

On a Customize API Settings panel, you can add a Community Name as shown in Figure 25-8.

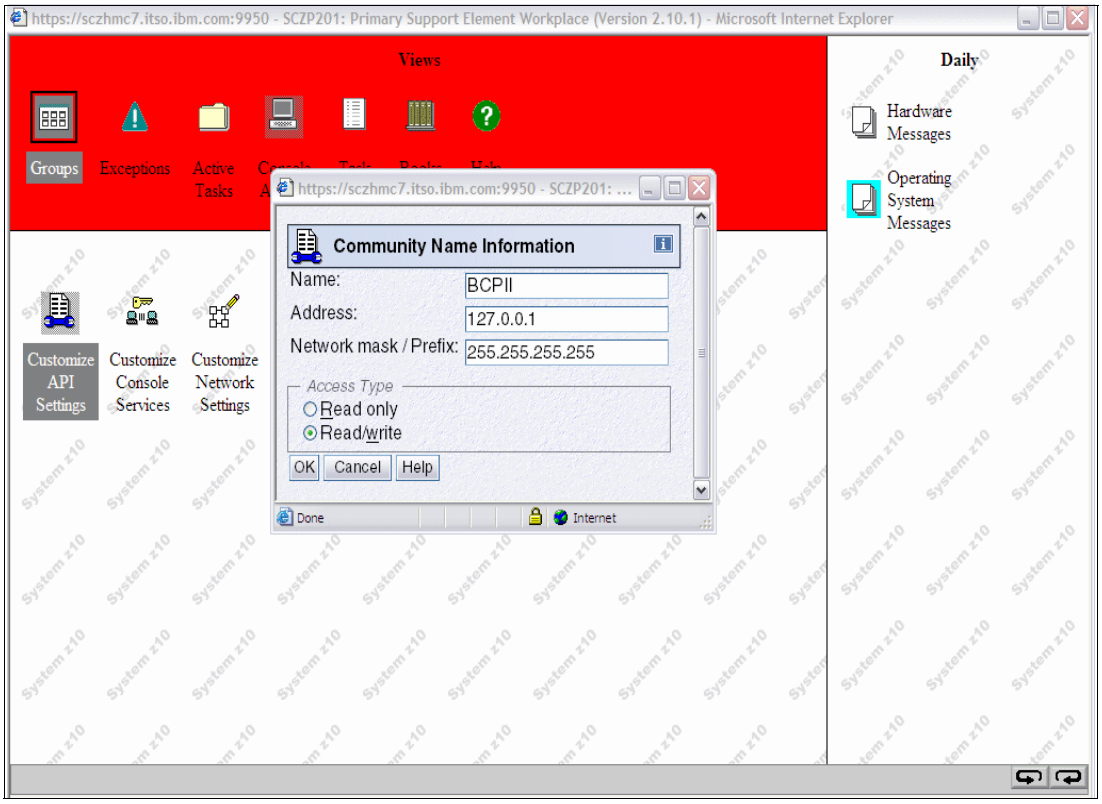


Figure 25-9 Community name in Customize API Settings

8. Make sure that the SNMP agent parameters are blank.
9. Add a BCPii community name, as shown in Figure 25-10. Click **Add**. When a window is prompted, fill in the following fields:

   Name: The actual SNMP community name. This value is a 1- to 16-character alphanumeric field. Because of restrictions with the security products on z/OS, the BCPii SNMP community name must not contain any lowercase characters.

   Important: The community name must be entered in uppercase on the local SE.

   Address: For BCPii, the address (referred as a “loop-back address”) must be 127.0.0.1.

   Network mask/Prefix: This must be 255.255.255.255.

   Note: If the community name entry is intended to be only used for BCPii purposes, then the address must be 127.0.0.1 and the mask 255.255.25.255. This limits the use of this community name to SNMP traffic that arrives over the loopback network interface, which is only used by BCPii.

In theory, an address of 0.0.0.0 and a mask of 0.0.0.0 will work as well, but it ends up creating a community name that is usable by any machine that has network connectivity to the SE. This is obviously not a desirable approach because it raises potential security concerns.

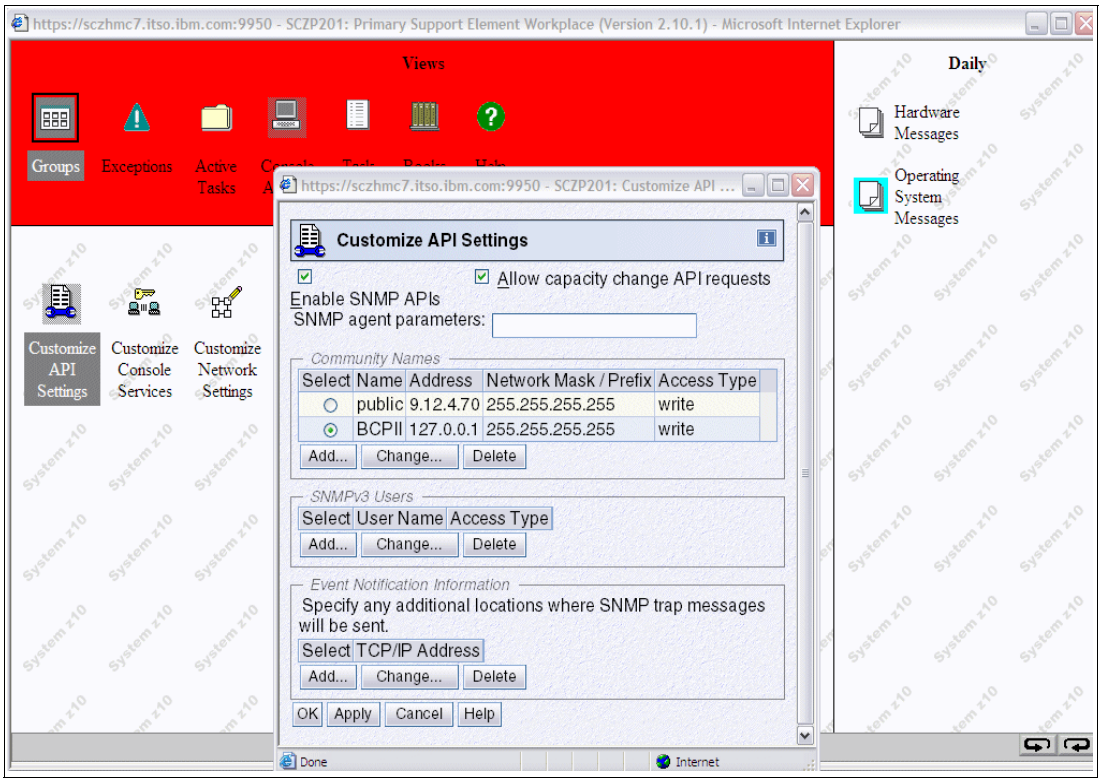


Figure 25-10 Custom API settings after apply

10. Make sure that the SNMP agent parameters are blank.

11. Add a BCPii community name, as shown in Figure 25-11. Click **Add**. When a window is prompted, fill in the following fields:

    Name: The actual SNMP community name. This value is a 1- to 16-character alphanumeric field. Because of restrictions with the security products on z/OS, the BCPii SNMP community name must not contain any lowercase characters.

    Important: The community name must be entered in uppercase on the local SE.

    Address: For BCPii, the address (referred to as a loop-back address) must be 127.0.0.1.

    Network mask/Prefix: This must be 255.255.255.255.

    Note: If the community name entry is intended to be only used for BCPii purposes, then the address must be 127.0.0.1 and the mask 255.255.25.255. This limits the use of this community name to SNMP traffic that arrives over the loopback network interface, which is only used by BCPii.

In theory, again, an address of 0.0.0.0 and a mask of 0.0.0.0 will work as well, but it ends up creating a community name that is usable by any machine that has network connectivity to the SE. This is obviously not a desirable approach because it raises potential security concerns.

When you select **Apply**, the new community name is reflected in the Customize API Settings as shown in Figure 25-10. Failing to set this properly on the local SE associated with the image of z/OS that is running BCPii results in a severe BCPii failure and you cannot start the

address space. Message HWI014I is issued if the community name defined on the SE for the local CPC does not match the definition in the security product for the local CPC.

Note: Make the same updates to all CPCs that you want BCPii to communicate with.

25.3.2 Authorize an application to use BCPii

Given the nature of the BCPii APIs and the capabilities of a BCPii application to potentially modify vital hardware resources:

- A number of authority validations are performed for each BCPii requestor.
- The API is not virtualizable.

A BCPii application needs to have program authority; general security product authority to be able to issue BCPii commands; authority to the particular resource that the application is trying to access; and a community name defined in the security product for each CPC to which communication is required.

Program authority

BCPii applications must be program-authorized, meaning that one of the following must be true of the application:

- It is running in supervisor state.
- It is running in an authorized key with PSW key mask (PKM) between 0 and 7.
- It resides in an APF-authorized library.

General security product authority

A BCPii application must have general authority to use BCPii. The profile HWI.APPLNAME.HWISERV in the FACILITY resource class controls which applications can use BCPii services. The security administrator must give at least read authority to this resource, in addition to granting authority to any specific resource that the application is attempting to access. In addition, BCPii requires that the FACILITY class to be RACLIST-specified.

Figure 25-11 shows a RACF example allowing user IBMUSER to use BCPii services in general.

```
RDEFINE FACILITY HWI.APPLNAME.HWISERV UACC(NONE)
PERMIT HWI.APPLNAME.HWISERV CLASS(FACILITY) ID(IBMUSER) ACCESS(READ)
SETROPTS RACLIST(FACILITY) REFRESH
```

Figure 25-11 Authority granting to BCPii resources

User IBMUSER is a user ID assigned by RACF to the HWISTART started task. Generic definitions may be created instead of specific users if the installation does not have specific definitions for every user.

Authorize an application to use BCPii specific resource

A BCPii application needs to have authority to the particular resource that it is trying to access. That particular resource can be the CPC itself, an image (LPAR) on a particular CPC, or a particular capacity record on a particular CPC. BCPii needs a profile defined in the FACILITY resource class that represents the target of the particular BCPii request.

The profile name required to be defined depends on the type of the particular resource required. Table 25-1 lists the specific resources to be authorized for.

Table 25-1 Specific resources to be authorized for

Request type	FACILITY class profile required
CPC	HWI.TARGET.netid.nau where netid.nau represents the 3- to 17-character SNA name of the particular CPC.
Image	HWI.TARGET.netid.nau.imagename where netid.nau represents the 3- to 17-character SNA name of the particular CPC and imagename represents the 1- to 8-character LPAR name.
Capacity record	HWI.CAPREC.netid.nau.caprec where netid.nau represents the 3- to 17-character SNA name of the particular CPC and caprec represents an 8-character capacity record name.
Activation profiles	HWI.TARGET.netid.nau where netid.nau represents the 3- to 17-character SNA name of the particular CPC the activation profile is defined.

The SNA name is composed of the Netid.name defined at the SE. The Netid is found in the panel Customize Network Settings, as shown in Figure 25-12.

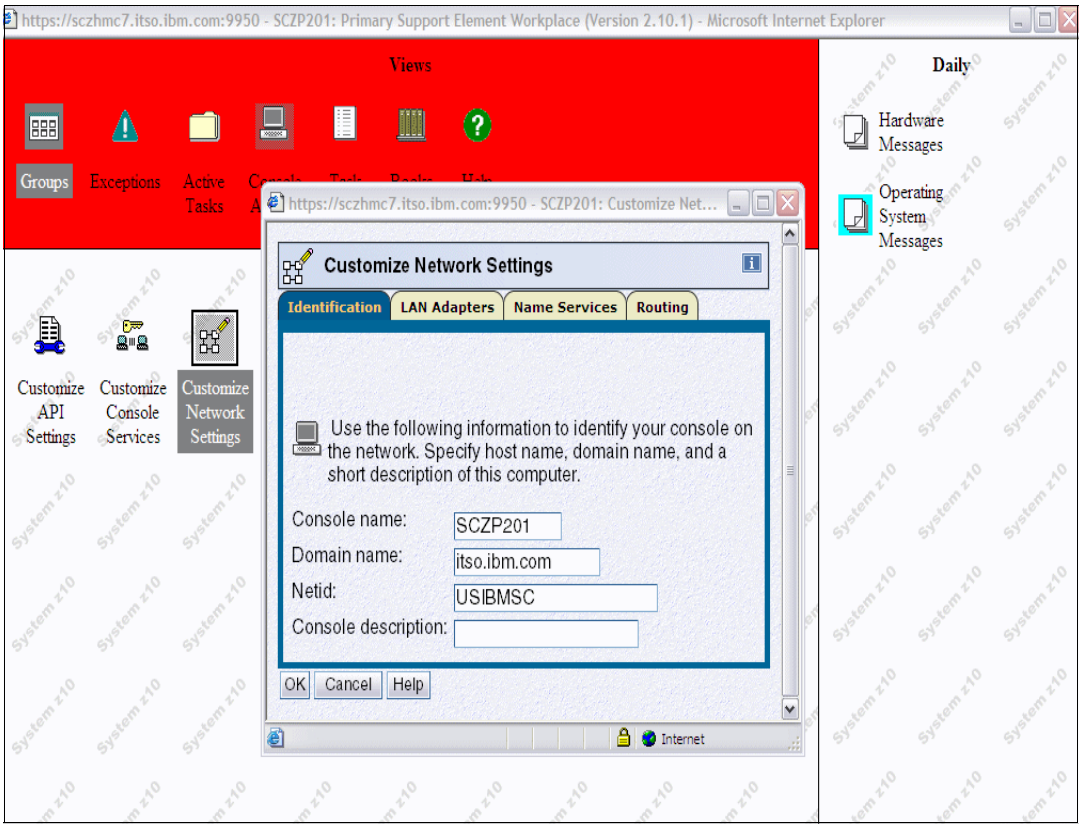


Figure 25-12 Netid on SE panel

The value for the name in the SNA name is the SNA name of the CPC from which Single Object Operations has been logged on, as highlighted in Figure 25-13.

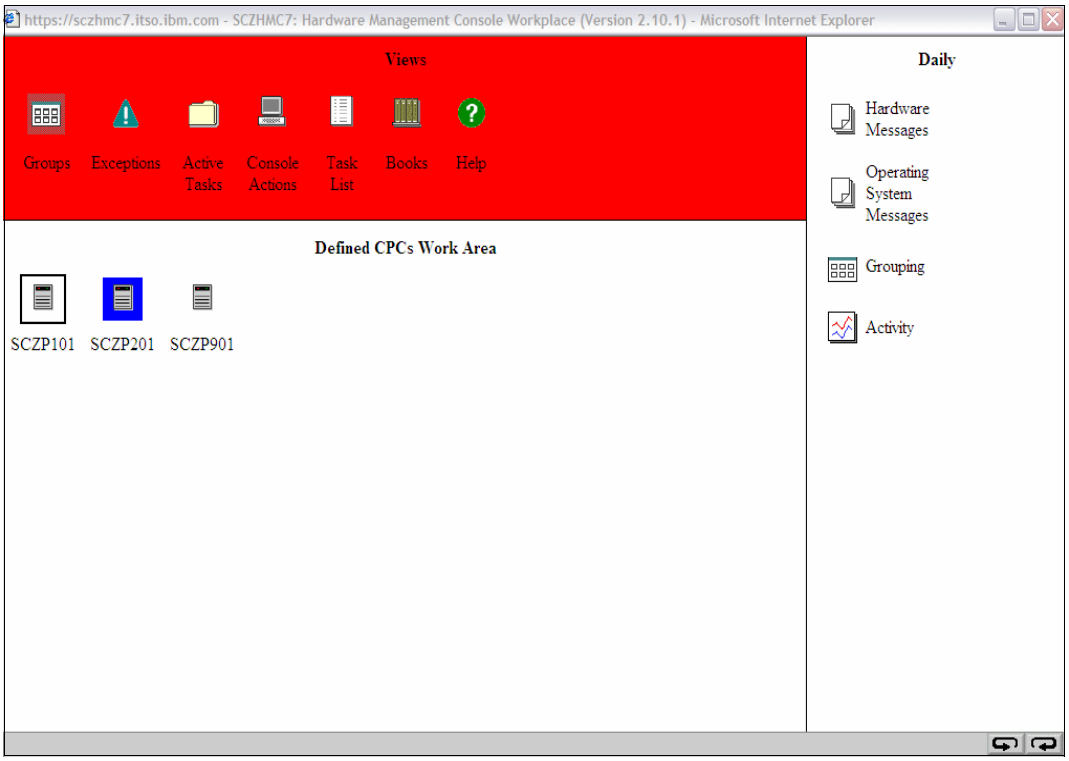


Figure 25-13 CPC name

The access level required for the particular profile depends on the service that the BCPii application attempts to issue. See the BCPii API documentation for specific information about the minimum access level required for each BCPii API service.

Figure 25-14 shows a RACF example allowing a user to have Connect, Event, List, and Query access to the SNA name of the CPC (USIBMSC.SCZP201, in our case) using the community name BCPII.

```
RDEFINE FACILITY HWI.TARGET.USIBMSC.SCZP201 UACC(NONE) APPLDATA('BCPII')
PERMIT HWI.TARGET.USIBMSC.SCZP201 CLASS(FACILITY) ID(IBMUSER) ACCESS(READ)
SETROPTS RACLIST(FACILITY) REFRESH
```

Figure 25-14 Authorizing access to CPC resource

Figure 25-15 shows a RACF example granting a user with Command, Connect, Event, List, Query, and Set access to any image (LPAR) on USIBMSC.SCZP201.

```
RDEFINE FACILITY HWI.TARGET.USIBMSC.SCZP201.* UACC(NONE)
PERMIT HWI.TARGET.USIBMSC.SCZP201.* CLASS(FACILITY) ID(IBMUSER) ACCESS(ALTER)
SETROPTS RACLIST(FACILITY) REFRESH
```

Figure 25-15 Authorizing access to LPAR resource

Community name defined in the security product for each CPC

BCPii uses an SNMP community name to provide a minimal level of security between the z/OS image executing the BCPii service and the SE. An SNMP community name is associated with a particular CPC. The same SNMP community name that was defined in the SE configuration for a particular CPC also must be defined in the security product for each CPC to which communication is required. This community name definition is extracted from the security product by BCPii and propagated to the SE. The SE validates that the community name passed by BCPii is correct before proceeding with the request.

To define the BCPii community name in the security product, use the `APPLDATA` field with the CPC profile definition to associate a community name with a particular CPC. The `APPLDATA` field for the BCPii community name contains a 1- to 16-character alphanumeric field. Because of restrictions with the security products on z/OS, the BCPii SNMP community name must not contain any lowercase characters.

Figure 25-16 shows a RACF example assigning a BCPii community name of `BCPII` to an existing CPC definition for a SNA name of CPC, `USIBMSC.SCZP201`.

```
RALTER FACILITY HWI.TARGET.USIBMSC.SCZP201 APPLDATA('BCPII')
SETROPTS RACLIST(FACILITY) REFRESH
```

Figure 25-16 Assigning a BCPii community name

25.3.3 Configuring the BCPii address space

As previously mentioned, the BCPii address space is the bridge between a z/OS application and the SE. The address space can perform the following tasks:

- Manage all application connections.
- Build and receive all internal communication requests to the SE.
- Provide an infrastructure for storage required by callers and by the transport communicating with the SE.
- Provide diagnostic capabilities to help with BCPii problem determination.
- Provide security authentication of requests.

The BCPii address space is mandatory for any BCPii API request. The system attempts to start the HWIBCPii address space during IPL. BCPii requires the high-level qualifier.SCEERUN2 and high-levelqualifier.SCEERUN data sets to be in the link list concatenation. IBM specifies these data sets in the default link list members (PROGxx parmlib member) in z/OS V1R10 and higher. Failure to have these two data sets in the link list means that BCPii cannot be started, and such failure is accompanied by error message HWI009I indicating that BCPii was unable to load a required Language Environment part.

25.3.4 Setting up the event notification mechanism for z/OS UNIX callers

Applications running in a started procedure, batch, TSO or other non-z/OS UNIX System Services environment can use the HWIEVENT service and provide their own ENF exit that receives control when the application-requested events occur on the target CPC or image.

Applications running in a z/OS UNIX System Services environment do not have normal ENF exit processing capabilities available and cannot readily listen for ENF signals. The Common Event Adapter (CEA) address space allows z/OS UNIX System Services applications to be able to receive such event notifications. BCPii provides several services that use the CEA

functionality to deliver these same events to z/OS UNIX System Services callers. Refer to 25.2.1, “BCPii services” on page 587, for more information about this topic.

CEA address space setup

The use of the CEA address space by BCPii requires minor CEA setup steps before z/OS UNIX System Services-only services of BCPii can work properly. The Common Event Adapter (CEA) address space must be active to allow the z/OS UNIX System Services-only services of BCPii to operate. CEA has two modes of operation, namely minimum mode and full-function mode. If the z/OS UNIX System Services-only services of BCPii are required to be available, then CEA must be running in full-function mode. CEA, like BCPii, starts as part of a system IPL.

CEA ENF security configuration

A z/OS UNIX System Services BCPii application must be granted authority to listen to ENF68 events. With the CEA ENF controls, it is also possible to fine-tune which BCPii events that a user is allowed to listen to.

Figure 25-17 shows a RACF example giving generic authority to the user ID associated with a z/OS UNIX System Services application authority to listen to any BCPii event.

```
AU user_id OMVS(Uid(n))
SETROPTS GENERIC(SERVAUTH)
RDEFINE SERVAUTH CEA.CONNECT UACC(NONE)
RDEFINE SERVAUTH CEA.SUBSCRIBE.ENF_0068* UACC(NONE)
PERMIT CEA.CONNECT CLASS(SERVAUTH) ID(user_id) ACCESS(READ)
PERMIT CEA.SUBSCRIBE.ENF_0068* CLASS(SERVAUTH) ID(user_id) ACCESS(READ)
SETROPTS RACLIST(SERVAUTH) REFRESH
```

Figure 25-17 Authorize ENF 68 access

To give specific authority to only certain BCPii events, use the event qualifier as part of the profile name. The event qualifier maps to the event mask for ENF68 (for more detail, see the ENFREQ information in *z/OS MVS Programming: Authorized Assembler Services Reference (EDT-IXG)*, SA22-7610). Hardware events are in the form ‘03xx00yy’ where xx is the event source (‘01’x = CPC, and ‘02’x =image) and yy denotes the particular event.

Figure 25-18 shows a RACF example allowing a user authority to only receive events related to CPC command responses (CmdResp = ‘01’x).

```
AU JOE OMVS(Uid(5))
RDEFINE SERVAUTH CEA.CONNECT UACC(NONE)
RDEFINE SERVAUTH CEA.SUBSCRIBE.ENF_006803010001 UACC(NONE)
PERMIT CEA.CONNECT CLASS(SERVAUTH) ID(JOE) ACCESS(READ)
PERMIT CEA.SUBSCRIBE.ENF_006803010001 CLASS(SERVAUTH) ID(JOE) ACCESS(READ)
SETROPTS RACLIST(SERVAUTH) REFRESH
```

Figure 25-18 Allowing a user to receive CPC events

BCPii startup and shutdown

The BCPii address space normally does not need to be started or shut down. BCPii initialization occurs during system IPL. If the configuration is correct, no further action is required. The address space remains active and ready to handle BCPii requests.

25.4 BCPii dependencies

If installed on a z/OS V1R10 system, BCPii is packaged as a separate deliverable and is enabled with APAR OA25426 (PTF UA47493). Only base functions are then available (no HWISET).

If installed on a z/OS V1R11 system, BCPii is shipped with the base operating system and additional functions are being made available:

- HWISET
- Support for IPL Token/Query PSWs
- Activation profiles support
- Minor internal serviceability enhancements.

With z/OS V1R12, BCPii comes with new functions:

- CTRACE enhancements
- Improved storage utilization and serviceability of BCPii transport code
- Additional CPC/Image attributes and commands

After installation of PTF (V1R10) or after V1R11 to V1R13 installation:

- BCPii will attempt to auto-start during IPL of z/OS
- If the proper configuration has not been made for BCPii, it will graciously terminate.
- BCPii can be restarted by the START command later, after the BCPii has been properly configured.

z/OS BCPii (unlike TSA BCPii) has no dependency on other software components or products. z/OS BCPii does, however, have dependencies on the hardware.

25.4.1 z/OS BCPii versus TSA BCPii

Tivoli System Automation (ProcOps) allows its automation product to use one of two transport protocols:

- SNMP over an IP network
- BCPii protocol (internal transport)

TSA BCPii implementation is similar to z/OS BCPii. It does, however, require TSA, IBM NetView®, and Communications Server. BCPii transport in TSA is for TSA usage only. In contrast, z/OS BCPii can run in any address space and has no other software product dependencies.

25.4.2 Hardware dependencies

To obtain the full functionality of BCPii on a z10, use the following recommended microcode levels:

- Driver 79
  - MCL034 in the N24415 (HMC-SYSTEM) EC stream - Bundle #22
  - MCL088 in the N24409 (SE-SYSTEM) EC stream - Bundle #22

For z/OS V1R13 new functions:

For zEnterprise: MCL220 in the N29802 EC stream – Bundle #22

For z10: check MCL number (and Bundle number) in the N24409 EC stream when the MCL will be built and the numbers are known.

On a z9 system, even with the recommended microcode levels, reduced functionality is made available (no IPLTOKEN, reduced attributes, no temporary capacity options)

On a machine lower than a z9 system, significantly reduced functionality (no HWICMD and reduced attributes) is made available.

Note: BCPii does not run in a z/VM virtual machine.

Capacity Provisioning

If more than one processor complex will be controlled by Capacity Provisioning, an HMC might then be required. The available options are listed in Table 25-2.

Table 25-2 SE or HMC requirement by Capacity Provisioning configuration

Configuration	Communication through System z API (SNMP)	Communication through BCPii
All observed systems and hosting system on one processor complex	HMC (preferred) or SE	SE
All observed systems on one processor complex; hosting system on a separate processor complex	HMC (preferred) or SE	HMC
Observed systems on multiple processor complexes	HMC	HMC

25.5 Exploiters of BCPii

These functions will use BCPii:

- Vendor applications use the Control center and systems management applications.
- In-house applications used by installations that need to manipulate System z resources:
  - In a more automated, algorithmic approach
  - From a z/OS environment rather than a hands-on HMC environment
  - To avoid having an intranet or Internet network connection to the process control (HMC) network.
- z/OS operating system components such as XCF
- With z/OS V1R13 - Common Information Model (CIM) and Capacity Provisioning

25.5.1 Current z/OS system BCPii exploiters

Current BCPii exploiters are:

- Capacity Provisioning Manager (CPM)

- CPM is used to enable adding or removing temporary capacity based on goals defined in the tool.
- It is available in z/OS V1R10 with APAR OA24945.

- Sysplex Failure Manager (SFM)
  - SFM is used to avoid the “Down error message” on the console when a missing XCF heartbeat occurs. It has been available since z/OS V1R11.

Migration and coexistence

BCPii is sysplex-unaware. It can run anywhere starting with z/OS V1R10 and further releases and has no migration or coexistence considerations.

25.6 BCP internal interface programming

The BCPii programming interface allows you to act on resources outside of the LPAR in which it is invoked.

Appendix C, “BCPii Metal C example” on page 823 contains a sample program, written in C, for a simple query to BCPii to obtain the number of other CPCs (see Figure 25-19).

From there, you can modify the code and invoke other BCPii functions that you want to use to manage the system.

```
JOB02749 ---- SUNDAY, 19 DEC 2010 ----
JOB02749  IRR010I  USERID LAFITTE  IS ASSIGNED TO THIS JOB.
JOB02749  ICH70001I LAFITTE LAST ACCESS AT 13:44:24 ON SUNDAY, DECEMBER 19
JOB02749  $HASP373 RUNHWI   STARTED - INIT 1 - CLASS A - SYS SC74
JOB02749   =>> BCPii C Sample starting ... <<=
JOB02749   =>> LIST all CPCs.    HWILIST Retcode = 0
JOB02749   =>> Number of CPCs found = 6
  ...................
JOB02749  $HASP395 RUNHWI ENDED
```

Figure 25-19 BCPii example program running in batch

25.6.1 A Metal C-example

C code can be compiled with the normal XL C/C++ compiler. Although this works, it implies that the module produced has to run in a z/OS UNIX System Services environment called a “process.”

As shown in Figure 25-20, however, a C program compiled with Metal C is simply the expression of an algorithm that has been translated into assembler, without any reference to POSIX such as a process, pipe, heap, and so on.

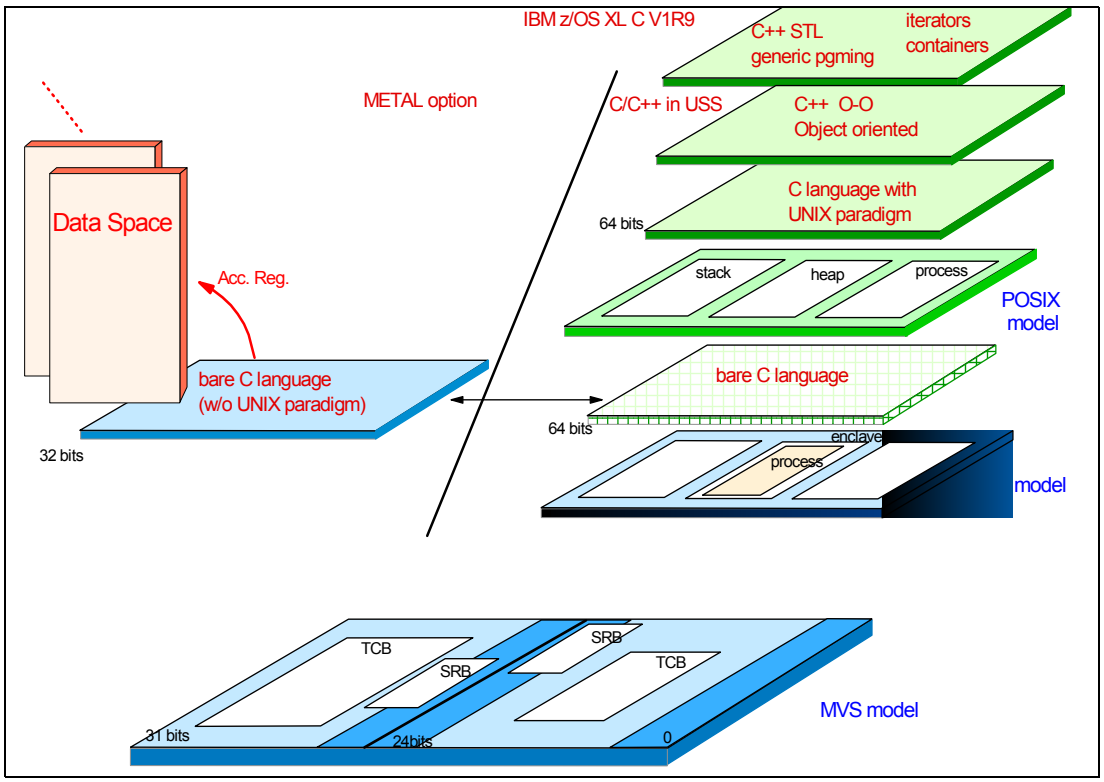


Figure 25-20 Metal C versus XL C/C++

As illustrated by Appendix C, “BCPii Metal C example” on page 823, this allows the module to run anywhere in z/OS. More specifically, it can be called as a command in a System REXX procedure.

25.6.2 BCPii as a set of new z/OS commands

When inserted in a System REXX procedure (as illustrated in “BCPii example as a System REXX” on page 844), the call to the same program can become like a z/OS command, directly suited for insertion in any automation tool. See Figure 25-21.

```
SC74 2010347 09:18:18.91 TSU02712 IEA630I  OPERATOR LAFITTE  NOW ACTIVE, SYSTEM
 SC74 2010347 09:18:25.68 LAFITTE   @Q CPC
 SC74 2010347 09:18:25.69 IRR812I PROFILE ** (G) IN THE STARTED CLASS WAS USED
865
                              865         TO START AXR03 WITH JOBNAME AXR03.
 SC74 2010347 09:18:25.71 STC02716 $HASP100 AXR03    ON STCINRDR
 SC74 2010347 09:18:25.76 STC02716 $HASP373 AXR03    STARTED
 SC74 2010347 09:18:25.81 STC02716 IEA630I OPERATOR *AXT0374 NOW ACTIVE, SYSTEM
 SC74 2010347 09:18:25.83 STC02716 =>> BCPii C Sample starting ... <<=
 SC74 2010347 09:18:26.33 STC02716 =>> LIST all CPCs.    HWILIST Retcode = 0
SC74 2010347 09:18:26.33 STC02716 =>> Number of CPCs found = 6
 SC74 2010347 09:18:25.82 STC02716 AXR0500I AXREXX OUTPUT DISPLAY 869EXECNAME=Q
REQTOKEN=00004000000000000C705BA95383ED8A6 869 Query zHybrid
TSU02712 IEA631I OPERATOR LAFITTE  NOW INACTIVE, SYST
```

Figure 25-21 BCPii program called as a System REXX procedure

Other codes can also be isolated from the example given in Appendix C, “BCPii Metal C example” on page 823. This can lead to a new set of z/OS commands with a general syntax.

```
<sentence>  ::=  <verb>  <vmid> "["  <object>  "]"   ¦
                 <verb>  <vmid> "("  <command> ")"

<verb>      ::=  'q'  ¦  'set'

<vmid>      ::=  { <n> "." ¦ <name> "." ¦ <id> "(" p ")." } +

<n>         ::=  '-1' ¦ <p>

<p>         ::=  '0'-'9'

<object>    ::=  .. any object recognized by the BCPi interface ...

<command>   ::=  .. any command to be issued unto the OS running in <vmid>
```

Figure 25-22 Backup Normal Form of a simple BCPii z/OS command set

As shown in Figure 25-23, this can raise the possibility of having commands such as:

@q 2[CPU]	Query the CPU type of LPAR 2 of the given system.
@q PROD0.VM02.ZLNX03[CPU]	Query the CPU of virtual machine ZLNX03 in virtual machine VM02 in LPAR PROD0.
@q 3.3.1[CPU]	Query the CPU of the same virtual machine, in a recursive manner.
@q CSS(0).MIF(3).MAINT[object]	Query an object in an XMP manner.
@set -1.IP_address[PTS]	Set system at IP_address as the Primary Timer Server in a coordinated time network.

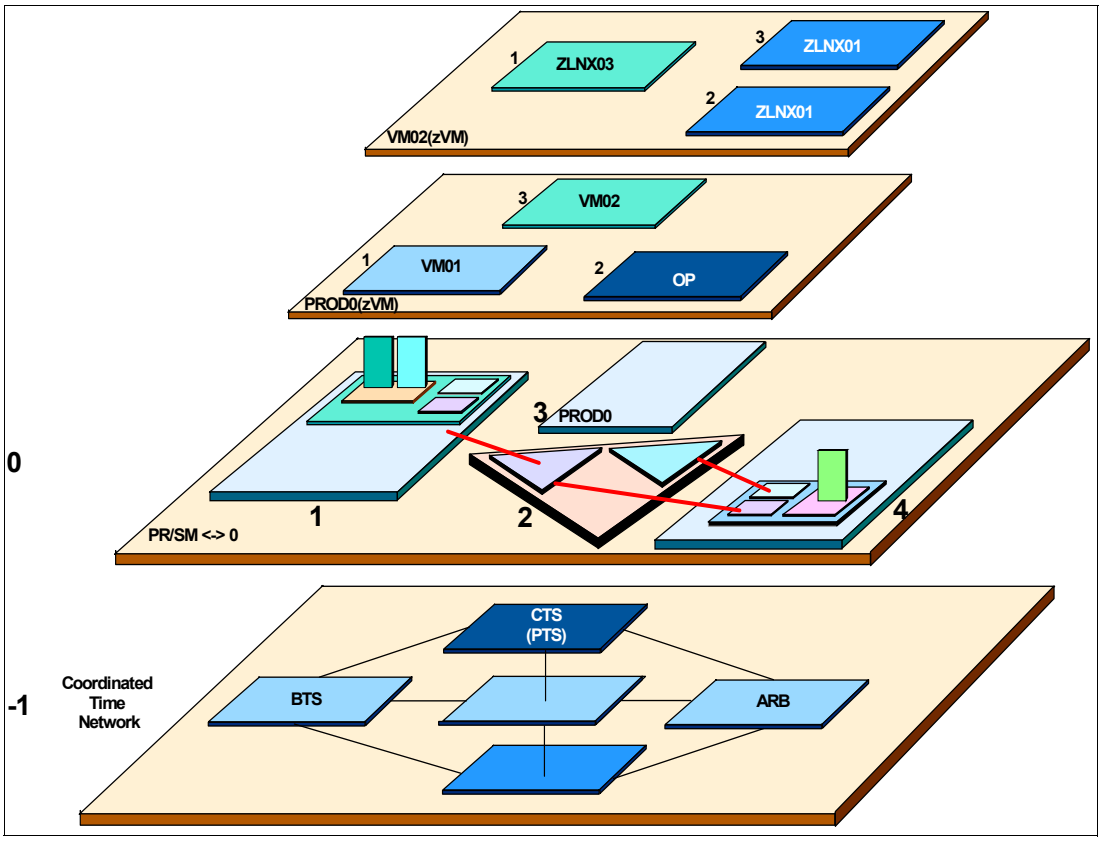


Figure 25-23 Sample context of the proposed BCPii commands

25.6.3 Potential benefits of a new command set

A new command set potentially can provide the following benefits with regard to BCPii:

- It can help to implement proper RACF security checking within the respective REXX procedures; see “BCPii services” on page 587.
- It can be the base for tracing what command is issued by which user on which system, thereby building a log record of the various actions issued on the BCPii interface.

26

Capacity Provisioning

z/OS Capacity Provisioning allows installations to manage processing capacity more reliably, more easily, and faster. Replacing manual monitoring with autonomic management, or supporting manual operation with recommendations, can assure that sufficient processing power will be available with the least possible delay. With the z/OS Capacity Provisioning function, it is possible to manage processing capacity through policy-based automation.

z/OS Workload Manager (WLM) manages the workload by goals and business importance (as defined by an installation WLM policy) on each z/OS system in a Parallel Sysplex. WLM metrics (such as resource delays and performance index) are available through existing interfaces and are reported through RMF Monitor III, with one RMF gatherer per z/OS system.

This chapter explains the latest enhancements provided with z/OS V1R13 for the Capacity Provisioning Manager (CPM), as listed here:

- User-defined capacity increments provide a better way to control the amount of capacity to be added.
- Recurring time conditions provide a better way to control when additional capacity can be added.
- Provisioning Manager command is enhanced, allowing you to filter the output of the workload report.
- New Control Center support - Windows 7 is now supported.
- Certain migrations are necessary now:
  - Support of SNMP will be removed in a later release; consider migrating to z/OS BCPii.
  - Java 5 is no longer supported and run time needs to be migrated.

26.1 Capacity Provisioning background

In releases prior to z/OS V1R9, WLM uses the IRD feature to control the distribution of physical processors among the z/OS systems contained in an LPAR cluster. This task was executed by the following WLM functions:

- WLM Vary Logical CPU Management, where the number of logical processors in a LP is dynamically controlled by WLM
- WLM Vary Weight Management, where based on whether goals are being achieved, LP weights are dynamically modified by taking processor resource from one LP and delivering to it others in the same server

With z10 EC, using provisioning capability combined with the Capacity Provisioning Management (CPM) component in z/OS makes it possible in a new, flexible, and automated process to control the activation of On/Off Capacity on Demand.

Furthermore, in earlier releases WLM was only able to take processor resource from an LP and give it to other LPs to decrease delayed transactions. Now CPM (WLM is included in that function) can activate spare processors through On/Off Capacity on Demand to avoid such delays.

Unexpected workload spikes can exceed available capacity so much that service level agreements (SLAs) cannot be met. Although business need might not justify a permanent upgrade, it might well justify a temporary upgrade. With the z10 EC, the hardware provides improved and integrated On/Off CoD and CBU, resulting in faster activation and improved robustness, and it can be partially activated and combined.

As mentioned, System z Capacity Provisioning allows an enterprise to manage processing capacity more effectively and efficiently. Using autonomic management instead of manual monitoring, or supporting manual operation with recommendations, can help to assure that sufficient processing power will be available with the least possible delay. Finally, Capacity Provisioning demonstrates the superior vertical scalability of System z.

Capacity provisioning infrastructure overview

The z/OS provisioning environment, with all its components, is shown in Figure 26-1.

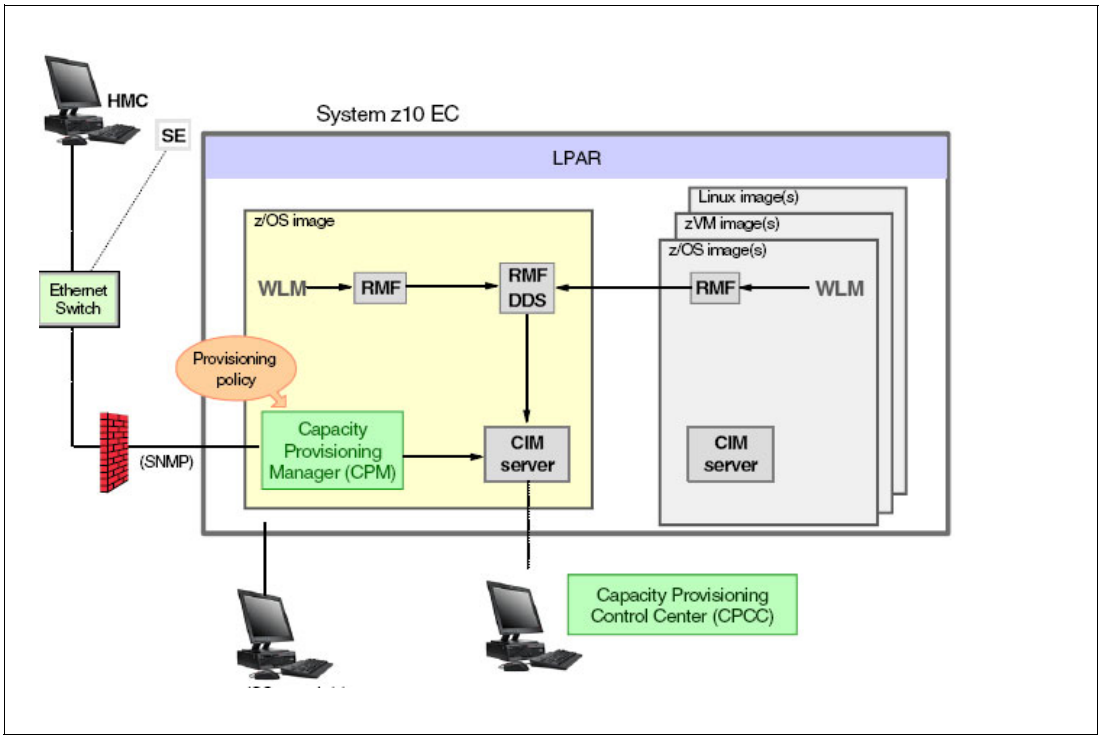


Figure 26-1 z/OS Capacity Provisioning

WLM manages by goals and makes its metrics available through existing interfaces, as explained here:

- There is one RMF gatherer per z/OS system.
- There is a RMF distributed data server (DDS) per sysplex. From the distributed data server (DDS) facility where all data captured, RMF can be send data to a RMF focal point.
- The RMF Common Interface Module (CIM) providers and associated CIM models publish the RMF Monitor III data.

  CIM is a standard data model developed by a consortium of major hardware and software vendors including IBM. The consortium, known as Distributed Management Task Force (DMTF), is a part of the Web Based Enterprise Management (WBEM) initiative. It includes a set of standards and technologies that provide management solutions for distributed network environments.

  CIM creates an interface (API) where applications can, for example, ask system management questions to determine how many jobs are running in the system. Such queries must be converted to an API known by the running operating system. In z/OS, CIM is implemented through the Common Event Adapter (CEA) address space.
- Then the Capacity Provisioning Manager (CPM), a function inside z/OS, retrieves critical metrics from one or more z/OS systems through the CIM structures and protocol. Depending on such metrics, CPM communicates to (local or remote) SEs and HMCs, respectively, through the SNMP protocol to access On/Off Capacity on Demand. The control over the Provisioning Infrastructure is executed by the CPM through a Capacity Provisioning Policy (CPP) that controls the Capacity Provisioning Domain (CPD).
- A Capacity Provisioning Policy is created and managed by the Capacity Provisioning Control Center (CPCC). The CPCC resides on a workstation, providing systems programmers with a front-end to administer such policies. CPCC is a graphical user

interface (GUI) component. (These policies are not the ones managed by WLM and kept in the WLM couple data set.) Note that CPCC is not required for regular CPM operation.

Capacity Provisioning Policy

A Capacity Provisioning Policy can consist of multiple rules based on a variety of items, such as specific applications like bank transactions, for example. Figure 26-2 illustrates a Capacity Provisioning Policy.

- Maximum Provisioning Scope defines the maximum additional capacity (expressed in MSUs, zIIPs, zAAPs) that may be activated at any time for all contained rules.
- Provisioning Condition is simply a group of time and workload conditions that can be referred to:
  - WLM Service Class conditions
  - Time condition (start/deadline/end)
  - Workload (critical workload conditions)
- Provisioning Scope defines the maximum capacity (expressed in MSUs, zIIPs, zAAPs) that may be activated.

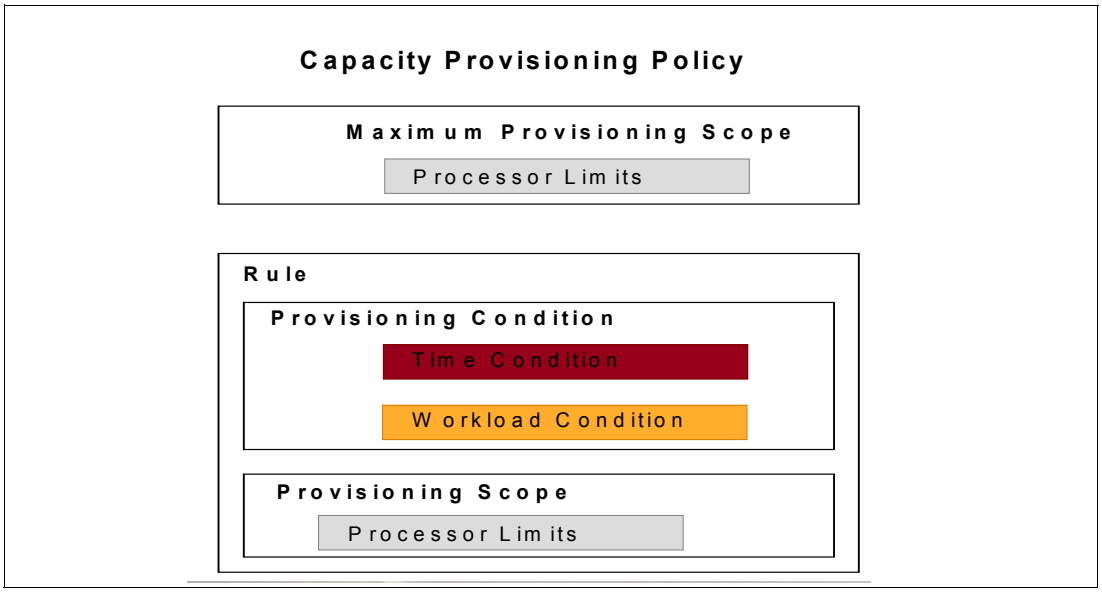


Figure 26-2 Capacity Provisioning Policy

Capacity management

CPM differentiates between several types of provisioning requests:

- Manual, through HMC or CPM (through commands)
- Scheduled (time condition without workload condition)
- Conditional (based on workload condition)

Separate capacity demands can be expressed as number of zAAPs, number of zIIPs, or general purpose capacity:

- Needs to consider various capacity levels (processor speeds) for subcapacity processors
- Speed demand for higher capacity levels
- Unqualified demand (capacity level or number of GCPs)

Capacity Provisioning components

Capacity Provisioning is mainly composed of the following components:

- As mentioned, the Capacity Provisioning Manager (CPM) is the server program that monitors the defined systems and CPCs and takes actions as appropriate and authorized by the policies.
- As also mentioned, the Capacity Provisioning Control Center (CPCC) is the graphical user interface (GUI) component. It is the interface through which administrators work with provisioning policies and domain configurations

  Optionally, you can use the CPCC to transfer provisioning policies and domain configurations files to the CPM, or to query CPM status.

  The CPCC is installed and used on a Microsoft Windows workstation. It is not required for regular operation of the CPM.

Capacity Provisioning enhancements in z/OS V1R12

In z/OS V1R12, the following Capacity Provisioning enhancements have been made:

- Support for CICS and IMS transaction service classes in workload conditions within a Capacity Provisioning policy
- Simplified setup of the Capacity Provisioning CIM provider
- RACF security definitions simplified and provided as samples
- Detailed status information about the Capacity Provisioning Manager within the Capacity Provisioning Control Center
- Support for management on behalf of the averaged Performance Index (PI)
- Support for Microsoft Windows Vista on the workstation running the Capacity Provisioning Control Center
- Support for new IBM zEnterprise System hardware with the Power Save Function

26.2 Capacity Provisioning Domain

Capacity Provisioning Domain (CPD) represents the set of servers (CECs) that are controlled by CPM the Capacity Provisioning Manager. The HMCs of the CPCs within a CPD must be connected to the same processor LAN. Parallel Sysplex members can be part of a CPD. There is no requirement that all z/OS members of a Parallel Sysplex must be part of the CPD. However, participating z/OS members must all be part of the same CPD.

Administrators work through the CPCC interface to define domain configurations and provisioning policies, but this is not needed during production. The CPCC is installed on a Microsoft Windows workstation. Figure 26-3 shows a Capacity Provisioning Domain.

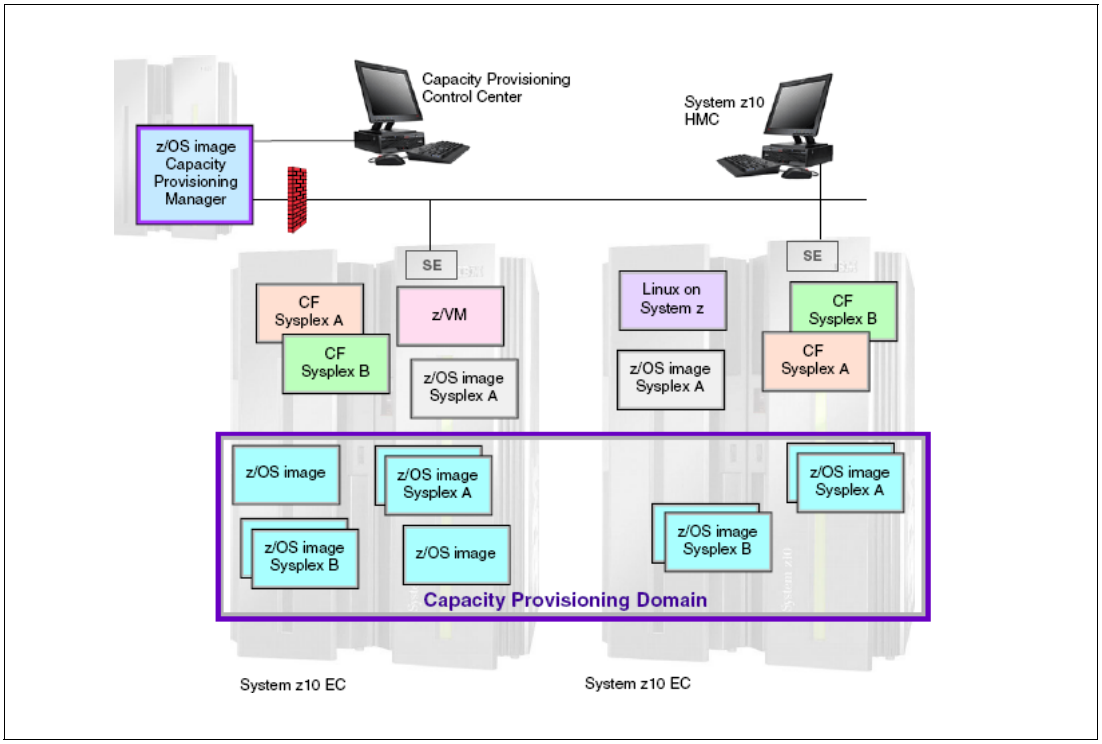


Figure 26-3 Capacity Provisioning Domain

Capacity Provisioning Manager processing modes

CPM operates in any of the following modes, thereby allowing for various degrees of automation:

- Manual mode - This is basically a command-driven mode where no CPM policy is active.
- Analysis mode - This works as described here:
  - CPM processes the capacity provisioning policy and informs the operator when a provisioning or deprovisioning action is due according to the criteria specified in the policy.
  - There is no checking done if On/Off CoD record is available.
  - Operators can choose to either ignore that information or to perform the upgrade or downgrade manually (using the HMC/SE or the available CPM commands).
- Confirmation mode - CPM processes the policy and the On/Off CoD record to be used for Capacity Provisioning. Every provisioning action needs to be authorized (confirmed) by the operator.
- Autonomic mode - Similar to confirmation mode, except that no human operator intervention is required.

In all modes, various reports will be available containing information about workload and provisioning status and the rationale for provisioning recommendations.

The CPM command user interface works through the z/OS system console. Note that configuration changes, rule changes, and so on cannot be executed from a console. Thus, CPCC and the console have no common commands.

The CPCC (only partially) defines configuration, policy, rules, and so on. The CPCC can upload them to CPM, but do nothing else. Even to activate these items, you need to use the MVS console and run a command.

The CPM server uses the following types of input data sets, which each contain separate types of information:

- The domain configuration defines the topology and connections, such as the CPCs and z/OS systems that are to be managed by the server.
- The policy contains the following information:
  - Which work is provisioning eligible, under which conditions and during which time frames.
  - How much capacity may be activated when the work suffers due to insufficient processing capacity.
- The PARM data set contains setup instructions such as UNIX environment variables and various processing options that can be set by an installation.

The provisioning policy defines the circumstances under which additional capacity may be provisioned. There are three elements in the criteria, namely the time condition, the workload condition, and the provisioning scope, as explained here.

- Time condition - when provisioning is allowed:
  - Start time - When provisioning can begin.
  - Deadline - Provisioning of additional capacity no longer allowed.
  - End time - Deactivation of additional capacity is to begin.
- Workload condition - Which work qualifies for provisioning; parameters include:
  - The z/OS systems that may execute eligible work.
  - Importance filter - Eligible service class periods, as identified by WLM importance.
  - Performance Indicator (PI) criteria.
    - Activation threshold - PI of service class periods must exceed the activation threshold for a specified duration before the work is considered to be suffering.
    - Deactivation threshold - PI of service class periods must fall below the deactivation threshold for a specified duration before the work is considered to no longer be suffering.
  - Included Service Classes - Eligible service class periods.
  - Excluded Service Classes - Service class periods that are not to be considered.
- Provisioning scope - How much additional capacity may be activated, expressed in MSUs:
  - Specified in MSUs, number of zAAPs, and number of zIIPs; one specification per CPC that is part of the CPD.

Capacity Provisioning Manager rules

The CPM includes the following rules:

- The domain configuration defines CPCs and z/OS systems that are controlled by a CPM instance.

- Sysplexes do not have to be completely contained in a domain, but must not belong to more than one domain.
- Multiple sysplexes and therefore multiple WLM service definitions may be involved.
- There can be only one active Capacity Provisioning Policy (CPP) per domain at a time.
- More than one policy can exist for separate purposes.

26.3 Workload condition

If time conditions are defined in a rule, but workload conditions are not defined in a rule, the Provisioning Manager schedules the activation and deactivation of additional capacity as specified. The maximum additional allowed resources in the provisioning scope are activated at start time and deactivated at the specified end time.

Capacity Provisioning allows you to define business-critical work to be eligible for provisioning using workload conditions. The concept of *workload condition* is based on the WLM service class model:

- In a WLM service definition, work is assigned to WLM service classes, which are associated with goals such as response time.
- A service class period describes how a group of work is to be managed by WLM. It includes a duration, a goal, and an importance. A service class can be composed of multiple periods. If work does not complete within the specified duration, it falls through to the next period.
- The importance of a service class period describes its business importance. If not all goals can be met, then service classes with a lower importance will grant to service classes with higher importance.
- Service classes are defined for the entire sysplex. On other sysplexes, the same or other service definitions may be active.

Workload condition identifies the work that can trigger the activation of additional capacity when that work does not achieve its goal due to insufficient capacity, and additional capacity can help. Its parameters are:

- Sysplex/Systems - The z/OS systems that may run eligible work.
- Importance filter - Eligible service class periods, identified by WLM importance.

  The filter includes the importance parameter, which indicates the relative importance of the service class periods. All service class periods with an importance value equal to or smaller than that specified match the filter unless another importance filter applies.

  An importance filter also includes Provisioning Criteria PI values indicating when service class periods matching the importance filter are considered to be suffering.

  For example, if you specify importance value 3 in a filter, all service class periods with importance values 3, 2, and 1 match the filter and the specified provisioning criteria will be applied to them. To define other provisioning criteria only for service class periods of importance value 1, you can define another importance filter with the new criteria. The filter for importance value 3 then applies only to service class periods with importance values 3 and 2, and the filter for importance value 1 only applied to service class periods with importance value 1.
- Service class period filter - Included and excluded service class periods are identified by service class period filters, which contain criteria that a service class period must match to be considered or ignored by the Provisioning Manager.

These filters include the following parameters:

- Service Definition - The name of the WLM service definition. The specified service class periods are only considered if this WLM service definition is installed. You may specify an asterisk (*) to include all WLM service definitions.
- Service Policy - The name of the service policy within the WLM service definition. The specified service class periods are considered if a service policy with that name is activated. You may specify an asterisk (*) to include all service policies matching the other criteria.
- Service Class - The name of the service class. You may specify an asterisk (*) to include all service classes matching the other criteria.
- Period - The period of the service class that is to be considered. In an included service class filter, this period and all periods with a lower period number are considered eligible to trigger provisioning. If a service class has periods lower than this number, all periods will be considered.

  In an excluded service class filter, this period and all periods with a higher period number are excluded from provisioning.

- Performance Index (PI) Criteria
  - Activation threshold - The PI of service class periods must exceed the activation threshold for a specified duration when the work is considered to require help.
  - Deactivation threshold - The PI of service class periods must fall below the deactivation threshold for a specified duration when the work is considered to no longer require help.
- Included Service Classes - Eligible service class periods
  - This extends the set of Service Class periods with qualified work (extends the default set of default-eligible service classes) and may specify various PI criteria.
- Excluded Service Classes - This identifies service class periods that are not to be considered.
- If specifications exist on multiple levels, then the service class periods as derived from the importance filter are merged with the explicitly defined (included) service class period. Finally, the excluded service class periods (if any) are removed from the previous set.

If no workload condition is specified, then the full capacity will be activated and deactivated unconditionally at the start and end times of the time condition (scheduled activation, deactivation).

Provisioning PI and provisioning PI duration are used by the Provisioning Manager to detect whether observed service class periods need help. Before any actions are taken, the Provisioning Manager considers the resource demand of the service class period to ensure that the activation of additional processing capacity can improve the PI. Deprovisioning PI and deprovisioning PI duration are used by the Provisioning Manager to detect when a service class period no longer needs help.

For example, assume a workload condition is specified including service class online. This condition is defined with one period of WLM service definition WLMSD, a provisioning PI of 1.8 and duration of 10 minutes, and a deprovisioning PI of 1.2 and duration of 10 minutes.

If the PI of the service class period changes within a defined time condition, as shown in Figure 26-4, the Provisioning Manager detects three instances where the provisioning PI criteria are fulfilled. At the first two instances, the Provisioning Manager will activate additional capacity. The third instance is ignored because it occurs after the deadline. The Provisioning Manager can also detect an instance when the deprovisioning PI criteria are fulfilled. The

Provisioning Manager then decides that service class ONLINE no longer needs additional capacity and deactivates it.

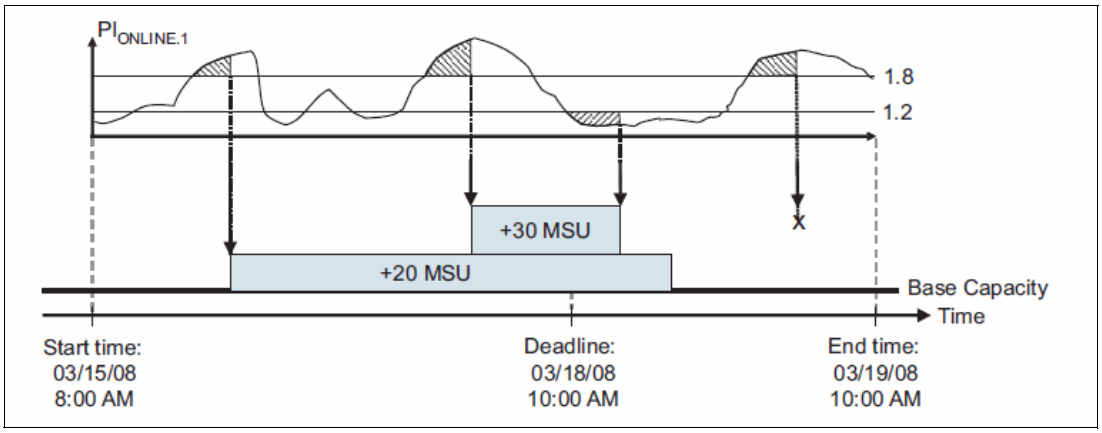


Figure 26-4 Provisioning criteria semantics

Moving average PI

The performance index of many workloads can change rapidly, for example because the amount of work varies, or because additional capacity becomes available in the system. If the provisioning duration includes several observation intervals, such as RMF MINTIME intervals, it can become unlikely to encounter a contiguous number of monitoring intervals such that the PI exceeds the provisioning PI for the entire provisioning duration.

Managing a workload by moving average PI allows accounting for that behavior. When the actual performance index of a workload decreases below the provisioning PI for a short time, the moving average PI may still exceed that limit. Consequently, high PI values in the provisioning duration can be recognized more reliably.

Figure 26-5 shows a fluctuating workload that does not meet the provisioning criteria because during the specified provisioning PI duration, the PI temporarily drops beneath the provisioning PI.

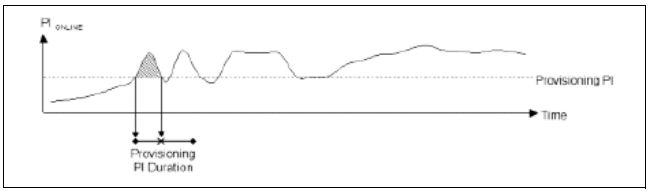


Figure 26-5 Fluctuating workload missing provisioning criteria

Optionally, the Provisioning Manager can average the PI provided by the monitoring component for a service class period. In this case the Provisioning Manager calculates a moving average PI. For this, the actual PI pattern is averaged through an exponentially weighted moving average (EWMA) function. The calculation considers current PI observations (in the interval `t`) and all preceding PI observations (as far as back to the first observed interval `0`) of a continuous time series. It weights the values with a user-specified smoothing factor ?; see Figure 26-6.

$$PI_{t}_{mov.avg.} = \omega \sum_{i=0}^{t-1} (1-\omega)^{i} PI_{t-i} + (1-\omega)^{t} PI_{0}$$

Figure 26-6 Exponentially weighted moving average (EWMA) PI formula

The graph resulting from such a moving average smoothening is characterized by a more even value pattern. This can allow for activations in situations that might not be taken into account without the additional smoothening. Figure 26-7 shows the PI pattern smoothened by the EWMA PI formula.

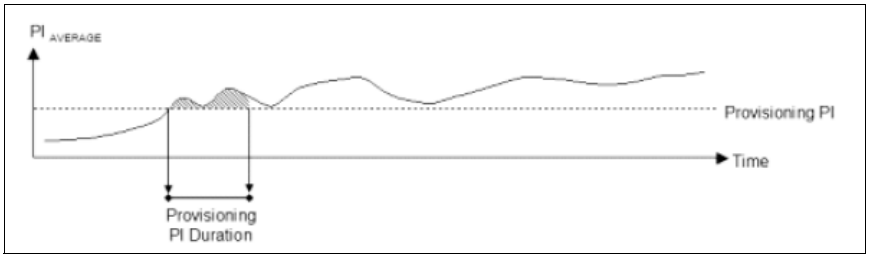


Figure 26-7 EWMA smoothened PI pattern with workload meeting provisioning criteria

A characteristic of the smoothening algorithm is that it delays the moments when the provisioning PI limit or the deprovisioning PI limit are being crossed; the smaller the smoothing factor ?, the more PI limit crossings will be delayed. Therefore, related capacity activations or deactivations will also be delayed. Reducing provisioning or deprovisioning PI durations adequately can compensate for this delay.

The formula assumes that a contiguous series of PI values (time interval i=0 to t) is available, meaning that PI values are reported for every interval. For patterns with short gaps without any workload at all, the formula will disregard all PI observations preceding the last gap. The series of contiguous PI values can also be interrupted after subcapacity changes of the managed hardware, when the monitoring component (such as RMF) needs to recalculate the provided data.

To avoid having high PI values distort the computed moving average PI pattern, specify a maximum capping PI value. This limits the PI values to be considered to the maximum value specified by the capping PI. PI values exceeding that limit will be replaced by the capping PI when computing the moving average PI. That protects the resulting moving average PI graph from high PI values having long-term effects.

By default, the PI is capped at 5.5. If necessary, it can be set to another value, or moving average PI capping can be disabled, but this is not desirable.

Management on behalf of moving average PI and the capping value of the maximum capping PI are set globally. Therefore, both the smoothing factor ? value and the maximum capping PI value apply to all observed workloads.

26.4 Capacity increments starting with z/OS V1R13

Capacity Provisioning activates capacity step by step. For subcapacity machines, and for full capacity machines with many processors, a one-step increment is usually insufficient capacity to resolve a bottleneck.

For instance, for full capacity machines with many processors, one additional processor is simply a small relative increment (for example, adding the 51st processor only increments the capacity by roughly 1%).

In consequence, using many activation steps to achieve your required capacity takes too much time. To alleviate such a scenario, z/OS V1R13 allows you to define the capacity increments for the first and all following activations of additional capacity in the policy.

Define capacity increments

The support is invoked by adding the primary and secondary activation capacity increments to the maximum provisioning scope for each CPC; see Figure 26-8.

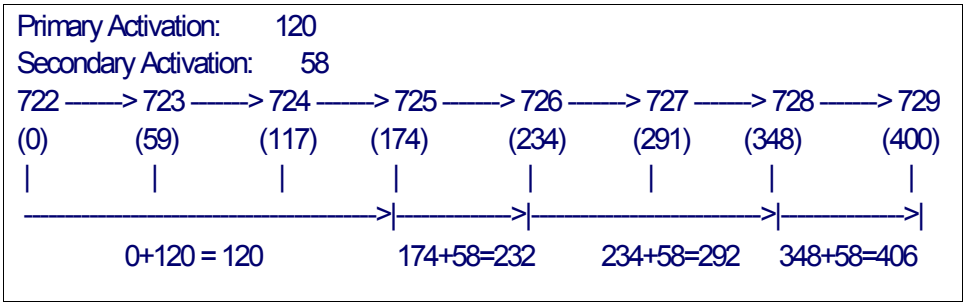


Figure 26-8 Capacity increments

The following new or changed external outputs are provided:

- The Policy report and policy status in the Control Center reports the capacity increments.
- Activation requests may skip the intermediate software model.

Adjusting capacity

With this support, you can now adjust capacity in the actual increments needed by the installation. Thus, it takes less time to achieve the capacity that suits the workload requirements.

You can specify the following primary and secondary capacity increments for each CPC:

Primary increment Capacity increment for the first activation

Secondary increments Capacity increments for all further activations

The increments are for general purpose capacity, and are specified in MSUs. Figure 26-9 shows primary and secondary capacity increments.

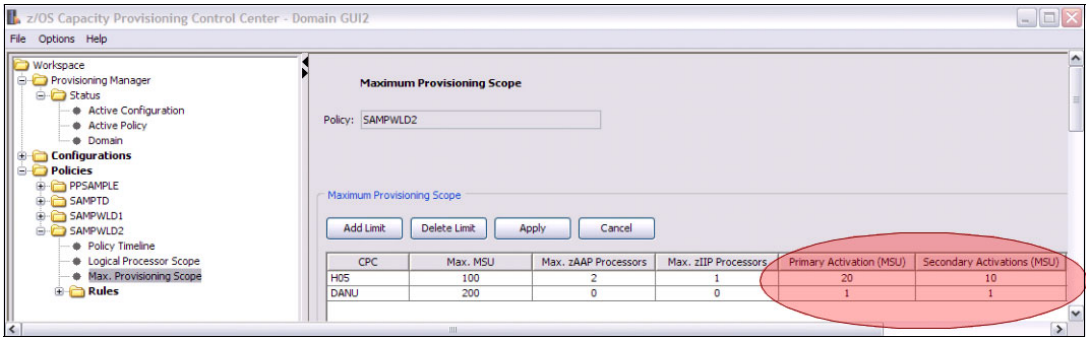


Figure 26-9 Capacity increments

Set the value for Primary Activation and for Secondary Activations at less than the maximum MSU value. The default for both values is 1. The allowed range is from 1 to 9999.

First activation The first model to be activated is to have more or equal MSU than the number of MSUs of the current model added by the number of MSUs specified as primary activation.

The number of MSUs of this model may not have more MSUs than specified in the maximum provisioning scope.

If no model fulfils the criteria, then the model with the most allowed MSUs will be used.

All other activations The next model to be activated is to have more or equal MSUs than the number of MSUs of the current model added by the number of MSUs specified with secondary activations.

The number of MSUs of this model may not have more MSUs than specified in the maximum provisioning scope.

If no model fulfils the criteria, then the model with the most allowed MSUs will be used.

Changes to the Details panel

Two groups have been added to the Details panel, one for Primary Activation and one for Secondary Activations, as shown in Figure 26-10.

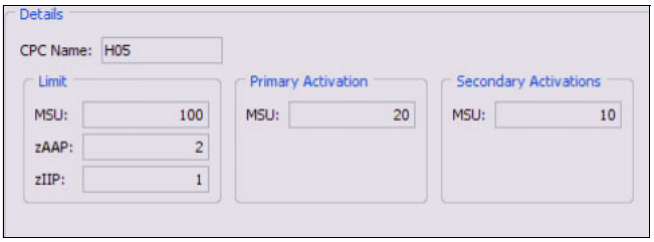


Figure 26-10 Capacity increment in Control Center

Policy report

The policy status of the Processor Limits details of the Maximum provisioning scope in the Control Center now additionally shows:

- The minimum capacity increment (in MSU) for the first activation
- The minimum capacity increment in (MSU) for all further activations

Reporting on the console is shown in Figure 26-11.

```
Policy EXAMPLE is enabled
Maximum provisioning scope:
 Limit for CPC H05 is 100 MSU, 3 zAAPs, 3 zIIPs
  activation of 20/10 MSU
```

Figure 26-11 Reporting on the console

26.5 Recurring time conditions starting with z/OS V1R13

In releases prior to z/OS V1R13, a time condition is defined using a start time, a deadline time, and an end time. To define prime shifts, you must define one time condition for each day, each with the start and end time during the day. The other possibility is to define one time condition for the whole period, but activate additional capacity during night shifts or at the weekend.

The new recurring time condition defines, for a range of day between a start date and an end date, when the condition for each day is to start (start time) and end (end time). You can also specify a deadline time in between, after which no additional capacity may be added, but

activated capacity may be left active. Then, you can select the days within the week for which the times apply, for example only Monday to Friday, but not on Saturday and Sunday.

This new support in z/OS V1R13 is invoked by adding the definitions to the Capacity Provisioning policy using the Control Center along with:

- The Policy report, which reports the recurring time conditions
- The Policy status in the Control Center, which displays the recurring time conditions

The Time Conditions Panel now includes a tab for nonrecurring time conditions, and a tab for recurring time conditions; see Figure 26-12.

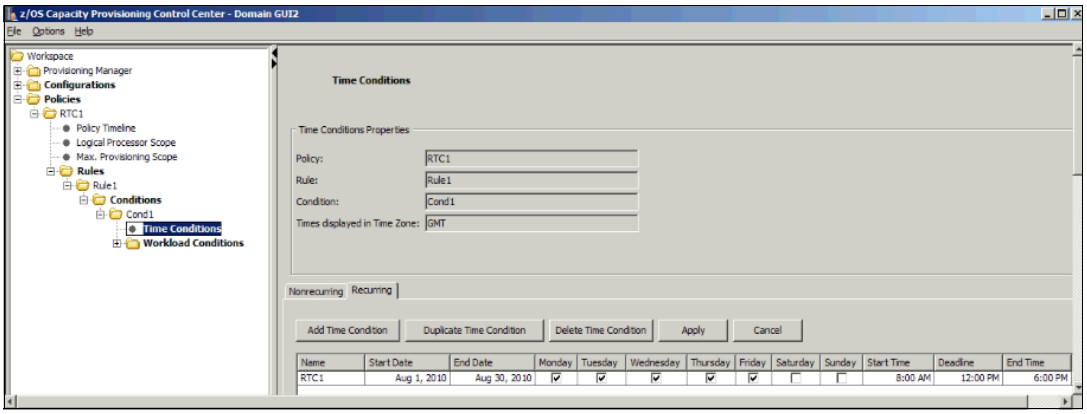


Figure 26-12 Recurring time conditions

Defining using a tab in the Time Conditions panel

Time conditions that are nonrecurring are now a tab in the same panel. The information reported is the same as in previous releases.

For recurring time conditions use the tab `Recurring`. Here you enter the name of the time condition, start date, end date, and allowed days of the week. For each allowed day, enter the start time, deadline time, and end time.

All dates and times are displayed using the time zone in the `Times displayed in Time Zone` field. When the time zone is changed, the difference to the previous time zone is added to the times shown. The dates may move to the day before or after the previous date. Accordingly, the selected days of the week may move by one position.

The active policy status panel now shows recurring time conditions. The Details consist of one group that contains name, status, and start and end date of the recurring time condition. A second group displayed contains the daily times, which are start time, deadline, and end time. A third group lists the selected days of the week.

Reporting on the console is shown in Figure 26-13. This new support of z/OS V1R13 allows you to define weekly recurring time conditions that occur for all allowed days in a large time range, for example, several months. As a benefit, fewer definitions are required and the Capacity Provisioning function is easier to define and maintain.

```
Recurring time condition RTCS001 is pending
 s/e/w days : 06/14/2010 /    06/15/2010 / XX-----
 s/d/e times:      15:20 / 17:20 / 18:20
```

Figure 26-13 Console report

26.6 Filtered workload report

Workload reports that produce monitoring reports about multiple systems or service classes can be confusing to understand, and the output is truncated if it exceeds 255 lines. With z/OS V1R13, you can now filter workloads using various criteria, such as:

- Workload with current PI readings
- Workload with current PI readings above the provisioning PI limit
- Workload of a specific system

The support is invoked by issuing the workload report with new or additional parameters:

- REPORT WORKLOAD TYPE=WithPIonly
- REPORT WORKLOAD TYPE=AbovePLonly
- REPORT WORKLOAD PLEX=xxx SYS=yyy

Such reports will only display service classes complying with the specified criteria. WithPIonly displays only service classes that have a current Provisioning Index (PI) reading. AbovePLonly displays only service classes that have a PI reading above the Provisioning Limit (PL) as specified in the CP policy, which means that the service class is actually suffering.

Both WithPIonly and AbovePIonly display detailed information (just as the TYPE=Detailed parameter does) as described here:

- PLEX=xxx SYS=yyy displays only the workload on a specific system. The syntax is the same as for enable/disable configuration plex=xxx sys=yyy.
- Filtering a specific system can be combined with the type-criteria filtering to display only workload on a single system that complies with the criteria of a type-filter such as AbovePLonly.

This enhancement to the Provisioning Manager command reduces output so that it fits on one window, while focusing on relevant information.

26.7 Windows 7 support

With z/OS V1R13, Windows NT and Windows Vista are still supported by the runtime environment. Support for Windows 7 is also added with Control Center behavior on Windows 7 being the same as on Windows Vista.

Support of Windows 7 for the Control Center is provided in 32-bit and 64-bit environments at installation and at run time.

26.8 SNMP removal

With z/OS V1R13, removal of SNMP as a supported protocol is announced in a future release. If you are currently using SNMP, keep the following migration considerations in mind regarding Capacity Provisioning:

- A CPO3060W message is issued by the Provisioning Manager automatically at startup if domain is configured to use SNMP protocol.
- An entry to z/OS Tracking Facility (CPO-W:SNMP usage <domain name>) is also created, as shown in Figure 26-14.

```
COMMAND INPUT ===> /DISPLAY OPDATA,TRACKING                    SCROLL ===
RESPONSE=IRD6
 CNZ1001I 14.09.28 TRACKING DISPLAY 184
 STATUS=ON       NUM=1    MAX=1000 MEM=n/a EXCL=0     REJECT=0
 ----TRACKING INFORMATION---- -VALUE-- JOBNAME  PROGNAME+OFF-- ASID NUM
 CPO-W:SNMP usage DOMAIN1           00 CPOSRV   *PATHNAM   2E0   5F   1
------------------------------------------------------------------------
TO REPORT THESE INSTANCES, SEND THIS MESSAGE VIA E-MAIL TO
CONSOLES@US.IBM.COM. FOR ADDITIONAL INFORMATION OR TO OBTAIN A CURRENT
EXCLUSION LIST, SEE APAR II13752.
```

Figure 26-14 SNMP removal in Tracking Facility

In the Tracking Facility entry text, <domain name> is substituted with the name of the actual domain that the Provisioning Manager is working for, as shown in the figure.

Tracking information can be activated using the **SETCON TRACKING=ON** command, and it can be viewed using the **DISPLAY OPDATA,TRACKING** command, giving clients timely awareness for migrating to BCPii.

Remove support for Java 5

Java Version 5 as a prerequisite is no longer supported. Support for Java Version 6 is provided as a runtime environment for the Provisioning Manager; it is already supported in earlier z/OS releases. An entry to the z/OS Tracking Facility indicates Java Version 5 usage on a previous release. It is no longer required to have Java Version 5 installed for Capacity Provisioning.

If you still have the Provisioning Manager installed and running with Java 5, you are required to migrate to Java 6 before you activate it in z/OS V1R13.

26.9 Capacity Provisioning hardware requirements

The following hardware is required to properly run Capacity Provisioning:

- One or more z10 or z196 servers.
- If temporary capacity is to be controlled by the Capacity Provisioning Manager, or if the manager is running in the confirmation or autonomic mode, or if provisioning actions are performed through CPM commands in either mode, then temporary capacity needs to be available.
  - This requires the CIU enablement feature, On/Off CoD enablement, and a valid On/Off CoD record for temporary general purpose processor, zAAP, or zIIP capacity.
- The workstation for the Control Center needs:
  - An Intel Pentium or equivalent processor with 512 MB memory (1 GB recommended).
  - Available disk space of 150 MB, running Microsoft Windows NT, Vista, or Windows 7, and a window resolution 1024 x 768 or higher.

Capacity Provisioning setup

See Appendix A.4, “Setting up Capacity Provisioning” on page 800 for details regarding setting up Capacity Provisioning.

Capacity Provisioning configuration dependencies and restrictions

Capacity Provisioning has the following dependencies and restrictions:

- Although observed systems must be running z/OS Release 9 or higher, other operating systems or the Coupling Facility Control Code (CFCC) can be in other LPARs.
- Observed systems running as guests under z/VM are not supported.
  - Run the Capacity Provisioning Manager on a system that is not running as a z/VM guest.
- An observed system may run in a shared or dedicated LPAR. An LPAR with dedicated processors can only generate demand for higher general purpose processor capacity levels. If the processor is not a sub-capacity processor, that is, if it is already operating at its maximum capacity level, then no additional demand will be recognized.
- For a dedicated LPAR, no demand for additional special purpose processors will be recognized.
- Demand for additional physical processors (as opposed to increased capacity level) for shared CP, zAAP, or zIIP processors can only be recognized if the current sum of logical processors is greater than or equal to the target number of physical processors in the respective pool.
  - Rationale: Capacity Provisioning currently does not configure reserved or offline processors online.
- Observed systems may have general purpose CPs, zAAPs zIIPs, or any combination of these configured. Other processor types in the physical configuration are allowable.
- Demand for zAAP processors can be recognized if at least one zAAP is already online to the system.
- Demand for zIIP processors can be recognized if at least one zIIP is already online to the system.
- The additional physical capacity will be distributed through IBM PR/SM™ and the operating system. In general, the additional capacity will be available to all LPARs.
  - Facilities such as defined capacity (soft capping) or initial capping (hard capping) can be used to control the use of capacity.
- For best results, avoid defining provisioning conditions for service classes that are associated with WLM resource groups for which a capacity maximum is in effect.

Prerequisites for installation

The following are prerequisites for installation:

- z/OS RMF must be set up and customized, including Distributed Data Server (DDS).
- z/OS CIM Server must be set up (z/OS Base element since z/OS V1R7).
- Customization of capacity provisioning must be performed as described in *z/OS MVS Capacity Provisioning User's Guide*, SA33-8299.

This customization is required on the following systems:

Observed z/OS systems These are the systems in one or multiple sysplexes that are to be monitored (see 26.2, “Capacity Provisioning Domain” on page 615).

Runtime systems These are the systems where the Capacity Provisioning Manager is running, or to which the server may fail over after server or system failures.

26.9.1 Capacity Provisioning communications

Capacity Provisioning communicates with the hardware to obtain information about the permanent and temporary capacity of the server. For this communication, two types of communication are supported:

SNMP A TCP/IP-based communication that requires a network connection from the z/OS system where the Provisioning Manager runs to the Support Element or the Hardware Management Console (HMC). For further information refer to *System z Application Programming Interfaces*.

> **Note:** If you use SNMP communication, then the SE and the HMC (when used) must be at driver level D73G or higher.

BCPii BCPii is a z/OS built-in communication that does not require a network connection; see Chapter 25, “Base Control Program internal interface” on page 581. If you use BCPii, refer to *z/OS MVS Programming: Callable Services for HLL*, or to the appropriate Preventive Service Planning (PSP) bucket for further information.

If more than one processor complex is to be controlled, an HMC is required. The available options are listed in Table 26-1 on page 628.

Table 26-1 SE or HMC requirement by Capacity Provisioning configuration

Configuration	Communication through System z API (SNMP)	Communication through BCPii
All observed systems and hosting system on one processor complex	HMC (preferred) or SE	SE
All observed systems on one processor complex; hosting system on another processor complex	HMC (preferred) or SE	HMC
Observed systems on multiple processor complexes	HMC	HMC

The Provisioning Manager must access a hardware console to obtain information about the available CPCs and their temporary capacity, and to activate and deactivate the temporary capacity, if required. You must specify access information for the HMC. The primary information is the protocol. It is specified using the configuration key Topology.Protocol.

The value can be either SNMP for the SNMP protocol, or INTERNAL for BCPii.

For BCPii the information looks as shown here:

```
# Topology settings
Topology.Protocol = INTERNAL
```

If the protocol is SNMP, you also need to specify the protocol. Additionally, you need to specify the host name or IP address of the HMC, and the community name under which all operations are performed. The HMC host address is specified using the configuration key Topology.Address. The community name is specified using the key Topology Community. The syntax for SNMP is shown in Figure 26-15.

```
# Topology settings
Topology.Protocol = SNMP
Topology.Address = HMC_address
Topology.Community = community_name
```

Figure 26-15 Topology setting for SNM

26.10 Capacity Provisioning software requirements

Capacity Provisioning requires the following software:

- Systems with, at minimum, z/OS Release 9 plus APAR OA20824 or above, can be monitored or used to run the Capacity Provisioning Manager.
- z/OS Resource Measurement Facility (RMF), an optional element of z/OS, must be enabled (or an equivalent product, including equivalent CIM RMF provider capability).
- The z/OS security product needs to support creation of passtickets (R_GenSec) and evaluation through the SAF interfaces.
- When using a security product other than IBM Security Server (Resource Access Control Facility, known as RACF), check with your vendor.
- BCP internal interface (BCPii) activated from the hosting system.
- IBM 31-bit SDK for z/OS, Java 2 Technology Edition, V6 (5655-R31); currently, no other levels are supported.
- Capacity Provisioning utilizes the CIM server (z/OS base element) and is used by z/OSMF.

26.10.1 Preparing the Provisioning Manager

Various prerequisites must be satisfied before you can successfully start the Provisioning Manager. These prerequisites include the runtime system and the systems that are observed by the Provisioning Manager. The runtime system may also be one of the observed systems. On the runtime system, complete the following tasks:

- Define data sets used for the runtime data.
- Set the configuration parameters to your chosen values.
- Create a started task procedure.
- Provide APF authorization.
- Define the security.
- Define a restart policy.

A.4, “Setting up Capacity Provisioning” on page 800, provides further details of an instance of this setup.

26.10.2 Migration and coexistence

Function and coexistence for recurring time conditions and capacity increments is available for z/OS V1R12 and V1R11 with APAR OA35284, as PTF UA59076 for V1R12 and PTF UA58996 for V1R11.

If you activate a policy with the new functionality, and you restart the Provisioning Manager on z/OS V1R10, then you need APAR OA35284, PTF UA58952 for V1R10, for coexistence. No recurring time conditions or capacity increments will be observed. Warnings for SNMP usage

and Java Version 5 usage are indicated using a message and Tracking Facility entry with APAR OA35284 for V1R12.

26.10.3 Coexistence of Control Center and policy versions

A V1R13 policy can be edited with the Control Center of an earlier version if the values for the capacity increments are the default values and the policy contains no recurring time condition.

All V1R13 policies can be edited with a V1R11 or V1R12 Control Center with coexistence support APAR OA35284 applied.

In all other cases, when a V1R13 policy is loaded into a Control Center of a previous release, then the error message is displayed as indicated in Example 26-1 on page 630.

Example 26-1 Incompatible policy

```
Message Text: CPO8110E Unable to parse the policy SAMPWLD.xml
```

This message is contained in the PTFs mentioned.

26.10.4 Installation of z/OS V1R13 Control Center

With z/OS V1R13, the location of the default workspace changes.

To continue using the existing (pre-z/OS V1R13) workspace, point to the desired workspace during the installation process; see Figure 26-16.

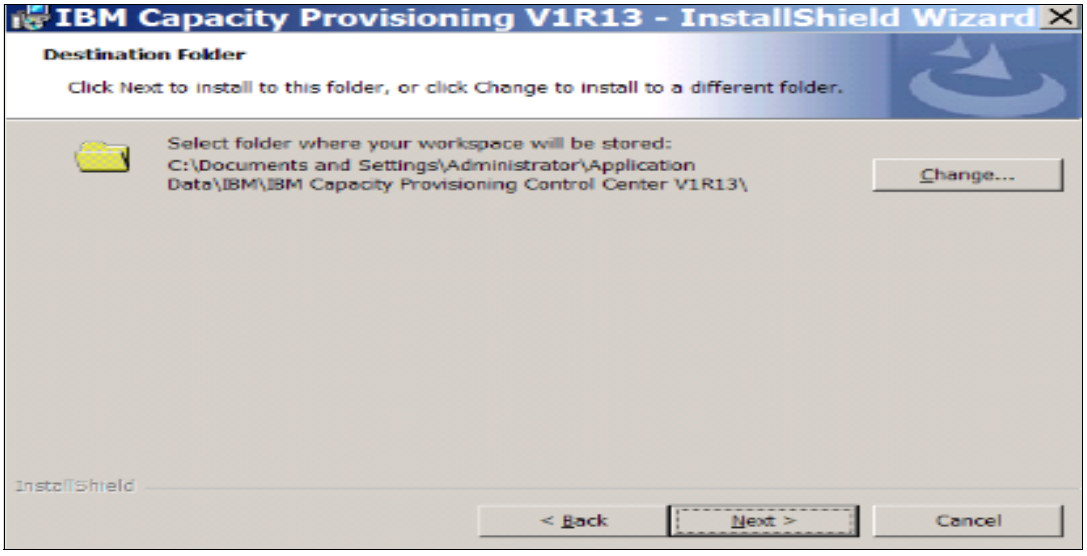


Figure 26-16 Destination folder

You can also switch the workspace at a later time by navigating to the desired workspace when the Control Center starts.

The actual directory depends on the Windows version that you are using.

- Windows XP

C:\Documents and Settings\user\Application Data\IBM\IBM Capacity Provisioning Control Center V1R13

- Windows Vista and Windows 7

  C:\Users\userName\AppData\Roaming\IBM\IBM Capacity Provisioning Control Center\V1R13\

Java 5 is no longer supported

As Java 5 is no longer supported, you need to migrate the Provisioning Manager run time by following these steps:

- Install IBM 31-bit SDK for z/OS, Java 2 Technology Edition, V6 (5655-R31).
- Change the LIBPATH variable in the ENV member of your Provisioning Manager parameter file to contain the installation directories of your Java V6 installation. See the following LIBPATH statement:

  ```
  LIBPATH=/usr/lib:/usr/lpp/java/J6.0/bin:usr/lpp/java/J6.0/bin/classic:/usr/lpp/cpo/lib
  ```

- If you install APAR OA35284 on V1R12, you will be informed of Java 5 usage by a Tracking Facility message:

  ```
  CPO-W:Java 1.5.0 <domain name>
  ```

26.11 Supported LPAR and z/OS environments

The IBM System z platform and z/OS allow for great flexibility. Capacity Provisioning supports a broad range of configurations, but certain configurations are not supported or are restricted. This section summarizes the restrictions that apply to Capacity Provisioning.

- Observed systems must be running z/OS Release 9 or higher. Other operating systems, or the Coupling Facility Control Code (CFCC), may be active in other LPARs.
- Observed systems running as guests under z/VM are not supported. Avoid using a z/OS system running as a z/VM guest to run the Provisioning Manager.
- An observed system may run in a shared or dedicated LPAR. An LPAR with dedicated processors, however, can only generate demand for higher general purpose processor capacity levels.

  If the processor complex is not a subcapacity model but is already operating at its maximum capacity level, no additional demand will be recognized. If the LPAR is dedicated, then no demand for additional special purpose processors will be recognized.
- Demand for additional physical processors (as opposed to increased capacity level) for shared CP, zAAP, or zIIP processors can only be recognized if the current sum of logical processors is greater than the number of physical processors in the respective processor pool, or the Capacity Provisioning policy allows for configuring logical processors online.
- Observed systems may have general purpose CPs, zAAPs, and zIIPs configured.
- The additional physical capacity provided through Capacity Provisioning will be distributed through PR/SM and the operating systems. In general the additional capacity will be available to all LPARs, but facilities such as defined capacity (soft capping) or initial capping (hard capping) can be used to control the use of capacity.
- Avoid defining provisioning conditions for service classes associated with resource groups for which a capacity maximum is in effect.

- If a system has IRD Vary CPU Management turned on, then no logical processors are configured by Capacity Provisioning.
- Logical processors are managed only for systems with shared processors.
- Avoid having monitored systems use “initial capping” (hard capping).

26.12 Capacity Provisioning Manager summary

Capacity Provisioning Manager provides faster and more reliable methods for activating On/Off Capacity on Demand. Note the following features of this z/OS V1R13 component:

- Manual mode allows activation and deactivation of physical general purpose capacity, zAAPs, and zIIPs.
- Analysis mode provides suggestions about how to address capacity bottlenecks.
- User-defined capacity increments provide an improved method of controlling the amount of capacity to be added.
- Recurring time conditions provide an improved method of controlling when additional capacity can be added.
- Confirmation and automation modes automate recognition and enable capacity changes with or without operator confirmation.
- The Provisioning Manager command allows filtering of the output of the workload report.
- Optimization of activated resources is enhanced.
- The solution integrates z10 and z196 enhancements, along with new and existing z/OS components Capacity Provisioning FMID, WLM, RMF, and CIM.
- z/OSMF V1R13 integrates this enhanced function.
- Various degrees of automation are now available, ranging from manual to autonomic.

27

System SSL enhancements

This chapter describes the System SSL enhancements for z/OS V1R13. The enhancements are:

- Elliptic Curve Cryptography (ECC) certificate creation and key agreement
- ECC support for TLS
- ECC key reference by ICSF key label

Note: ECC is an emerging public-key algorithm. It provides the same or better security with much shorter key lengths than RSA keys. It is appropriate for use in resource-constrained environments such as smart cards and mobile phones, which might have limited space to hold storage keys.

ECC augments end-to-end encryption for data in flight by helping to maintain data privacy and prevent data leakage of sensitive information when providing the next generation of security-level requirements.

This implementation will allow exploiters to keep current with industry standards protocols and security features. It continues to ensure interoperability between client and server applications.

27.1 ECC certification and key agreement

Elliptic Curve Cryptography (ECC) offers significant benefits over other asymmetric cryptographic algorithms, particularly when providing the next generation of security-level requirements. To take advantage of these benefits, System SSL needs to support the creation of ECC-based certificates.

With z/OS V1R13, System SSL now has the functionality to generate an ECC-based public/private key pair for use in an x.509 certificate. This certificate then can be used in TLS session negotiations. ECC offers equivalent security with smaller key sizes than RSA.

By using this new support, users can:

- Create ECC certificates or certificate requests through the **gskkyman** utility
- Create ECC certificates or certificate requests through the Certificate Management APIs
- Store the certificates into key database files or PKCS#11 tokens

Refer to 27.6, “System SSL ECC example” on page 639 to learn how to use this new function.

27.2 ECC support for Transport Layer Security (TLS)

System SSL has been updated to enable TLS V1.0 and TLS V1.1 handshakes to utilize ECC cipher suites and digital certificates during the secure connection negotiation. System SSL will now support twenty new RCC cypher suites. The RFC that describes the implementation of ECC for TLS is RC 4492. It can be found at the following site:

`http://www.ietf.org/rfc.html`

Users can specify ciphers that use the Elliptic Curve Diffie-Hellman (ECDH) key agreement scheme to establish TLS secure connections. The key agreement can either use fixed or ephemeral keys. Fixed ECDH certificates and Elliptic Curve Digital Signature Authentication (ECDSA) can be used to for authenticate TLS partners.

By default, ECC-based TLS connections are not negotiated. For TLS V1.0 or V1.1 connections to negotiate using ECC, the System SSL application must specify:

- An ECC cipher
- An appropriate certificate to be used during the negotiation
- Optionally the supported elliptic curves, when acting as a client

The ECC ciphers are not included in the default cipher list.

27.2.1 Cipher suites

Prior to the ECC support, ciphers were specified through the GSL_V3_CIPHER_SPECS environment variable or connection attribute. TLS RFC 2246 specifies that cipher suites are identified either by a 2-byte binary value or by four hexadecimal digits; see Figure 27-1.

```
TLS_RSA_WITH_AES_256_CBC_SHA = {0x00, 0x35};
```

Figure 27-1 Cipher suite sample

A shorthand notation can be used for cipher suite definition where the initial two characters of the cipher suite specification have always been assumed to be zero (0x00); see Figure 27-2.

```
export GSK_V3_CIPHER_SPECS=35
```

Figure 27-2 Cipher suite shorthand notation sample

The ECC cipher suites defined by RFC 4492 (ECC Cipher Suites for TLS) have their first byte set to x'C0'; see Figure 27-3.

```
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA = {0xC0, 0x14}
```

Figure 27-3 ECC cipher suite sample

The two-character shorthand notation cannot be used to specify ECC cipher suites. A new buffer ID (environment variable) has been added to system SSL called GSK_V3_CIPHER_SPECS_EXPANDED.

Users may use GSK_V3_CIPHER_SPECS_EXPANDED to define a cipher specification string using 4-character cipher suite definitions; see Figure 27-4.

```
export GSK_V3_CIPHER_SPECS_EXPANDED=C014C005002F
```

Figure 27-4 GSK_V3_CIPHER_SPECS-EXPANDED sample

A new enum (enumerated type) ID GSK_V3_CIPHERS has also been added to allow users to specify which cipher specification string will be used by their application.

The options GSK_V3_CIPHER_SPECS_EXPANDED and GSK_V3_CIPHERS can be specified at either the environment level or at the connection level.

Supported ECC cipher suites

The supported ECC ciphers with the RFC 4492 symbolic names are listed here:

- Ephemeral Elliptic Curve Diffie-Hellman
  - C006 TLS_ECDHE_ECDSA_WITH_NULL_SHA
  - C007 TLS_ECDHE_ECDSA_WITH_RC4_128_SHA
  - C008 TLS_ECDHE_ECDSA_WITH_3DES_EDE_CBC_SHA 1
  - C009 TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA 1
  - C00A TLS_ECDHE_ECDSA_WITH_AES_256_CBC_S 1
  - C010 TLS_ECDHE_RSA_WITH_NULL_SHA
  - C011 TLS_ECDHE_RSA_WITH_RC4_128_SHA
  - C012 TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA 1
  - C013 TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA 1
  - C014 TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA 1
- Fixed Elliptic Curve Diffie-Hellman
  - C001 TLS_ECDH_ECDSA_WITH_NULL_SHA
  - C002 TLS_ECDH_ECDSA_WITH_RC4_128_SHA
  - C003 TLS_ECDH_ECDSA_WITH_3DES_EDE_CBC_SHA 1
  - C004 TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA 1
  - C005 TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA 1
  - C00B TLS_ECDH_RSA_WITH_NULL_SHA
  - C00C TLS_ECDH_RSA_WITH_RC4_128_SHA

– C00D TLS_ECDH_RSA_WITH_3DES_EDE_CBC_SHA 1
– C00E TLS_ECDH_RSA_WITH_AES_128_CBC_SHA 1
– C00F TLS_ECDH_RSA_WITH_AES_256_CBC_SHA 1

1 These cipher suites are supported when running in a mode designed for FIPS. The ICSF has to be configured to run in FIPSMODE. In FIPSMODE, only the strongest cipher suites are allowed.

Client specification of supported elliptic curves can now be specified with a new environment variable named GSK_CLIENT_ECURVE_LIST. This can be done at either the environment level or at the connection level. When the list is not mentioned for the connection, all supported curves will be used. The supported elliptic curves and their names in each of the standard organizations are shown in Table 27-1.

Table 27-1 Supported elliptic curves

IANA Elliptic Curve Enumerators	SECG	ANSI x.92	NIST
0019	secp192r1	prime192v1	NIST P-192
0021	secp224r1		NIST P-224
0023	secp256r1	prime256v1	NIST P-256
0024	secp384r1		NIST P-384
0025	secp512r1		NIST P-512

Note: 27.6, “System SSL ECC example” on page 639 explains how to use the cipher suites.

27.3 ECC key reference by ICSF key label

With z/OS V1R13, System SSL has been enhanced to support private keys stored in the ICSF PKDS data set. RACF RACDCERT has now the capability to create and add certificates with their ECC keys stored in the PKDS data set. This new functionality is described in 27.6, “System SSL ECC example” on page 639.

27.4 RACF and hardware ECC support for RACDCERT

In z/OS V1R12, through ICSF PKCS#11 support, RACF and PKI Services started the support on Elliptic Curve Crypto (ECC)-based certificates. However, the implementation was based on software only, which cannot provide the level of protection for keys as the hardware.

With z/OS V1R13, RACF can now generate keys with the ECC algorithm through the hardware using a new crypto function that exploits the new Crypto Express 3 Cryptographic Coprocessor (CEX3C) and stores the key in ICSF public key data set (PKDS) and is now protected by the master key.

The method of naming various key types in RACF was not clear. For example, key types ICSF, Non-ICSF, and PCICC are all RSA keys. Adding support for more key types made the

situation worse. The key types are now restructured by the **RACDCERT** command to make them more intuitive and more consistent for input and output.

There are four public/private key types supported in RACF: RSA, DSA, NIST ECC, and Brainpool ECC. Except for DSA, you have a choice to generate/store the key in ICSF PKDS, which is protected by the master key. Both key types are used and they are stored as listed in Table 27-2.

Table 27-2 Storing key types

Input key type in RACDCERT GENCERT	Key type displayed in RACDCERT LIST
NISTECC	Key Type: NISTECC (no PKDS label entry)
NISTECC(PKDS)	Key Type: NISTECC PKDS Label: <system generated label>
NISTECC(PKDS(<specified label>))	Key Type: NISTECC PKDS Label: <specified label>
BPECC	Key Type: BPECC (no PKDS label entry)
BPECC(PKDS)	Key Type: BPECC PKDS Label: <system generated label>
BPECC(PKDS(<specified label>))	Key Type: BPECC PKDS Label: <specified label>
RSA = no key type specified	Key Type: RSA (no PKDS label entry)
RSA(PKDS) = PCICC	Key Type: RSA PKDS Label: <system generated label>
RSA(PKDS(<specified label>))	Key Type: RSA PKDS Label: <specified label>

This new support allows you to expand your use of certificates with ECC keys protected by hardware, and simplify the key type usage using the **RACDCERT** command. Consolidating key types will make it more comprehensive, and permit easy expansion for key types in the future.

For the RACF **RACDCERT** command, a new sub keyword PKDS is added to indicate the key is a hardware key. Note the following examples:

- Generating a certificate with an NIST ECC key stored in PKDS with a system-generated key label is shown in Figure 27-5.

```
RACDCERT GENCERT SUB(CN('Company A')) WITHLABEL('New NISTECC cert')
NISTECC(PKDS)
```

Figure 27-5 RACF RACDCERT system-generated label sample

- Generating a certificate with a Brainpool ECC key stored in PKDS with key label BPECCFORA is shown in Figure 27-6.

```
RACDCERT GENCERT SUB(CN('Company A')) WITHLABEL('New BPECC cert')
BPECC(PKDS(BPECCFORA))
```

Figure 27-6 RACF RACDCERT user-generated label sample

The RACF TSO interface panels are updated to support this new function. The new format of the Generate a Digital Certificate panel is shown in Figure 27-7.

```
RACF - Generate a Digital Certificate
COMMAND ===>

                                                                More:   - +

   Organization(in quotes): (ex: 'IBM' )
   ____________________________________________________________________

   Locality(in quotes): (ex: 'Poughkeepsie' )
   ____________________________________________________________________

   State/Province(in quotes): (ex: 'New York'  )
   ____________________________________________________________________

   Country(in quotes): (ex: 'US'  )
   ____________________________________________________________________

 Enter the decimal size of the private key:

   ______  (Default is 1024 for non ECC keys, 192 for ECC keys)

 Select the key type to be generated:

   _  RSA(default)
   _  RSA in PKDS with an optional PKDS label or *:
      ____________________________________________________________
   _  RSA Modulus-Exponent in PKDS with an optional PKDS label or *:
      ____________________________________________________________
   _  DSA
   _  NIST ECC
   _  NIST ECC in PKDS with an optional PKDS label or *:
      ____________________________________________________________
   _  Brainpool ECC
   _  Brainpool ECC in PKDS with an optional PKDS label or *:
      ____________________________________________________________



 Enter when the certificate is valid:

   NOTBEFORE Date: YYYY-MM-DD  (Default is current local date)
   NOTBEFORE Time: HH:MM:SS    (Default is 00:00:00)
```

Figure 27-7 RACF - Generate a Digital Certificate panel

Note that 27.6, “System SSL ECC example” on page 639, illustrates the RACF **RACDCERT** command on the RACF panels.

27.5 Prerequisites, migration, and coexistence considerations

The software prerequisites are:

- ICSF Web Deliverable #10 (HCR7780) has to be installed and operating on the system.
- RACF (HRF7780) has to be installed and operating on the system (or equivalent ESM that supports elliptic curves in ICSFs PKDS).

The hardware prerequisite is a zEnterprise Server (2817, z196) with a Crypto Express3 Coprocessor Card (feature 0864).

The toleration APAR OA34156 is needed on z/OS 1.11 and 1.12 systems running in a sysplex with z/OSV1R13. This toleration APAR is needed for proper functioning of the sysplex-wide session ID caching function within System SSL.

27.6 System SSL ECC example

SSL connections use public/private key mechanisms for authenticating each side of the SSL session and for agreeing on bulk encryption keys to be used for the SSL session. To use public/private key mechanisms (known as PKIs), public/private key pairs must be generated. In addition, X.509 certificates (which contain public keys) might need to be created, or certificates must be requested, received, and managed.

System SSL supports the following methods for managing PKI private keys and certificates:

- Using the z/OS shell-based program called **gskkyman**

  This creates, fills in, and manages either a z/OS file or z/OS PKCS #11 token that contains PKI private keys, certificate requests, and certificates. The z/OS file is called a “key database” and, by convention, has a file extension of .kdb.

- Using the z/OS Security Server (RACF) **RACDCERT** command

  RACDCERT installs and maintains PKI private keys and certificates in RACF. See *z/OS Security Server RACF Command Language Reference*, SA22-7687, for details about the **RACDCERT** command. RACF supports multiple PKI private keys and certificates to be managed as a group. These groups are called “”key rings or “z/OS PKCS #11 tokens.”

Using RACF key rings or z/OS PKCS #11 tokens is the preferred method for managing PKI private keys and certificates for System SSL.

The System SSL application uses the GSK_KEYRING_FILE parameter of the gsk_attribute_set_buffer() API or the GSK_KEYRING_FILE environment variable to specify the locations of the PKI private keys and certificates to System SSL.

- If you are using a z/OS key database, the key database file name is passed in this parameter.
- If you are using a RACF key ring or z/OS PKCS #11 token, the name of the key ring or token is passed in this parameter.

The following examples are discussed:

- Creating a certificate authority (CA) certificate using the ECC option
- Creating a certificate request for a client and a server and signing it with the CA certificate
- Configuring the HTTP server and a client to use those certificates in a TLS session

The **gskkyman** utility is used together with the RACF **RACDCERT** command to create the certificates. The RACF certificates are used only to configure the servers and clients for the example.

27.6.1 Checking the environment

In the system used in this example, the System SSL, RACF, and ICSF are already configured and running to support the demonstration. To perform several functions, various RACF authorizations had to be implemented so that commands and resource access can be performed without issue.

Checking System SSL cipher suites

To determine what System SSL cipher suites are available in our environment we issued the **F GSKSRVR,DISPLAY CRYPTO** command; see Figure 27-8.

```
F GSKSRVR,DISPLAY CRYPTO 1
ICH408I USER(GSKSRVR ) GROUP(SYS1    ) NAME(####################) 2
  CSF1TRL CL(CSFSERV )
  WARNING: INSUFFICIENT AUTHORITY - TEMPORARY ACCESS ALLOWED
  FROM * (G)
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
ICH408I USER(GSKSRVR ) GROUP(SYS1    ) NAME(####################) 2
  CSFIQA CL(CSFSERV )
  WARNING: INSUFFICIENT AUTHORITY - TEMPORARY ACCESS ALLOWED
  FROM * (G)
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
ICH408I USER(GSKSRVR ) GROUP(SYS1    ) NAME(####################) 2
  CSFIQA CL(CSFSERV )
  WARNING: INSUFFICIENT AUTHORITY - TEMPORARY ACCESS ALLOWED
  FROM * (G)
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
GSK01009I Cryptographic status 005
Algorithm       Hardware    Software
DES                 56          56
3DES               168         168
AES                256         256
RC2                 --         128
RC4                 --         128
RSA Encrypt       4096        4096
RSA Sign          4096        4096
DSS                 --        1024
SHA-1              160         160
SHA-2              512         512
ECC                521         521 3
```

Figure 27-8 F GSKSRVR,DISPLAY CRYPTO command

Note the following points in the figure:

1 The command **F GSKSRVR, DISPLAY CRYPTO** showed what cipher algorithms were available and whether they are performed by hardware or by software.

2 This shows the RACF profile authorizations needed for this command work properly: CLASS CSFSERV and profiles CSF1TRL (PKCS11 token record list callable service) and

CSFIQA (ICSF Query Algorithm callable service). Several RACF profiles and classes were defined in warning mode to enable us to trace the proper authorizations levels required to perform various functions.

3 The ECC cipher algorithms were available by software and hardware.

Checking ICSF startup options and initialization messages

Figure 27-9 shows the ICSF startup options from the CSFPRM00 used in our environment.

```
CKDSN(CSF.CSFCKDS)
PKDSN(CSF.CSFPKDS)
TKDSN(CSF.CSFTKDS)
COMPAT(NO)
SSM(NO)
KEYAUTH(YES)
CHECKAUTH(YES)
TRACEENTRY(10000)
USERPARM(USERPARM)
REASONCODES(ICSF)
FIPSMODE(YES,FAIL(YES))
```

Figure 27-9 ICSF startup options

The ICSF initialization messages are shown in Figure 27-10.

```
CSFO0166 DEFAULT CICS WAIT LIST WILL BE USED.
CSFM608I A CKDS KEY STORE POLICY IS DEFINED.
CSFM608I A PKDS KEY STORE POLICY IS DEFINED.
CSFM610I GRANULAR KEYLABEL ACCESS CONTROL IS ENABLED.
CSFM611I XCSFKEY EXPORT CONTROL FOR AES IS ENABLED.
CSFM611I XCSFKEY EXPORT CONTROL FOR DES IS ENABLED.
CSFM612I PKA KEY EXTENSIONS CONTROL IS DISABLED.
CSFM015I FIPS 140 SELF CHECKS FOR PKCS11 SERVICES SUCCESSFUL.
CSFM111I CRYPTOGRAPHIC FEATURE IS ACTIVE. CRYPTO EXPRESS3 ACCELERATOR H00,
SERIAL NUMBER N/A.
CSFM530I I/O INTERRUPT SUPPORT HAS BEEN ENABLED FOR CRYPTO EXPRESS3 ACCELERATOR
H00, SERIAL NUMBER N/A.
CSFM530I I/O INTERRUPT SUPPORT HAS BEEN ENABLED FOR CRYPTO EXPRESS3 COPROCESSOR
SERIAL NUMBER 90003937.
CSFM530I I/O INTERRUPT SUPPORT HAS BEEN ENABLED FOR CRYPTO EXPRESS3 COPROCESSOR
SERIAL NUMBER 90003961.
CSFM129I MASTER KEY DES ON CRYPTO EXPRESS3 COPROCESSOR G06, SERIAL NUMBER
90003937, IS CORRECT.
CSFM129I MASTER KEY RSA ON CRYPTO EXPRESS3 COPROCESSOR G06, SERIAL NUMBER
90003937, IS CORRECT.
CSFM129I MASTER KEY AES ON CRYPTO EXPRESS3 COPROCESSOR G06, SERIAL NUMBER
90003937, IS CORRECT.
CSFM129I MASTER KEY ECC ON CRYPTO EXPRESS3 COPROCESSOR G06, SERIAL NUMBER
90003937, IS CORRECT.
CSFM129I MASTER KEY DES ON CRYPTO EXPRESS3 COPROCESSOR G07, SERIAL NUMBER
90003961, IS CORRECT.
CSFM129I MASTER KEY RSA ON CRYPTO EXPRESS3 COPROCESSOR G07, SERIAL NUMBER
90003961, IS CORRECT.
CSFM129I MASTER KEY AES ON CRYPTO EXPRESS3 COPROCESSOR G07, SERIAL NUMBER
90003961, IS CORRECT.
CSFM129I MASTER KEY ECC ON CRYPTO EXPRESS3 COPROCESSOR G07, SERIAL NUMBER
90003961, IS CORRECT.
CSFM111I CRYPTOGRAPHIC FEATURE IS ACTIVE. CRYPTO EXPRESS3 COPROCESSOR G06,
SERIAL NUMBER 90003937.
CSFM111I CRYPTOGRAPHIC FEATURE IS ACTIVE. CRYPTO EXPRESS3 COPROCESSOR G07,
SERIAL NUMBER 90003961.
CSFM400I CRYPTOGRAPHY - SERVICES ARE NOW AVAILABLE.
CSFM130I CRYPTOGRAPHY - RSA SERVICES ARE AVAILABLE.
CSFM127I CRYPTOGRAPHY - AES SERVICES ARE AVAILABLE.
CSFM130I CRYPTOGRAPHY - ECC SERVICES ARE AVAILABLE.
CSFM126I CRYPTOGRAPHY - FULL CPU-BASED SERVICES ARE AVAILABLE.
CSFM001I ICSF INITIALIZATION COMPLETE
```

Figure 27-10 ICSF initialization messages

27.6.2 Creating the certificates using the gskkyman utility

The **gskkyman** utility is a z/OS shell-based program that creates, fills in, and manages a z/OS file or z/OS PKCS #11 token that contains PKI private keys, certificate requests, and certificates. As mentioned, z/OS file is known as a key database and has a file extension of .kdb. There is also an .rdb file that is a counterpart to the .kdb file.

Using the interface to the gskkyman utility: The interface to the **gskkyman** utility, although command line-based, is an interactive dialog between you as the user and the program. At each step the interactive **gskkyman** program prompts you with one or more lines of output, and it expects a numeric choice to be supplied as input at the prompt.

After you make a choice, the **gskkyman** program prompts you for the individual pieces of information needed to fulfill the request. You are prompted for each piece of information.

Many times there is a default choice that is listed between parentheses ( ) at the end of the command prompt.

- If the default choice is acceptable, press Enter to select the default.
- If a choice other than the default is desired, enter the value at the prompt and press Enter.

If a value is entered that is outside of the acceptable range of inputs, you will be prompted again for the information.

The certificates we created to be used in our example used the **gskkyman** utility in the z/OS UNIX System Services environment. To begin the demonstration, follow these steps:

- Start in the OMVS session using the **OMVS** command in TSO ISPF.
- In the OMVS session, run the **gskkyman** utility by typing `gskkyman` on the command line.

The output in the OMVS session is shown in Figure 27-11.

```
       Database Menu

   1 - Create new database
   2 - Open database
   3 - Change database password
   4 - Change database record length
   5 - Delete database
   6 - Create key parameter file
   7 - Display certificate file (Binary or Base64 ASN.1 DER)

  11 - Create new token
  12 - Delete token
  13 - Manage token
  14 - Manage token from list of tokens

   0 - Exit program

Enter option number:
```

Figure 27-11 **gskkyman** *utility*

- Select option **2 - Open the database**
  - Using an existing database, ecctest1, open the database and follow the instructions shown in Figure 27-12.

```
Enter option number: 2
Enter key database name (press ENTER to return to menu):
===> ecctest1
Enter database password (press ENTER to return to menu):
( password entered )
       Key Management Menu
       Database: /u/rodolfi/ecctest1
       Expiration: None
   1 - Manage keys and certificates
   2 - Manage certificates
   3 - Manage certificate requests
   4 - Create new certificate request
   5 - Receive requested certificate or a renewal certificate
   6 - Create a self-signed certificate
   7 - Import a certificate
   8 - Import a certificate and a private key
   9 - Show the default key
  10 - Store database password
  11 - Show database record length
   0 - Exit program
Enter option number (press ENTER to return to previous menu): 4
```

Figure 27-12 Database is opened, select option 4 to create new certificate

As shown, option 4 was selected to create a new certificate request. Next, select option **5 - Certificate with an ECC key**; see Figure 27-13.

```
       Certificate Type

   1 - Certificate with 1024-bit RSA key
   2 - Certificate with 2048-bit RSA key
   3 - Certificate with 4096-bit RSA key
   4 - Certificate with 1024-bit DSA key
   5 - Certificate with an ECC key

Enter certificate type (press ENTER to return to menu): 4
Enter request file name (press ENTER to return to menu): 5
Enter label (press ENTER to return to menu):
```

Figure 27-13 Choose certificate with an ECC key

Figure 27-14 is displayed next. Select option **5** and type in the request file name, in this example: `server`.

```
       NIST Recommended Curve Type
   1 - secp192r1
   2 - secp224r1
   3 - secp256r1
   4 - secp384r1
   5 - secp521r1
Select NIST recommended curve type (press ENTER to return to menu): 5
Enter request file name (press ENTER to return to menu):
===> server
```

Figure 27-14 Choose NIST Recommended Curve Type

Figure 27-15 is displayed. Type in a label, in this example: `ca`.

```
Enter request file name (press ENTER to return to menu): server
Enter label (press ENTER to return to menu):
===> ca
```

Figure 27-15 Enter label

Figure 27-16 is displayed. Type in a subject name, in this example: `server1`.

```
Enter subject name for certificate
  Common name (required):
 ===> server1
```

Figure 27-16 Enter a subject name for certificate

Figure 27-17 is displayed. Type in the Organizational unit, in this example: `ITSO`.

```
 Common name (required): server2
  Organizational unit (optional):
 ===> ITSO
```

Figure 27-17 Enter Organizational unit

Figure 27-18 is displayed showing a certificate has been created.

```
Organizational unit (optional): itso
Organization (required): itso
City/Locality (optional):
State/Province (optional):
===> us
Country/Region (2 characters - required): us
Enter 1 to specify subject alternate names or 0 to continue:
 ===> 0
Please wait .....
Certificate created.
```

Figure 27-18 Certificate created

```
Enter label (press ENTER to return to menu): ca
Enter subject name for certificate
  Common name (required): ca
  Organizational unit (optional):
  Organization (required): itso
  City/Locality (optional):
  State/Province (optional):
  Country/Region (2 characters - required): us
Enter number of days certificate will be valid (default 365):
Enter 1 to specify subject alternate names or 0 to continue: 0
Please wait .....
Certificate created.
```

Figure 27-19 CA certificate created

At this point, the CA certificate has been created. It will be used to sign the server and the client certificate requests.

Note: Our database was created with the NIST option. Because of this option, when choosing the recommended curve type, the Brainpool option is not available. In a database without the NIST option, the Brainpool curves will show for selection.

The certificate requests have been created for the server and the client. Two files, named `server` and `client`, have been created in the directory; see Figure 27-20.

```
ROGERS @ SC75:/u/rodolfi>ls -l
total 256
-rw-r--r--   1 RODOLFI  SYS1         806 Apr 21 15:35 cliend.signed
-rw-r--r--   1 RODOLFI  SYS1         582 Apr 21 15:34 client
-rw-------   1 RODOLFI  SYS1       70088 Apr 22 12:08 ecctest1
-rw-------   1 RODOLFI  SYS1          88 Apr 21 16:09 ecctest1.rdb
-rw-------   1 RODOLFI  SYS1         129 Apr 22 12:08 ecctest1.sth
-rw-r--r--   1 RODOLFI  SYS1         582 Apr 21 16:03 server
-rw-r--r--   1 RODOLFI  SYS1         806 Apr 21 16:09 server.signed
```

Figure 27-20 Files created by gskkyman

27.6.3 Signing a certificate using the command line

To sign a certificate request, the **gskkyman** command must be issued using command-line options. The **gskkyman** command must be issued with these parameters:

```
gskkyman -g -x num-of-valid-days -cr certificate-request-file-name -ct
signed-certificate-file-name -k CA-key-database-file-name -l label
```

In our example we issue the **gskkyman** command twice, once for each certificate request that was created; see Figure 27-21.

```
RODOLFI:/u/rodolfi: >gskkyman -g -x 365 -cr client -ct client.signed -k
ecctest1 -l ca
Enter database password (press ENTER to cancel):

Certificate created.
RODOLFI:/u/rodolfi: >gskkyman -g -x 365 -cr server -ct server.signed -k
ecctest1 -l ca
Enter database password (press ENTER to cancel):

Certificate created.
```

Figure 27-21 Signing the certificates

Next, we received the signed certificates into the database; see Figure 27-22.

```
       Key Management Menu

       Database: /u/rodolfi/ecctest1
       Expiration: None

   1 - Manage keys and certificates
   2 - Manage certificates
   3 - Manage certificate requests
   4 - Create new certificate request
   5 - Receive requested certificate or a renewal certificate
   6 - Create a self-signed certificate
   7 - Import a certificate
   8 - Import a certificate and a private key
   9 - Show the default key
  10 - Store database password
  11 - Show database record length

   0 - Exit program

Enter option number (press ENTER to return to previous menu): 5

Enter certificate file name (press ENTER to return to menu): client.signed

Certificate received.

Press ENTER to continue.

Enter option number (press ENTER to return to previous menu): 5

Enter certificate file name (press ENTER to return to menu): server.signed

Certificate received.

Press ENTER to continue.
```

Figure 27-22 Receiving the signed certificates

Verifying the certificates

Then we verified the certificates, as shown in Example 27-1.

Example 27-1 Verifying the certificates

```
       Key Management Menu

       Database: /u/rodolfi/ecctest1
       Expiration: None

   1 - Manage keys and certificates
   2 - Manage certificates
   3 - Manage certificate requests
   4 - Create new certificate request
   5 - Receive requested certificate or a renewal certificate
   6 - Create a self-signed certificate
   7 - Import a certificate
   8 - Import a certificate and a private key
   9 - Show the default key
  10 - Store database password
  11 - Show database record length

   0 - Exit program

Enter option number (press ENTER to return to previous menu): 1

       Key and Certificate List

       Database: /u/rodolfi/ecctest1

   1 - ca
   2 - client
   3 - server

   0 - Return to selection menu

Enter label number (ENTER to return to selection menu, p for previous list): 1

       Key and Certificate Menu

       Label: ca

   1 - Show certificate information
   2 - Show key information
   3 - Set key as default
   4 - Set certificate trust status
   5 - Copy certificate and key to another database/token
   6 - Export certificate to a file
   7 - Export certificate and key to a file
   8 - Delete certificate and key
   9 - Change label
  10 - Create a signed certificate and key
  11 - Create a certificate renewal request

   0 - Exit program
```

```
Enter option number (press ENTER to return to previous menu): 1

                        Certificate Information

                Label: ca
            Record ID: 15
      Issuer Record ID: 15
              Trusted: Yes
              Version: 3
        Serial number: 4db08127000d0f44
          Issuer name: ca
                       itso
                       US
         Subject name: ca
                       itso
                       US
       Effective date: 2011/04/21
      Expiration date: 2012/04/20
   Signature algorithm: ecdsaWithSha512
     Issuer unique ID: None
    Subject unique ID: None
  Public key algorithm: ecPublicKey
      Public key size: 521
           Public key: 04 00 9F 36 5F 60 A0 00 DC 5D 97 CA 20 50 6B B3
                       6B 49 26 A2 9C B9 2B AB 2A 76 B6 9C 78 C8 DA 32
                       09 07 83 57 CC FC A1 92 3B C0 B9 4F 1B 73 FF 4D
                       1C D6 DF 73 4F 14 B0 10 C7 FE A1 B6 92 91 7C 08
                       A7 B3 CE 01 2B 8E CD FE D5 C5 95 C0 C4 FA 02 C1
                       05 D1 89 E8 65 CE E4 A9 02 B6 D7 D6 97 1B 87 59
                       27 7C FE 90 8E FF D4 C8 77 C3 37 F3 1D 00 AA B0
                       55 E8 C1 AF 6C 36 F8 1F F7 49 86 79 4E CD 81 E0
                       AD D8 1C 9F 1E
  Number of extensions: 4

Certificate Information

                Label: client
            Record ID: 16
      Issuer Record ID: 15
              Trusted: Yes
              Version: 3
        Serial number: 4db087120002a265
          Issuer name: ca
                       itso
                       US
         Subject name: client
                       itso
                       US
       Effective date: 2011/04/21
      Expiration date: 2012/04/20
   Signature algorithm: ecdsaWithSha512
     Issuer unique ID: None
    Subject unique ID: None
  Public key algorithm: ecPublicKey
      Public key size: 521
```

```
          Public key: 04 00 2B 43 D4 CA 60 AC 49 DF 6B BC 00 67 0C 32
                      04 8F 0A 32 5A 12 9B 73 9A 7F F0 CE 2A 76 3C C2
                      27 07 30 FA BD 20 85 FE EC 42 42 A8 55 50 00 A3
                      49 55 63 B3 F0 01 A0 8E 48 C7 B7 97 83 75 72 FB
                      B2 5F B8 01 B4 4C 9E 05 1C D5 55 1A 36 FB 5B 85
                      9D 89 E7 65 31 0B 30 B1 D7 32 C8 29 74 13 5A CA
                      77 FE C4 3F 39 57 6B 30 E9 1B D3 93 8E E9 1E B3
                      D0 68 BD 81 4E E7 3B F4 19 75 E7 37 1C 73 8B 89
                      16 77 E7 40 AA
Number of extensions: 3

                      Certificate Information

              Label: server
          Record ID: 17
    Issuer Record ID: 15
            Trusted: Yes
            Version: 3
      Serial number: 4db08eef000374b3
        Issuer name: ca
                     itso
                     US
       Subject name: server
                     itso
                     US
     Effective date: 2011/04/21
     Expiration date: 2012/04/20
 Signature algorithm:
    Issuer unique ID: None
   Subject unique ID: None
Public key algorithm: ecPublicKey
     Public key size: 521
          Public key: 04 00 A9 5D 29 16 DE BA 11 BF 07 3A 16 B8 D3 F1
                      E5 6F 1D A9 76 C4 8E 89 9E 40 AC BB 6C A3 75 CF
                      22 FB 51 79 61 06 88 BD 1C D5 DD F6 9D BD 6E CE
                      2A F5 2E 8D 78 16 86 26 17 1E 23 9C 69 F9 E0 FD
                      FA E1 0E 01 79 8C 39 96 FC 03 0D CC 6A C5 5A 09
                      33 64 08 6C 0D 27 7B 93 AE 37 A3 FD 64 96 0C 2F
                      EB 20 AB F2 6B BD C7 D6 73 77 34 85 D4 4B CD 6C
                      6A D1 93 8F 2D AF C5 43 82 67 6C 6C 54 00 42 15
                      9D 57 FF E1 EE
Number of extensions: 3
```

CA certificate extensions

The CA certificate is self-signed; the subject and issuer names are the same. The client and server certificates are signed by the CA certificate. All three certificates contain ECC private and public keys with 521 bits. The signature algorithm is ecdsaWithSha512.

The CA certificate has extensions that differ from the other certificates. Figure 27-23 shows the CA certificate extensions.

```
       Certificate Extensions List

   1 - subjectKeyIdentifier
   2 - authorityKeyIdentifier
   3 - keyUsage (critical)
   4 - basicConstraints (critical)

Enter extension number (press ENTER to return to previous menu): 3

Certificate signature
CRL signature

Press ENTER to continue.


       Certificate Extensions List

   1 - subjectKeyIdentifier
   2 - authorityKeyIdentifier
   3 - keyUsage (critical)
   4 - basicConstraints (critical)

Enter extension number (press ENTER to return to previous menu): 4

Certification authority: Yes
```

Figure 27-23 CA certificate extensions

Figure 27-24 shows the server and client extensions.

```
       Certificate Extensions List

   1 - subjectKeyIdentifier
   2 - keyUsage (critical)
   3 - authorityKeyIdentifier

Enter extension number (press ENTER to return to previous menu): 2

Digital signature
Non-repudiation
Key agreement
```

Figure 27-24 Server and client extensions

27.6.4 Creating certificates using the RACF RACDERT command

You can create the certificates using the RACF **RACDCERT** command in the TSO environment. To create a CA certificate, issue the command shown in Figure 27-25.

```
racdcert gencert certauth withlabel('caitsoecc') subjectsdn(cn('ca')
o('itso') c('us')) size(521) nistecc(pkds(caitsoecc)) keyusage(certsign)
```

Figure 27-25 RACF racdcert ca certificate

Next, generate the server and client certificates as shown in Figure 27-26.

```
racdcert gencert site withlabel('serverecc') subjectsdn(cn('server')
o('itso') c('us')) size(521) nistecc(pkds(srvitsoecc))
keyusage(handshake keyagree) signwith(certauth label('caitsoecc'))

racdcert gencert id(rodolfi) withlabel('clientecc')
subjectsdn(cn('client') o('itso') c('us')) size(521) nistecc(pkds(cliitsoecc))
keyusage(handshake keyagree) signwith(certauth label('caitsoecc'))
```

Figure 27-26 RACF racdcert server and client certificates

The list shown in the following figures indicates all three certificates to verify the definitions using the **racdcert list** command. Figure 27-27 shows the label caitsoecc.

```
(1). racdcert list(label('caitsoecc')) certauth
 Digital certificate information for CERTAUTH:
   Label: caitsoecc
   Certificate ID: 2QiJmZmDhZmjgYOBiaOiloWDgOBA
   Status: TRUST
   Start Date: 2011/04/22 00:00:00
   End Date:   2012/04/22 23:59:59
   Serial Number:
        >00<
   Issuer's Name:
        >CN=ca.O=itso.C=us<
   Subject's Name:
        >CN=ca.O=itso.C=us<
   Key Usage: CERTSIGN
   Key Type: NIST ECC
   Key Size: 521
   Private Key: YES
   PKDS Label: CAITSOECC
   Ring Associations:
   *** No rings associated ***
```

Figure 27-27 Using racdcert list command to verify the ca, server, and client certificate (1)

Figure 27-28 shows the label serverecc.

```
(2). racdcert list(label('serverecc')) site
   Digital certificate information for SITE:

   Label: serverecc
   Certificate ID: 2QiJmZmiiaOFg6KFmaWFmYWDgOBA
   Status: TRUST
   Start Date: 2011/04/22 00:00:00
   End Date:   2012/04/22 23:59:59
   Serial Number:
        >02<
   Issuer's Name:
        >CN=ca.O=itso.C=us<
   Subject's Name:
        >CN=server.O=itso.C=us<
   Key Usage: HANDSHAKE, KEYAGREE
   Key Type: NIST ECC
   Key Size: 521
   Private Key: YES
   PKDS Label: SRVITSOECC
   Ring Associations:
   *** No rings associated ***
```

Figure 27-28 racdcert list command to verify the ca, server and client certificate (2)

Figure 27-29 shows the label `clientecc`.

```
(3). racdcert list(label('clientecc')) id(rodolfi)
 Digital certificate information for user RODOLFI:
   Label: clientecc
   Certificate ID: 2QfZ1sTWO8bJg5OJhZWjhYOD
   Status: TRUST
   Start Date: 2011/04/22 00:00:00
   End Date:   2012/04/22 23:59:59
   Serial Number:
        >03<
   Issuer's Name:
        >CN=ca.O=itso.C=us<
   Subject's Name:
        >CN=client.O=itso.C=us<
   Key Usage: HANDSHAKE, KEYAGREE
   Key Type: NIST ECC
   Key Size: 521
   Private Key: YES
   PKDS Label: CLIITSOECC
   Ring Associations:
   *** No rings associated ***
```

Figure 27-29 racdcert list command to verify the ca, server and client certificate (3)

Next, we created two key rings, one for server usage and another for client usage. The key rings in RACF are like the databases for the **gskkyman** utility. They contain all certificates necessary to authenticate and be used by the server and client applications running on a z/OS system.

The key rings are created and connected to the certificates as shown in Figure 27-30.

```
racdcert addring(clientecc) id(rodolfi) 1

racdcert addring(serverecc) id(rodolfi) 1

racdcert connect(certauth label('caitsoecc') ring(serverecc) usage(certauth))
id(rodolfi) 2

racdcert connect(site label('serverecc') ring(serverecc) default usage(site))
id(rodolfi) 3

racdcert connect(certauth label('caitsoecc') ring(clientecc) usage(certauth))
id(rodolfi) 2

racdcert connect(id(rodolfi) label('clientecc') ring(clientecc) default
usage(personal)) id(rodolfi) 3
```

Figure 27-30 Creating the server and client key rings and connecting the certificates

Note the following points in Figure 27-30:

1 We used a personal user ID to create the key rings. When creating the key rings, the user ID must be the user ID for the server and user application.

2 In this step we connect the CA certificate to both key rings to be able to verify the authenticity of server and client certificates, if required.

3 In this step we connect the server and client certificate to the specific key ring and make it the default. An application can select a specific certificate from a key ring. Making the certificate the default in the key ring means that it will be selected if a specific certificate is not specified.

27.6.5 Application support

Currently, the only application supporting the ECC algorithm is the Policy Agent IKE daemon for the IKE version 2 protocol. For more information about this topic, see *IBM z/OS V1R12 Communication Server TCP/IP Implementation Volume 4: Security and Policy-Based Networking,* SG24-7899.

It is also available in the System SSL API for applications written in C++.

Reference material

For more detailed information about the topics discussed in this chapter, refer to the following sources:

- *z/OS ICSF System Programmer's Guide*, SA22-7520
- *z/OS ICSF Administrator's Guide*, SA22-7521
- *z/OS ICSF Messages*, SA22-7523
- *z/OS V1R13.0 Security Server RACF System Programmer's Guide*, SA22-7681
- *z/OS Security Server RACF Security Administrator's Guide*, SA22-7683
- *z/OS Security Server RACF Command Language Reference*, SA22-7687

- *z/OS ICSF Writing PKCS #11 Applications*, SA23-2231
- *z/OS System SSL Programming*, SC24-5901
- RFC4492 “Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security”

  http://tools.ietf.org/html/rfc4492
- Standards for Efficient Cryptography Group (SECG) “SEC 1: Elliptic Curve Cryptography”

  http://www.secg.org/download/aid-780/sec1-v2.pdf
- Standards for Efficient Cryptography Group (SECG) “SEC 2: Recommended Elliptic Curve Domain Parameters”

  http://www.secg.org/download/aid-386/sec2_final.pdf
- RFC3279 “Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile”

  http://www.rfc-editor.org/rfc/rfc3279.txt
- RFC5480 “Elliptic Curve Cryptography Subject Public Key Information”

  http://www.rfc-editor.org/rfc/rfc5480.txt
- RFC5480 “Elliptic Curve Cryptography Subject Public Key Information”

  http://www.rfc-editor.org/rfc/rfc5480.txt
- FIPS 186-2 “Digital Signature Standard (DSS)”

  http://csrc.nist.gov/publications/fips/archive/fips186-2/fips186-2.pdf

28

UNICODE support

The Unicode Standard is the universal character encoding standard used for representation of text for computer processing. It is fully compatible with the second edition of International Standard ISO/IEC 10646-1:2000, and contains all the same characters and encoding points as ISO/IEC 10646. The Unicode Standard also provides additional information about the characters and their use. Any implementation that is in conformance to Unicode is also in conformance with ISO/IEC 10646.

Unicode provides a consistent way of encoding multilingual plain text, and it brings order to a chaotic state of affairs that has made it difficult to exchange text files internationally. Computer users who deal with multilingual text, such as business people, linguists, researchers, scientists, and others, will find that the Unicode Standard greatly simplifies their work. Mathematicians and technicians, who regularly use mathematical symbols and other technical characters, will also find the Unicode Standard valuable.

This chapter describes the following enhancements to UNICODE with IBM z/OS V1R13:

- New Information APIs
- BiDirectonial Phase II enhancements

28.1 New Information APIs

In prior releases, the Unicode Services Info API does not provide sufficient information for the supported CCSIDs. Starting with z/OS V1R13, IBM enhanced the Information API to provide additional information needed. For example, the API will now return the value to use for the space control character. This type of information can be useful when padding a field.

With this enhancement, the following information will also be returned on the Information API:

- The control character definitions associated with a given CCSID and its associated subCCSIDs

  These are the supported control characters:

  - Space
  - Substitution
  - New Line
  - Line Feed
  - Carriage Return
  - End of File
- Type of conversion (direct or indirect)
- Suffix letters for a given CCSID and its associated subCCSIDs (for example, PG for CCSID 13488) as given in the Unicode book, SA22-7649, under Description of CCSIDs.
- Description (CCSID Description) for the user knowledge base (USERKBS)

You can obtain the new, additional information through a call to the z/OS API Conversion Tables info service.

- CUNLINFO for 31-bit callers
- CUN4LINF for 64-bit callers

A new version (version 2) of the Information API parameter area is needed for this new function, as shown in Figure 28-1. The parameter area is defined in CUNBIIDF and CUN4BIID (31-bit and 64-bit, respectively) for PL/X and HLASM callers. For C/C++ callers, there is CUNHC for both 31-bit and 64-bit.

```
/*********************************************************************/
/* Additional information for Version 2 Parameter Area               */
/*********************************************************************/
char CCSID1_SUFFIX[2];              /* Suffix for CCSID1. The suffix*/
                                    /* for subCCSIDs are returned in*/
                                    /* subCCSIDs_info               */
char CCSID2_SUFFIX[2];              /* Suffix for CCSID1. The suffix*/
                                    /* for subCCSIDs are returned in*/
                                    /* subCCSIDs_info               */
unsigned char  Conversion_Type;     /* type of conversion for       */
                                    /* CCSID1 to CCSID2             */
                                    /* 1 = direct conversion        */
                                    /* 2 = indirect conversion      */
char Resi1[3];                      /* Reserved                     */
/*********************************************************************/
/* Additional information for Version 2 Parameter Area               */
/*********************************************************************/
void *        CCSID1_CTLDEF_Ptr ;   /* Pointer to CTLF (optional)   */
unsigned int  CCSID1_CTLDEF_ALET;   /* ALET for CTLF_Ptr            */
unsigned char CCSID1_CTLDEF_Num ;   /* Num of entries in CTLF       */
char Resi2[3];                      /* Reserved                     */
void *        CCSID2_CTLDEF_Ptr ;   /* Pointer to CTLF              */
unsigned int  CCSID2_CTLDEF_ALET;   /* ALET for CTLF_Ptr            */
unsigned char CCSID2_CTLDEF_Num ;   /* Num of entries in CTLF       */
```

Figure 28-1 Extended version 2 parameter fields

With this implementation, z/OS Unicode Services Character Conversion Service exploiters will be able to obtain additional CCSID information programmatically before calling the conversion service. CCSID information can be used separately by calling z/OS Conversion Information Service (Info API) for any CCSID background purpose.

28.2 BiDirectonial Phase II Enhancements

Unicode Services Character Conversion Service support for bidirectional transformation and character shaping (Bidi) is outdated. The Unicode Services Character Conversion Service API is being enhanced with the highest level of Bidi support available. The new Bidi support meets several of the standards set forth in the Unicode Consortium's Standard Annex #9.

The annex can be found at the Unicode Consortium's web site at:

http://www.unicode.org/reports/tr9

The extended Bidi support does not implement the full Unicode Consortium's Bidi standard. It implements the highest level of support available. It allows users of Unicode Services Character Conversion Service's bidirectional transformation and character shaping (Bidi) support to utilize the highest level of support available. Prior to z/OSV1R13, two Unicode Services APIs supported bidirectional transformation and character shaping:

- Bidi transformation API
- Character conversion API "B" technique

Support for the Bidi transformation API and the character conversion API "B" technique remain "as is" to avoid migration issues. No enhancements to this support will be made.

Use the “extended Bidi support” for all new development and for clients who want to use the highest level of Bidi support. This design introduces “extended” Bidi support in the form of a new “extended Bidi parameter area” and an “extended Bidi parameter area pointer” in the existing character conversion parameter area.

IBM provides two samples that demonstrate how to use the Unicode Services:

CUNSISM7 Data set SYS1.SAMPLIB contains a sample that demonstrates how to use the Unicode Services Extended Bidi Support.

CUNSISM8 Data set SYS1.SAMPLIB is a sample that demonstrates how to use the Unicode Services Extended Bidi Support using the Open Group's standard “Portable Layout Services” interface functions, as listed in Table 28-1.

Table 28-1 Portable Layout Service functions

Function	Description
m_create_layout()	Create and Initialize a Layout Object
m_getvalues_layout()	Query Layout Values of a Layout Object
m_setvalues_layout()	Set Layout Values of a Layout Object
m_getoptions_layout()	Query the current setting of Layout Options of a Layout Object
m_setoptions_layout()	Change the Layout Options of a Layout Object
m_transform_layout()	Layout Transformation for Character Strings
m_wtransform_layout()	Layout Transformation for Wide-Character Strings
m_getprocessedlength_layout()	Query the Length of Source Text Processed by the LastTransform Operation on a Layout Object
m_destroy_layout()	Destroy a Layout Object

The Unicode Services Character Conversion Service does not support the use of user-customized character conversion tables while using extended Bidi support. The Unicode Services extended Bidi support operates on CCSID 01200. If the source and target CCSIDs are not both 01200 (or equivalent CCSIDs), the Bidi algorithm will cause a two-stage conversion to be performed regardless of other considerations. The source buffer is first converted to CCSID 01200, Bidi transformations are performed, then the characters are converted to the target CCSID. The work buffer (Wrk_Buf) buffer is required for this.

Use of the Unicode Services extended Bidi support for CCSIDs other than Arabic or Hebrew CCSIDs 00420, 00424, 00425, 00856, 00862, 00864, 00916, 01046, 01089, 01255, or 01256 will result in the following indications:

```
RC = CUN_RC_USER_ERR
RS = CUN_RS_CCSID_NOT_SUPP
```

Note: This support is intended to run in a Language Environment as a replacement to Language Environment Bidi support.

All the new functions are called by the z/OS Unicode Services Character Conversion Service API:

- CUNLCNV for 31-bit callers
- CUN4LCNV for 64-bit callers

Version 3 of the Character Conversion Service API parameter area is needed for this new function. The parameter area is defined in CUNBDPRM and CUN4BDPR (31-bit and 64-bit, respectively) for PL/X and HLASM callers, and in CUNBCPRM and CUN4BCPR for C/C++ callers.

Set the Extended_Bidi_Parm_Area_Ptr and remove B from the technique search order parameter. Fill in the extended Bidi parameter area CUNBDPRM and CUN4BDPR (31-bit and 64-bit, respectively) for PL/X and HLASM callers as appropriate, and CUNBCPRM and CUN4BCPR for C/C++ callers. Figure 28-2 shows the modified tag CUNBCPRM structure.

```
typedef struct tagCUNBCPRM {
 long Version; /* Structure version number */

 ...

 long         Return_Code;
 long         Reason_Code;
 unsigned int Res6;
 struct {
 int       ETF3E_Behavior        :  1,
                                 : 15;

 } Flag3;
 char         Res7[2];
 CUNBDPRM *   Extended_Bidi_Parm_Area_Ptr;
 char         Res8[64];
 } CUNBCPRM;
```

Figure 28-2 Modified tagCUNBCPRM structure

Figure 28-3 shows the new CUNBCPRM structure.

```
/* The extended bidi parameter area *
typedef struct tagCUNBDPRM {
int             Version;
int             Length;
struct {
   int          XOpen_Defaults        :  1,
                KBS_Defaults          :  1,
                Keyword               :  1,
                From_wtransform       :  1,
                                      :  4,
}               InFlags;
struct {
   int          Layout_Roundtrip      :  1,
                Layout_WinCompat      :  1,
                Layout_ImpToImp       :  1,
                Layout_Remove_Marks   :  1,
                Layout_Insert_Marks   :  1,
                Layout_Streaming      :  1,
                                      :  2;
}               Layout_Options;
struct {
   int          ActiveShapeEditing    :  1,
                ActiveDirectional     :  1,
                                      : 14;
}               OutFlags;
int             Orientation_Src;
int             Orientation_Targ;
int             Context_Src;
int             Context_Targ;
int             TypeOfText_Src;
int             TypeOfText_Targ;
int             ImplicitAlg_Src;
int             ImplicitAlg_Targ;
int             Swapping_Src;
int             Swapping_Targ;
int             Numerals_Src;
int             Numerals_Targ;
int             TextShaping_Src;
int             TextShaping_Targ;
int             ShapeCharsetSize;
int             ShapeCharsetSize_Front;
int             ShapeCharsetSize_Back;
int             CheckMode;
unsigned int    InpBufIndex;
unsigned long   Streaming_Processed_Length;
int             ArabicOneCellShaping_Src;
int             ArabicOneCellShaping_Targ;
int             WordBreak_Src;
int             WordBreak_Targ;
int             LamAlefEditMode_Src;
int             LamAlefEditMode_Targ;
int             YehHamzaMode_Src;
int             YehHamzaMode_Targ;
int             TailEditMode_Src;
int             TailEditMode_Targ;
int             TashkeelEditMode_Src;
int             TashkeelEditMode_Targ;
unsigned int  * InpToOut_Ptr;
unsigned int  * OutToInp_Ptr;
unsigned char * BidiLvl_Ptr;
char            Layout_Streaming_State[64];
char            Bidi_Keyword[128];
char            Res2[64];

} CUNBDPRM;
```

Figure 28-3 New CUNBDPRM structure

Figure 28-4 shows a modified CUNBDPRM structure.

```
typedef struct tagCUN4BCPR {

unsigned int  Version; /* Structure version number */

...

unsigned int  Return_Code;
unsigned int  Reason_Code;
int           Res4;
long          Res5;
struct {
   int       ETF3E_Behavior        :  1,
                                   : 15;
} Flag3;

char         Res7;
CUN4BDPR *   Extended_Bidi_Parm_Area_Ptr;
char         Res8[64];

} CUN4BCPR;
```

Figure 28-4 Modified CUN4BCPR structure

Figure 28-5 shows the new CUN4BDPR structure.

```
/* The extended bidi parameter area */

typedef struct CUN4BDPR {
int             Version;
int             Length;
struct {
   int          XOpen_Defaults        :  1,
                KBS_Defaults          :  1,
                Keyword               :  1,
                From_wtransform       :  1,
                                      :  4,
}               InFlags;
struct {
   int          Layout_Roundtrip      :  1,
                Layout_WinCompat      :  1,
                Layout_ImpToImp       :  1,
                Layout_Remove_Marks   :  1,
                Layout_Insert_Marks   :  1,
                Layout_Streaming      :  1,
                                      :  2;
}               Layout_Options;
struct {
   int          ActiveShapeEditing    :  1,
                ActiveDirectional     :  1,
                                      : 14;
}               OutFlags;
int             Orientation_Src;
int             Orientation_Targ;
int             Context_Src;
int             Context_Targ;
int             TypeOfText_Src;
int             TypeOfText_Targ;
int             ImplicitAlg_Src;
int             ImplicitAlg_Targ;
int             Swapping_Src;
int             Swapping_Targ;
int             Numerals_Src;
int             Numerals_Targ;
int             TextShaping_Src;
int             TextShaping_Targ;
int             ShapeCharsetSize;
int             ShapeCharsetSize_Front;
int             ShapeCharsetSize_Back;
int             CheckMode;
unsigned long   InpBufIndex;
unsigned long   Streaming_Processed_Length;
int             ArabicOneCellShaping_Src;
int             ArabicOneCellShaping_Targ;
int             WordBreak_Src;
int             WordBreak_Targ;
int             LamAlefEditMode_Src;
int             LamAlefEditMode_Targ;
int             YehHamzaMode_Src;
int             YehHamzaMode_Targ;
int             TailEditMode_Src;
int             TailEditMode_Targ;
int             TashkeelEditMode_Src;
int             TashkeelEditMode_Targ;
unsigned int  * InpToOut_Ptr;
unsigned int  * OutToInp_Ptr;
unsigned char * BidiLvl_Ptr;
char            Layout_Streaming_State[64];
char            Bidi_Keyword[128];
char            Res2[64];
} CUN4BDPR;
```

Figure 28-5 New CUN4BDPR structure

Note: Both CUNBIDF and CUN4BCID macros, which resided in the SYS1.MACLIB, were changed to reflect these changes.

29

Language Environment enhancements

Language Environment provides a common run-time environment for IBM versions of certain high level languages (HLLs), namely C, C++, COBOL, Fortran, and PL/I, in which you can run existing applications written in previous versions of these languages and in the current Language Environment-conforming versions.

Prior to Language Environment, each of the HLLs had to provide a separate run-time environment. Language Environment combines essential and commonly used run-time services (such as routines for run-time message handling, condition handling, storage management, date and time services, and math functions) and makes them available through a set of interfaces that are consistent across programming languages.

With Language Environment, you can use one runtime environment for your applications, regardless of the application's programming language or system resource needs because most system dependencies have been removed. Language Environment provides compatible support for existing HLL applications. Most existing single-language applications can run under Language Environment without being recompiled or relink-edited. POSIX-conforming C applications can use all Language Environment services.

This chapter describes the following enhancements:

- CEEPIPI multi-main and user word
- LE/C BSAM > 64 K tracks (binary and text)
- LE C-RTL I/O ABEND recovery
- Metal C qsort()

29.1 CEEPIPI multi-main and user word

The preinitialization compatibility interface (PICI) is an older form of PreInit that is supported but is no longer being enhanced. Users who want to take advantage of newer functionality need to change their assembler programs to use the PreInit interfaces.

In zOSV1R13, IBM provides a second PreInit Interface. Conversion from the Preinitialization compatibility interface (PICI) to PreInit would be easier if additional interfaces were provided in PreInit, specifically:

- Support for multiple main environments on one TCB
- Support for a user word that can be accessed from both outside and within a PreInit environment

The additional PreInit Interfaces facilitate conversion from PICI to PreInit so that users can move to the more strategic interface.

The new support also contains support for multiple main environments on one TCB. The new initialization routine CEEPIPI function init_main_dp allows the Preinit assembler driver to create multiple main CEEPIPI environments on the same TCB.

Main programs can be called on these environments, but only one call can be active at a time on a given TCB. The CALL CEEPIPI(init_main_dp,ceexptbl_addr,service_rtns,token) parameters are shown in Table 29-1.

Table 29-1 Parameters for CEEPIPI multi-main call

parameter	Description
init_main_dp (input)	A fullword containing the init_main_dp function code (integer value = 19)
ceexptbl_addr (input)	A fullword containing the address of the PreInit table to be used during initialization of the new environment
service_rtns (input)	A fullword containing the address of the service routine vector or 0, if there is no service routine vector
token (output)	A fullword containing a unique value used to represent the environment

Support for Preinit user word

The support for PreInit user word facilitates communication between the Preinit assembler driver and the user code running within a Preinit environment Preinit assembler driver uses CEEPIPI interfaces to access the user word:

- CEEPIPI(set_user_word,...) sets the user word value
- CEEPIPI(get_user_word,...) retrieves the user word value from the last set_user_word call

The code running within the Preinit environment accesses the user word from within the CAA control block. The Field CEECAA_USER_WORD in the assembler CEECAA mapping 4-byte field is located at offset +3F0x.

Important: Modifications to this field by the user code running in the Preinit environment are not saved between CEEPIPI calls. The next CEEPIPI call will use the value from last set_user_word call.

Table 29-2 lists the appropriate parameters for the CEEPIPI set user word call.

Table 29-2 Parameters for CEEPIPI set user word call

Parameter	Description
set_user_word (input)	A fullword containing the set_user_word function code (integer value = 17)
token (input)	A fullword with the value of the token of the environment
value (input)	A fullword value that will be used to initialize the user word in the initial thread CAA when the application is invoked

Table 29-4 lists the appropriate parameters for the CEEPIPI get user word call.

Table 29-3 Parameters for CEEPIPI get user word call

Parameter	Description
get_user_word (input)	A fullword containing the get_user_word function code (integer value = 18)
token (input)	A fullword with the value of the token of the environment
value (output)	A fullword that will be returned containing the current value that will be used to initialize the CAA user word when the next application is invoked.

29.2 LE/C BSAM > 64 K tracks (binary and text)

In prior releases of z/OS, sequential data sets can only use a maximum of 59 volumes. There is a limit to the number of tracks a volume can contain. After the limit has been reached, there is no room for the data set to grow.

In z/OS V1R7, DFSMSdfp provided support for large format sequential data sets, removing the size limit of 65,535 tracks/volume for QSAM, BSAM, and EXCP. Therefore, C/C++ applications need to be able to exploit the functionality added by DFSMSdfp. The basic forms of ftell() and fseek() cannot handle the offsets that are possible when working with large format sequential data sets. These functions only support the reporting of and repositioning (directly or relatively) to offsets that are 2 GB - 1 or less.

Starting with z/OS V1R13, IBM provides XL C/C++ Run-time Library support for large format sequential data sets greater than 65,535 tracks/volume when opened for BSAM (seek) under binary and text I/O. This allows clients to use ftello() and fseeko() to work on large format sequential data sets. The used data sets are now able eligible to grow beyond the 65,535 tracks/volume limit.

The support is invoked by calling the fopen() function on a pre-existing large format sequential data set (DNSTYPE=LARGE was specified).

Note: You can allocate a new large format sequential data set by specifying the keyword DSNTYPE=LARGE on a JCL DD statement, or by using the dynamic allocation equivalent.

New macro __DSNT_LARGE is added for use with the dynalloc() function. After the data set is opened, other OS I/O functions can be used to process the stream. The XL C/C++ Run-time Library OS I/O routines can be used on large format sequential data sets that are single volume or multivolume. Data sets can reside on SMS-managed or non-SMS managed storage devices. They can also be cataloged or uncataloged.

An ABEND during OPEN/CLOSE/EOV processing is no longer generated when seek is used with large format sequential data sets greater than 65535 tracks under binary or text I/O. Furthermore, attempts to open a large format sequential data set for read (r,rb,rt) with seek under binary or text I/O while the data set is already open for write (w,wb,wt) with noseek will no longer fail.

Clients can also use the capability of using the large files versions of the ftello() and fseeko() functions on MVS data sets that are opened for any type of I/O (record, binary or text). This support allows reporting of and repositioning to, either directly or relatively, offsets greater than 2 GB - 1.

To use that functionality for AMODE 31 C/C++ applications, the fseek() function accepts a signed 4-byte offset and therefore cannot be used to directly or relatively position to offsets beyond 2 GB - 1. To avoid repositioning limitations, have AMODE 31 C/C++ applications define the #define _LARGE_FILES 1 feature test macro before any headers are included, and replace the fseek() function with the fseeko() function. For AMODE 64 C/C++ applications, there are no restrictions on using the fseek() function with large files. The AMODE 64 version automatically accepts a signed 8-byte offset.

29.3 LE C-RTL I/O ABEND Recovery Part 1

The Language Environment C/C++ Run-time Library (C-RTL) always attempts to ignore abend conditions that occur during low-level OS I/O processing. However, sometimes situations are encountered where a condition cannot be ignored. DFSMS will issue an abend and Language Environment condition handling takes over. The C-RTL does not recover (return gracefully to the application) in these instances. Therefore, application developers have to write condition handlers or SIGABND handlers to attempt to recover when the C-RTL cannot ignore these types of abend conditions. Restrictions apply regarding what can be done in the abend handler. Having the C-RTL recover in certain situations (when the abend can be ignored) and not recover in others is not user-friendly behavior.

With zOSV1R13, IBM provides a mechanism to enable the C-RTL to recover gracefully from an abend condition during output or CLOSE processing, when the abend cannot be ignored. The C-RTL function that triggered the abend condition will gracefully return to the application instead of bringing down the enclave (if condition handling is not in effect). Now clients have improved reliability in knowing that an abend during a low-level OS I/O operation will not terminate the Language Environment.

Clients have two ways to invoke abend recovery behavior:

- New fopen()/freopen() keyword
- New environment variable

Either method can be used to control how the C-RTL treats abend conditions that cannot be ignored. When either method is set up to recover from the abend, the C-RTL will instruct the function to return a failing value to the application and set errno to 92. When using the new behavior, detailed diagnostic information will also be set in the __amrc structure.

New fopen()/freopen()

A new parameter is provided by the fopen/freopen function:

```
abend=abend | recover
```

The value `abend` instructs the runtime library to ignore abend conditions that can be ignored. No attempt is made to recover from abend conditions that cannot be ignored.

The value `recover` instructs the runtime library to attempt to recover from an abend issued during certain low-level I/O operations (WRITE / CHECK sequence and CLOSE). Figure 29-1 shows an example of fopen using recovery services.

```
fopen("//'myfile.data'","wb,type=record,abend=recover");
```

Figure 29-1 Sample fopen with recovery

Note: This method specifies the behavior for only the stream being opened.

New environment variable _EDC_IO_ABEND

The second method to use this recovery behavior is by using the new Environment variable _EDC_IO_ABEND. The variable can contain either ABEND or RECOVER values. The values invoke the same behavior as their relative fopen() keyword values. When unset or set to something other than RECOVER, the default behavior is ABEND. The setting of the environment variable defines the behavior for the life of an open stream The environment variable can be overridden by the fopen() keyword. Figure 29-2 shows an example of setting the environment variable.

```
setenv("_EDC_IO_ABEND", "RECOVER", 1);
```

Figure 29-2 Sample of setting env variable

CEECAA updates

The CEECAA is used to generate a common anchor area (CAA) mapping. This macro has no parameters, and no label can be specified. CEECAA is required for the CEEENTRY macro. Beginning with z/OSV1R13, IBM added two new fields to this macro as listed in Table 29-4.

Table 29-4 New added CEECAA fields

Field	Offset	Description
CEECAASHAB_RECOVER_IN_ESTAE_MODE	+30C	[Bit in the CEECAAFLAG1 field] When ON, the Language Environment ESTAE resumes to the abend shunt in the mode and key in which the Language Environment ESTAE was established.
CEECAASHAB_KEY	+30D	[Character] IPK result when CEECAASHAB is set.

29.4 Metal C qsort()

The qsort() function is an array sorting function that is part of many C standard libraries. In prior releases of z/OS there was no equivalent function available in the Metal C Runtime Library, such that it can be used within their applications.

Starting with zOSV1R13, a qsort() is available to the Metal C Runtime Library. The support is usable for both ILP32-compiled and LP64-compiled programs. You are able to use sort in Metal C applications. Figure 29-3 shows the function and parameter.

```
#include <stdlib.h>

void qsort(void *base, size_t num, size_t width,
           int(*compare)(const void *element1, const void *element2));
```

Figure 29-3 A qsort function definition

Table 29-5 lists the parameters passed to the qsort function.

Table 29-5 Calling parameters for qsort() function

Parameter	Description
base	Address of first element of the array
Num	Number of elements in the array
Width	Size of each element
compare	Function that compares two elements in the array Returns: <0: element1 is less than element2 =0: element1 is equal to element2 >0: element1 is greater than element2

30

SDSF enhancements

This chapter describes the enhancements to SDSF in z/OS V1R13.

It discusses the following topics:

- JES2 only
- JES3 only
- Both JES2 and JES3
- SDSF/REXX

30.1 Enhancements for JES2

SDSF is enhanced to support the new JES2 functions introduced in z/OS V1R13.

30.1.1 Spool migration support

The Spool Volumes (SP) panel has added new columns in support of spool migration processes. Spool migration is a new function available in JES2 on z/OS V1R13. Figure 30-1 shows an example of the new columns.

```
  Display  Filter  View  Print  Options  Search  Help
------------------------------------------------------------------------------
SDSF SPOOL DISPLAY SC74      42% ACT   9975 FRE   5783 LINE 1-1 (1)
COMMAND INPUT ===>                                        SCROLL ===> CSR
NP   NAME    LgFree HiUsed Comp% Phase        MigSys Target   MigVol MigDSName
     BH5SP1    9215  40660
```

Figure 30-1 New columns in the SP panel provide JES2 spool migration information

The meanings of the new columns are as follows:

Comp%	Completion percentage for the current action against the volume
Phase	Migration phase
MigSys	Member performing the spool migration
Target	The target volume for migration
MigVol	Volume this extent is migrating to
MigDSName	Data set this extent is migrating to

30.1.2 Job RC display

The Job Class (JC) panel is enhanced to show the new JES2 job class parameter JOBRC. This is an overtypable field that takes the values of MAXRC or LASTRC, in correspondence with the **$TJOBCLASS(B),JOBRC=MAXRC** or **$TJOBCLASS(B),JOBRC=LASTRC** commands; see Figure 30-2.

```
 Display  Filter  View  Print  Options  Search  Help
------------------------------------------------------------------------------
SDSF JOB CLASS DISPLAY ALL CLASSES                    LINE 1-36 (38)
COMMAND INPUT ===>                                       SCROLL ===> CSR
ACTION=//-Block,=-Repeat,+-Extend,D-Display,ST-Status
NP   CLASS    Tp26 CPr MC Scheduling-Env   JesLog        XBMProc  DupJob JobRC
     A        YES                          (NOSPIN)               NO     MAXRC
     B        YES                          (NOSPIN)               NO     LASTRC
     C        YES                          (NOSPIN)               NO     MAXRC
     D        YES                          (NOSPIN)               NO     MAXRC
     E        YES                          (NOSPIN)               NO     MAXRC
     F        YES                          (NOSPIN)               NO     MAXRC
```

Figure 30-2 New JOBRC column on the JC panel

In addition, the Max-RC field in the O, H, I, and ST panels is updated to show the value **CONV ERR** if the job failed at the conversion phase, and the value **SYS FAIL** if the job ended due to an IPL.

30.1.3 JES2 $EJ,STEP command support

Two new line actions are added to the DA, ST, and I panels in support of the new JES2 job restart after step function:

ES This restarts the job at the next step after the current step completes, which corresponds to the **$EJ,STEP** command.

ESH This holds the job and re-queues it for execution after the current step completes, which corresponds to the **$EJ,STEP,HOLD** command.

The new actions are shown in Figure 30-3.

```
 Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
SDSF DA SC74     SC74     PAG  0  CPU/L/Z   1/  1/  0  LINE 1-2 (2)
COMMAND INPUT ===>                                          SCROLL ===> CSR
ACTION=//-Block,=-Repeat,+-Extend,?-JDS,A-Release,C-Cancel,CA-CancelARM,
ACTION=CD-CancelDump,CDA-CancelARMDump,D-Display,DL-DisplayLong,E-Restart,
ACTION=EC-RestartCancel,ES-RestartStep,ESH-RestartStepHold,H-Hold,K-SysCancel,
ACTION=KD-SysCancelDump,L-List,LL-ListLong,P-Purge,PP-PurgeProtected,Q-OutDesc,
ACTION=R-Reset,RQ-ResetQuiesce,S-Browse,SB-ISPFBrowse,SE-ISPFEdit,SJ-JCLEdit,
ACTION=W-Spin,X-Print,XC-PrintClose,XD-PrintDS,XDC-PrintDSClose,XF-PrintFile,
ACTION=XFC-PrintFileClose,XS-PrintSysout,XSC-PrintSysoutClose,Y-SysStop,
ACTION=Z-SysForce
NP   JOBNAME  StepName ProcStep JobID    Owner    C Pos DP Real Paging    SIO
     RMF      RMF      IEFPROC  STC03934 RMF        NS  FE 535T   0.00   0.00
     RMFGAT   RMFGAT   IEFPROC  STC03958 RMFGAT     NS  FE 8020   0.00   0.00
```

Figure 30-3 New line actions in support of the restart job step function

30.1.4 Spin line action in the JDS panel

The Job Data Set (JDS) panel now supports a new **W** line action to spin any spinnable job data set. The **W** action corresponds to the JES2 **$TJ,SPIN,DDNAME=** command.

Note: The **W** line action is only available when the JDS panel is accessed through the DA, ST, or I panels.

A new column is added to the JDS panel to indicate whether a job data set is spinnable, as shown in Figure 30-4. On that panel the spin option is NO for that job. You must scroll to get to the correct columns to view Spin.

```
 Display  Filter  View  Print  Options  Search  Help
------------------------------------------------------------------------------
SDSF JOB DATA SET DISPLAY - JOB HEUSERK  (JOB07570)    LINE 1-5 (5)
COMMAND INPUT ===>                                        SCROLL ===> HALF
PREFIX=*  DEST=(ALL)  OWNER=*  SYSNAME=*
ACTION=//-Block,=-Repeat,+-Extend,Q-OutDesc,S-Browse,SB-ISPFBrowse,SE-ISPFEdit,
ACTION=SJ-JCLEdit,V-View,W-Spin,X-Print,XC-PrintClose,XD-PrintDS,
ACTION=XDC-PrintDSClose,XF-PrintFile,XFC-PrintFileClose,XS-PrintSysout,
ACTION=XSC-PrintSysoutClose
NP   DDNAME               Spin Sel TP  TPJName  TPJobID  TPAcct   TRd-Time TRd
     JESMSGLG             NO   YES NO                               0:00:00 000
     JESJCL               NO   YES NO                               0:00:00 000
     JESYSMSG             NO   YES NO                               0:00:00 000
     SYSPRINT             NO   YES NO                               0:00:00 000
     SYSOUT               NO   YES NO                               0:00:00 000
```

Figure 30-4 The new Spin column in the JDS panel

30.1.5 Sysplex-wide view in more panels

In z/OS V1R13, JES2 has been enhanced to provide sysplex-wide information through the subsystem interface (SSI) function codes 82 and 83. SDSF uses SSI 82 and 83 to request information from JES2. This allows SDSF to display sysplex-wide information in more panels, without requiring the SDSF server to be active in all systems in the sysplex.

The following panels are updated to use SSI 82 and 83 to request JES2 information. They no longer require the SDSF server to display sysplex-wide data:

NO Node display
NC Network Connection display
NS Network Server display
PU Punch display
RDR Reader display
INIT Initiator display

30.2 Enhancements for JES3

Prior to z/OS V1R10, SDSF supported only JES2 environments. z/OS V1R10 SDSF included initial support for the JES3 environment. The JES3 job-related Display Active Users (DA), Input Queue (I), and Status (ST) panels were available for JES3 displays. Other SDSF panels that do not depend on the JES type were also available in the JES3 environment.

z/OS V1R11 added function in the JES3 environment to include the SYSLOG, Job Class (JC), Spool Volumes (SP), and JESPLEX (JP) displays. Support was also added to display and modify output descriptors for JES3 jobs through the Output Descriptor (OD) panel and the Job Data Set (JDS) panel. The JES3 browse of a job that is running on a system other than the one you are logged on shows data from buffers not yet written to the spool.

In z/OS V1R12, SDSF introduced support for JES3 local printers, under the Printers (PR) panel. The support included the call, start, and restart device functions.

z/OS V1R13 expands SDSF function in the JES3 environment to include Initiator (INIT), Job zero (J0), Line (LI), Node (NO), Punch (PUN), Reader (RDR), Held Output Queue (H), and

Output Queue (O) panels for JES3 objects. Network Connect (NC) and Network Server (NS) panels show information about JES3 networking. The Printers (PR) panel now also supports remote JES3 printers.

30.2.1 Printer (PR) display

The Printer panel allows authorized users to display information about JES printers and jobs being printed. The panel now supports both local and remote JES3 printers; see Figure 30-5.

```
 Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
SDSF PRINTER DISPLAY  (ALL)                            LINE 1-14 (14)
COMMAND INPUT ===>                                      SCROLL ===> HALF
ACTION=//-Block,=-Repeat,+-Extend,BC-BackCkpt,BCnP-BackNumCkpt,BD-BackTop,
ACTION=BN-BackCkptN,BNnP-BackNumCkptN,C-Cancel,CG-CancelGroup,CJ-CancelJob,
ACTION=CP-CancelPosition,CT-CancelStop,D-Display,DL-DisplayLong,
ACTION=E+ADHJLMRTX-RestartOptions,E-Restart,EH-RestartHold,EJ-RestartJob,
ACTION=ER-RestartRescan,Fn-ForwardNum,FC-ForwardCkpt,FCnP-ForwardNumCkpt,
ACTION=FN-ForwardCkptN,FNnP-ForwardNumCkptN,K-ForceFSS,L-Fail,LD-FailDump,
ACTION=S+ADMTX-StartOptions,S-Start,V-VaryOn,VF-VaryOff,X+DRTX-CallWtrOptions,
ACTION=X-CallWtr,XR-CallWtrResched
NP     PRINTER    Status   Group    SForms   SClass          JobName  JobID
       IAZFSS     AC       LOCAL    STD      CD              VAINI    JOB20994
       IPDPOK     AC       IPDS     STD      I
       IPDWAY     OFF      IPDS     STD      K
       NS         OFF      TCPIP    VTAM     J
       PRTWAY     OFF      TCPIP    STD      J
       PRTWA2     OFF      TCPIP    STD      J
       RM001PR1   AV       RM001    STD
```

Figure 30-5 PR display in a JES3 environment

It is also possible to invoke the PR panel by using **PR LCL** to list only local printers, or by using **PR RMT** to list only remote printers.

30.2.2 Punch (PU) and Reader (RDR) displays

The Punch (PU) panel allows authorized users to display information about JES punches and jobs being punched; see Figure 30-6.

```
 Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
SDSF PUNCH DISPLAY  SC75                                LINE 1-7 (7)
COMMAND INPUT ===>                                        SCROLL ===> HALF
ACTION=//-Block,=-Repeat,+-Extend,BC-BackCkpt,BCn-BackNumCkpt,BD-BackTop,
ACTION=BN-BackCkptN,BNn-BackNumCkptN,C-Cancel,CG-CancelGroup,CJ-CancelJob,
ACTION=CP-CancelPosition,CT-CancelStop,D-Display,DL-DisplayLong,
ACTION=E+ADHJMRTX-RestartOptions,E-Restart,EH-RestartHold,EJ-RestartJob,
ACTION=ER-RestartRescan,FC-ForwardCkpt,FCn-ForwardNumCkpt,FN-ForwardCkptN,
ACTION=FNn-ForwardNumCkptN,L-Fail,LD-FailDump,S+ADMTX-StartOptions,S-Start,
ACTION=V-VaryOn,VF-VaryOff,X+DRTX-StartOptions,X-CallWtr,XR-CallWtrResched
NP   PUNCH      Status   Group    SForms   JobName  JobID    Owner    Rec-Cnt R
     RM002PU1   AV       RM002    STANDARD
     RM003PU1   AV       RM003    STANDARD
     RM004PU1   AV       RM004    STANDARD
     RM005PU1   AV       RM005    STANDARD
```

Figure 30-6 PU display in a JES3 environment

The Reader (RDR) panel allows authorized users to display information about JES readers and jobs being processed by readers; see Figure 30-7.

```
 Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
SDSF READER DISPLAY  SC75                               LINE 1-8 (8)
COMMAND INPUT ===>                                        SCROLL ===> HALF
ACTION=//-Block,=-Repeat,+-Extend,C-Cancel,CH-CancelHold,CHN-CancelNoHold,
ACTION=CK-CancelAlloc,CKN-CancelPurge,D-Display,DL-DisplayLong,L-Fail,
ACTION=LD-FailDump,S-Start,SH-StartHold,SHN-StartNoHold,SK-StartAlloc,
ACTION=SKN-StartPurge,V-VaryOn,VF-VaryOff,X-Call,XC-CallCardImage,XH-CallHold,
ACTION=XHN-CallNoHold,XK-CallAlloc,XKN-CallPurge
NP   READER     Status   Group    JobName  JobID    Owner    Rec-Cnt    Rec-Pro
     RM001RD1   AV       RM001
     RM002RD1   AC       RM002     (NONE)
     RM003RD1   AV       RM003
     RM004RD1   AV       RM004
     RM005RD1   AV       RM005
     RM006RD1   AV       RM006
     RM007RD1   AV       RM007
```

Figure 30-7 RDR display in a JES3 environment

Note that in z/OS V1R13, the PU/RDR device name, which is a fixed field, is expanded to be 10 characters wide.If this change is undesirable, SDSF allows you to revert to the behavior before z/OS V1R13 by setting the following two properties to true in the ISFPRMxx member:

Panel.PUN.DevnameAlwaysShort For the Punches panel

Panel.RDR.DevnameAlwaysShort For the Readers panel

30.2.3 Line (LI) display

The Line (LI) panel allows authorized users to display information about JES lines and their associated transmitters and receivers; see Figure 30-8.

```
 Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
SDSF LINE DISPLAY  SC75                                  LINE 0-0 (0)
COMMAND INPUT ===>                                         SCROLL ===> HALF
ACTION=//-Block,=-Repeat,+-Extend,C-Cancel,D-Display,DE-DisplayErrors,
ACTION=DL-DisplayLong,DS-DisplayStatus,E-Restart,I-Interrupt,L-Fail,
ACTION=LD-FailDump,S-Start,SL-StartLog,SNL-StartNoLog,SNR-StartNoRcv,
ACTION=SR-StartRcv,SRJP-StartRJP,V-VaryOn,VF-VaryOff
NP   DEVICE       Status   Unit  Node     JobName  JobID    Owner    Proc-Lines
```

Figure 30-8 LI display in a JES3 environment

By default, the LI panel lists lines and their associated NJE transmitters and receivers. To suppress the NJE transmitters and receivers from the list, invoke the LI display with the SHORT parameter, for example **LI SHORT**.

30.2.4 Node (NO) display

The Nodes (NO) panel allows authorized users to display information about JES nodes.; see Figure 30-9.

```
 Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
SDSF NODE DISPLAY  SC75     WTSC75J3                  LINE 1-10 (10)
COMMAND INPUT ===>                                         SCROLL ===> HALF
ACTION=//-Block,=-Repeat,+-Extend,A-Release,D-Display,DL-DisplayLines,
ACTION=EL-ResetLines,H-Hold,SN-Start
NP   NODENAME Status               Path     PType Hold VerifyP  SendP    SysNa
     WTSCMXA  CONNECTED            WTSCNET        NONE NOTSET   NOTSET   SC75
     WTSCNET  CONNECTED            WTSCNET  TCPIP NONE NOTSET   NOTSET   SC75
     WTSCPLX1 UNCONNECTED          WTSCPLX1 TCPIP NONE NOTSET   NOTSET   SC75
     WTSCPLX2 UNCONNECTED          WTSCPLX2 SNA   NONE NOTSET   NOTSET   SC75
     WTSCPLX3 UNCONNECTED          WTSCPLX3 SNA   NONE NOTSET   NOTSET   SC75
     WTSCPLX4 CONNECTED            WTSCPLX4 TCPIP NONE NOTSET   NOTSET   SC75
     WTSCPLX7 CONNECTED            WTSCPLX7 TCPIP NONE NOTSET   NOTSET   SC75
     WTSCPLX9 CONNECTED            WTSCPLX9 TCPIP NONE NOTSET   NOTSET   SC75
     WTSCPOK  UNCONNECTED          WTSCPLX1       NONE NOTSET   NOTSET   SC75
     WTSC75J3 OWNNODE              WTSC75J3       NONE NOTSET   OWNNODE  SC75
```

Figure 30-9 Node (NO) display in a JES3 environment

30.2.5 Initiator (INIT) display

The Initiator (INIT) panel allows authorized users to display information about JES-managed and WLM-managed initiators; see Figure 30-10.

```
  Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
SDSF INITIATOR DISPLAY  (ALL)                          LINE 1-7 (7)
COMMAND INPUT ===>                                          SCROLL ===> HALF
PREFIX=VAIN*  DEST=*  OWNER=*  FILTERS=3  SYSNAME=*
ACTION=//-Block,=-Repeat,+-Extend,D-Display,DL-DisplayLong,P-Stop,S-Start
NP   ID       ResType Status       SysName  JobName  Stepname JobID    C
     S        GROUP   ON           SC74
     S        CLASS   ON           SC74                                 S
     T        CLASS   ON           SC74                                 T
     S        GROUP   ON           SC75
     S        CLASS   ON           SC75                                 S
     S        INIT    ACTIVE       SC75     CLASSS2  XYZZY    JOB21135 S
     S        INIT    ACTIVE       SC75     CLASSS   XYZZY    JOB20889 S
     S        INIT    ACTIVE       SC75     CLASSS3  XYZZY    JOB21136 S
     T        CLASS   ON           SC74                                 T
```

Figure 30-10 INIT display in a JES3 environment

30.2.6 Job Zero (J0) display

The Job Zero (J0) panel allows authorized users to display information about JES3 job 0. It is available only in a JES3 environment. With this panel, you can work with data sets that were created by JES3. See Figure 30-11.

```
  Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
SDSF JOB 0 DISPLAY                  LINES 187,048      DATA SET DISPLAYED
COMMAND INPUT ===>                                          SCROLL ===> HALF
ACTION=//-Block,=-Repeat,+-Extend,?-JDS,C-Cancel,D-Display,H-Hold,O-Release,
ACTION=P-Purge,Q-OutDesc,S-Browse,SB-ISPFBrowse,SE-ISPFEdit,X-Print,
ACTION=XC-PrintClose,XD-PrintDS,XDC-PrintDSClose,XF-PrintFile,
ACTION=XFC-PrintFileClose,XS-PrintSysout,XSC-PrintSysoutClose
NP   DSPNAME  DSID Owner    C CC PrMode   Burst Forms    FCB  UCS  Wtr      Fla
     DC         18 JES3     A  1 LINE     C     STD      STD3 ANY           NON
     JOB17005   20 JES3     A  1 LINE     C     STD      STD3 ANY           NON
     DISPLAY    21 JES3     A  1 LINE     C     STD      STD3 ANY           NON
     DC         23 JES3     A  1 LINE     C     STD      STD3 ANY           NON
     JOB20658   81 JES3     A  1 LINE     C     STD      STD3 ANY           NON
     DISPLAY   157 JES3     A  1 LINE     C     STD      STD3 ANY           NON
```

Figure 30-11 Job zero (J0) display

30.3 New panels for both JES2 and JES3

In z/OS V1R13, SDSF adds two new panels for displaying information about networking devices and connections. The displays are available for both JES2 and JES3.

Both new panels use SSI function code 83 to obtain the information from JES and therefore do not rely on the SDSF server address space for sysplex-wide data.

30.3.1 Network Server (NS) display

The Network Server (NS) panel allows authorized users to display information about server-type networking devices on the node:

- NETSERV devices used to communicate between JES and TCP/IP
- LOGON devices used to communicate between JES2 and VTAM
- BDT instances used to communicate between JES3 and VTAM

Figure 30-12 shows an example of the NS panel.

```
 Display  Filter  View  Print  Options  Search  Help
------------------------------------------------------------------------------
SDSF NS DISPLAY  SC75                                    LINE 1-2 (2)
COMMAND INPUT ===>                                          SCROLL ===> HALF
ACTION=//-Block,=-Repeat,+-Extend,C-Cancel,D-Display,E-Restart,K-SysCancel,
ACTION=KD-SysCancelDump,L-Fail,LD-FailDump,S-Start,X-CallTCP,Z-SysForce
NP   DEVICE       Status   DSPName  Stack    CTr VTr JTr ASID  SrvJobNm IPName
     JES3NS       ACTIVE   JOB20820 TCPIP    NO  NO  NO  003B  JOB20941 WTSC75.
     JES3NS9      INACTIVE          TCPIP    NO  NO  NO                 WTSC75.
```

Figure 30-12 NS display in a JES3 environment

30.3.2 Network Connection (NC) display

The Network Connection (NC) panel allows authorized users to display information about networking connections to an adjacent node:

- SOCKET devices that represent a TCP/IP networking connection
- APPL devices that represent a SNA connection (JES2 only)
- Active BSC NJE lines (JES3 only)
- Associated NJE transmitters and receivers

The NC panel is shown in Figure 30-13.

```
 Display  Filter  View  Print  Options  Search  Help
------------------------------------------------------------------------------
SDSF NC DISPLAY  SC75                                    INVALID COMMAND
COMMAND INPUT ===>                                          SCROLL ===> HALF
ACTION=//-Block,=-Repeat,+-Extend,C-Cancel,D-Display,SN-StartNetComm
NP   DEVICE       Status   Type ANode    JobName  JobID    JType    Owner    Pr
     @0000002     ACTIVE   TCP  WTSCPLX9
     @0000002.JR1 INACTIVE
     @0000002.JT1 INACTIVE
     @0000002.OR1 INACTIVE
     @0000002.OT1 INACTIVE
     WTSCNET      ACTIVE   TCP  WTSCNET
     WTSCNET.JR1  INACTIVE
     WTSCNET.JT1  INACTIVE
     WTSCNET.OR1  INACTIVE
     WTSCNET.OT1  INACTIVE
     WTSCPLX1     INACTIVE TCP  WTSCPLX1
     WTSCPLX4     ACTIVE   TCP  WTSCPLX4
     WTSCPLX4.JR1 INACTIVE
     WTSCPLX4.JT1 INACTIVE
     WTSCPLX4.OR1 INACTIVE
     WTSCPLX4.OT1 ACTIVE                 VAINI    JOB20994 JOB
     WTSCPLX7     INACTIVE TCP  WTSCPLX7
```

Figure 30-13 NC display in a JES3 environment

By default, the NC panel lists the associated NJE transmitters and receivers. To suppress the NJE transmitters and receivers from the list, invoke the NC display with the SHORT parameter; for example **NC SHORT**.

30.4 Using XCF for sysplex-wide data

Until z/OS V1R13, SDSF servers communicated through WebSphere MQ to provide sysplex-wide data on SDSF panels. WebSphere MQ must be up and operational on each system in the SDSF server group. The WebSphere MQ queue managers must be configured to communicate using channels, and special queues must be defined for SDSF use.

In z/OS V1R13, most of the SDSF panels that display JES data are changed to request data from JES through SSI function codes, which provide sysplex-wide data in return. Other panels, which mostly display system information, now use XCF rather than WebSphere MQ to communicate between systems. Using XCF for the sysplex-wide data is preferable because XCF is always present and no configuration is required. When running in a JES3 environment or a JES2 environment with all systems at z/OS V1R13 and later, the following SDSF device panels provide sysplex-wide data, without requiring the SDSF server address space to be running:

INIT Initiator display
LI Line display
NO Node display
PR Printer display
PUN Punch display
RDR Reader display
SO Spool offload display (JES2 only)

An active SDSF server address space is required on each system in the sysplex to provide sysplex-wide data for the following panels:

CK Health checker display
ENC Enclave display
PS z/OS UNIX process display
RM Resource monitor display (JES2 only)

Figure 30-14 shows the health checker (CK) display, with data from multiple systems. For example, it shows the time that the VSM_CSA_THRESHOLD check is next scheduled to run in systems SY1, SY3, and SY4.

```
SDSF HEALTH CHECKER DISPLAY  (ALL)              LINE 1-35 (41)
COMMAND INPUT ===>                             SCROLL ===> CSR
PREFIX=*  DEST=(ALL)  OWNER=*  SORT=Interval/A  SYSNAME=*
NP   NAME                     CheckOwner SysName  NextSch-Int
     VSM_CSA_THRESHOLD        IBMVSM     SY1          0:00:13
     VSM_CSA_THRESHOLD        IBMVSM     SY3          0:01:28
     VSM_CSA_THRESHOLD        IBMVSM     SY4          0:03:59
     CNZ_TASK_TABLE           IBMCNZ     SY1          0:05:06
     RSM_HVSHARE              IBMRSM     SY1          0:05:06
     RSM_MAXCADS              IBMRSM     SY1          0:05:06
     VSM_SQA_THRESHOLD        IBMVSM     SY1          0:05:06
     CNZ_TASK_TABLE           IBMCNZ     SY3          0:11:28
     RSM_HVSHARE              IBMRSM     SY3          0:11:28
     RSM_MAXCADS              IBMRSM     SY3          0:11:28
     VSM_SQA_THRESHOLD        IBMVSM     SY3          0:11:28
     CNZ_TASK_TABLE           IBMCNZ     SY4          0:13:59
```

Figure 30-14 Example of an SDSF panel showing data from multiple systems

30.4.1 Enabling SDSF server XCF communication

XCF communications is enabled by default when all SDSF servers are running on z/OS V1R13 or higher. If not all systems are at z/OS V1R13, SDSF falls back to WebSphere MQ communications. SDSF does not merge data from the XCF and WebSphere MQ protocols. All target systems must be accessible through XCF, otherwise WebSphere MQ is used. The interaction between the SDSF user, the SDSF servers and XCF is shown in Figure 30-15.

By default, when all systems that you want to include are at the z/OS V1R13 level, SDSF uses XCF to communicate between SDSF servers, and does not use a server group defined in ISFPARMS.

When one or more systems that you want to include is at the z/OS V1R12 or lower level, the server group defined in ISFPARMS is also required, along with WebSphere MQ.

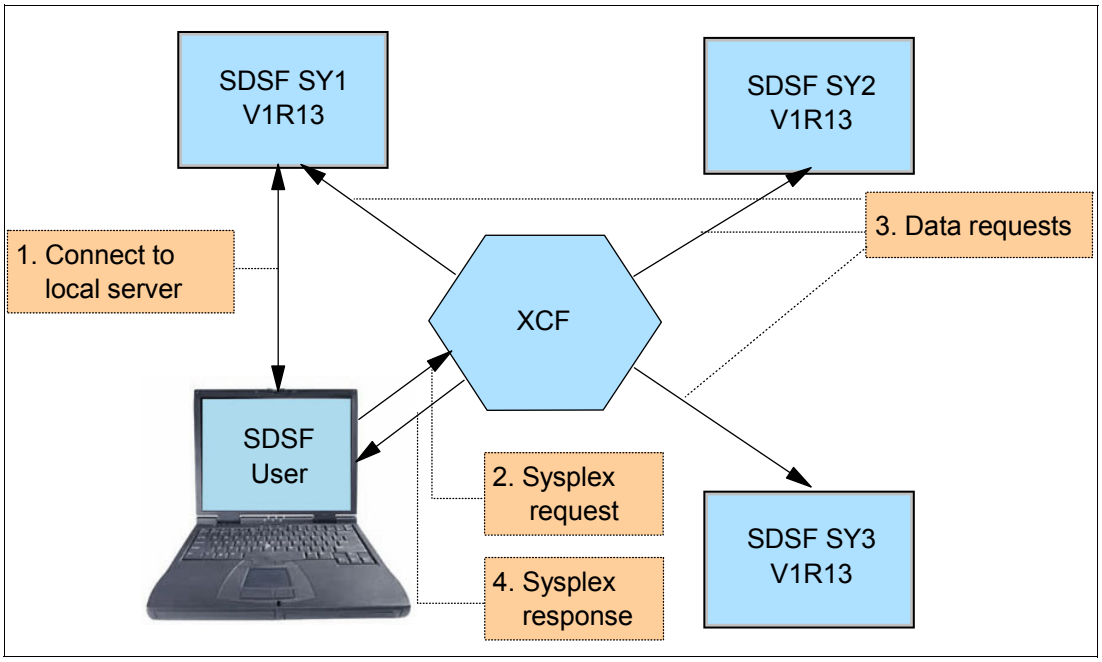


Figure 30-15 SDSF server XCF communications

ISFPARMS statements for XCF communication

XCF communication between SDSF servers requires a common server name for all systems. If all your SDSF servers do not have the same name, you can use the XCFSRVNM parameter on the CONNECT statement in ISFPARMS to meet this requirement. You can exclude a system from a sysplex-wide panel by specifying XCFSRVNM(NONE) on the CONNECT statement in the ISFPARMS for that system.

The CONNECT statement defines the server connection, including whether the server is the default SDSF server and the XCF application server name. It can also request that XCF not be used to provide sysplex data. The format of the CONNECT statement is shown in Figure 30-16.

```
CONNECT DEFAULT(YES|NO|COND),
        XCFSRVNM(SAME|server-name|NONE)
```

Figure 30-16 Format of the CONNECT statement in ISFPARMS

Use the DEFAULT parameter to indicate whether the server is identified as the default server. Users are connected to the default server unless another server is specified, either with the SERVER keyword in the assembler ISFPARMS, or with the SERVER parameter on the SDSF command used to access SDSF. The DEFAULT parameter takes the following values:

YES This indicates that this server is to be made the default server unconditionally, replacing any other default server if necessary.

NO This indicates that the server is not to be made the default. If the server had previously been made the default, it will remain the default until it terminates. This is the case if the DEFAULT parameter is not specified.

COND This indicates that the server will be made the default unless another default server is already defined.

Use the XCFSRVNM parameter to define the XCF application server name, or to request that XCF is not to be used to provide sysplex-wide data. Explanation of the possible XCFSRVNM values is as follows:

SAME This indicates that the XCF application server name is derived from the SDSF server name. This is the default, and so is the case if the XCFSRVNM parameter is omitted.

When you use SAME, all SDSF servers that are to participate in sysplex requests must have the same name. (The server name is either the job name or the started task ID.)

server-name This specifies the customized portion of the XCF application server name, ISFSRVR.server-name. server-name can be up to 8 characters, and can consist of alphabetic characters, numeric characters, and the national characters @, #, or $.

When you use server-name, the names of the SDSF servers that are to participate in sysplex requests do not need to be the same.

NONE This indicates that the server should not identify itself to XCF and thus will not respond to sysplex requests through XCF.

A value of NONE for the local server connection properties system (the system the user is logged on to) causes SDSF to attempt to revert to using server groups and WebSphere MQ to provide sysplex-wide data.

A value of NONE for a remote system requests that this remote system not be included in the sysplex-wide data.

The example in Figure 30-17 shows how to define that the server processing ISFPRMxx is the default server, replacing any other default server that might have been defined previously, and that the XCF application server name is derived based on the SDSF server name.

```
CONNECT DEFAULT(YES),
        XCFSRVNM(SAME)
```

Figure 30-17 Example CONNECT statement in ISFPARMS

Checking the status of XCF communications

A new COMMX keyword is added to the **WHO** command in z/OS V1R13. The COMMX keyword displays the current XCF communication mode status. The status can be either Enabled, Disabled, Suspended, or Not-Available; see Figure 30-18.

```
 Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
HQX7780 -----------------  SDSF PRIMARY OPTION MENU  -------------------------
COMMAND INPUT ===>                                            SCROLL ===> CSR
USERID=PELEG,PROC=IKJACCNT,TERMINAL=SC38TC99,GRPINDEX=1,GRPNAME=ISFSPROG,
MVS=z/OS 01.13.00,JES=z/OS1.13,SDSF=HQX7780,ISPF=6.3,RMF/DA=NOTACC,SERVER=YES,
SERVERNAME=SDSF,JESNAME=JES2,MEMBER=SC74,JESTYPE=JES2,SYSNAME=SC74,
SYSPLEX=PLEX75,COMM=NOTAVAIL,COMMX=ENABLED
H     Held output queue                 RDR   Readers
ST    Status of jobs                    LINE  Lines
                                        NODE  Nodes
LOG   System log                        SO    Spool offload
SR    System requests                   SP    Spool volumes
MAS   Members in the MAS                NS    Network servers
JC    Job classes                       NC    Network connections
SE    Scheduling environments
RES   WLM resources                     RM    Resource monitor
ENC   Enclaves                          CK    Health checker
PS    Processes
                                        ULOG  User session log
END   Exit SDSF
```

Figure 30-18 Output of the WHO command with the COMMX= keyword

Displaying XCF communications configuration

The SDSF server **DISPLAY** operator command is enhanced to indicate whether XCF communication is configured, and to display XCF usage statistics. Figure 30-19 shows an example of the SDSF server **DISPLAY** command, which displays information about the server, including the status of the server and server communications.

```
SY1           f sdsf,d
SY1 SO000006  ISF304I Modify DISPLAY command accepted.
SY1 SO000006  ISF312I SDSF Display
   Server status:  Active       Default: Yes
   Communications: Active
   Parms: ISFPRM02 / SYS2.PARMLIB
   XCF Communications: Configured
```

Figure 30-19 SDSF server DISPLAY command

In addition, Figure 30-19 shows an example of the SDSF server **DISPLAY COMM** operator command. The command displays information about the servers, including their ID, status, and the system they are processing.

```
SY1           f sdsf,d,c
SY1 S0000006  ISF304I Modify DISPLAY command accepted.
SY1 S0000006  ISF310I SDSF Communications
   Id Server   Status       System   JESN Member ReqsProc
   01 SDSF     Active/L     SY1      JES2 SY1           4
   02 SDSF     Defined      SY2      JES2 SY2           1
SY1 S0000006  ISF315I SDSF XCF Communications
   Application server name: ISFSRVR SDSF
   Tasks Active: 000  Idle: 010
   Sends: 0000000027  Receives: 0000000038
```

Figure 30-20 SDSF server DISPLAY COMM operator command

30.4.2 Using z/OS V1R12 compatibility mode

When one or more systems that you want to include is at the z/OS V1R12 or lower level, you must obtain sysplex-wide data for the CK, ENC, PS, and RM panels as in z/OS V1R12 SDSF and earlier releases. This requires an SDSF server on each system, a server group defined in ISFPARMS, and WebSphere MQ for communication between servers.

Perform these steps:

1. Configure WebSphere MQ for SDSF sysplex support.
2. If both the system you log on to and at least one other system are at the z/OS V1R12 or lower level, request that SDSF run in Z12 compatibility mode. When running in Z12 compatibility mode, SDSF reverts to using the server group and WebSphere MQ for communications.

Attention: If you do not set the compatibility mode to Z12 and both the system you log on to and at least one other system are at the z/OS V1R13 level, then SDSF will use XCF for communications and any lower-level systems will be omitted.

The SET CMODE command

The **SET CMODE** command controls the mode that SDSF uses for communication to provide sysplex-wide data on SDSF panels. It can be issued from any SDSF panel. Under ISPF, the value is saved across sessions. The format of the **SET CMODE** command is shown in Figure 30-21.

```
SET CMODE (Z12|Z13|?)
```

Figure 30-21 Format of the SET CMODE command

Issuing **SET CMODE** with no parameters specifies the default for the current release. The parameters are used as follows:

? This opens the Set Communications Mode pop-up panel.

Z12 This specifies the SDSF will revert to using WebSphere MQ for communications if one or more systems is z/OS V1R12 or lower. Systems must be in the server group.

Z13 This specifies the sysplex support that was introduced in z/OS V1R13 SDSF. It uses XCF for communications and does not use the server

group. Systems that you want to be included must be at least z/OS V1R13. This is the default.

For example, to set the communications mode to z/OS V1R13, use the command shown in Figure 30-22.

```
SET CMODE Z13
```

Figure 30-22 Example of using the SET CMODE command

The Comm.Release.Mode property in ISFPARMS

System programmers can use the **Comm.Release.Mode** property in ISFPARMS to set the mode that SDSF uses for communication to provide sysplex-wide data on SDSF panels. The possible values are 1 or 2.

A value of 1 sets the communication mode to Z12, which requests that SDSF revert to using WebSphere MQ for communications if one or more systems is z/OS V1R12 or lower. Systems must be in the server group.

A value of 2 sets the communication mode to Z13, which requests that SDSF use the sysplex support that was introduced in z/OS V1R13 SDSF. SDSF uses XCF for communications and does not use the server group. Systems that you want to be included must be at least z/OS V1R13. This is the default.

30.5 OPERLOG colors

In z/OS V1R13 SDSF, messages are displayed on the OPERLOG panel in the same color, highlighting, and intensity that was assigned to them when they were issued. An example is shown in Figure 30-23 on page 686. The message colors can be customized using the **SET SCREEN** command.

```
 Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
 SDSF OPERLOG  DATE 04/25/2011     0 WTORS                      COLUMNS 52- 131
 COMMAND INPUT ===>                                            SCROLL ===> CSR
000090  IAT6395 00002 REQUEST(S) - WAITING FOR A MAIN, CLASS, OR GROUP
000090  DSNA679I  DSNA6BUF THE ADMIN SCHEDULER DB9YADMT CANNOT ACCESS TASK
        LIST SYSIBM.ADMIN_TASKS, REASON=
000090  SQLCODE -981, SQLSTATE 57015, 00C12219
000090  HZS0001I CHECK(IBMCSV,CSV_APF_EXISTS): 023
000090  CSVH0957E Problem(s) were found with data sets in the APF list.
000090 *HZS0003E CHECK(IBMRACF,RACF_SENSITIVE_RESOURCES): 024
000090  IRRH204E The RACF_SENSITIVE_RESOURCES check has found one or
000090  more potential errors in the security controls on this system.
000290  IEA631I  OPERATOR ZANON    NOW INACTIVE, SYSTEM=SC74    , LU=SC38TC32
000090  DSNA679I  DSNA6BUF THE ADMIN SCHEDULER DB9YADMT CANNOT ACCESS TASK
        LIST SYSIBM.ADMIN_TASKS, REASON=
000090  SQLCODE -981, SQLSTATE 57015, 00C12219
```

Figure 30-23 OPERLOG panel in color

Customizing OPERLOG colors

To customize the message colors on the OEPRLOG panel, use the **SET SCREEN** command. The Set Screen Characteristics window pops up, as shown in Figure 30-24.

```
  Display  Filter  View  Print  Options  Search  Help
 -------------------------
 SDSF OPERLOG  DATE 04/25/2 ┌──────────────────────────────────────────┐
 COMMAND INPUT ===> SET SCR │          Set Screen Characteristics      │
 00090  IAT6100 ( DEMSEL )  │                                          │
 00290  BPXP024I BPXAS INIT │ Select the elements that you want to customize.
        0030                │                                          │
 00090  AIR022I REQUEST TO  │ 2   1.  Basic settings and tabular panels
 00090  CHECK NAME= PFA_LOG │     2.  OPERLOG panel                    │
 00090  UNIX SIGNAL RECEIVE │                                          │
 00090  IAT6395 00002 REQUE │                                          │
 00090  DSNA679I  DSNA6BUF  │ F1=Help  F12=Cancel                      │
        LIST SYSIBM.ADMIN_TASKS, REASON=
000090  SQLCODE -981, SQLSTATE 57015, 00C12219
000090  HZS0001I CHECK(IBMCSV,CSV_APF_EXISTS): 023
000090  CSVH0957E Problem(s) were found with data sets in the APF list.
000090 *HZS0003E CHECK(IBMRACF,RACF_SENSITIVE_RESOURCES): 024
```

Figure 30-24 The Set Screen Characteristics pop-up window

Select option **2** for the OPERLOG panel. Another pop-up window appears, as shown in Figure 30-25 on page 687. This window allows you to select the color, highlight, and intensity for each message descriptor code. It also allows you to completely turn off OPERLOG colors.

```
  Display  Filter  View  Print  Options  Search  Help
 -------
 SDSF OP ┌──────────────────────────────────────────────────────────┐
 COMMAND │           Set Screen Characteristics: OPERLOG Panel      │
 00090   │                                                  More:   +
 00290   │ Use color and highlighting    1   1. Yes   2. No         │
         │                                                          │
 00090   │ Type values to override the original color and highlighting.
 00090   │ Press F5/17 to see changes.                              │
 00090   │                                                          │
 00090   │ Descriptor code                 Color    Highlight  Intensity
 00090   │  1 - System failure                                      │
         │  2 - Immediate action required   red                     │
 00090   │  3 - Eventual action required                            │
 00090   │  4 - System status                                       │
 00090   │  5 - Immediate command response                          │
 00090 * │  6 - Job status                                          │
 00090   │  7 - Task-related                                        │
 00090   │  8 - Out of line                                         │
 00290   │  9 - Operator's request                                  │
 00090   │ 10 - Not defined                                         │
         │ 11 - Critical eventual action                            │
 00090   │ 12 - Important information                               │
 00090   │  F1=Help  F5=Refresh  F6=Default  F11=Cuaattr  F12=Cancel │
         └──────────────────────────────────────────────────────────┘
 00090  SQLCODE -981, SQLSTATE 57015, 00C12219
000090  DSNA679I  DSNA6BUF THE ADMIN SCHEDULER DB9YADMT CANNOT ACCESS TASK
```

Figure 30-25 Customize OPERLOG panel colors

30.6 Reading the OPERLOG from SDSF/REXX

In z/OS V1R12, SDSF added support to access the SYSLOG panel from REXX through the ISFLOG command. However, the ISFLOG command did not support reading the OPERLOG.

In z/OS V1R13, the ISFLOG SDSF/REXX command is enhanced to provide access to the OPERLOG panel and the SYSLOG panel. The support is provided using a new TYPE parameter for the ISFLOG command, which indicates whether to read the SYSLOG or OPERLOG. The full format of the ISFLOG command is shown in Figure 30-26.

```
Address SDSF "ISFLOG ALLOC|READ TYPE(log-type) (option)"
```

Figure 30-26 Format of the ISFLOG SDSF/REXX command

The value of **log-type** can be either SYSLOG or OPERLOG. The code sample in Figure 30-27 shows how to use the new TYPE parameter to read the OPERLOG from REXX.

The example also demonstrates the use of the special REXX variables that are supported for the ISFLOG command. The example uses the special variables to limit the OPERLOG output to a maximum of 1,000 lines, starting yesterday.

```
/* REXX */

rc = isfcalls('on')

isfdate         = "mmddyyyy /"
yesterday       = date("C") - 1             /* yesterday */
isflogstartdate = date("U", yesterday, "C") /* yesterday mm/dd/yy */
isflogstarttime = time("N")                 /* current time */
isflogstopdate  = date("U")                 /* current date mm/dd/yy */
isflogstoptime  = time("N")                 /* current time */
isflinelim      = 1000

address SDSF "isflog read type(operlog)"
do ix = 1 to isfline.0
    say isfline.ix
end

/* print debug messages */
do ix = 1 to isfmsg2.0
    say isfmsg2.ix
end

rc = isfcalls('off')
```

Figure 30-27 Sample SDSF/REXX program to read the OPERLOG

Message ISF757I in Figure 30-28 shows that SDSF/REXX processed the special variables as specified in the program (highlighted in bold). The figure shows a sample output generated by the preceding REXX program. We ran the REXX program on 2011.116 at 10:16. The output starts from 2011.115 at 10:16 and is limited to 1000 lines.

```
N 4020000 SC75     2011115 10:16:49.00 DB9YADMT 00000090  DSNA679I  DSNA6BUF TH
E ADMIN SCHEDULER DB9YADMT CANNOT ACCESS TASK
S                                                         LIST SYSIBM.ADMIN_TAS
KS, REASON=
N 4020000 SC75     2011115 10:16:49.00 DB9YADMT 00000090  SQLCODE -981, SQLSTAT
E 57015, 00C12219:
N 4000000 SC75     2011115 10:17:23.04 HZSPROC  00000090  IAT1613 JOB HZSPROC
(JOB21274) SYSTEM MESSAGE COUNT IS 65,000
M 4040000 SC75     2011115 10:17:23.59 HZSPROC  00000090  HZS0001I CHECK(IBMCSV
,CSV_APF_EXISTS): 012
E                                          012 00000090  CSVH0957E Problem(s)
were found with data sets in the APF list.
M 4040000 SC75     2011115 10:17:25.59 HZSPROC  00000090 *HZS0003E CHECK(IBMRAC
F,RACF_SENSITIVE_RESOURCES): 013
...
ISF754I Command 'SET DATE MMDDYYYY /' generated from associated variable ISFDAT
E.
ISF757I Variable ISFLINELIM being processed with value '1000'.
ISF757I Variable ISFLOGSTARTTIME being processed with value '10:16:49'.
ISF757I Variable ISFLOGSTARTDATE being processed with value '04/25/11'.
ISF757I Variable ISFLOGSTOPTIME being processed with value '10:16:49'.
ISF757I Variable ISFLOGSTOPDATE being processed with value '04/26/11'.
ISF770W Request limit 1000 from variable ISFLINELIM reached, processing
stopped.
ISF767I Request completed.
```

Figure 30-28 OPERLOG messages read by a SDSF/REXX program

31

JES2 enhancements

JES2, or Job Entry Subsystem 2, is a functional extension of the HASP II program that receives jobs into the system and processes all output data produced by the job. So, what does all that mean? Simply stated, JES2 is that component of MVS that provides the necessary functions to get jobs into, and output from, the MVS system. It is designed to provide efficient spooling, scheduling, and management facilities for the MVS operating system.

This chapter describes the enhancements to JES2 for z/OS V1R13, as follows:

- Managing data sets on volumes for the JES2 spool
  - JES2 spool migration
- JES2 Batch Modernization enhancements
  - JCL instream data sets in PROCs
  - Support for JOBRC
  - Evict job on a step boundary
  - SPIN any SPIN data set
- Enhancements to SSI in support of SDSF
- JES device and property SSI enhancements

31.1 Spool migration

A JES2 spool migration moves an existing JES2 spool volume (an extent or data set) to a new spool volume, or merges an existing volume with another existing spool volume. Migrating a spool volume to a new spool volume is called a *move migration*. Migrating a spool volume to an existing spool volume is called a *merge migration*.

As described here, the **$MSPL(volser,)** command queues a request to move one or more existing spool volumes to a new spool volume, or to merge them with an existing spool volume:

- For move requests, the target spool volume specified as a volser is not recognized by JES2. When the move completes, the source spool volumes are no longer recognized by JES2.
- For merge requests, the specified target must be a spool volume that is recognized by JES2. When the merge completes, the source spool volumes remain in the MAPPED state until all users of the source spool volumes have been purged.

Note: This command requires system authority. The description of the **$T RDRnn** command explains how to ensure that the authority of the appropriate MVS system command group is assigned.

Spool migration types

Both types of spool migration enhance JES2 spool configuration by providing the following functionality:

- Increasing or reducing the total number of spool volumes
- Increasing or reducing the size of spool volumes
- Removing spool volumes without altering any spool pointers (MTTRs or MQTRs)
- Copying of data from one spool volume to another while address spaces are actively reading and writing data to the spool volumes
- Merging of volumes and track group map onto another spool volume
- Creating a new spool volume

MERGE migration

A MERGE migration copies an existing source volume to free space on a target volume, as shown in Figure 31-1. Upon completion, the source volume becomes a mapped volume. The volume remains mapped until all jobs and SYSOUT that have space on the Source Volume are purged. It then goes away (no longer exists).

Restrictions: The following restrictions need to be considered:

- The Target Volume must be Active (can be Reserved).
- The Records Per Track of the Target Volume cannot be less than the Source Volume.
- The Target Volume must use relative addressing.
- The Target Volume cannot be actively extending.

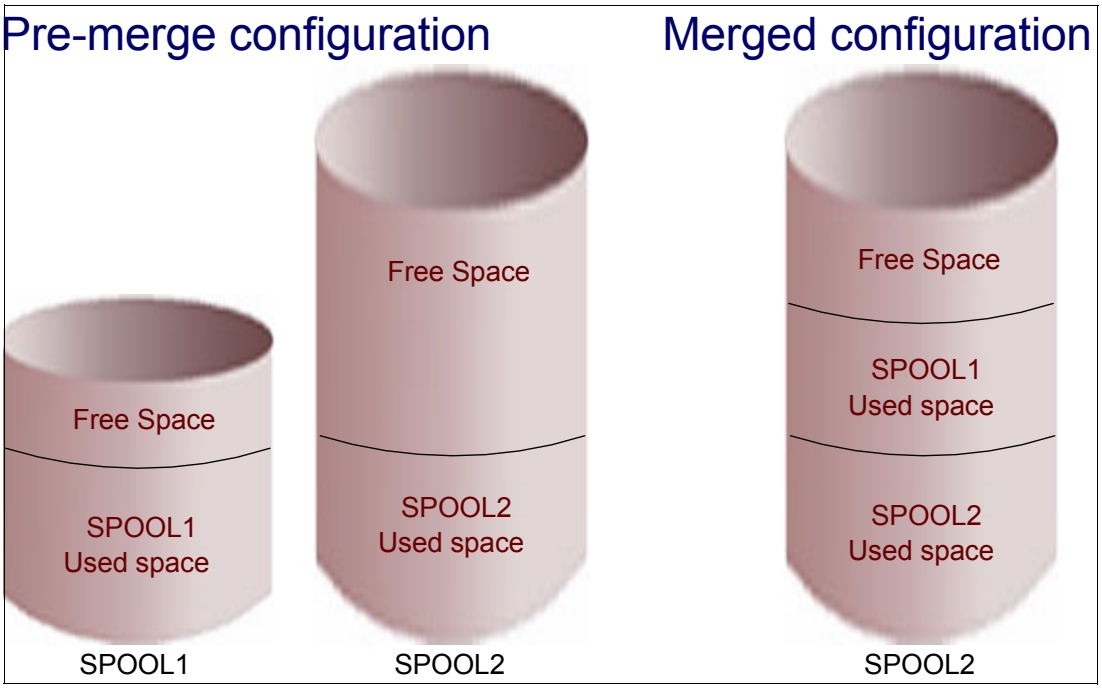


Figure 31-1 Copy an existing source volume to free space on a target volume

COPY migration

A MOVE migration copies a single Inactive (halted) source volume to a new target volume. Upon completion, the target volume inherits the $DAS structure of the source volume.

Restriction: The source volume must be Inactive. You cannot move an absolute volume (instead, do a merge). You need to use the SPACE= parameter. The target volume will inherit the source volume tracks per track group value.

31.1.1 Managing the migration

To manage the migration of spool volume data and the access to that data, JES2 runs various operations on all members of a MAS. These operations communicate messages through JESXCF services. JESXCF services require one XCF group per active migration to identify the messages that are specific to each active migration.

Note: JES2 limits the number of simultaneous migrations to five due to XCF considerations. If a spool migration is requested and an XCF group is not available, JES2 refuses the request and issues a $HASP808 message with return code 57 or 58. To determine the number of available XCF groups, run the **`D XCF,COUPLE`** command.

Be aware of the following restrictions:

- The source volume cannot be a mapped target.
- The source volume cannot be actively migrating or extending.
- The source volume cannot be stunted.
- You must be at z11 checkpoint mode.
- All members must be at release V1R13.

Spool migration states with the $DSPL command

The new spool volume proceeds through the following states:

MIGRATING The spool volume is a source of an active migration.

MAPPED The spool volume has been migrated and the corresponding data set is eligible for deletion. The volume will remain MAPPED until all jobs and SYSOUT that have space on the volume have been purged.

The new spool volume goes through the following migration phases. PHASE is a parameter of the **$DSPL** command that filters the migrating volumes by current phase types as follows:

PENDING Awaiting start of migration.

INITIALIZING General migration configuration work is being done, such as creating subtasks, data structures, and XCF mailboxes.

SETUP Setup for a migration is being done. All MAS members participate in this process.

COPY The data set on the source spool volume has been migrated to the target spool volume. Runtime changes are coordinated and tracked by the migration.

CATCHUP Tracks that were changed by runtime operations during the COPY phase are being recopied.

CLEANUP General cleanup is being done at the end of the migration.

CANCEL Migration subtask cleanup is being done because an operator cancelled the active migration, or the migration process encountered an error.

BACKOUT Updates are being backed out because an operator cancelled the active migration, or the migration process encountered an error.

Note: An active Migration can be cancelled using the **$MSPL(source),CANCEL** command. However, a migration cannot be cancelled during the CATCHUP phase or any CLEANUP phase. Upon completion of a cancel, the source volume will remain in draining state.

Check volume space for migration

You can use the **$D SPOOL,MIGDATA** command to determine if enough space exists on a target volume to accommodate a planned source volume or volumes. The command displays the largest contiguous free area on each spool volume, as shown here:

```
$D SPOOL,MIGDATA
$HASP893 VOLUME (SPOL3)
$HASP893 VOLUME (SPOL3) MIGDATA=(SPACE_USED=40000,LARGEST_FREE=10000)
$HASP893 VOLUME (SPOL5)
$HASP893 VOLUME (SPOL5) MIGDATA=(SPACE_USED=20000,LARGEST_FREE=10000)
```

To display all volumes having contiguous free space greater than 3000 cylinders, issue the following command:

```
$D SPOOL,MIGDATA=LARGEST_FREE>3000,MIGDATA
$HASP893 VOLUME (SPOL2)
$HASP893 VOLUME (SPOL2) MIGDATA=(LARGEST_FREE=4000)
$HASP893 VOLUME (SPOL3)
$HASP893 VOLUME (SPOL3) MIGDATA=(LARGEST_FREE=7000)
```

Important: The extended volume can become stunted if the JES2 Track Group Map (TGM) space is not available. Stunted volumes cannot be the source volume or target volume of a migration. Use the **$D SPOOLDEF,TGSPACE** command to expand the TGM size.

A checkpoint reconfiguration can be needed if there is not enough free space in the checkpoint. Use the **$DCKPTSPACE** command to see how much free space is available.

31.1.2 Spool migration examples

The example shown in Figure 31-2 demonstrates the use of the **$MSPL(volser,)** command usage. The **$MSPL(volser,)** command can be used to migrate one or more existing spool volumes (volsers) to a new spool volume. Note the following points regarding this example:

- When you issue the **$MSPL** command, JES2 creates a new data set on the new spool volume J2SPL1. The default data set name is defined by the **$SPOOLDEF** command.
- The data set size is specified by the SPACE=MAX parameter, indicating the largest size possible considering the available disk space and JES2 architecture limits.
- After the spool volume J2SPL1 is formatted, its state is RESERVED and it remains in that state until the **$TSPOOL RESERVED=NO** command is executed.
- The spool volume SPOOL6 is in the INACTIVE state prior to the migration and transitions to the MIGRATING state during the migration. SPOOL6 is removed from the SPOOL configuration (drained) at the completion of the migration.
- The data set previously associated with the spool volume SPOOL6 can now be deleted.

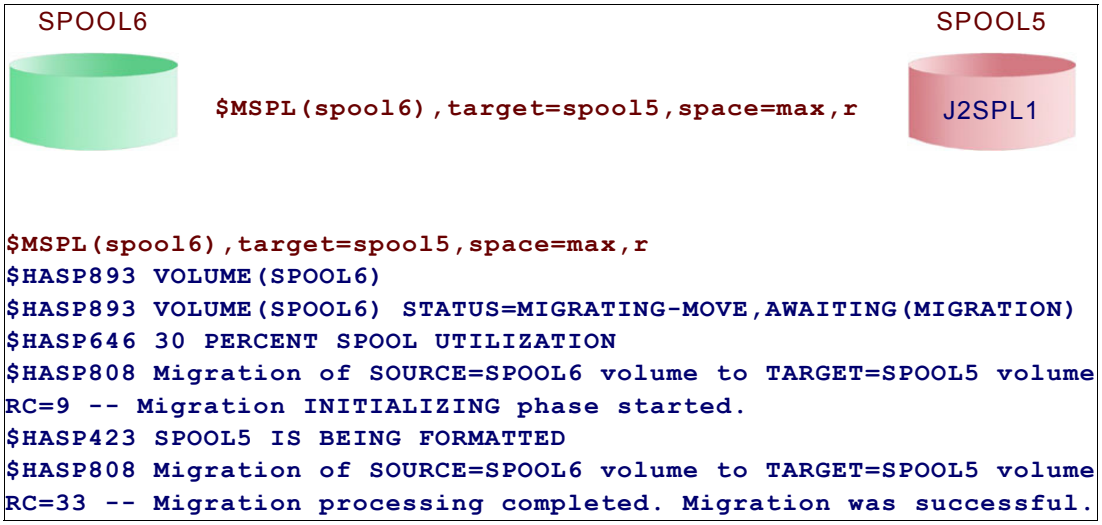


Figure 31-2 $MSPL command example

In the figure, the $HASP808 message displays the migration progress. The **$DSPL** command displays the status of the target volume:

```
$DSPL(spool5),status,percent,reserved
$HASP893 VOLUME(SPOOL5) STATUS=ACTIVE,PERCENT=0,RESERVED=YES
$HASP646 30 PERCENT SPOOL UTILIZATION
```

Spool migration example

The **$MSPL(volser,)** command can also be used to merge one or more existing spool volumes (volsers) with a single existing spool volume, as shown in Figure 31-3. Note the following points regarding this example:

- When the **$MSPL(spool2,spool3,spool4),target=spool5** command is used as shown in the figure, JES2 merges spool volumes spool2, spool3, and spool4 with the existing active spool volume spool5.
- Spool volume spool2 is merged first, followed by spool3 and then spool4.
- Spool volumes spool3 and spool4 migration phases are pending while spool2 is migrating.
- After migration is complete, spool2, spool3, and spool4 are left in the MAPPED state.
- At this point, the data sets previously associated with volumes spool2, spool3, and spool4 can be deleted.
- Both spool volumes will remain in a MAPPED state until all jobs using the volumes are purged. At that point spool2, spool3, and spool4 are removed from the spool configuration (drained) and the corresponding spool extent number can be reused.

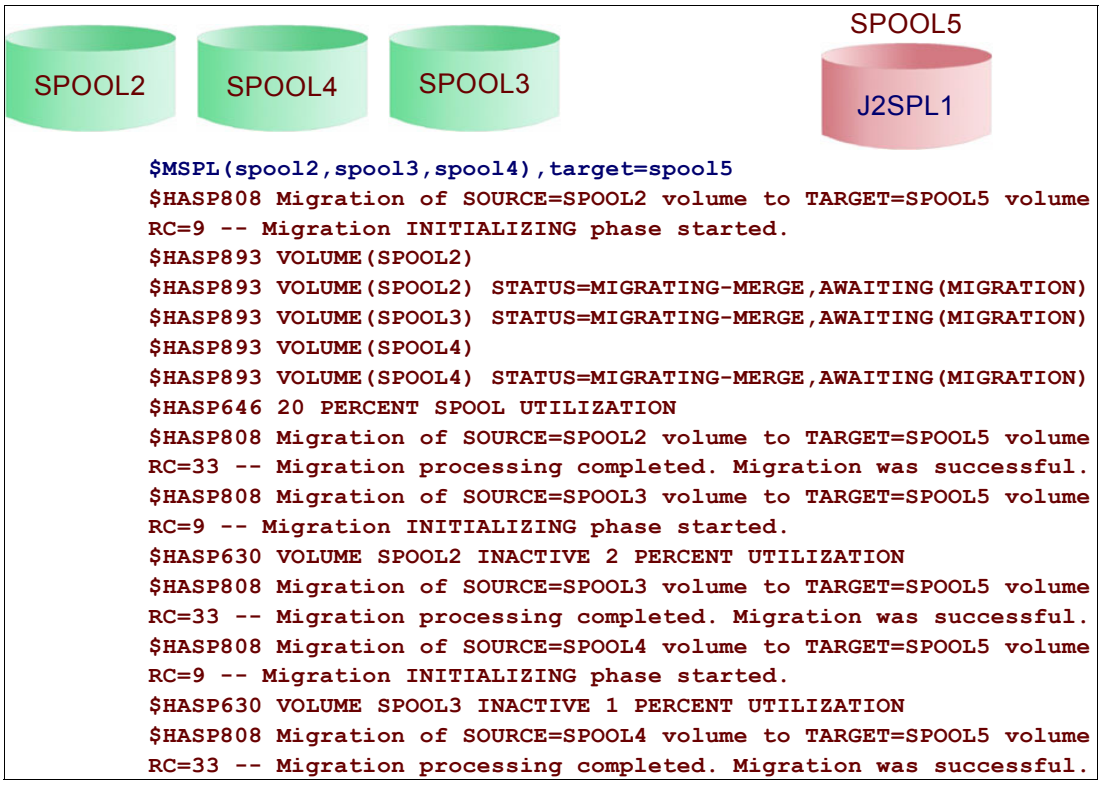


Figure 31-3 $MSPL command example

Use the **$DSPL** command to see the status of the target volume, and the source volumes if they are MAPPED, as shown in Figure 31-4.

```
$dspl
$HASP893 VOLUME(SPOOL1) STATUS=ACTIVE,PERCENT=1
$HASP893 VOLUME(SPOOL2) STATUS=MAPPED,PERCENT=2
$HASP893 VOLUME(SPOOL3) STATUS=MAPPED,PERCENT=1
$HASP893 VOLUME(SPOOL4) STATUS=MAPPED,AWAITING(JOBS),PERCENT=1
$HASP893 VOLUME(SPOOL5) STATUS=ACTIVE,PERCENT=1
$HASP646 0.5714 PERCENT SPOOL UTILIZATION
```

Figure 31-4 $dspl command

The $HASP893 messages, shown in Figure 31-5, display the status for each spool volume defined to JES2. The $HASP646 message then displays the total percent spool utilization for the complex of all active spool volumes.

```
$dspl,status,target,maptarget
$HASP893 VOLUME(SPOOL1) STATUS=ACTIVE,MAPTARGET=NO
$HASP893 VOLUME(SPOOL2) STATUS=MAPPED,TARGET=SPOOL5,MAPTARGET=NO
$HASP893 VOLUME(SPOOL3) STATUS=MAPPED,TARGET=SPOOL5,MAPTARGET=NO
$HASP893 VOLUME(SPOOL4) STATUS=MAPPED,AWAITING(JOBS),TARGET=SPOOL5,
MAPTARGET=NO
$HASP893 VOLUME(SPOOL5) STATUS=ACTIVE,MAPTARGET=YES
$HASP646 30 PERCENT SPOOL UTILIZATION
```

Figure 31-5 $dspl,status,target,maptarget command

The **$dspl** command new parameters are shown here in bold text.

- **MAPTARGET=Yes | No**

  This indicates if the volume is the target of a MAPPED spool volume.

  - **Yes** indicates the volume is the target of a MAPPED volume.
  - **No** indicates the volume is not the target of a MAPPED volume.

- **MIGDATA=([LARGEST_FREE | SPACE_USED])**

  On a per-spool volume basis, this displays the largest contiguous free space or the highest used location. The unit of measurement is tracks.

- **MIGRATOR**

  This displays the name of the JES2 MAS member that is performing the migration of a MIGRATING spool volume.

- **MPERCENT**

  This displays the percentage of the migration that has completed.

- STATUS=[ACTIVE | DRAINING | EXTENDING | HALTING | INACTIVE | **MAPPED** | **MIGRATING** | STARTING]

  - **MAPPED** - This indicates that the spool volume has been migrated and the corresponding data set is eligible for deletion.

    Note: The spool volume extent number will persist until all jobs and SYSOUT that have space on the volume have been purged.

  - **MIGRATING** - This indicates that the spool volume is a source of an active migration. The associated target volume is also displayed.

31.1.3 $DSPL command processing considerations

The **$DSPL** command displays active migrations and provides filtering capability.

Examples of the $dspl command with new parameters

The following commands illustrate the new parameters on the **$dspl** command.

```
$D SPOOL,MIGDATA
$HASP893 VOLUME (SPOL1)
$HASP893 VOLUME (SPOL1) MIGDATA=(SPACE_USED=40000,LARGEST_FREE=10000)
$HASP893 VOLUME (SPOL2)
$HASP893 VOLUME (SPOL2) MIGDATA=(SPACE_USED=20000,LARGEST_FREE=10000)
$HASP646 43.2679 PERCENT SPOOL UTILIZATION
The largest contiguous free space for each spool volume, with highest location
used in tracks, is displayed.

$D SPOOL,MIGDATA=LARGEST_FREE
$HASP893 VOLUME (SPOL1)
$HASP893 VOLUME (SPOL1) MIGDATA=(LARGEST_FREE=10000)
$HASP893 VOLUME (SPOL2)
$HASP893 VOLUME (SPOL2) MIGDATA=(LARGEST_FREE=20000)
$HASP646 43.2679 PERCENT SPOOL UTILIZATION
The largest contiguous free space for all spool volumes is displayed.

$D SPOOL(SPOL2),MIGDATA=LARGEST_FREE
$HASP893 VOLUME (SPOL2)
$HASP893 VOLUME (SPOL2) MIGDATA=(LARGEST_FREE=10000)
$HASP646 43.2679 PERCENT SPOOL UTILIZATION
The largest contiguous free space for spool SPOL2 is displayed.

$D SPOOL,MIGDATA=LARGEST_FREE>3000,MIGDATA
$HASP893 VOLUME (SPOL2)
$HASP893 VOLUME (SPOL2) MIGDATA=(LARGEST_FREE=4000)
$HASP893 VOLUME (SPOL5)
$HASP893 VOLUME (SPOL5) MIGDATA=(LARGEST_FREE=7000)
$HASP646 43.2679 PERCENT SPOOL UTILIZATION
All volumes having contiguous free space greater than 3000 tracks are
displayed.

$D SPOOL,TARGET=SPOL10,TARGET
$HASP893 VOLUME(SPOL1) TARGET=SPOL10
$HASP893 VOLUME(SPOL3) TARGET=SPOL10
$HASP646 80.0000 PERCENT SPOOL UTILIZATION
All spool volumes that have spool SPOL10 as a target are displayed. This
includes both migrating and mapped source spool volumes.

Note: The $HASP646 message displays the percentage of total spool space in
use within the MAS.
```

Figure 31-6 New parameter examples on the $dspl command ($d spool)

31.1.4 Extending spool data sets

With z/OS V1R13, this function is implemented to enhance JES2 to expand a single spool extent into free space contiguous to the extent. This JES2 support provides a way to exploit the increased z/OS addressable disk storage of dynamically resized volumes within the spool data set.

An operator command requests the extend through a command, and can ask for the maximum available size or a specific larger size. DFSMS services are used to extend the data set into adjacent free space on the spool volume. JES2 will then update the spool volume size information based on new extent size.

This extension can occur without impacting any running jobs. The extension of the spool data set is limited by the following conditions:

- The spool volume must be STATUS=ACTIVE with no commands or migration active or pending against it, and the extension must use relative addressing.
- A singe JES2 spool extent per volume restriction applies.
- The available free space contiguous to the JES2 spool extent is used.
- The JES2 data set maximum sizes is based on the current available support.
- All members of the MAS must be at JES2 z/OS V1R13 level.
- Downlevel members can later join the MAS and use the extended data set on the spool volume.

Note: The following requirements exist for this support:

- If large data set support is not active, LARGEDS=ALLOWED or ALWAYS must be specified on the SPOOLDEF statement, the data set extension is limited to a maximum of 65,535 tracks.
- If large data set support is active, the data set extension is limited to a maximum of 1,048,575 tracks.
- If the spool volume is an EAV volume, then the data set may extend into the EAS storage on the EAV volume if:
  - The data set is currently EAS-eligible (EATTR=OPT)
  - The correct DSCB format is specified.

If all of these conditions are true, the maximum space is determined by LARGEDS support. If any one condition is false, the maximum space is the smaller of 65,520 cylinders and the maximum is determined by LARGEDS support.

Operator commands

Note the following operator commands:

```
$D SPOOL,MIGDATA=LARGEST_FREE>3000,MIGDATA
$HASP893 VOLUME (SPOL2)
$HASP893 VOLUME (SPOL2) MIGDATA=(LARGEST_FREE=4000)
$HASP893 VOLUME (SPOL5)
$HASP893 VOLUME (SPOL5) MIGDATA=(LARGEST_FREE=7000)
$HASP646 43.2679 PERCENT SPOOL UTILIZATION
All volumes having contiguous free space greater than 3000 tracks are
displayed.
```

```
$D SPOOL,TARGET=SPOL10,TARGET
```

```
$HASP893 VOLUME(SPOL1) TARGET=SPOL10
$HASP893 VOLUME(SPOL3) TARGET=SPOL10
$HASP646 80.0000 PERCENT SPOOL UTILIZATION
All spool volumes that have spool SPOL10 as a target are displayed. This
includes both migrating and mapped source spool volumes.
```

The $HASP646 message displays the percentage of total spool space in use within the MAS.

31.1.5 Command support

The **$TSPL** command is used to extend the spool volume using the SPACE= parameter; see Figure 31-7.

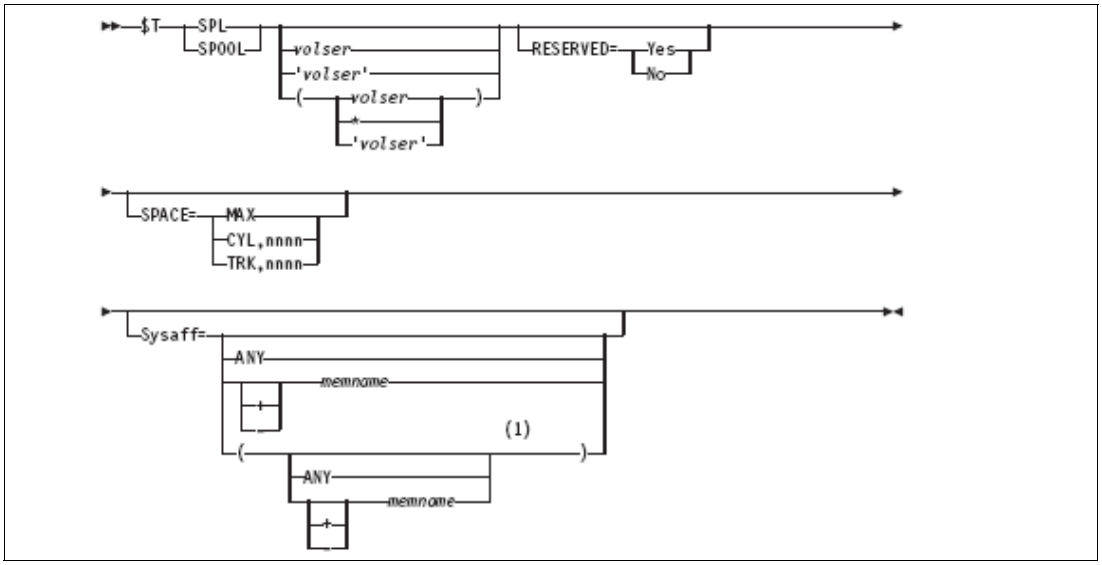


Figure 31-7 $TSPL command parameters

The maximum allowed is influenced by the largest contiguous free space next to the current JES2 spool extent, and the configuration values LARGEDS and CYL_MANAGED. You can specify SPACE=(CYL,xx) or SPACE=(TRK,xx) for a new larger size. SPACE=MAX is used to obtain the largest possible data set.

The following command extends the spool volume, volser SPOOL5, data set to the maximum size allowed.

```
$TSPL(SPOOL5),SPACE=MAX
```

If CYL_MANAGED=ALLOWED is specified on the SPOOLDEF statement for the volume, the extent can be extended into EAS storage on the EAV volume. Otherwise, it has to stay in track-managed storage.

You cannot extend a spool volume data set for a volume that is part of an active migration, an extend, or does not use relative addressing.

The following command extends the spool volume, volser SPOOL6, data set to 200 cylinders, if there is that much contiguous space available.

```
$TSPL(SPOOL6),SPACE=(CYL,200)
```

Note: Extending a spool volume's data set can cause the track group map to become full.

Commands to display results

A new message, $HASP740, indicates that the extend is successful. Use the $DSPL command to display the results of the extend, as explained here:

- Use **$DSPL,TGNUM** to display the number of track groups in the data set after the extend.
- Use **$DSPL,UNITDATA** to display the track range (TRKRANGE) of the data set after the extend.

31.2 Naming spool data sets

An existing problem with JES2 is that it uses the same spool data set name for all spool volumes. With z/OS V1R13, JES2 is enhanced to allow specification of a spool data set name to be used for a volume. This allows for unique naming of spool data sets.

The spool data set name can be specified as shown here:

- A **$SSPL** command for a new volume
  - **$SSPL(x),DSName=**
- New spool initialization statement
  - SPOOL(x) DSName=
- SPOOLDEF statement
  - DSNAME= is the default name used
- SPOOLDEF
  - DSNMASK= limits possible data set names and can contain generics. Supplied spool data set names must match the mask. If not specified, DSNAME= must match SPOOLDEF DSNAME=.

Note: The use of a nonstandard DSNAME= requires a z11 checkpoint mode. You cannot activate this function in a z2 mode if using nonstandard DSNAMEs.

Bind spool volumes

To bind spool volumes, the SPOOL initialization statement binds the specified spool volumes for cold start scan processing. Binding can enhance performance if the spool prefix defined by the **$T SPOOLDEF** command includes generic characters. Multiple spool initialization statements can be specified; see Figure 31-8.

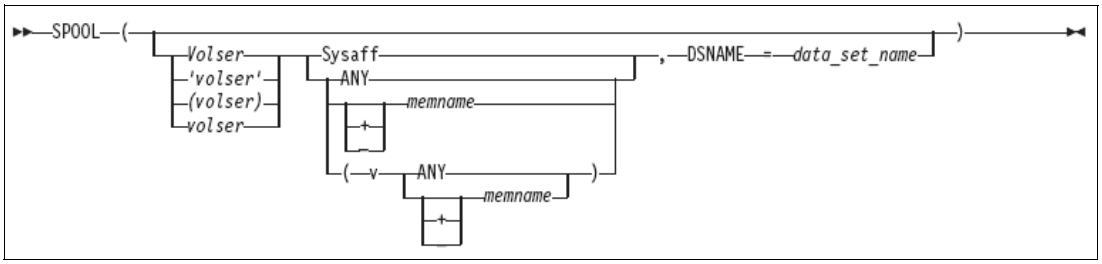


Figure 31-8 Format description for SPOOL statement

Following are the parameter descriptions for SPOOL bind:

volser This specifies a list of spool volumes, by serial number (volser), to be scanned at cold start processing.

DSNAME= data_set_name - This specifies the 1-44 character name of a JES2 spool volume data set. If not specified, the default data set name that is set using the **$T SPOOLDEF** command is used.

Sysaff= This specific JES2 members to associate the specified spool volume with. The member list can include any combination of the following keywords, separated by commas:

ANY - This associates the spool volume with all members in the JES2 multi-access spool configuration.

[+ | –] - memname * specifies the JES2 member names (memname of 1-4 characters) to be added to (+) or deleted from (–) the list of members that the spool volume is associated with. If the add nor the delete character is specified, the system affinities of the spool volume is explicitly set to the specified member names. The asterisk character (*) specifies the JES2 member that the command was issued on.

( ) - If system affinity is not a criterion for job selection, you can specify a null parameter of (Sysaff= ()). You must specify the empty parentheses.

Note: Spool volumes started on a z/OS V1R13 member will be usable by downlevel members. Downlevel members then must have compatibility APAR OA31806 applied.

Spool volume prefix

The current JES2 spool volume prefix is too restrictive. With z/OS V1R13, an enhancement is made for the JES2 spool prefix to support generics. This provides more flexibility in identifying spool volumes that can be used in the MAS.

SPOOLDEF VOLUME= allows generics to be used. However, you can still provide a specific four- or five-character prefix. An asterisk (*) allows any volume to be used for spool.

A spool prefix using generics skips scanning UCBs at cold start time. It is possible to start without spool volumes by not specifying a SPOOL initialization statement. If a SPOOL prefix contains generics, downlevel members cannot start. Thus, you can set a spool volume prefix to a non-generic value to allow downlevel members to start.

31.3 SPIN for any data set

The existing JESLOG spin only deals with job logs and system message data sets. In addition, other spin data sets can exist that cannot be spun off.

With JES2 V1R13, JES2 provides the ability to “spin” any spin spool data sets. A benefit of this implementation is that it allows the system to free spool space associated with log data sets created by long-running jobs. Because the JES2 JESLOG function supports spinning a job’s JESMSGLG and JESYSMSG which is based on command, size, or time (interval or absolute time), similar spin processing for any spin data set can now be done.

Note: This function is also supported in dynamic allocation and TSO ALLOC. The **$TJnnn,SPIN** command can optionally have an operand of DDNAME=xxxx.

If DDNAME=xxxx is supplied, then the file whose DDNAME matches xxxx will be spun and that file.

If DDNAME= is not specified, then all files that are spinnable will be spun, along with JESMSGLG and JESYSMSG, if spinnable.

31.3.1 SPIN= JCL parameter operator command

The SPIN= DD JCL is enhanced to support operands similar to JESLOG=.

The $T JOB command is enhanced with a new DDNAME= operand when specified with the SPIN operand.

- **$T JQ(xxx),SPIN,DDNAME=ddname** command

SPIN parameter

Use the SPIN parameter to specify that the output for the SYSOUT data set is to be made available for processing:

- Immediately upon unallocation; or
- At the end of the job.

DDname parameter

This parameter requests SPIN processing for the specified job. If DDname= is not specified, all eligible data sets (including JESMSGLG and JESYSMSG) are immediately spun off for the job. If DDname= is specified, the specified SYSOUT DD is immediately spun off for the job.

The DDname= value specified must represent a SYSOUT data set that was allocated with a SPIN= attribute that allows it to be spun.

```
SPIN[,DDname=jxxxxxxx]
```

Note: When the JESLOG data sets are spun, the message `$HASP138 JESLOG SPIN REQUESTED BY OPERATOR` is placed in the JESMSGLG and JESYSMSG data sets. $HASP138 is not placed in the console or SYSLOG.

31.3.2 JCL SYSOUT DD statement

If you specify the output to be immediately available upon unallocation, you can also specify for the data set to be capable of being spun by operator command, when the data set reaches a certain size, or when the data set has been active for a specified time period. Figure 31-9 shows the SPIN options.

```
SPIN= {NO } - old operand
      {UNALLOC } - old operand
---------- New operands with V1R13 ------------------------
      {(UNALLOC,'hh:mm') }
      {(UNALLOC,'+hh:mm') }
      {(UNALLOC,nnn [K|M])}
      {(UNALLOC,NOCMND) }
      {(UNALLOC,CMNDONLY) }
```

Figure 31-9 SPIN= options

Where:

NO This indicates that the system makes the sysout data set available for processing as a part of the output at the end of the job, regardless of when the data set is unallocated.

UNALLOC This indicates that the system makes the data set available for processing immediately when the data set is unallocated. If you dynamically unallocate the sysout data set, either explicitly or by specifying FREE=CLOSE, the system makes the data set available for processing immediately.

If you do not dynamically unallocate it, the sysout data set is unallocated at the end of the step, and the system will make it available for processing then.

(UNALLOC,'hh:mm') This indicates that the data set is to be spun at time 'hh:mm' each 24-hour period. hh is hours and has a range of 00 through 23. mm is minutes and has a range of 00 through 59. Note that the time must be specified within apostrophes.

(UNALLOC,'+hh:mm') This indicates that the data set is to be spun every hh:mm' time interval, where hh is hours and has a range of 00-23 and mm is minutes and has a range of 00-59. The minimum interval that can be specified is 10 minutes (mm). Hours hh must be specified even if zero. For example, SPIN=(UNALLOC,'+00:20') specifies that the data set be spun at 20 minute intervals. Note that the time interval must be specified within apostrophe characters.

(UNALLOC,nnn[K|M]) This indicates that the data set is to be spun when it has the specified number of lines, where nnn is lines. A minimum of 500 lines must be specified. Specify the optional characters K for thousands of lines and M for millions of lines.

(UNALLOC,NOCMND) This indicates that the data set cannot be spun before it is unallocated.

(UNALLOC,CMNDONLY) This indicates data set is only to be spun when an operator issues a command to spin the data set.

Note: If you specify SPIN=UNALLOC, the following defaults apply:

- A data set that is closed by the application program is available for processing immediately.
- A data set that is closed as part of the end-of-step cleanup, such as for a program abend, is available for processing at the end of the job.
- A data set can be spun as the result of an operator command. This is the same processing as SPIN=(UNALLOC,CMNDONLY).

SPIN example

Specify a DD statement in the JCL as follows:

```
//DD5 DD SYSOUT=A,SPIN=(UNALLOC,5K)
```

In this example, the system splits the data set into 5000 record segments and makes the SYSOUT data set available for printing every 5000 records. Whatever remains in the data set at the end of the STEP is available for printing at the end of step.

31.4 JCL instream data sets in procedures

With JES2 V1R13, new support is added to allow instream data sets to be created when processing a DD DATA or DD * JCL statement within procedures or INCLUDEs. This support is added for both cataloged and instream procedures.

This support works the same like instream data in a normal JCL stream, but does not support generating //SYSIN DD *. This support can only be used if the job converts on a z/OS V1R13 system, but it can execute on a downlevel system.

The following data sets are supported and other considerations are of concern:

- New SYSIN data sets are included in extended status DSLIST function.
- New SYSIN data sets are *not* included in spool data set browse of JCLIN which are part of the original JCL submitted.
- New SYSIN are not transmitted to other nodes or offloaded which are not part of the original JCL submitted.
- This support also works for batch jobs and started tasks.

Figure 31-10 is an example that shows a JES2 instream data set within an INCLUDE statement.

```
//ABC     JOB
//INCLUDE MEMBER=HELLO
//STEPA EXEC PGM=IEBGENER
//SYSIN DD DUMMY
//SYSPRINT DD SYSOUT=A
//SYSUT2 DD SYSOUT=A
//SYSUT1 DD DATA
HELLO WORLD
/*
```

Figure 31-10 Sample job stream with INCLUDE statement

31.5 Job completion code support

In various cases, the highest completion code of a job might not be the most meaningful. With both JES2 and JES3, a new JCL keyword is added to control the reported completion code. With this implementation, you can select the last step, a specific step, or the highest step completion code. This allows the user to better determine the success of a job without having to look at the job output.

31.5.1 JOBRC parameter

Use the JOBRC parameter to control how the job completion code (presented by JES2 or JES3) is set. By default (when JOBRC is not specified), the job completion code is set to the highest return code of any step, or if the job's execution fails because of an ABEND, the job completion code is set to the last ABEND code. However, this parameter can be used to request that the job completion code be set to the return code of the last executed step or a particular step that more accurately reflects the success or failure of the job.

The new JOBRC parameter can now be specified on a JOB card. The possible values for this keyword are as follows:

```
JOBRC= {MAXRC}| {LASTRC} | {(STEP,stepname[.procstepname]}
```

Where:

MAXRC This is the default. The job completion code is set to the highest return code of any step in the job, or if the completion of the job fails because of an ABEND, the job completion code is set to the last ABEND code.

LASTRC The job completion code is set to the return code or ABEND code of the last step that is executed in the job.

(STEP,stepname.procstepname) The job completion code is set to the return code or ABEND code of the step that is indicated by the stepname.[.procstepname] parameter. If this step does not exist, a JCL error is generated. If this step does not run, the processing is the same as though MAXRC is specified.

Note: The $T JOBCLASS(x) command is enhanced with a new JOBRC operand, as follows:

```
JOBRC=MAXRC|LASTRC
```

The JOBRC keyword on job card takes precedence.

Examples of the JOBRC parameter

In this example, the specification indicates to use the return code of the last executed step as the completion code for the job:

```
JOBRC=LASTRC
```

In this example, use the return code for the C step in the HLASM procstepname as the completion code for the job:

```
JOBRC=(STEP,C.HLASM)
```

31.6 Device and property SSI 82

Because JES2 SSIs do not report information associated with the spool migration changes. JES2 V1R13 is enhanced by using new JES2 JES Properties SSI 82 spool information support to provide information for migration planning and active migrations. JES2 SSI exploiters such as SDSF will be able to provide users with information pertinent to any spool migration use.

Figure 31-11 is an SDSF display of the new fields that are highlighted for the new spool migration changes, by using the SP command from the primary panel. The new fields are shown by scrolling to the right on this panel.

```
SDSF SPOOL DISPLAY SC74      53% ACT   9975 FRE   4640 LINE 1-1 (1)
COMMAND INPUT ===>                                    SCROLL ===> HALF
PREFIX=ROGERS*  DEST=(ALL)  OWNER=*  SYSNAME=*
ACTION=//-Block,=-Repeat,+-Extend,D-Display,DL-DisplayLong,J-Jobqueue,P-Purge,
ACTION=PC-PurgeCancel,S-Start,Z-Halt
NP   NAME   e HiUsed Comp% Phase        MigSys Target   MigVol MigDSName
     BH5SP1 5  40430
```

Figure 31-11 SDSF display of the new fields for spool migration

The highlighted fields shown in this figure are explained here:

Phase Migration phase

MigSys JES2 member performing the spool migration

Target Volume name in JES2 where this extent is migrating to or has migrated to

MigVol Volume to which this extent is migrating

MigDSName 44 data set name to which this extent is migrating

Node information SSI 82

This sub-function of the JES properties SSI (SSI 82) is enhanced to provide information about NJE nodes from all active members of JES2 MAS. The node information is available from JES2 MAS members starting from z/OSV1R11 and requires coexistence APAR on z/OS V1R11 and z/OS V1R12. This new function is also exploited by SDSF.

For more information, see *z/OS MVS Using the Subsystem Interface*, SA22-7642.

Device information SSI 83

This enhancement supports all types of devices managed by JES2 (readers, punches, transmitters, receivers, lines, and offloads). It provides extensive filtering capabilities, by device state, device name, and a variety of device attributes. The information is provided about all devices managed by all active members of JES2 MAS.

The device information is available from JES2 MAS members starting from z/OS V1R11 and requires a coexistence APAR on z/OS V1R11 and V1R12. This new function is exploited by SDSF.

For more information, see *z/OS MVS Using the Subsystem Interface*, SA22-7642.

31.7 Using the network resource monitor

The JES2 network resource monitor allows you to automatically start and restart JES2 networking devices (NJE and RJE) and NJE connections, eliminating any requirement to customize these networking tasks. The network resource monitor can also run JES2 commands to control the entire JES2 network, which can facilitate a transition to TCP/IP usage.

31.8 Evict job on step boundary

JES2 V1R13 provides an orderly way to remove jobs out of execution by providing a new operand on the **$E Jxxxx** command that forces a job out of execution when the current step ends. The job resumes execution from the next step.

The new JES2 command option is $EJxxxx,STEP[,HOLD], which can also deal with cross-member requests to force a job out of execution. The job is requeued for execution or is held if requested and then can utilize existing restart logic. This function requires the JES journal to be active which is specified on the JOBCLASS(x) JOURNAL=YES initialization statement.

Note: You can use the new step end SSI to communicate with the initiator to requeue the job. There is also a new JES2 exit 58 called in the step end SSI that can be used to inhibit or trigger this function.

$EJxxxx command

The following command $EJxxxx has a new operand:

```
Step [,Hold]
```

Using this operand starts the job at the next step after the current step completes. By default, the job is placed back in the execution queue in the ready state, and will reexecute immediately. However, if the current step is the final step in the job, the Step parameter has no effect and the job will exit the execution phase.

The optional Hold parameter causes the job to be held and requeued for execution after the current step completes.

Command processing verifies that the requested jobs are in the execution phase, and marks the job for the appropriate restart processing. The job is not required to be executing on the member where the command was issued. If the job is not executing, the command has no effect.

Note: Be aware of the following points:

- The Step parameter requires that the job use journaling. If the job does not have a journal, the **$E** command will fail.
- The Step parameter is designed to move a job to another system for reexecution at a step boundary. To move a job, it must no longer be eligible to run on the original system, which can be achieved by altering the affinity of the job, or by draining the job class, service class or member that the job is currently executing on.

31.9 Migration and coexistence considerations

With a JES2 V1R13 system in the sysplex, a downlevel member will not be allowed to join the MAS. A HASP720 message will be issued in the following situations:

- A spool migration is active or pending.

  To request a spool migration, the MAS must be running in z11 checkpoint mode.

- An extend spool is active or pending.

  To request an extend spool data set or spool migration, all members of the MAS must be at JES2 V1R13.
- A spool volume has STATUS=MAPPED.
- A spool prefix has been defined using generics.

HASP720 message

The first $HASP720 message condition encountered will cause the message to display and the WARM start to fail. The other $HASP720 conditions might also exist. The $HASP720 message might be preceded by a `$HASP896 DAS VERIFICATION AND REBUILD HAS COMPLETED` message, which is typical and does not indicate an error in the DAS.

System action: The WARM start is terminated.

```
HASP720 WARM START DENIED - reason
```

The warm start was denied, where reason is one of the following:

- EAV EXTENDED ADDRESSING SUPPORT HAS BEEN ACTIVATED IN THE MAS. THIS RELEASE DOES NOT SUPPORT EAV EXTENDED ADDRESSING.
  - JES2 members prior to z/OS V1R12 are not allowed to join the MAS after EAV Extended Addressing Support has been enabled with SPOOLDEF,CYL_MANAGED=ALLOWED.

    Even if CYL_MANAGED=FAIL is later set, members prior to JES2 z/OS V1R12 will not be allowed to join the MAS.
- SPOOL VOLUME VOLSER IS IN A MIGRATING STATE. THIS RELEASE DOES NOT SUPPORT MIGRATING SPOOL VOLUMES.
  - JES2 members prior to JES2 z/OS V1R13 are not allowed to join the MAS if a **$MSPL** migrate spool command is active or pending, or if the spool volume is MAPPED. A z/OS V1R13 member existing in a mixed MAS will reject a migrate SPOOL command.
- SPOOLDEF VOLUME= PREFIX CONTAINS GENERIC CHARACTERS. THIS RELEASE DOES NOT SUPPORT GENERICS IN THE SPOOL VOLUME PREFIX.
  - JES2 members prior to JES2 z/OS V1R13 are not allowed to join the MAS because they do not support generic characters in the spool volume prefix. A z/OS V1R13 member existing in a mixed MAS will reject a **$T SPOOLDEF** command that tries to utilize a generic SPOOL volume prefix.
- SPOOL VOLUME VOLSER DATA SET IS BEING EXTENDED. THIS RELEASE DOES NOT SUPPORT EXTENDING SPOOL VOLUME DATA SETS.
  - JES2 members prior to JES2 z/OS V1R13 are not allowed to join the MAS if a **$TSPL,SPACE=** extend SPOOL command is active or pending. A z/OS V1R13 member existing in a mixed MAS will reject an extend spool command.

Migration from JES2 V1R9 or V1R10

From these earlier releases, an all-member warm is needed to go to z/OS V1R13 because there is no coexistence support. For a possibility of a fallback to a previous release after V1R13 is installed, note the following implications:

- Various new data structures created by z/OSV1R13 JES2 can result in problems in z/OS V1R10 and earlier systems.
- Systems prior to z/OS V1R10 might not be able to use spool volumes with non-standard data set names.

- From JES2 z/OSV1R11 or z/OS V1R12 systems, a compatibility APAR OA31806 is needed on a z/OS V1R11 or z/OS V1R12 member to coexist in a MAS with z/OS V1R13. This APAR is also highly recommended for fallback.
- Various new data structures created by z/OS V1R13 JES2 can result in problems if APAR OA31806 is not installed.

32

JES3 enhancements

This chapter describes the enhancements to JES3 in z/OS V1R13.

In this chapter, the following topics are discussed:

- Adding a new spool volume during a JES3 hot start with refresh
- Adding a new spool volume using the **`*MODIFY CONFIG`** operator command
- Performing automatic flush after a local main failure

32.1 Adding spool volumes without a warm start

With JES3, you can increase or decrease your installation's spool capacity without performing a cold start by adding or deleting a spool data set. To determine whether your installation's spool capacity is appropriate for your installation, you can monitor spool usage using the ***I,Q,S** operator command.

Prior to z/OS V1R13, the only way to add, delete, or replace a spool volume was during a JES3 warm start (W), warm start with replace (WR), warm start with analysis (WA) or a cold start (C). However, a cold start or any type of a warm start requires a reIPL of the global and all locals.

Starting with z/OS V1R13, a spool volume can also be added using either of the following:

- A hot start with refresh (HR)
- The ***MODIFY CONFIG** command

Deleting or replacing a spool volume still requires a warm start.

Coexistence with previous versions of JES3

To add a spool volume using a hot start with refresh or the ***MODIFY CONFIG** command, the JES3 global must be at the z/OS V1R13 level or higher. In addition, all the JES3 local members must also be at the z/OS V1R13 level or higher. The new spool volume is made available only after all currently connected local members reconnect to the global or are flushed from the complex.

If your complex has JES3 systems at the z/OS V1R12 or earlier level, JES3 issues message IAT3426 during a configuration change to indicate down-level systems were found and must be reset. The down-level systems have to be reIPLed before they can reconnect to the global and use the new spool volume. Message IAT2061 is issued to list the systems that prevent the configuration change from completing. JES3 also issues WTOR message IAT2064 which allows you to cancel the configuration change, in case you cannot reset the down-level systems at that time.

32.1.1 Adding a spool volume using a hot start with refresh

This section describes the procedure for adding a new spool data set during a JES3 hot start with refresh. In our environment, we have two spool data sets allocated, as shown by the inquiry command on Figure 32-1.

```
*I Q DD=ALL
IAT8513 SPOOL1   JES3PART  15,000 GRPS,  7,770 LEFT ( 52%), STT
IAT8513 SPOOL2   JES3PART  1,800K GRPS, 1,793K LEFT (100%), STT
IAT8611 INQUIRY ON SPOOL DATA SET STATUS COMPLETE
```

*Figure 32-1 Output of *I Q DD=ALL before adding a third spool volume*

To allocate a new spool data set, we use the JCL shown in Figure 32-2. This job is expected to complete with a return code of zero. We specify the DSNTYPE=LARGE parameter to allocate the data set on a 3390-9 volume in a single contiguous extent of 10,016 cylinders. The JES3 support for large spool data sets was added in z/OS V1R12.

```
//ALLOCSP3 JOB ACCNT#,MSGLEVEL=(1,1),NOTIFY=&SYSUID
//ALLOCATE EXEC PGM=IEFBR14
//SPOOL3   DD DISP=(NEW,CATLG),DSN=SYS1.JES3SPL3,
//  DCB=(RECFM=U,BLKSIZE=4084),SPACE=(CYL,10016,,CONTIG),
//  DSNTYPE=LARGE,EATTR=OPT,UNIT=3390,VOL=SER=BH5J30
//
```

Figure 32-2 JCL to allocate a JES3 spool data set

We format the new spool data set using the IEBDG utility before starting JES3. If you do not format the spool data set before starting JES3, the startup time can be elongated, especially if you are using a large spool data set on an EAV volume. The JCL in Figure 32-3 shows how to use the IEBDG utility to format a spool data set. The job is expected to complete with a SD37 ABEND. This is normal for IEBDG.

```
//FORMATJ3 JOB ACCNT#,MSGLEVEL=(1,1),NOTIFY=&SYSUID
//FORMAT   EXEC PGM=IEBDG
//SPXTNT   DD DISP=OLD,DSN=SYS1.JES3SPL3
//SYSPRINT DD SYSOUT=X
//SYSIN    DD *
  DSD    OUTPUT=(SPXTNT)
  FD     NAME=SPOOL,FILL=X'FF',LENGTH=4084
  CREATE NAME=(SPOOL),QUANTITY=2000000000
  END
//
```

Figure 32-3 JCL to format a JES3 spool data set

Now that the spool data set is allocated and formatted, we update the JES3 initialization deck to include it. Figure 32-4 is not a complete initialization deck. It only shows the initialization statements relevant to the JES3 spool configuration. DDs SPOOL1 and SPOOL2 are statically allocated in our JES3 startup procedure. The new spool data set is dynamically allocated using the DYNALLOC statement. Because the new spool data set is already formatted, we use the TRACK statement to define it to JES3.

```
DYNALLOC,DDN=SPOOL3,DSN=SYS1.JES3SPL3
TRACK,DDNAME=SPOOL1
TRACK,DDNAME=SPOOL2
TRACK,DDNAME=SPOOL3
BUFFER,BUFSIZE=4084,PAGES=(1000,16,500),SPLIM=(5,5),GRPSZ=10
OPTIONS,DUMP=PRDMP,WANTDUMP=YES,JOBNO=(1,32767,32767),
XCFGRPNM=WTSCPLX4
ENDJSAM
```

Figure 32-4 JES3 inish deck statements to add a third spool volume

After the initialization deck is updated, we restart JES3. Figure 32-5 shows the JES3 initialization sequence. Notice that we reply to WTOR number 193 (message IAT3011) with HR, specifying the hot start with refresh startup type. The suffix of our initialization deck member is GL, which contains the statements shown in Figure 32-4. Message IAT4030 is issued to indicate that three spool data sets are now available to JES3.

```
 S JES3
...
IAT3040 STATUS OF JES3 PROCESSORS IN JESXCF GROUP WTSCPLX4
 IAT3040 SC74    (  ), SC70    (  ), SC75    +  +
 IAT3011 SPECIFY JES3 START TYPE
*193 IAT3011 (C, L, H, HA, HR, HAR, W, WA, WR, WAR, OR CANCEL)

 R 193,HR
 IEE600I REPLY TO 193 IS;HR
*194 IAT3012 SELECT JES3 INISH ORIGIN (N OR M=), AND OPTIONAL EXIT
 PARM (,P=) OR CANCEL
 R 194,M=GL
 IEE600I REPLY TO 194 IS;M=GL
 IEF196I IEF237I DB4F ALLOCATED TO SPOOL3
 IAT4030 0003 SPOOL DATA SETS IN USE
 IAT4075 MAXIMUM NUMBER OF JOBS IS LIMITED TO 0021461 BY JCT DATA SET
...
*IAT3100 JES3 z 1.13.0  SYSTEM  HOTSTART ON 2011.108 AS SC75
...
 IAT2645 ****** SC75     CONNECT COMPLETE ******
```

Figure 32-5 JES3 startup messages after hot start with refresh to add a spool volume

Finally, we display our spool configuration again. As shown in Figure 32-6, we now have three spool volumes in use.

```
*I Q DD=ALL
IAT8513 SPOOL1   JES3PART  15,000 GRPS,  7,690 LEFT ( 51%), STT
IAT8513 SPOOL2   JES3PART  1,800K GRPS, 1,793K LEFT (100%), STT
IAT8513 SPOOL3   JES3PART 150,240 GRPS,150,233 LEFT (100%)
IAT8611 INQUIRY ON SPOOL DATA SET STATUS COMPLETE
```

*Figure 32-6 Output of *I Q DD=ALL after adding a third spool volume*

32.1.2 Adding a spool volume using the *MODIFY CONFIG command

JES3 performs the following actions when it processes the ***MODIFY CONFIG** command:

- It parses the statements and compares them to the current configuration.
- It adds new spool extents, if requested.
- It add new spool partitions, if requested.
- It rejects requests to delete spool volumes.
- It changes other parameters, if possible. Currently only the parameters on the OPTIONS statement are supported.
- It commits the changes if no errors are found.

In this section, we describe a procedure for dynamically adding a spool volume to JES3 using the ***MODIFY CONFIG** command. We allocate the spool data set using a JCL similar to the one in Figure 32-2. We do not format the spool data set before adding it to JES3.

Figure 32-7 shows the configuration statements used to add a fourth spool data set to JES3. Notice that the fourth spool data is defined using the FORMAT statement, which tells JES3 to

format the spool data set before adding it. We save the statements in member JES3INGS. This member name will be used in a ***MODIFY CONFIG ADD=*** command. The member must contain all JES3 initialization deck statement relevant to the spool configuration, starting with DYNALLOC statements and ending with the ENDJSAM statement.

```
DYNALLOC,DDN=SPOOL3,DSN=SYS1.JES3SPL3
DYNALLOC,DDN=SPOOL4,DSN=SYS1.JES3SPL4
TRACK,DDNAME=SPOOL1
TRACK,DDNAME=SPOOL2
TRACK,DDNAME=SPOOL3
FORMAT,DDNAME=SPOOL4
BUFFER,BUFSIZE=4084,PAGES=(1000,16,500),SPLIM=(5,5),GRPSZ=10
OPTIONS,DUMP=PRDMP,WANTDUMP=YES,JOBNO=(1,32767,32767),
XCFGRPNM=WTSCPLX4
ENDJSAM
```

Figure 32-7 JES3 inish deck statements to add a fourth spool volume

After the member is ready, we use the ***MODIFY CONFIG,ADD=*** command to add the new spool volume to JES3. Figure 32-8 shows the log messages issued by JES3.

```
 *F CONFIG,ADD=JES3INGS
 IEF196I IEF237I D85B ALLOCATED TO JES3IN
 IAT6369  JES3 WAITING FOR CHECKPOINT DATA SET RESERVE -  CHKPNT ,
 BH5JS3,872E.
 IAT6369  JES3 WAITING FOR CHECKPOINT DATA SET RESERVE -  CHKPNT2,
 BH5JS4,DD65.
 IEF196I IEF237I DC0F ALLOCATED TO SPOOL4
 IAT8348 WARNING LEVEL MESSAGE(S) ISSUED DURING INITIALIZATION
 STATEMENT PROCESSING
*195 IAT8337 CONFIRM "*F CONFIG" COMMAND (CONTINUE(U), CANCEL, OR LOG)
 R 195,U
 IEE600I REPLY TO 195 IS;U
*IAT4031 FORMATTING OF SPOOL DATA SET "SPOOL4  " IN PROGRESS
 IAT4032 FORMATTING OF SPOOL DATA SET "SPOOL4  " COMPLETE,   NO ERRORS
 IAT4030 0004 SPOOL DATA SETS IN USE
 IAT8069 MESSAGES WERE GENERATED - SEE LOG FOR DETAILS
 IAT8090 SPOOL DATA SET SPOOL1   MOVED TO PARTITION JES3PART
 IAT8090 SPOOL DATA SET SPOOL2   MOVED TO PARTITION JES3PART
 IAT8090 SPOOL DATA SET SPOOL3   MOVED TO PARTITION JES3PART
 IAT8090 SPOOL DATA SET SPOOL4   MOVED TO PARTITION JES3PART
 IEF196I IEF285I   SYS1.PARMLIB                                KEPT
 IEF196I IEF285I   VOL SER NOS= BH5CAT.
 IAT8350 CONFIGURATION MODIFICATION IS COMPLETE - WARNING MESSAGES
```

*Figure 32-8 JES3 messages after a *MODIFY CONFIG to add a spool volume*

Important: After using the ***MODIFY CONFIG*** command, remember to update the regular JES3 inish deck members to reflect the additional spool data set. Otherwise, the definitions will be lost in case of a future cold start, warm start, or hot start with refresh.

Finally, we display our spool configuration again. As shown in Figure 32-9, we now have four spool volumes in use.

```
*I Q DD=ALL
IAT8513 SPOOL1   JES3PART  15,000 GRPS,  7,690 LEFT ( 51%), STT
IAT8513 SPOOL2   JES3PART  1,800K GRPS, 1,793K LEFT (100%), STT
IAT8513 SPOOL3   JES3PART 150,240 GRPS,150,223 LEFT (100%)
IAT8513 SPOOL4   JES3PART 150,240 GRPS,150,224 LEFT (100%)
IAT8611 INQUIRY ON SPOOL DATA SET STATUS COMPLETE
```

*Figure 32-9 Output of *I Q DD=ALL after adding a fourth spool data set*

32.2 Performing automatic flush after a local main failure

With z/OS V1R13, JES3 issues an automatic flush to flush active jobs on a JES3 local main processor that has failed or been stopped. If jobs remain active on a main processor that has been flushed, errors on the JES3 spool can occur and a cold start is required. For this reason, you cannot flush the global processor.

JES3 uses XCF services to determine whether a local main processor is no longer responding. After it is determined that the member is no longer in the sysplex, JESXCF informs JES3 and the flush is automatically performed to flush all active jobs on the failed main. Note that after an automatic flush, the main remains in the online state. This is different from the regular behavior after a flush command is issued by an operator, where the local main is also varied offline.

Figure 32-10 shows the log messages after system SY2 has failed and is removed from the sysplex by the operator. XCF begins cleanup actions after the operator replies to WTOR message IXC102A. During that time, JES3 automatically flushes all active jobs on member SY2. Message IAT2006 provides an indication that JES3 flushed active jobs on SY2.

```
SY1          *36 IXC409D SIGNAL PATHS BETWEEN SY2 AND SY1 ARE LOST. REPLY RETRY
OR SYSNAME=SYSNAME OF THE SYSTEM TO BE REMOVED.
36,sysname=sy2
SY1      IEE600I REPLY TO 36 IS;SYSNAME=SY2
SY1          *43 IXC417D CONFIRM REQUEST TO REMOVE SY2 FROM THE SYSPLEX. REPLY
SYSNAME=SY2 TO REMOVE SY2 OR C TO CANCEL.
43,sysname=sy2
SY1      IEE600I REPLY TO 43 IS;SYSNAME=SY2
SY1    IXC101I SYSPLEX PARTITIONING IN PROGRESS FOR SY2 REQUESTED BY
XCFAS. REASON: LOSS OF CONNECTIVITY
SY1          *44 IXC102A XCF IS WAITING FOR SYSTEM SY2 DEACTIVATION. REPLY DOWN
WHEN MVS ON SY2 HAS BEEN SYSTEM RESET
44,down
SY1       IEE600I REPLY TO 44 IS;DOWN
SY1    IXC808I ELEMENTS FROM TERMINATED SYSTEM SY2 WERE NOT PROCESSED BY
THIS SYSTEM. ARM COUPLE DATA SET IS NOT AVAILABLE TO THIS SYSTEM.
SY1     IXC105I SYSPLEX PARTITIONING HAS COMPLETED FOR SY2
- PRIMARY REASON: LOSS OF CONNECTIVITY
- REASON FLAGS: 000002
IAT2006 PREMATURE JOB TERM - JOB SYSLOG  (JOB00039) - CANCELED  - SY2
IAT2006 PREMATURE JOB TERM - JOB SDSF   (JOB00040) - CANCELED  - SY2
IAT2006 PREMATURE JOB TERM - JOB RACF   (JOB00042) - CANCELED  - SY2
IAT2006 PREMATURE JOB TERM - JOB MOUNT  (JOB00038) - CANCELED  - SY2
SY1      ISG011I SYSTEM SY2 - BEING PURGED FROM GRS COMPLEX
SY1      ISG013I SYSTEM SY2 - PURGED FROM GRS COMPLEX
SY1     CNZ4200I CONSOLE C3E0SY2 HAS FAILED. REASON=SYSFAIL
SY1     CNZ4200I CONSOLE *AXR01Y2 HAS FAILED. REASON=SYSFAIL
SY1     CNZ4200I CONSOLE *AXR04Y2 HAS FAILED. REASON=SYSFAIL
SY1     CNZ4200I CONSOLE *AXR02Y2 HAS FAILED. REASON=SYSFAIL
SY1     CNZ4200I CONSOLE *AXR03Y2 HAS FAILED. REASON=SYSFAIL
SY1            IEA258I CONSOLE PARTITION CLEANUP COMPLETE FOR SYSTEM SY2.
```

Figure 32-10 JES3 messages after a local main fails and leaves the sysplex

33

Resource Management Facility enhancements

Many activities are required to keep your system running smoothly and provide the best service on the basis of available resources and workload requirements. These activities are variously performed by operators, administrators, systems programmers, or performance analysts. Resource Management Facility (RMF) is the tool that helps these specialists perform their activities effectively. RMF consists of several components:

- Monitor I, Monitor II, Monitor III
- Postprocessor
- RMF performance monitoring
- Client/Server enabling
- Spreadsheet Reporter
- Sysplex data server
- Distributed Data Server

These components work together in providing the capabilities needed for performance management:

- Gathering data
- Reporting data
- Accessing data across the sysplex

This chapter describes the changes to RMF in z/OS V1R13:

- Workload Activity reporting enhancements
- GRS and supervisor delay monitoring enhancements
- Redesign of the postprocessor Paging Activity report
- Enhanced RMF Monitor II OPT Settings report
- Integrated ensemble performance monitoring on systems running Linux or AIX
- z/VM guest support

33.1 WLM reporting enhancements

In z/OS V1R13, the Postprocessor Workload Activity report is extended by the following features:

- Supervisor promotion

  The report is enhanced to show the CPU time consumed for work units while they have been promoted by the supervisor to a higher dispatching priority than assigned by Workload Manager (WLM). In addition, RMF provides new overview conditions for the Postprocessor based on SMF record 72-3.

- Response time distribution for velocity and discretionary goals

  The Postprocessor Workload Activity report is enhanced to display new tabular response time distributions for service and report class periods that have defined velocity or discretionary goals.

- Resource group capacity values

  Minimum and maximum values for CPU service units per second up to 99.999.999 (8-digit capacity values) are supported in the Service Policy Page of the Postprocessor Workload Activity report and in the Monitor III Resource Group Data report. This report also displays the actual value in 8 digits.

You can also display the minimum, maximum, and actual resource group capacity in 8-digit values in the Monitor III Sysplex Summary report using the Monitor III report format definition utility (RMF UTIL).

33.1.1 WLMGL postprocessing report

Postprocessor reports can be generated using JCL that you developed and submitted as a batch job, as described in *z/OS Resource Management Facility User's Guide*, SC33-7990. Alternatively, you can use the ISPF Postprocessor interface.

The ISPF Postprocessor interface consists of a series of panels that are presented in sequence. It allows you to define and save commonly used Postprocessor options.

Before you use the ISPF Postprocessor interface, you need to create a report profile data set that defines the following report options:

EXCEPT Exception condition definitions.

EXRPTS Interval reports to be written if an exception in an associated EXCEPT statement is met.

REPORTS Selected interval and duration reports for single systems.

SYSRPTS Selected interval and duration reports for a sysplex.

You can create the following types of Postprocessor reports based on data gathered as SMF records by Monitor I, Monitor II, and Monitor III:

Interval reports Show system activity during a specified interval using data from Monitor I, Monitor II, and Monitor III gatherers. Each report shows data for one data gatherer interval.

Duration reports Summarize a particular system activity over a specified reporting period.

Summary reports Present an overview of system activity over a specified reporting period using data from the Monitor I gatherer.

Exception reports Present a summary of the values that exceeded the installation-defined thresholds over a specified period of time using data from the Monitor I gatherer.

Overview reports Provide enhanced summary and exception reporting capabilities.

The ISPF Postprocessor can generate interval reports based on data gathered as SMF records by RMF (Monitor I, Monitor II, and Monitor III), by web servers, and by IBM Lotus® Domino® servers.

The Postprocessor can either get its input from data sets with SMF records from all systems in the sysplex, or it can access all current SMF records in the Sysplex automatically using the RMF Sysplex Data Server. For details about how to call the Postprocessor with various options and capabilities, see *z/OS Resource Management Facility User's Guide*, SC33-7990.

Because the majority of WLM reporting enhancements have been made to the WLMGL postprocessor report, we can use the ISPF Postprocessor interface to generate the report with the new data.

ISPF Postprocessor interface

The ISPF Postprocessor interface consists of a series of panels that are presented in sequence and allow you to set Postprocessor options in an easy way. For example, the panels help you to specify the input data source, namely the RMF SMF BUFFER, SMF data sets, or SMF log streams. The ISPF Postprocessor interface creates and submits a Postprocessor job based on the user input, as shown on the following steps:

1. To invoke the ISPF Postprocessor interface, enter `RMF` on ISPF Option 6. The ISPF RMF Performance Management menu is displayed, as shown in Figure 33-1.

```
 RMF - Performance Management                 z/OS V1R13 RMF
Selection ===>

Enter selection number or command on selection line.


  1 Postprocessor   Postprocessor reports for Monitor I, II, and III      (PP)
  2 Monitor II      Snapshot reporting with Monitor II                    (M2)
  3 Monitor III     Interactive performance analysis with Monitor III     (M3)

  U USER            User-written applications (add your own ...)          (US)

  R RMF SR          Performance analysis with the Spreadsheet Reporter
  P RMF PM          RMF PM Java Edition
  N News            What's new in z/OS V1R13 RMF

                             T TUTORIAL    X EXIT

  RMF Home Page:    http://www.ibm.com/systems/z/os/zos/features/rmf/

        5694-A01 Copyright IBM Corp. 1994, 2011. All Rights Reserved
                 Licensed Materials - Property of IBM
```

Figure 33-1 ISPF RMF Postprocessor Interface

2. Define the input data and point to the member that contains the reports options on the Report Profile field; see Figure 33-2.

```
 RMF - Postprocessor Setup
Command ===>

 Input Data         ===> SDS____   DATASET, SDS (Sysplex Data Server Buffers)
                                   or SMFLOG (SMF Log Streams)
 Output Data        ===> NO_       YES or NO (NO to route output to SYSOUT)

 Report Profile     ===> TEST1.JCL(RMFPOST)_____________________________

 Edit generated JCL ===> YES       YES or NO




 Job Statement Information:
  ===> //RMFPOST  JOB (999,POK),'RMF TEST',CLASS=A,MSGCLASS=T,
  ===> // NOTIFY=&SYSUID,TIME=1440,MSGLEVEL=(1,1),REGION=0M
  ===> //*
  ===> //*
```

Figure 33-2 Select the input data and the report profile member

3. On the panel shown in Figure 33-3, define the date on which the data will be used, and the interval that is relevant.

```
 Command ===>

  Reporting (DATE) Start ===> __ . ___        End ===> __ . ___        yy.ddd
               or Start ===> 04 / 21 / 2011 End ===> 04 / 21 / 2011 mm/dd/yyyy
  Duration  (DINTV)      ===> 00 : 30
  Exception (ETOD) Start ===> 00 : 00        End ===> 24 : 00  hh:mm
  Interval  (RTOD) Start ===> 00 : 00        End ===> 24 : 00  hh:mm
  Summary   (STOD) Start ===> 00 : 00        End ===> 24 : 00  hh:mm

  Summary   (SUMMARY)    ===> TOT,INT       NO, INT, TOT, or TOT,INT
  Overview  (OVERVIEW)   ===> REPORT_______ RECORD, REPORT (or both)

  DELTA     ===> NO_  YES or NO
  EXITS     ===> NO_  YES or NO      SESSION   ===> __   Session ID Monitor II
  SYSOUT    ===> T    Sysout Class   SYSID     ===> ____ System identifier

  To (edit and) submit Postprocessor job, press ENTER.
  To return to previous panel, press END.
  To return to the Postprocessor Setup Menu, enter CANCEL.
```

Figure 33-3 Define the data intervals

4. On Figure 33-3, you can see the final job generated by the panel with the field `SYSRPTS(WLMGL(SCPER(TSO1)))` that was in the member pointed to on the first panel.

```
//RMFPOST  JOB (999,POK),'RMF TEST',CLASS=A,MSGCLASS=T,
// NOTIFY=&SYSUID,TIME=1440,MSGLEVEL=(1,1),REGION=0M
//*****************************************************************
//RMFPP    EXEC PGM=ERBRMFPP,REGION=0M
//MFPMSGDS DD   SYSOUT=*
//*****************************************************************
//SYSIN    DD   *
SYSRPTS(WLMGL(SCPER(TSO1)))
  DATE(04212011,04212011)
  ETOD(0000,2400)
  RTOD(0000,2400)
  STOD(0000,2400)
  SUMMARY(TOT,INT)
  NODELTA
  NOEXITS
  SYSOUT(T)
  DINTV(0030)
  OVERVIEW(REPORT)
/*
```

Figure 33-4 Final job generated before execution

If needed, you can press the Help button on any field to obtain more information about it, or refer to *z/OS Resource Management Facility User's Guide*, SC33-7990.

33.1.2 Supervisor promotion

The report is enhanced to show the CPU time consumed for work units while they were promoted by the supervisor to a higher dispatching priority than that assigned by WLM. RMF also provides new overview conditions for the Postprocessor based on SMF record 72-3.

In Figure 33-5, you can see the new field `PROMOTED`. This field contains the CPU time in seconds that transactions in this group were running at a promoted dispatching priority, separated by the reason for the promotion, as explained here:

BLK CPU time in seconds consumed while the dispatching priority of work with low importance was temporarily raised to help blocked workloads.

ENQ CPU time in seconds consumed while the dispatching priority was temporarily raised by enqueue management because the work held a resource that other work needed.

CRM CPU time in seconds consumed while the dispatching priority was temporarily raised by chronic resource contention management because the work held a resource that other work needed.

LCK In HiperDispatch mode, the CPU time in seconds consumed while the dispatching priority was temporarily raised to shorten the lock hold time of a local suspend lock held by the work unit.

SUP CPU time in seconds consumed while the dispatching priority for a work unit was temporarily raised by the z/OS supervisor to a higher dispatching priority than that assigned by WLM.

```
1                                        W O R K L O A D   A C T I V I T Y
                                                                                                    PAGE   2
       z/OS V1R13              SYSPLEX PLEX75             START 04/25/2011-09.30.00 INTERVAL 000.30.00   MODE = GOAL
                               RPT VERSION V1R13 RMF      END   04/25/2011-10.00.00

                                          POLICY ACTIVATION DATE/TIME 04/20/2011 14.30.10

  REPORT BY: POLICY=TEST1      WORKLOAD=SYSTEM     SERVICE CLASS=SYSSTC     RESOURCE GROUP=*NONE
                                                   CRITICAL     =NONE
                                                   DESCRIPTION  =STARTED TASK DEFAULT

  -TRANSACTIONS-  TRANS-TIME HHH.MM.SS.TTT  --DASD I/O--  ---SERVICE---   SERVICE TIME  ---APPL %---  --PROMOTED--  ----STORAGE-
  AVG      61.09  ACTUAL                0  SSCHRT   9.2  IOC    108488   CPU   38.196  CP      2.30  BLK    0.000  AVG    15776
  MPL      61.09  EXECUTION             0  RESP     1.4  CPU     18772K  SRB    2.962  AAPCP   0.00  ENQ    0.000  TOTAL  96382
  ENDED        0  QUEUED                0  CONN     1.2  MSO         0   RCT    0.070  IIPCP   0.00  CRM    0.000  SHARED  2081
  END/S     0.00  R/S AFFIN             0  DISC     0.0  SRB      1362K  IIT    0.088                 LCK    0.122
  #SWAPS     759  INELIGIBLE            0  Q+PEND   0.1  TOT     20243K  HST    0.000  AAP     0.00  SUP    0.000  -PAGE-IN RAT
  EXCTD        0  CONVERSION            0  IOSQ     0.0  /SEC    11246   AAP    0.000  IIP     0.00                SINGLE
  AVG ENC   0.00  STD DEV               0                               IIP    0.000                              BLOCK
  REM ENC   0.00                                       ABSRPTN   184                                             SHARED
  MS ENC    0.00                                       TRX SERV  184                                             HSP
```

Figure 33-5 New PROMOTED field on WLMGL postprocessor report

33.1.3 Response time distribution for velocity and discretionary goals

For workloads with execution velocity goals, WLM reporting service IWMRCOLL does not provide a response time distribution (ended transactions). But it is desirable to show a response time distribution if this is a transactional workload. You can do this either directly with IWMRCOLL, or with the RMF Workload Activity report (WLMGL – Service Class Period Report).

To solve this, in z/OS V1R13, RMF Monitor I data gatherer collects new response time information in SMF 72 subtype record 3, as shown in Table 33-1. See *z/OS Resource Management Facility User's Guide*, SC33-7990, for more information.

Table 33-1 SMF 72 Subtype 3 record

SMF record type 72 subtype 3 (Workload Activity) - Service/Report Class Period Data section				
Offset	**Name**	**Length**	**Format**	**Description**
608 x260	R723RTDM	4	Binary	Midpoint of all response times that were collected in the response time distribution buckets in milliseconds. For response time goals, the midpoint is always the response time goal For execution velocity goals, the midpoint is the average of all response times that were collected in the response time distribution buckets.
612 x264	R723RTDC	4	Binary	Number of midpoint changes that occurred during interval. The field equals zero (0) for response time goals.
616 x268	R723RTDT	4	BInary	Time stamp in STCK format, showing the last point in time when a midpoint change occurred in R723RTDM.

The RMF Postprocessor Workload Activity report now also displays response time distribution for response time and execution velocity goals, one merged distribution for workloads with response time goals per sysplex, and one distribution for workloads with execution velocity goals per system in the sysplex.

This new information provides a better picture of workload behavior, helping you to detect problems before they occur. Also, clients deploying new software products are now able to obtain better analysis data and can migrate their goal definitions to a more meaningful response time goal.

Figure 33-6 shows a report with the new RESPONSE TIME DISTRIBUTIONS field for a service class TSO1 with a velocity goal. This report was generated with the report options ‘SYSRPTS(WLMGL(SCPER(TSO1))).

Important: To see the new field, you must use the SCPER or the RCPER sub-options within the SYSRPTS (WLMGL) for each of the workloads that you need the distribution. If the WLMGL is used alone, it will not show the new field.

```
1                                        W O R K L O A D   A C T I V I T Y
                                                                                                                PAGE   1
        z/OS V1R13              SYSPLEX PLEX75             START 04/25/2011-09.30.00 INTERVAL 000.30.00   MODE = GOAL
                                RPT VERSION V1R13 RMF      END   04/25/2011-10.00.00

                                          POLICY ACTIVATION DATE/TIME 04/20/2011 14.30.10

   ------------------------------------------------------------------------------------------------------------------- SERVICE CLASS PER

   REPORT BY: POLICY=TEST1      WORKLOAD=TEST1      SERVICE CLASS=TSO1       RESOURCE GROUP=*NONE      PERIOD=1 IMPORTANCE=2
                                                    CRITICAL     =NONE

   -TRANSACTIONS-  TRANS-TIME HHH.MM.SS.TTT  --DASD I/O--  ---SERVICE---   SERVICE TIME  ---APPL %---  --PROMOTED--  ----STORAGE-
   AVG       0.00  ACTUAL                83  SSCHRT   1.4  IOC     11580   CPU    0.213  CP      0.01  BLK    0.000  AVG     1772
   MPL       0.00  EXECUTION             71  RESP     0.5  CPU    124566   SRB    0.008  AAPCP   0.00  ENQ    0.000  TOTAL      5
   ENDED       79  QUEUED                12  CONN     0.3  MSO         0   RCT    0.002  IIPCP   0.00  CRM    0.000  SHARED     1
   END/S     0.04  R/S AFFIN              0  DISC     0.1  SRB      4903   IIT    0.002                LCK    0.000
   #SWAPS      75  INELIGIBLE             0  Q+PEND   0.1  TOT    141049   HST    0.000  AAP     0.00  SUP    0.000  -PAGE-IN RAT
   EXCTD        0  CONVERSION             0  IOSQ     0.0  /SEC       78   AAP    0.000  IIP     0.00                SINGLE
   AVG ENC   0.00  STD DEV              257                                 IIP    0.000                              BLOCK
   REM ENC   0.00                                          ABSRPTN    25K                                          SHARED
   MS ENC    0.00                                          TRX SERV   25K                                          HSP

   GOAL: EXECUTION VELOCITY 70.0%     VELOCITY MIGRATION:   I/O MGMT  90.9%     INIT MGMT  100%

            RESPONSE TIME EX   PERF  AVG   --EXEC USING%--  -------------- EXEC DELAYS % -----------  -USING%-  --- DELAY % ---
   SYSTEM                  VEL% INDX ADRSP  CPU AAP IIP I/O  TOT                                        CRY CNT  UNK IDL CRY CNT

   *ALL           --N/A--     100  0.7   6.6  0.0 0.0 0.0 0.0  0.0                                       0.0 0.0  0.0 100 0.0 0.0
   SC74                       100  0.7   5.6  0.0 0.0 0.0 0.0  0.0                                       0.0 0.0  0.0 100 0.0 0.0
   SC75                       0.0  0.0   1.0  0.0 0.0 0.0 0.0  0.0                                       0.0 0.0  0.0 100 0.0 0.0

                                        ----------RESPONSE TIME DISTRIBUTIONS----------
   SYSTEM: SC74 -----INTERVAL: 30.00.000 -----MRT CHANGES:  0 ---
      ----TIME----  -NUMBER OF TRANSACTIONS-  ------PERCENT------
      HH.MM.SS.TTT  CUM TOTAL      IN BUCKET  CUM TOTAL IN BUCKET
   <  00.00.00.030         63             63       79.7      79.7
   <= 00.00.00.036         64              1       81.0       1.3
   <= 00.00.00.042         64              0       81.0       0.0
   <= 00.00.00.048         65              1       82.3       1.3
   <= 00.00.00.054         66              1       83.5       1.3
   <= 00.00.00.060         66              0       83.5       0.0
   <= 00.00.00.066         67              1       84.8       1.3
   <= 00.00.00.072         67              0       84.8       0.0
   <= 00.00.00.078         67              0       84.8       0.0
   <= 00.00.00.084         67              0       84.8       0.0
   <= 00.00.00.090         67              0       84.8       0.0
   <= 00.00.00.120         69              2       87.3       2.5
   <= 00.00.00.240         71              2       89.9       2.5
   >  00.00.00.240         79              8        100      10.1
```

Figure 33-6 New field RESPONSE TIME DISTRIBUTIONS on WLMGS report

33.1.4 Resource group capacity values

In z/OS V1R13, the minimum and maximum values for CPU service units per second can be up to 99.999.999 (8-digit capacity values). They are supported in the Service Policy page of the Postprocessor Workload Activity report and in the Monitor III Resource Group Data report. This report also displays the actual value in 8 digits.

In addition, you can display the minimum, maximum, and actual resource group capacity in 8-digit values in the Monitor III Sysplex Summary report using the Monitor III report format definition utility (RMF UTIL).

Starting the report utility (RMF UTIL)

As a prerequisite for the invocation of the Monitor III report format definition utility, SYS1.SERBCLS must be concatenated to your SYSPROC library. For more information, refer to “Setting up RMF” in *z/OS RMF User's Guide*.

To start the utility, use one of the following commands:

- From TSO/E ready mode: RMF UTIL
- From within ISPF: TSO RMF UTIL

If you have the Kanji version of RMF, you start the Monitor III utility by entering:

- RMFJPN UTIL

For details about all values that can be displayed for all Monitor III reports and how to use each of the RMF UTIL panels, see *z/OS Resource Measurement Facility Programmer's Guide,* SC33-7994.

Important: Do not use a 3270 session with a window size smaller than 32 x 80. Do not try to access the report format definition utility in split-screen mode when you are in an active RMF Monitor III reporter session.

33.2 GRS and supervisor delay monitoring enhancements

In large systems, it can be difficult to detect and debug performance problems due to resource contention. System dumps or traditional performance reports might not be adequate tools to identify the address space that is causing a contention.

For this purpose, in z/OS V1R13, RMF obtains enqueue and latch performance statistics, as well as system suspend lock contention information at a system level and at address space levels from the z/OS global resource serialization (GRS) component and from the z/OS supervisor. The retrieved statistics are presented in two sections of the SDELAY report.

The new Postprocessor Serialization Delay report is available in XML format which provides system suspend lock, enqueue, and latch contention information. This report helps to detect serialization-related performance issues, including latch contention. RMF also stores this information in a new subtype 5 of SMF record 72.

33.2.1 Contents and division of the SDELAY report

The Serialization Delay report provides contention information at a system level and at an address space level for various types of suspend locks, GRS latches, and GRS ENQs.

Reported suspend lock types are CMS, CMS EQDQ, CMS Latch, CMS SMF, CML, and Local.

The Serialization Delay Report consists of two sections:

- The Serialization Delay Summary
- The Serialization Delay Details

Serialization Delay Summary section

The Serialization Delay Summary section, shown in Figure 33-7, contains system-wide summary data for all address spaces. It is divided into three subsections (*RMF User's Guide* provides more information about each field in the sections):

- The System Locks subsection displays summary data for system suspend locks.
- The GRS Latch subsection displays summary data about GRS latches.
- The GRS Enqueue subsection displays summary data about GRS enqueue requests.

Serialization Delay Summary

System Locks

Lock Type	Total Contention Time	Avg Contention Time	Total Contention Count	Contention Count with QLen>1
CMS	0		0	0
CMSEQDQ	0		0	0
CMSLatch	0		0	0
CMSSMF	0		0	0
Local	10	0.02	462	70
CML Owner	2	0.03	53	2

GRS Latch

GRS Mode: STAR

	Total Contention Time	Avg Contention Time	Std Dev of Contention Time	Total Contention Count
GRS Latch Set Creator	0	0.00	0.00	4

GRS Enqueue

GRS Mode: STAR

	Scope	Total Contention Time	Avg Contention Time	Std Dev of Contention Time	Total Request Count	Total Contention Count
GRS Enqueue	Step	0			544	0
GRS Enqueue	System	0			1857	0
GRS Enqueue	Systems	0			930	0

Figure 33-7 Serialization Delay Summary report sample

The field explanations for this report are listed in Table 33-2.

Table 33-2 Field explanation of Serialization Delay Summary report

Field heading	Meaning
System Locks – contains system-wide summary data on system suspend locks for all address spaces	
Lock Type	Displays the system suspend lock type: ► **CMS** - CMS lock ► **CMS EQDQ -** CMS Enqueue Dequeue lock ► **CMS Latch -** CMS Latch lock ► **CMS SMF -** CMS SMF lock ► **Local -** Local lock ► **CML Owner -** CML lock owner
Total Contention Time	The total amount of time in milliseconds that a unit of work was suspended by a lock of the indicated type.

Field heading	Meaning
Avg Contention Time	The average amount of time in milliseconds that a unit of work was suspended by a lock of the indicated type.
Total Contention Count	The total number of times that a unit of work was suspended by a lock of the indicated type.
Contention Count with QLen > 1	The total number of times that a unit of work was suspended by a lock of the indicated type when there was already at least one other unit of work suspended for the lock.
GRS Latch – contains summary data about GRS latches for all address spaces.	
GRS Mode	The operation mode of GRS: ► NONE ► RING ► STAR
Total Contention Time	The total amount of time in milliseconds that latch requests were suspended.
Avg Contention Time	The average amount of time in milliseconds that latch requests were suspended.
Std Dev of Contention Time	The standard deviation of the total contention time in milliseconds.
Total Contention Count	The total number of suspended latch requests.
GRS Enqueue – contains summary data about GRS enqueue requests for all address spaces.	
GRS Mode	The operation mode of GRS: ► NONE ► RING ► STAR
Scope	The scope of an GRS enqueue request: ► STEP ► SYSTEM ► SYSTEMS One line is displayed for requests of a certain scope.
Total Contention Time	The total amount of time in milliseconds that the GRS enqueue requests with the specified scope were suspended.
Avg Contention Time	The average amount of time in milliseconds that the GRS enqueue requests were suspended.
Std Dev of Contention Time	The standard deviation of the total contention time in milliseconds.
Total Request Count	The total number of GRS enqueue requests.
Total Contention Count	The total number of GRS enqueue requests that were suspended for this address space.

Serialization Delay Details section

The Serialization Delay Details section and the field descriptions provide the following information in four subsections, namely, the CMS Lock Details subsection; the CML and Local Lock Details subsection; the GRS Latch Details subsection; and the GRS Enqueue Details subsection. These subsections are explained here:

- The CMS Lock Details subsection contains detail data about CMS/CMSEQDQ/CMSLatch/CMSSMF locks per address space; see Figure 33-8.

Serialization Delay Details

Additional Information: No CMS Lock Details data available

CML and Local Lock Details

Address Space ID	Job Name	Service Class Name	Service Class Period	CML Lock Owner - Total Contention Time	CML Lock Owner - Avg Contention Time	CML Lock Owner - Total Contention Count	CML Lock Owner - Contention Count with QLen>1	Local Lock - Total Contention Time	Local Lock - Avg Contention Time	Local Lock - Total Contention Count	Local Lock - Contention Count with QLen>1	CML Lock Requestor - Total Contention Time	CML Lock Requestor - Avg Contention Time	CML Lock Requestor - Total Contention Count	CML Lock Requestor - Contention Count with QLen>1
002D	RMF	SYSSTC	1					3	0.02	148	0				
002F	HZSPROC	SYSSTC	1	2	0.04	45	2	2	0.05	40	3				
0016	IOSAS	SYSTEM	1					2	0.14	14	5	0	0.00	4	1
000A	SMSVSAM	SYSTEM	1					1	0.00	165	60				
0030	PFA	SYSSTC	1					0	0.00	40	0	1	0.02	41	1
0020	RMFGAT	SYSSTC	1					0	0.00	17	0	0	0.00	2	0
0024	RRS	SYSSTC	1					0	0.00	7	0	0	0.00	4	0
0025	TCPIP	SYSSTC	1					0	0.00	6	0				
0006	XCFAS	SYSTEM	1	0	0.00	4	0	0	0.00	7	0				
005E	DB9YDIST	SYSSTC	1					0	0.00	3	0				
0007	GRS	SYSTEM	1					0	0.00	2	0				
000C	WLM	SYSTEM	1					0	0.00	1	0				
0008	SMSPDSE	SYSTEM	1					0	0.00	2	0				
0013	JESXCF	SYSTEM	1					0	0.00	3	0	0	0.00	1	0
0017	IXGLOGR	SYSTEM	1					0	0.00	3	2				
005A	DB9YMSTR	SYSSTC	1					0	0.00	2	0				
0021	NET	SYSSTC	1					0	0.00	1	0				
002A	APPC	SYSSTC	1					0	0.00	1	0				
000B	CONSOLE	SYSTEM	1	0	0.00	2	0								
0010	OMVS	SYSTEM	1	0	0.00	2	0								
0027	JES3DLOG	SYSTEM	1									0	0.00	1	0

Figure 33-8 Serialization Delay Detail sample - CML report

- The CML and Local Lock Details subsection contains detail data about CML and local locks per address space descriptions; see Table 33-3.

Table 33-3 Field explanation of Serialization Delay Detail - CMS and CML

Field Heading	Meaning
CMS Lock Details – contains detail data about CMS/CMSEQDQ/CMSLatch/CMSSMF locks per address space	
Address Space ID	The address space identifier (ASID) of the job requesting the lock or waiting for it.
Jobname	The name of the job.
Service Class	The name of the service class that the job has been running in.
CMS - Total Contention Time	The total amount of time in milliseconds that a unit of work of the indicated address space was suspended on a CMS lock.
CMS - Avg Contention Time	The average amount of time in milliseconds that a unit of work of the indicated address space was suspended on a CMS lock.
CMS - Total Contention Count	The number of times that a unit of work of the indicated address space was suspended on a CMS lock.
CMS - Contention Count with Qlen > 1	The number of times that a unit of work of the indicated address space was suspended on a CMS lock when there was already at least one other unit of work suspended for the lock.
CML and Local Lock Details – contains detail data about CML and local locks per address space	
Address Space ID	The address space identifier (ASID) of the job requesting the lock or waiting for it.
Jobname	The name of the job.
Service Class Name	The name of the service class that the job has been running in.
Service Class Period	Service class period that the job has been running in.
CML Lock Owner - Total Contention Time	The total amount of time in milliseconds that a unit of work from another address space was suspended when requesting the local lock of the indicated address space.

Field Heading	Meaning
CML Lock Owner - Avg Contention Time	The average amount of time in milliseconds that a unit of work from another address space was suspended when requesting the local lock of the indicated address space.
CML Lock Owner - Total Contention Count	The number of times that a unit of work from another address space was suspended when requesting the local lock of the indicated address space.
CML Lock Owner - Contention Count with Qlen > 1	The number of times that a unit of work from another address space was suspended when requesting the local lock of the indicated address space and there was already at least one other unit of work waiting for this lock.
Local Lock - Total Contention Time	The total amount of time in milliseconds that a unit of work of the indicated address space was suspended on a local lock.
Local Lock - Avg Contention Time	The average amount of time in milliseconds that a unit of work of the indicated address space was suspended on a local lock.
Local Lock - Total Contention Count	The number of times that a unit of work of the indicated address space was suspended on a local lock.
Local Lock - Contention Count withQlen > 1	The number of times that a unit of work of the indicated address space was suspended on a local lock when there was already at least one other unit of work suspended.
CML Lock Requestor - Total Contention Time	The total amount of time in milliseconds that a unit of work of the indicated address space was suspended when requesting the local lock of another address space.
CML Lock Requestor - Contention Time	The average amount of time in milliseconds that a unit of work of the indicated address space was suspended when requesting the local lock of another address space.
CML Lock Requestor - Total Contention Count	The number of times that a unit of work from this address space was suspended when requesting the local lock of another address space.
CML Lock Requestor - Contention Count with Qlen > 1	The number of times that a unit of work from this address space was suspended when requesting the local lock of another address space and there was already at least one other unit of work waiting for that lock.

GRS Mode: STAR

Address Space ID	Job Name	Service Class Name	Service Class Period	Latch Set Creator - Total Contention Time	Latch Set Creator - Avg Contention Time	Latch Set Creator - Std Dev Contention Time	Latch Set Creator - Total Contention Count	Latch Requestor - Total Contention Time	Latch Requestor - Avg Contention Time	Latch Requestor - Std Dev Contention Time	Latch Requestor - Total Contention Count
0010	OMVS	SYSTEM	1	0	0.00	0.00	4				
0020	RMFGAT	SYSSTC	1					0	0.00	0.00	2
001D	BPXOINIT	SYSTEM	1					0	0.00	0.00	2

GRS Enqueue Details

GRS Mode: STAR

Address Space ID	Job Name	Service Class Name	Service Class Period	ENQ STEP - Total Contention Time	ENQ STEP - Avg Contention Time	ENQ STEP - Std Dev Contention Time	ENQ STEP - Request Count	ENQ STEP - Contention Count	ENQ SYSTEM - Total Contention Time	ENQ SYSTEM - Avg Contention Time	ENQ SYSTEM - Std Dev Contention Time	ENQ SYSTEM - Request Count	ENQ SYSTEM - Contention Count	ENQ SYSTEMS - Total Contention Time	ENQ SYSTEMS - Avg Contention Time	ENQ SYSTEMS - Std Dev Contention Time	ENQ SYSTEMS - Request Count	ENQ SYSTEMS - Contention Count
0033	CATALOG	SYSTEM	1	0			100	0						0			250	0
0030	PFA	SYSSTC	1	0			120	0										
0020	RMFGAT	SYSSTC	1	0			12	0	0			606	0					
001D	BPXOINIT	SYSTEM	1	0			80	0										
000C	WLM	SYSTEM	1	0			192	0						0			600	0
000B	CONSOLE	SYSTEM	1	0			30	0	0			14	0					
0001	*MASTER*	SYSTEM	1	0			10	0	0			6	0					
0031	SMS	SYSSTC	1						0			80	0	0			40	0
002D	RMF	SYSSTC	1						0			40	0					
002A	APPC	SYSSTC	1						0			240	0					
0019	CEA	SYSTEM	1						0			1	0					
0010	OMVS	SYSTEM	1						0			5	0					
0009	SMSPDSE1	SYSTEM	1						0			433	0					
0008	SMSPDSE	SYSTEM	1						0			432	0					
0024	RRS	SYSSTC	1											0			40	0

Figure 33-9 Serialization Delay Detail sample - GRS report

- Figure 33-9 shows the detail data about GRS enqueue requests contained in the GRS Enqueue Details subsection.
- The GRS Latch Details subsection contains detail data about GRS latches; see Table 33-4.

Table 33-4 Field explanation of Serialization Delay Detail - GRS

Field heading	Meaning
GRS Latch Details – contains detail data about GRS latches.	
GRS Mode	The operation mode of GRS: ► NONE ► RING ► STAR
Address Space ID	The address space identifier (ASID) of the job requesting the lock or waiting for it.
Jobname	The name of the job.
Service Class Name	The name of the service class that the job has been running in.
Service Class Period	Service class period that the job has been running in.
Statistics for latch obtain requests against latch sets created by this address space:	
Latch Set Creator - Total Contention Time	The amount of contention time in milliseconds that was caused by latch set creator requests.
Latch Set Creator - Avg Contention	The average amount of contention time in milliseconds that was caused by latch set creator requests.
Latch Set Creator - Std Dev of Contention Time	The standard deviation of the total contention time.
Latch Set Creator - Total Contention Count	The number of times a latch obtain request was suspended.
Statistics for latch obtain requests issued from this address space:	
Latch Requestor - Total Contention Time	The amount of contention time in milliseconds that was caused by latch obtain requests.
Latch Requestor - Avg Contention Time	The average amount of contention time in milliseconds that was caused by latch obtain requests.
Latch Requestor - Std Dev of Contention Time	The standard deviation of the total contention time.
Latch Requestor - Total Contention Count	The number of times a latch obtain request was suspended.
GRS Enqueue Details – contains detail data about GRS enqueue requests.	
GRS Mode	The operation mode of GRS: ► NONE ► RING ► STAR
Address Space ID	The address space identifier (ASID) of the job requesting the lock or waiting for it.
Jobname	The name of the job.
Service Class Name	The name of the service class that the job has been running in.

Field heading	Meaning
Service Class Period	Service class period that the job has been running in.
ENQ STEP - Total Contention Time	The total amount of contention time in milliseconds that was caused by GRS enqueue requests of SCOPE = STEP for this address space.
ENQ STEP - Avg Contention Time	The average amount of contention time in milliseconds.
ENQ STEP - Std Dev of Contention Time	The standard deviation of the total contention time.
ENQ STEP - Request Count	The total number of GRS enqueue requests of SCOPE = STEP for this address space.
ENQ STEP - Contention Count	The total number of GRS enqueue requests of SCOPE = STEP that were suspended for this address space.
ENQ SYSTEM - Total Contention Time	The total amount of contention time in milliseconds that was caused by GRS enqueue requests of SCOPE = SYSTEM for this address space.
ENQ SYSTEM - Avg Contention Time	The average amount of contention time in milliseconds.
ENQ SYSTEM - Std Dev of Contention Time	The standard deviation of the Total Contention Time.
ENQ SYSTEM - Request Count	The total number of GRS enqueue requests of SCOPE = SYSTEM for this address space.
ENQ SYSTEM - Contention Count	The total number of GRS enqueue requests of SCOPE = SYSTEM that were suspended for this address space.
ENQ SYSTEMS - Total Contention Time	The total amount of contention time in milliseconds that was caused by GRS enqueue requests of SCOPE = SYSTEMS for this address space.
ENQ SYSTEMS - Avg Contention Time	The average amount of contention time in milliseconds.
ENQ SYSTEMS - Std Dev of Contention Time	The standard deviation of the Total Contention Time.
ENQ SYSTEMS - Request Count	The total number of GRS enqueue requests of SCOPE = SYSTEMS for this address space.
ENQ SYSTEMS - Contention Count	The total number of GRS enqueue requests of SCOPE = SYSTEMS that were suspended for this address space.

Note: For more information about each SDELAY report, see *z/OS Resource Measurement Facility Report Analysis*, SC33-7991.

33.3 Redesign of the Postprocessor Paging Activity report

The Postprocessor Paging Activity report is redesigned to provide a more current and complete picture of storage distribution.

The obsolete Swap Placement Activity section is no longer available. Instead, the report contains a new Memory Objects and Frames section with values to provide information about storage occupation.

The Paging Activity report is also available in XML format. For more information about how to create a XML report, see *z/OS Resource Management Facility User's Guide*, SC33-7990.

33.3.1 Purpose and uses of the information given on the report

The Paging Activity report provides information about the demands made on the system paging facilities and the use of main storage and external page storage during the interval.

If the non-swap, non-VIO page fault rate (page-ins) is excessively high, it can be the result of overcommitment of main storage.

Other problems to look for are high pageable system area non-swap page-in rates, which can be caused by a poor pack list or a large number of fixed LPA modules. A period of high VIO slot use can be a sign that a specific job is making excessive use of VIO. Always be alert for bad slots because they can cause executing jobs to end abnormally.

33.3.2 New contents of the report

Monitor I gathers data for this report automatically. If you want to suppress gathering, you need to specify NOPAGING.

To produce this report, specify REPORTS(PAGING) on the job used to generate the report. You can also use the ISPF panel to create the job, as seen on 33.1.1, “WLMGL postprocessing report” on page 720.

Figure 33-10 shows a sample of the Memory Objects and Frames report section if Enhanced DAT architecture is supported. If Enhanced DAT architecture is not supported, the fields MEMORY OBJECTS- 1MB and FRAMES - 1 MB are set to N/A.

```
                                        P A G I N G   A C T I V I T Y
                                                                                                                                  PAGE 0
            z/OS V1R13              SYSTEM ID SC74            START 05/02/2011-09.30.00  INTERVAL 000.29.59
                                    RPT VERSION V1R13 RMF     END   05/02/2011-10.00.00  CYCLE 1.000 SECONDS
OPT = IEAOPT00   LFAREA SIZE = 104857600   MEMORY OBJECTS AND FRAMES
-------------------------------------------------------------------------------------------------------------------------------------------
MEMORY OBJECTS     COMMON      SHARED       1 MB
--------------  ---------  ----------  ---------
      MIN              32           3          0
      MAX              32           3          0
      AVG              32           3          0
FRAMES             COMMON  COMM FIXED     SHARED       1 MB
--------------  ---------  ----------  ---------  ---------
      MIN           4,719       3,102     25,521          0
      MAX           4,719       3,102     25,521          0
      AVG           4,719       3,102     25,521          0
```

Figure 33-10 PAGING Report sample- New Memory Objects and Frames

Table 33-5 lists and explains each field in the report.

Table 33-5 Fields in the Paging Activity report - Memory Objects and Frames

Field heading	Meaning
MEMORY OBJECTS	**COMMON:** Number of memory objects allocated in the high virtual common storage of the system. **SHARED:** Number of memory objects allocated in the high virtual shared storage of the system. **1MB:** Number of 1 MB memory objects allocated in the system (only available if the Enhanced DAT architecture is supported).
FRAMES	**COMMON:** Number of high virtual common memory 4 K frames that are backed in main storage. **COMMON FIXED:** Number of high virtual common memory 4 K frames that are fixed in main storage. **SHARED:** Number of high virtual shared memory 4 K frames that are backed in main storage. **1 MB:** Number of 1 MB frames that are backed in main storage (only available if the Enhanced DAT Architecture is supported).

33.4 Enhanced RMF Monitor II OPT Settings report

RMF provides a Monitor II OPT Settings report that displays information about the active OPT settings in the IEAOPTxx parmlib member and the settings of all OPT parameters.

In z/OS V1R13, the OPT Settings report displays information about four additional OPT parameters.

You can request an OPT report in two ways:

- In ISPF, specify **L** on the Monitor II Primary Menu. This leads you to the Library List and OPT Settings Selection Menu, where you can select **4** “IEAOPTxx - OPT Settings”. See Figure 33-11.
- Alternatively, in the command interface of an ISPF or TSO/E display session, you can type the command `OPT`.

```
          RMF Monitor II Library List and OPT Settings Selection Menu
Selection ===> 4

Enter selection number or command on selection line.

  1 Link list         LNKLSTxx - Link Library list                      (LLI)
  2 LPA list          LPALSTxx - LPA Library List                   (LLI LPA)
  3 APF list          IEAAPFxx - Authorized Program List            (LLI APF)

  4 OPT               IEAOPTxx - OPT Settings                           (OPT)
```

Figure 33-11 IEAOPTxx parmlib member - OPT settings

33.4.1 Purpose and content of the report

The currently active OPT member is shown in Figure 33-12.

```
 RMF - OPT Settings                    Line 1 of 34
Command ===>                                          Scroll ===> PAGE

                       CPU=  1/  1 UIC= 65K PR=   0         System= SC74 Total

OPT: 00            Time: N/A
-- Parameter --  - Default - -- Value -- Unit ---------- Description ----------

ABNORMALTERM             Yes         Yes Y/N  Abnormal terminations in routing
BLWLINTHD                 20          20 sec  Time blocked work waits for help
BLWLTRPCT                  5           5 0/00 CPU cap. to promote blocked work
CCCAWMT                12000        3200 usec Alternate wait management time
ZAAPAWMT               12000        3200 usec AWM time value for zAAPs
ZIIPAWMT               12000        3200 usec AWM time value for zIIPs
CCCSIGUR                  45          33 msec Min. mean-time-to-wait threshold
CNTCLIST                  No          No Y/N  Clist commands count individually
CPENABLE           10,30|0,0       10,30 %    Threshold for TPI (low,high)
DVIO                     Yes         Yes Y/N  Directed VIO is active
ERV                      500      500/CB SU   Enqueue residency CPU Service/DP
FULLPRESYSTEM             No          No Y/N  System AS can preempt other work
HIPERDISPATCH             No     Yes/Yes Y/N  Hiperdispatch is desired/active
IFAHONORPRIORITY         Yes         Yes Y/N  Allows CPs to help zAAPs
IIPHONORPRIORITY         Yes         Yes Y/N  Allows CPs to help zIIPs
INITIMP                    0        0/FE #    INITIMP value/DP for initiators
IRA405I             70,50,50    70,50,50 %    Fixed storage of <16M,16M-2G,tot
MANAGENONENCLAVE          No          No Y/N  Manage non-enclave work
MAXPROMOTETIME             6           6 *10s Holder allowed to run promoted
MCCAFCTH           3993,7986   3992,7984 #    Threshold for storage (low,ok)
MCCFXEPR                  92          92 %    Fixed storage threshold < 16 MB
MCCFXTPR                  80          80 %    Fixed online storage threshold
PROJECTCPU                No          No Y/N  CPU projection for zAAPs, zIIPs
RCCFXET                82,88       82,88 %    Physical MPL threshold (low,high)
RCCFXTT                66,72       66,72 %    Logical MPL threshold (low,high)
RMPTTOM            1000|3000        3000 msec SRM invocation interval
RTPIFACTOR               100         100 %    PI affects server routing weights
STORAGENSWDP             Yes         Yes Y/N  Sets non-swap. ASID non-dispatch.
STORAGESERVERMGT          No          No Y/N  Storage I/O priority management
STORAGEWTOR              Yes         Yes Y/N  WTOR to cancel AS in shortage
TIMESLICES                 1           1 #    Time slices for discretionary wrk
VARYCPU                  Yes          No Y/N  VARYCPU is enabled
VARYCPUMIN                 1           1 #    VARYCPUMIN value
WASROUTINGLEVEL            0           0 #    WebSphere routing level
```

Figure 33-12 OPT settings

The field descriptions are listed and explained in Table 33-6.

Table 33-6 Fields in the OPT Settings report

Field heading	Meaning
OPT	Suffix xx in the name of the active option member IEAOPTxx. The option member contains parameters that affect System Resource Manager (SRM) decisions.
Time	Time stamp when the IEAOPTxx member was activated. If the systems programmer did not change the active IEAOPTxx member of SYS1.PARMLIB since the last IPL, then N/A is shown.
Parameter	Name of the WLM OPT parameter.
Default	Default value or values of the parameter. If more than one default exists, the values are separated by a vertical bar (|).
Value	Current value or values of the parameter. This value can differ from the value originally specified. With two values displayed, separated by a forward (/) slash, the second value is provided by SRM. Also, parameters that are not set in the IEAOPTxx member are shown with the default value, if not changed otherwise. For information about how SRM handles the settings of OPT parameters, refer to *z/OS MVS Initialization and Tuning Reference*. When RMF cannot obtain any data for a parameter, 'No Data' is shown
Unit	Unit in which the parameter value is measured.
Description	Basic description of the purpose of the parameter. For detailed information, refer to *z/OS MVS Initialization and Tuning Reference.*

33.5 Integrated ensemble performance monitoring

The IBM zEnterprise integrates the System z platform with System x and System p under a single Unified Resource Manager (URM).

With the z/OS V1R13 RMF Monitor, RMF provides new CIM-based performance data gatherers for Linux on System z, Linux on System x, and AIX on System p. Exploiters of the HTTP API provided by the DDS can send an HTTP request to retrieve performance data from the endpoints running the Linux or AIX operating systems. The DDS returns the requested data as a structured XML document.

With the Resource Monitoring plug-in for IBM z/OS Management Facility (z/OSMF), performance metrics from connected Linux or AIX systems can be displayed in the same way, and together with z/OS in heterogeneous client environments, with a single point of control for performance monitoring; see Figure 33-13. Now you can identify performance problems at a glance using a common look and feel for all platforms running on zEnterprise.

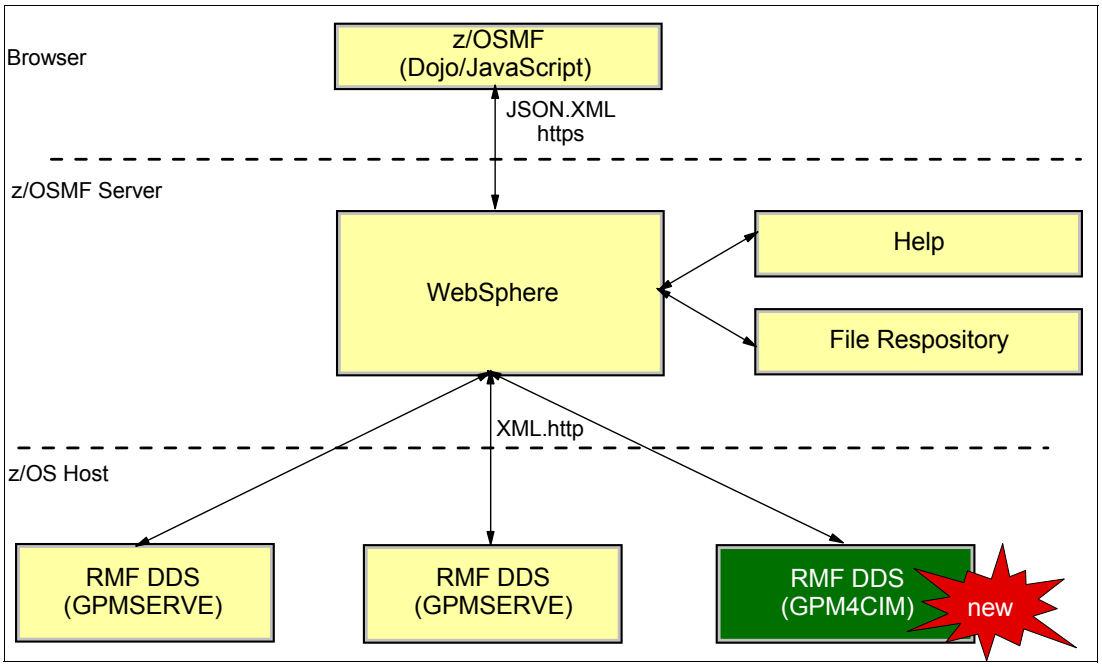


Figure 33-13 z/OSMF resource monitoring architecture

The Common Information Model (CIM) is a standard data model developed by a consortium of major hardware and software vendors (including IBM), called the Distributed Management Task Force (DMTF). It provides a common definition for describing and accessing systems management data in an heterogeneous environments. It allows vendors and system programmers to write applications (CIM monitoring clients) that measure system resources in a network with various operating systems and hardware, and to manage those systems.

The z/OS base element Common Information Model (z/OS CIM) implements the CIM server, based on the OpenPegasus open source project. A CIM monitoring client invokes the CIM server, which in turn collects z/OS metrics from the system and returns it to the calling client. A part of z/OS CIM is the eServer OS management profile. It provides access to, and management of, the base IT resources, like zSeries LPARs, z/OS operating system images, and z/OS address spaces.

A part of z/OS RMF is the eServer OS monitoring profile. It provides monitoring data for the IT resources exposed by the eServer OS management profile. This profile allows read access to monitoring data that pertains to the current time, and also provides navigation between the resources and their monitoring data. If a CIM client requests the CIM server to obtain z/OS metrics, the CIM server invokes the appropriate z/OS RMF monitoring provider which retrieves these metrics associated to z/OS system resources. The z/OS RMF monitoring providers use RMF Monitor III performance data.

Figure 33-14 shows the RMF ensemble architecture of a zEnterprise environment with z/OS and AIX running on a z/BX.

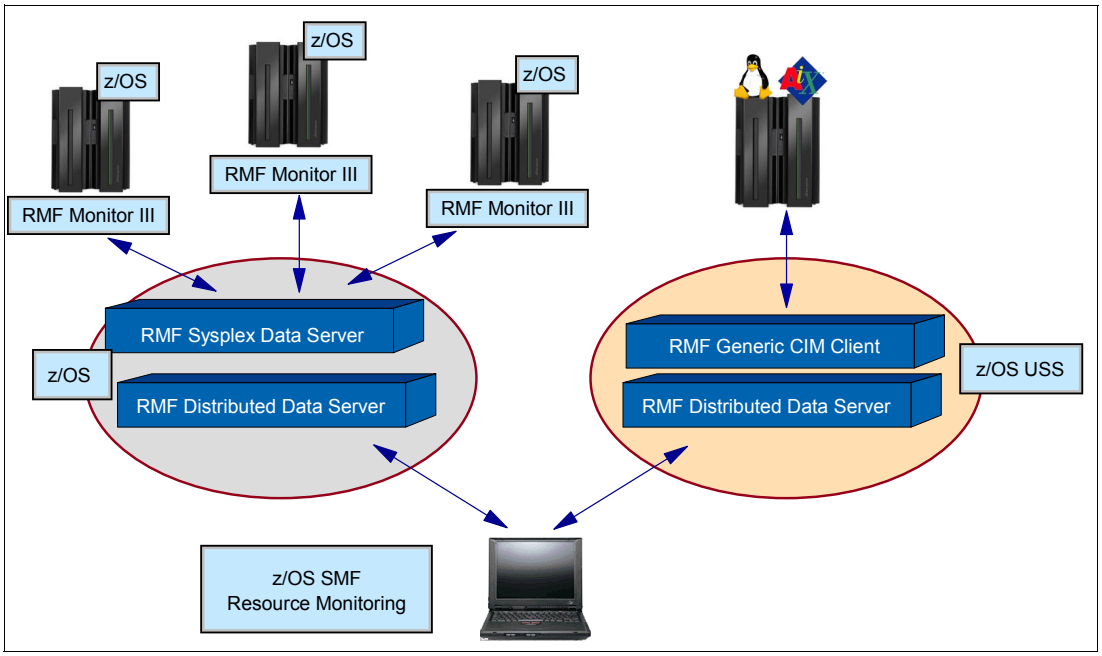


Figure 33-14 RMF Ensemble architecture on zEnterprise (z/OS, z/BX, and AIX)

Figure 33-15 displays the ensemble on a Linux environment.

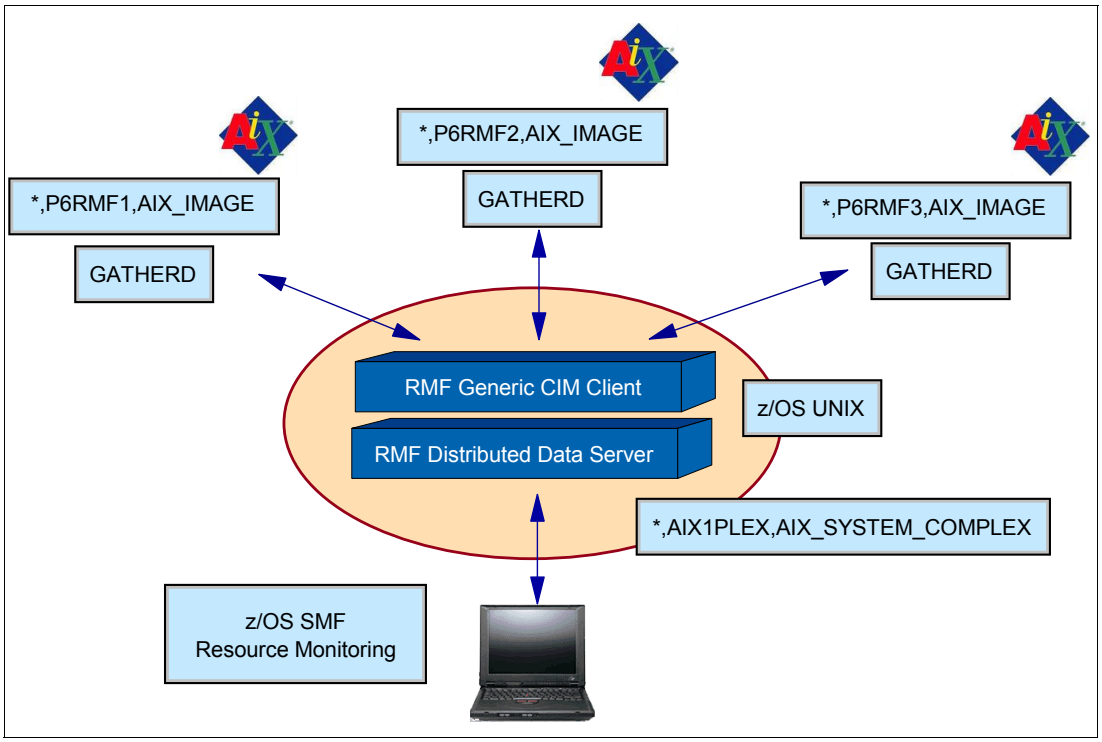


Figure 33-15 RMF ensemble architecture with Linux agents (zSeries on z/VM, LPAR, and IBM xSeries®)

33.5.1 Interactions and dependencies

To install and use the RMF ensemble you need to install an agent on each system that you want to monitor and start the z/OS client task (GPM4CIM) that collects the data on each of them. You also need to have the RMF Distributed Data Server for z/OS (GPMSERVE) started, but the client and the server do not have to be on the same system because each of them use a dedicated TCP/IP port number.

The agents software dependencies are listed here:

- AIX
  - AIX 5.3+, 6.1+
  - Packages
    - sysmgt.cimserver.pegasus
    - sysmgt.cim.providers
    - sysmgt.cim.smisproviders
- Linux
  - RED HAT Linux 5.3, 5.4, 6.0 32- and 64-bit
  - SUSE Linux 10, 11 32- and 64-bit
  - Packages
    - tog-pegasus or sblim-sfcb
    - sblim-gather-provider
    - (sblim = standard based linux instrumentation for manageability)
- The hardware must be able to run any of these operating systems.

33.5.2 Installation considerations

To use the z/OS RMF ensemble client task, you need to create a task that runs in the OMVS shell and collects the data on the agents of the other systems.

- The SYS1.IBM.PROCLIB data set has member GPM4CIM as a sample procedure. Copy that member to your system proclib data set.

  Note: To run this task, the user ID that GPM4CIM runs under must have READ access to the resource BPX.WLMSERVER in the FACILITY class.

- The `bin` directory contains the `gpm4cim_setup.sh` script. This script creates all the other necessary directories and copies the needed files.

  ```
  /usr/lpp/gpm/bin
  ```

- The JCL shown in Figure 33-16, with modifications for your installation, can be used start the z/OS ensemble. Also create a specific filesystem for at least the following file:

  ```
  /var/gpm/logs
  ```

- Copy the configuration file from `/usr/lpp/gpm/etc/gpm4A.cfg` to `/etc/gpm`.

```
//GPM4CIM PROC OS=A
//*******************************************************************
//STEP1    EXEC PGM=BPXBATCH,TIME=NOLIMIT,REGION=0M,
//       PARM='PGM /usr/lpp/gpm/bin/gpm4cim cfg=/etc/gpm/gpm4&OS..cfg'
//STDENV   DD   PATH='/etc/gpm/gpm4cim.env'
//STDOUT   DD   PATH='/var/gpm/logs/gpm4cim&OS..out',
//         PATHOPTS=(OWRONLY,OCREAT,OTRUNC),
//         PATHMODE=(SIRUSR,SIWUSR,SIRGRP)
//STDERR   DD   PATH='/var/gpm/logs/gpm4cim&OS..trc',
//         PATHOPTS=(OWRONLY,OCREAT,OTRUNC),
//         PATHMODE=(SIRUSR,SIWUSR,SIRGRP)
///*******************************************************************
//STEP2    EXEC PGM=BPXBATCH,
//         PARM='PGM /bin/cat /var/gpm/logs/gpm4cim&OS..out'
//STDOUT   DD   SYSOUT=*
//STDERR   DD   SYSOUT=*
///*******************************************************************
//STEP3    EXEC PGM=BPXBATCH,
//         PARM='PGM /bin/cat /var/gpm/logs/gpm4cim&OS..trc'
//STDOUT   DD   SYSOUT=*
//STDERR   DD   SYSOUT=*
//*******************************************************************
//
```

Figure 33-16 The procedure to start the z/OS ensemble

- Customize the `/etc/gpm/gpm4A.cfg` file and change the default listener port, if needed. Point to the TCP/IP address of each CIM instrumentation client on the other systems (AIX and Linux), as shown in Figure 33-17.

```
/*******************************************************************/
MAXSESSIONS_HTTP(20)            /* MaxNo of concurrent HTTP requests */
HTTP_PORT(8805)                 /* Port number for HTTP requests     */
HTTP_ALLOW(*)                   /* Mask for hosts that are allowed   */
HTTP_NOAUTH()                   /* No server can access without auth.*/
/*******************************************************************/
INTERVAL(300)                   /* Monitoring interval (seconds)     */
AIX_COMPLEX(SAPPLEX)            /* User defined name of AIX complex  */
                                /* AIX images following here         */
AIX_IMAGE(SAP1.US.IBM.COM:5988,SAP2.US.IBM.COM:5988)
AIX_IMAGE(SAP3.US.IBM.COM:5988)
```

Figure 33-17 Part of the file gpm4A.cfg

- After this is done, you can start an ensemble client on z/OS. Note that it is also necessary to install and start the packages listed in 33.5.1, “Interactions and dependencies” on page 739 on each system from which you want to collect data.

33.6 RMF data portal and z/OSMF integration

All the data collected is also available on the RMF Distributed Data Server for z/OS (also known as GPMSERVE); see Figure 33-18. The ensemble client and GPMSERVE do not need to run on the same system. Each of them can use a separate TCP/IP port.

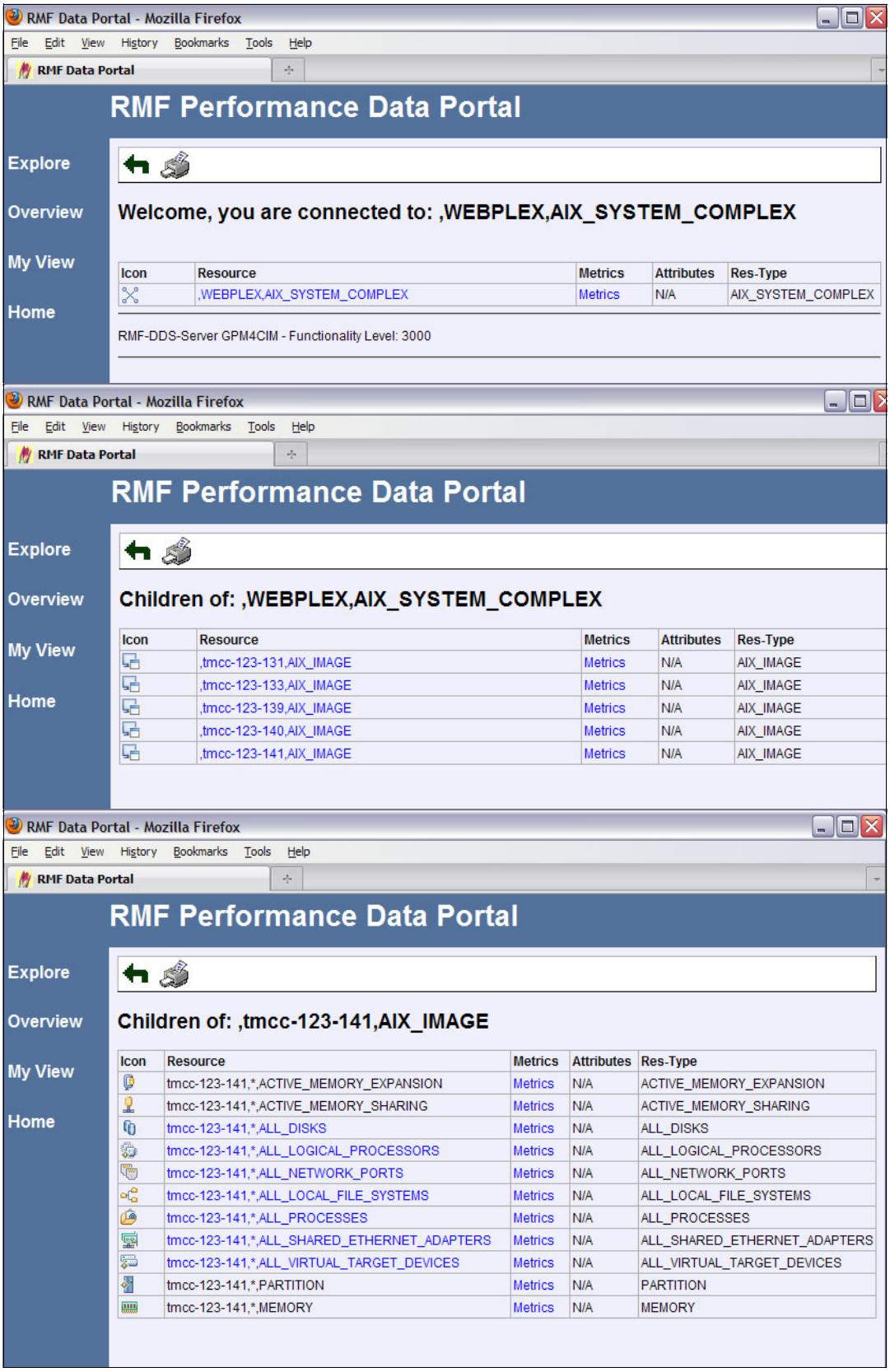


Figure 33-18 z/OS RMF Data Portal sample screens on a hybrid environment

33.6.1 z/OSMF Resource Monitoring plug-in

For a more centralized and consistent installation monitoring solution for zEnterprise, you can also use z/OSMF in this hybrid computing environment scenario that is new with z/OS V1R13, in parallel with RMF providing new CIM-based performance data gatherers for Linux on System z, Linux on System x, and AIX systems.

Together with the Resource Monitoring plug-in for the z/OS Management Facility, which was first made available with z/OSMF V1R12, this displays performance metrics from those platforms and combines them with z/OS metrics in common graphic views.

This monitoring solution can be achieved without making changes to z/OSMF after it has been configured to display the RMF DDS data (apart from adding the new systems, dashboards, and metrics, as with another z/OS system).

This new scenario contains new wording but has the same meaning, because resource monitoring across various hardware platforms provides added value within an ensemble, as follows:

- Ensemble Resource Monitoring
- Hybrid Monitoring
- Cross Platform Resource Monitoring (RMF XP)
- Classic RMF Distributed Data Server
  - RMF DDS
  - GPMSERVE
- RMF Distributed Data Server for Hybrid Monitoring
  - RMF XP DDS
  - GPM4CIM

z/OSMF Resource Monitoring task

The Resource Monitoring task in z/OSMF provides a web-based user interface that you can use to monitor the performance of the z/OS sysplexes, AIX system complexes (System p), Linux system complexes (System z and System x), or Linux images (System z and System x) in your environment.

With the Resource Monitoring task, you can monitor most of the metrics supported by RMF Monitor III, create and save custom views of the metrics, and display real-time data as bar charts.

For z/OS sysplexes, the Resource Monitoring task takes its input from a single data server on one system in the sysplex. That data server collects data from the RMF Monitor III data gatherer on each image in the sysplex. This function is called the Distributed Data Server (DDS). To allow monitoring of several sysplexes, ensure that each sysplex has an active DDS.

Similarly for Linux or AIX system complexes, the Resource Monitoring task collects input from a Cross Platform Distributed Data Server on a z/OS system that gathers data from CIM servers on the systems to be monitored. The Resource Monitoring task can also monitor single Linux images or guests. Here, the task collects input from the RMF Linux data gatherer (rmfpms).

The Resource Monitoring task “Add Metric action” is shown in Figure 33-19.

- In the Resource tab, the AIX and Linux system complexes are listed, together with the z/OS sysplexes.
- The behavior did not change; the user uses the resource tree to navigate to the desired resource, then switches to the Metric tab and selects the desired metric (depending on the selected resource).

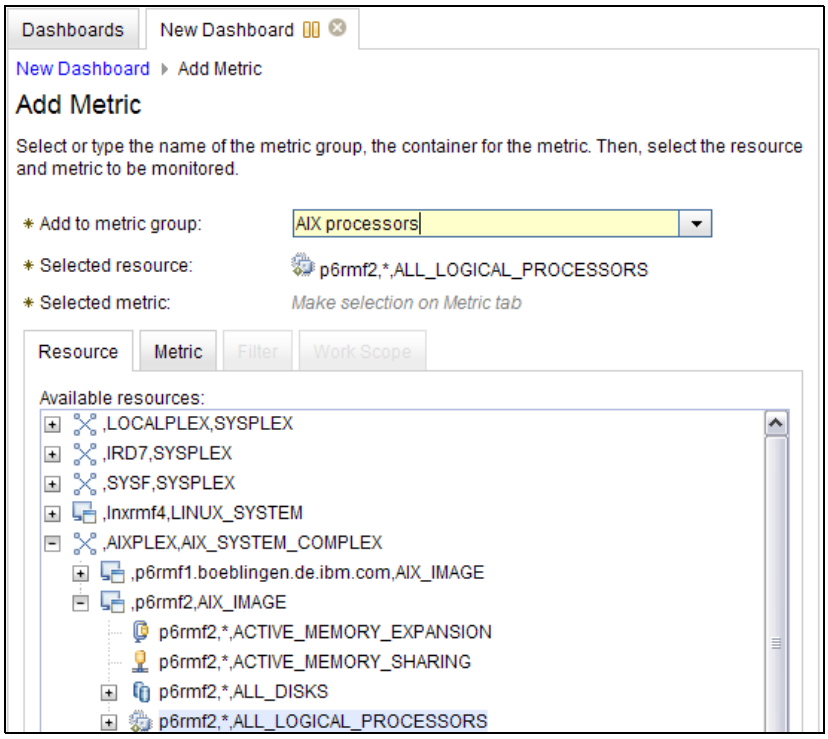


Figure 33-19 z/OSMF - Adding a metric based on an AIX source

The metrics of various platforms can be combined in the same dashboard. The sample in Figure 33-20 shows the CPU time spent by AIX processes and the percentage of processor time used by the z/OS job.

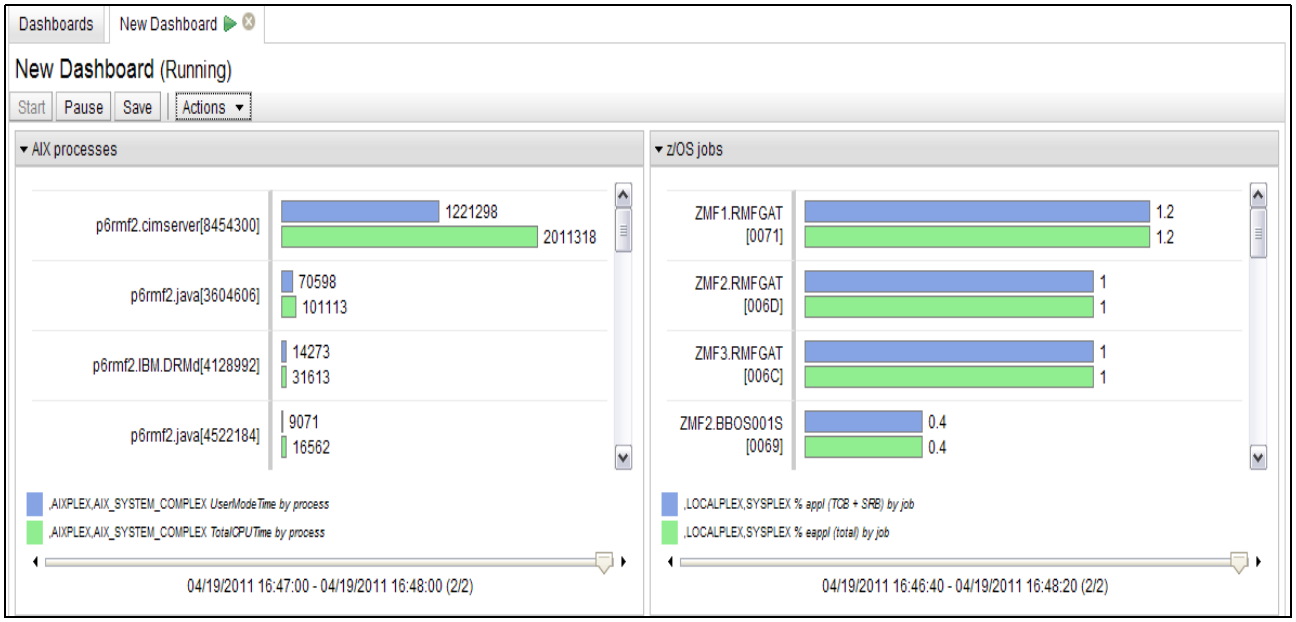


Figure 33-20 z/OSMF - Dashboard monitoring AIX clients

33.6.2 System Status task overview

The System Status task in z/OSMF combines data from an entire sysplex into one performance indicator so that you can quickly assess the performance of the workloads running on the z/OS sysplexes in your environment. The System Status task also provides a single location where you can define the z/OS sysplexes to be monitored in the Resource Monitoring task.

You can also manage the AIX system complexes (System p) and Linux system complexes (System z and System x) to be monitored in the Resource Monitoring task. To do so, the RMF Cross Platform Distributed Data Server must be installed and configured properly. Further, you can manage single Linux images using the Linux data gatherer (rmfpms).

z/OSMF user interfaces

There are also changes to the z/OSMF user interface, with several fields changing names and other fields with new values, as shown in Figure 33-21:

- System Status task, Add Entry/Modify Entry actions:
  - Field “Operating system” is renamed to “Target system type”
  - Field “Port” is moved below the “Target system type”
- Default ports:
  - z/OS: 8803
  - AIX: 8805
  - Linux on System x: 8806
  - Linux on System z: 8807
  - Linux (rmfpms): 8803

Note: If you use default port numbers and you change the target system type, the port number changes to the default port of that target system type.

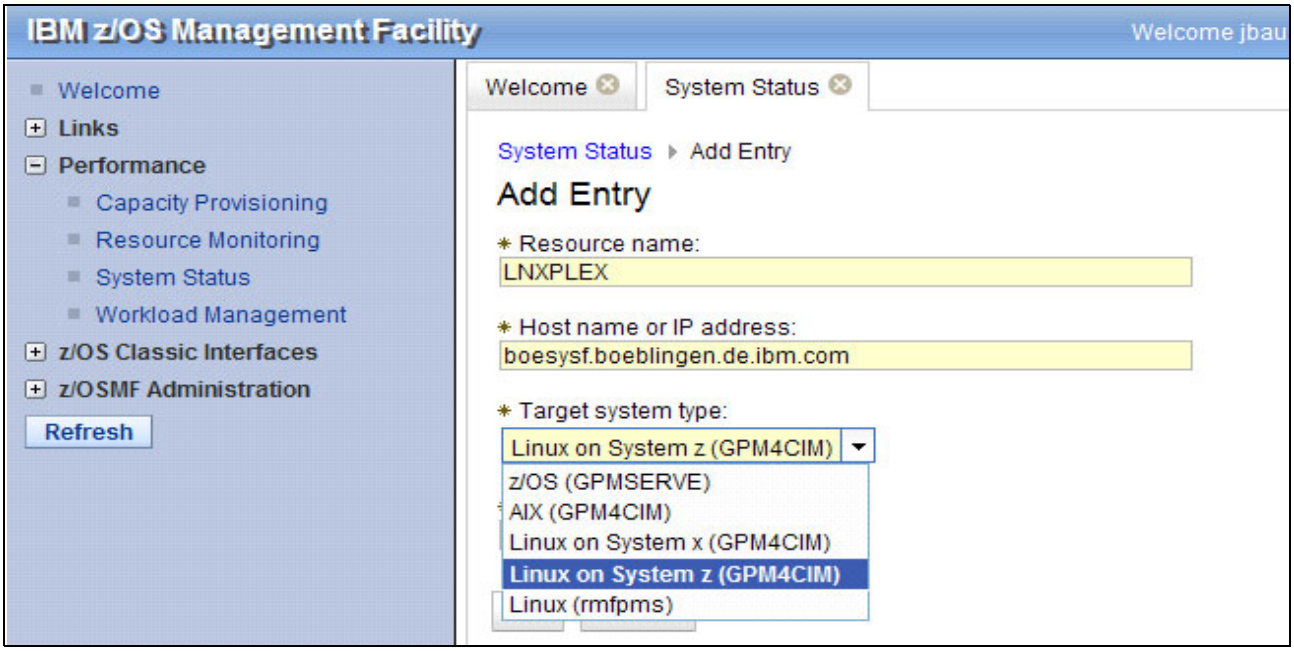


Figure 33-21 z/OSMF - Adding a CIM Agent on Linux

33.7 z/VM guest support

RMF enhances its CPU reporting capabilities for systems running as z/VM guests. SMF record type 70 subtype 1 (CPU activity) provides simplified PR/SM partition and logical processor data sections for the guest system.

This enables the Partition Data Report section of the CPU Activity report to provide dispatch times and processor utilizations for this guest system. The additional CPU data for z/VM guests is measured as soon as the new gatherer option VMGUEST for Monitor I is specified.

Defining parameters for Monitor I

RMF ships a default ERBRMF00 parmlib member and an alternative one ERBRMF02 parmlib member to specify the Monitor I gatherer options, as follows:

ERBRMF00 This is the default parmlib member for Monitor I gatherer sessions. It contains the options that RMF defaults to anyway, if none were specified in a parmlib member.

ERBRMF02 This is the alternative parmlib member for Monitor I gatherer sessions. It contains options appropriate for monitoring all resources in the system.

Note: To use this new support, specify the new gatherer option VMGUEST in ERBRMF00, because the default is NOVMQUEST.

34

z/OS Batch Runtime

In current distributed z/OS environments, installations with a need to re-engineer existing native z/OS COBOL applications want to incorporate the Java language to take advantage of its larger developer skill base and many language features.

Moreover, a mixed z/OS COBOL and Java batch application must share a DB2 connection and use relevant language APIs to communicate with DB2 in the same unit of work (UOW).

Additionally, during the running of such a re-engineered program, Embedded Structured Query Language (SQL) DB2 access in Enterprise COBOL and Java Database Connectivity (JDBC), Structured Query Language for Java (SQLJ) DB2 access in Java, or both, must coexist transparently.

This chapter describes the IBM z/OS Batch Runtime product. z/OS Batch Runtime provides the ability to update the DB2 database from both COBOL and Java in a single transaction.

This chapter discusses the following topics:

- Java COBOL interoperability
- JCL instream data sets
- Evict jobs on a step boundary
- Support for JOBRC (return code)

For information about the SDSF perspective regarding several of these topics, see “JES2 enhancements” on page 691.

34.1 Batch runtime - from punch cards to Java batch

The past 45 years of evolution in batch processing on the IBM mainframe has provided the foundation for heavy-duty, reliable, and efficient batch for most large companies in the world. The foundation includes:

- WLM batch initiators
- Batch and print subsystems, JES3, JES2, and PSF
- Job control language (JCL)
- Batch management interfaces (that is, SDSF)
- Step and Job dependencies through condition codes and job networks
- Online and batch in parallel
- Time-driven job execution
- Job/Step restart functions, start, submit, remote submit, syntax scanner
- Accounting based on Job/USER, job statistics and RMF reports
- Pre-loaded address spaces (initiators)
- All mainframe programming languages can be used in batch

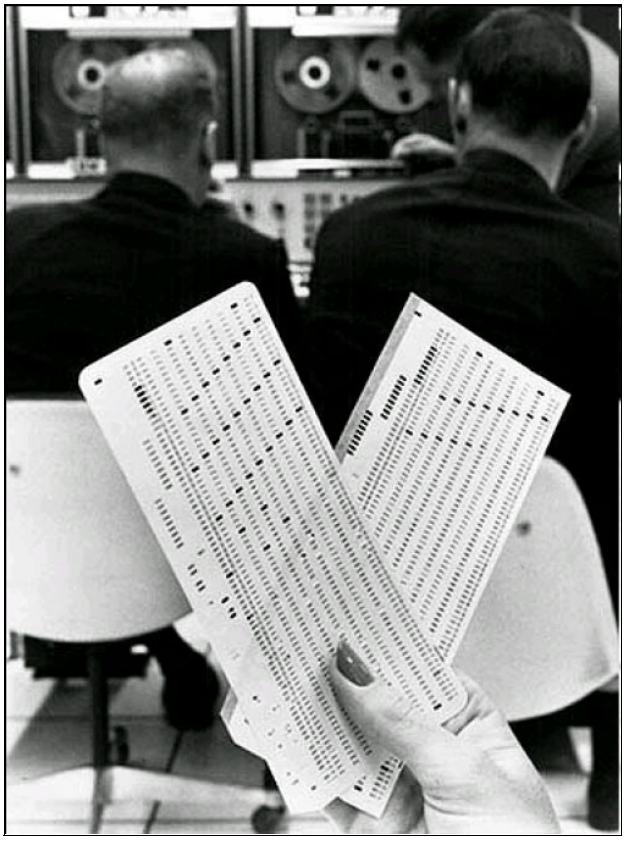


Figure 34-1 z/OS 45 years of batch

Data processing

Mainframe computers continue to play a central role in the daily operations of many of the world's largest corporations. Although other forms of computing are used extensively in various business capacities, the mainframe occupies a coveted place in the current

ebusiness environment. In banking, finance, health care, insurance, public utilities, government, and a multitude of other public and private enterprises, the mainframe computer continues to form the foundation of modern business.

For the majority of enterprises in every industry, batch processing is still a fundamental, mission-critical component. This chapter describes aspects of modern batch processing and points out why it might be necessary to change the current batch process to support modern business requirements.

34.1.1 Differences between OLTP and batch processing

Data processing on the mainframe can be grouped into two main categories:

- Online transaction processing (OLTP)
- Batch processing

OLTP is triggered by a user with a direct response. To initiate OLTP, users typically complete an entry form or select other appropriate actions through a user interface application component. The user interface component then initiates the associated online transaction, with the business logic in the background. When the transaction is complete, the same user interface or other user interface component presents the result of the transaction to the user.

The response can be data or it can be a message regarding the success or failure of the processing of the input data.

During the processing time of the online transaction, the user typically has to wait and cannot work with the user interface. Therefore, it is important to have transactions finish in the shortest time possible. Thus, the scope of online transactions and the amount of data that is processed has to be relatively small to minimize locking resources on the system.

Batch processes, in contrast, require no user activity. Most batch programs read data from various sources (for example, databases, files, and message queues), process that data, and then store the result. On the mainframe, batch programs are often designed to handle large amounts of data within a short time (such as millions of records in an elapsed time of a few minutes).

Figure 34-2 on page 750 illustrates the difference between OLTP and batch processing.

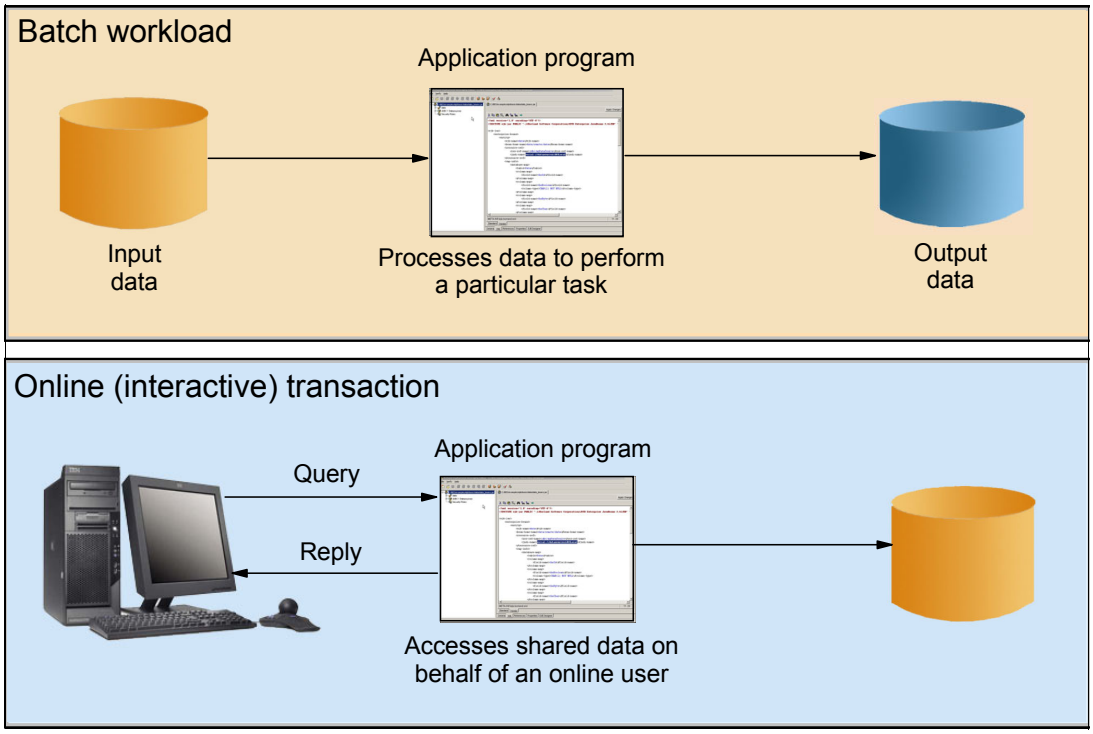


Figure 34-2 Batch and OLTP processing differences

Current batch processing

Generally, one batch program generates output data that then is sometimes used as input by another batch program. Because of this type of dependency, a “network” of batch jobs can grow and become quite complex rather quickly. Appropriate tools, such as Tivoli Workload Scheduler, help when executing and monitoring batch programs within a large job network.

In large IT environments, batch jobs usually run overnight. During this batch window, as shown in the top of Figure 34-3, online activity is restricted or even completely forbidden.

Because the online capability is extremely relevant for almost all companies, the batch window must end as soon as possible, or at least before the committed start time of the OLTP window.

The start and times of the batch window (which can be different on different days of the week or month) are specified in the service level agreement (SLA), which also includes additional information regarding management of errors, recovery, and so forth.

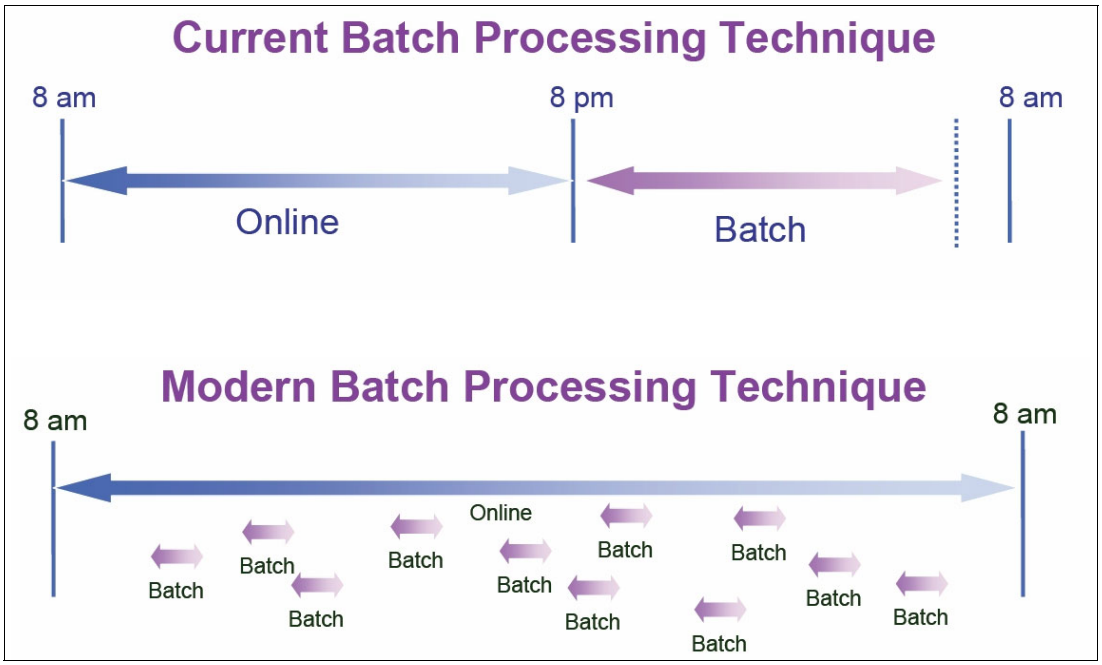


Figure 34-3 Current and future batch windows

The batch window can vary greatly, depending on the day of the week or month. For example, most companies have other job networks on weekdays, weekends, the last workday of the week, and the last day of the month. End of week and end of month processing can make a job network significantly more complex and longer to run. These factors are all included in the SLA.

Batch activities

Examples of batch activities include:

- Generating daily reports of processed data
- Paying salaries for employees
- Archiving historical data at the end of month
- Reorganizing data
- Processing files with large amounts of data from a partner

Batch jobs do not always include an application program, as such. For example, many times a utility is used to extract data, reorganize a database, create a backup copy of data, or replicate a file.

Using batch processing provides the following benefits:

- Traditionally, a batch window and an online window allow balanced system usage with almost constant 100% utilization. This helps to optimize IT resources and save costs. The mainframe is used during the non-office hours for work that does not need immediate response, thereby freeing up cycles during the office hours for transactions that need immediate response.
- Batch programs are more efficient performing repetitive logic.
- Postponing transactions can help to achieve more business security, for example intervening into money transfers that might damage the business.

As shown at the bottom of Figure 34-3, as more web-triggered transactionals are being developed, the more fined-grained batch processing is being used, changing the importance of the traditional 8 pm-to-8 am batch window.

34.2 Batch launcher and toolkit for Java applications

The IBM JZOS Batch Toolkit for z/OS SDKs is a set of tools that improves many of the functional and environmental characteristics of the current Java batch capabilities on z/OS.

The toolkit includes a native launcher for running Java applications directly as batch jobs or started tasks, and a set of Java methods that make access to traditional z/OS data and key system services directly available from Java applications. Additional system services include console communication, multiline WTO (write to operator), and return code passing capability.

In addition, JZOS provides facilities for flexible configuration of the run-time environment. It also allows intermediate data to be seen through z/OS System Display and Search Facility (SDSF). Java applications can be fully integrated as job steps to augment existing batch applications.

Batch launcher and toolkit

The combination of the launcher, data access, added system services, and environmental enhancements make running Java on z/OS as batch jobs easier, particularly for traditional z/OS programmers. The net result of these enhancements is that the “look and feel” of running Java applications is much closer to other z/OS batch jobs, and the way Java batch can be managed is now like other z/OS batch applications written in COBOL, PL/I, or other compiled languages.

The batch launcher and toolkit extends the z/OS SDK products with a set of Java classes and additional C++ code. Java applications can be launched directly as batch jobs on z/OS along with using the custom launcher. The JZOS set of Java class libraries extends the function available in the standard Java product. The extensions provide APIs for Java access to z/OS operating system services and access to z/OS-specific data types, including VSAM data.

34.2.1 z/OS V1R13 enhancements

The enhancements to the JZOS Batch Launcher and Toolkit in z/OS V1R13 are:

- Support for z/OS Java SDK 1.4.2 (31 bit), z/OS Java SDK 5.0, SDK 6.0.0, (31-bit and 64-bit) and SDK 6.0.1 (31-bit and 64-bit).
- The ability to access z/OS Workload Manager (WLM) services
- The ability to submit z/OS batch jobs from Java

34.3 Java and COBOL interoperability

As mentioned, in the distributed z/OS environments of today, installations that have to re-engineer existing native z/OS COBOL applications want to incorporate the Java language to benefit from its larger developer skill base and many language features. As well, mixed z/OS COBOL and Java batch applications must share a DB2 connection and use relevant language APIs to communicate with DB2 in the same unit of work (UOW).

Note: For complete documentation about building COBOL applications, including Object Oriented (OO) COBOL, see *Enterprise COBOL for z/OS, V4R2, Programming Guide*, SC23-8529.

For compiling with JCL, IBM provides a set of cataloged procedures to reduce the amount of JCL coding that you need to write. If the cataloged procedures do not meet your needs, you can write your own JCL. Using JCL, you can compile a single program or compile several programs as part of a batch job.

The compiler translates your COBOL program into language that the computer can process (object code). The compiler also lists errors in your source statements and provides supplementary information to help you debug and tune your program. Use compiler-directing statements and compiler options to control your compilation. After compiling your program, review the results of the compilation and correct any compiler-detected errors.

To build Java programs, use the **`javac`** command to create the classes and use the **`jar`** command for packaging. This documentation focuses on building a typical use case that updates a traditional COBOL program to call out to Java methods in which either or both can use DB2.

Additionally, as mentioned, when you run such a re-engineered program, be aware that Embedded Structured Query Language (SQL) DB2 access in Enterprise COBOL and Java Database Connectivity (JDBC), Structured Query Language for Java (SQLJ) DB2 access in Java, or both, must coexist transparently.

Note: With z/OS V1R13, updates to multiple databases are not supported.

Non-Java batch job

The non-Java batch job shown in Figure 34-4 is based on the classic non-interpretive nature of programming language such as COBOL flows:

1. The source code is compiled to produce the object module,
2. The external references are resolved by the binder in a second phase known as the link-edit.
3. This produces the load module stored into a library like the well-known partitioned data set.
4. After the initiator allocates all resources that will be required by the applications, the loader loads the load module into memory for execution.

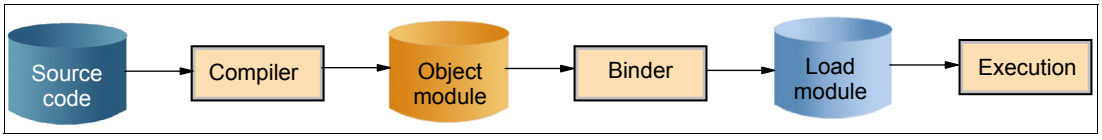


Figure 34-4 Classical batch preparation

COBOL and Java intermixed in batch

The steps of the classical non-Java batch job process are not suitable for the batch processing of an interpreted language such as Java.

z/OS Batch Runtime in z/OS V1R13 allows interoperability of COBOL applications and Java applications where required within the same unit of work (UOW), as it provides database integrity. It runs the application within the Batch Runtime environment provided with z/OS

V1R13, and preserves COBOL assets while migrating to Java applications where required. It is a program designed to provide a managed environment that enables shared access to a DB2 connection by both COBOL and Java programs. Updates to DB2 are committed in a single transaction.

Figure 34-5 shows the insertion of a Java program execution as a step into a batch job to handle the following tasks:

- To launch a special environment, called Batch Runtime, to be able to run the desired Java program
- To handle return codes which differ from those of traditional batch, as shown in the figure. This difference in handling return codes necessitates the modification provided in JES2 of z/OS V1R13, as described in “JES2 batch modernization” on page 761.

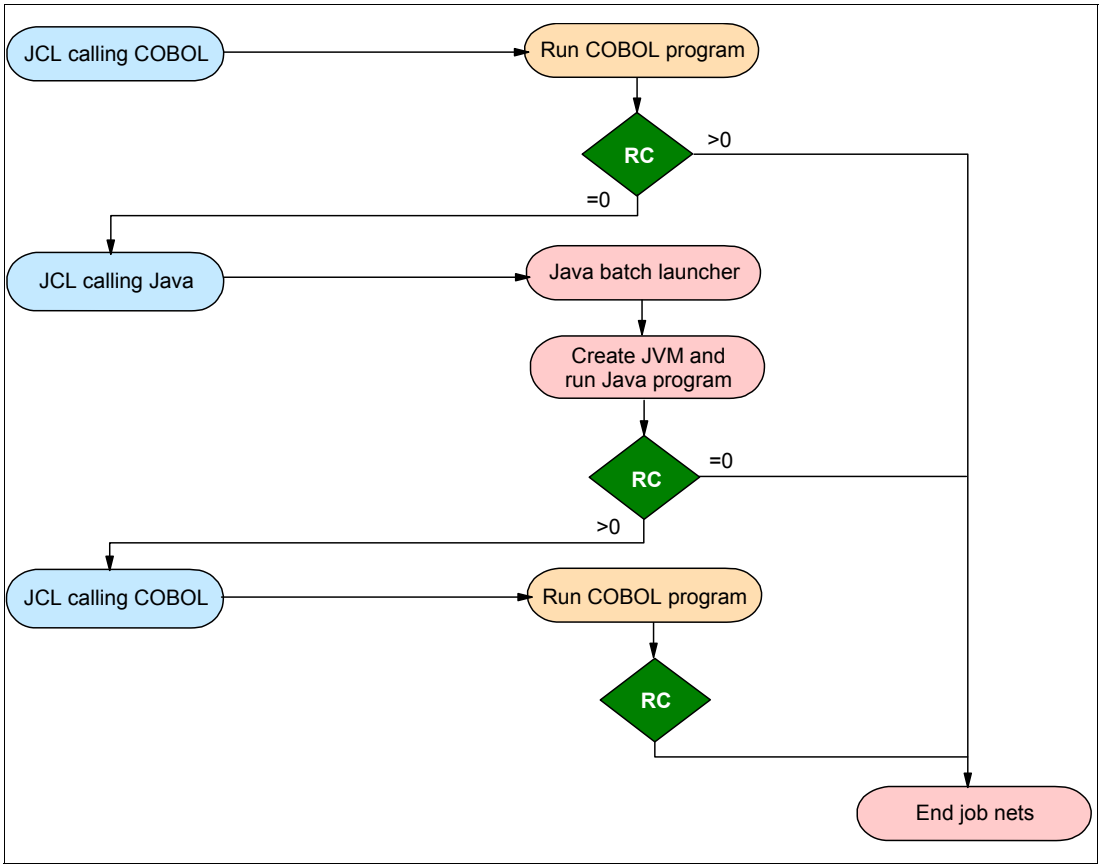


Figure 34-5 COBOL and Java intermixed in batch

The newly introduced Batch Runtime environment is started by an initiator, like any other program. It is then being dubbed into a UNIX System Services process in which a JVM is started to run the desired Java program. This entire address space is coined a JZOS address space, which is conceptually the equivalent of the overall process shown in Figure 34-6 on page 755 for non-interpreted languages such as C or COBOL.

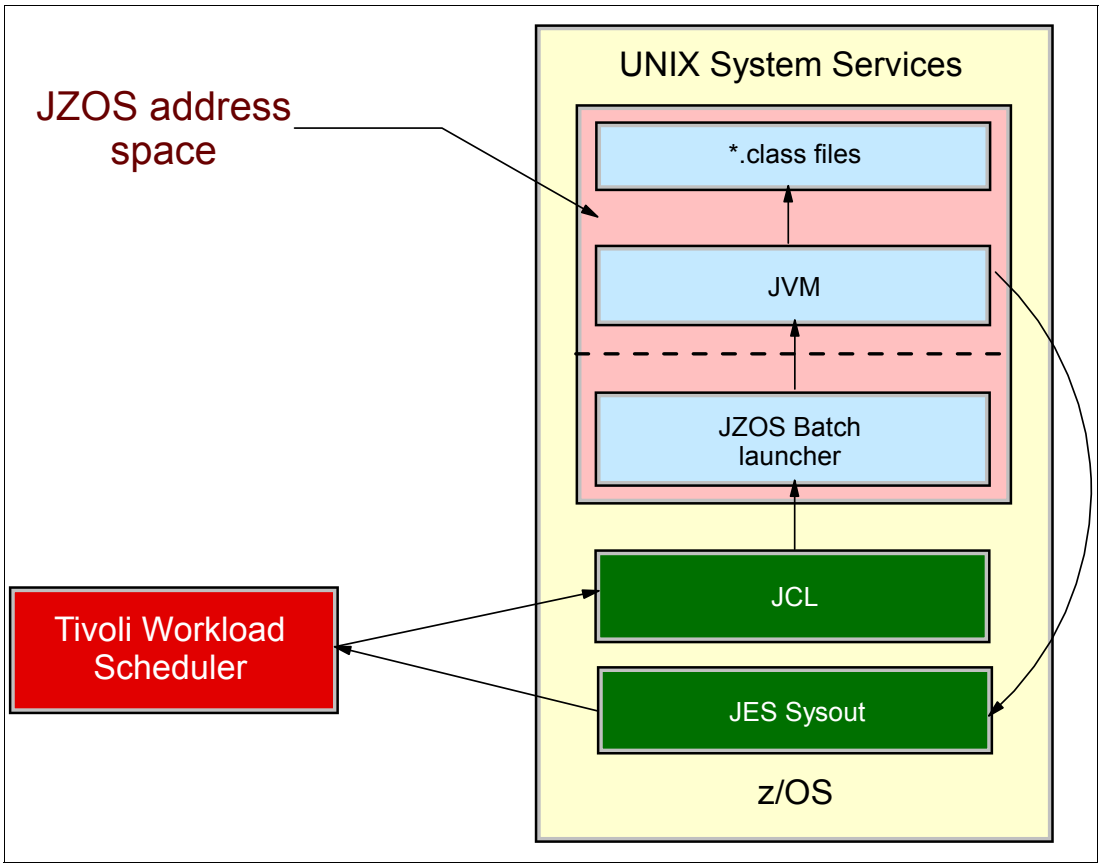


Figure 34-6 JZOS address space

The z/OS Batch Runtime environment is a new option for running batch work in z/OS V1R13. It provides a managed environment for the integration of Java and COBOL, and is consistent with IBM WebSphere-based batch.

34.3.1 Resource Recovery Services

Resource Recovery Services (RRS) provides a global sync point manager that any resource manager (RM) on z/OS can benefit from. RRS is increasingly becoming a prerequisite for new resource managers and for new capabilities in existing resource managers. Rather than having to implement their own two-phase commit protocol, these products can use the support provided by RRS.

RRS exploiters

There are many exploiters of RRS, and each has its own resource manager. With the increasing number of resource managers now available on z/OS, there was a need for a general sync point manager on z/OS that any resource manager can benefit from. This is the role of RRS, a component of z/OS. It enables transactions to update protected resources managed by many resource managers. Applications must be sure that changes to data have been made or backed out and so the applications must access this data through the RM. Figure 34-7 illustrates how RRS acts as a sync point manager in the z/OS system.

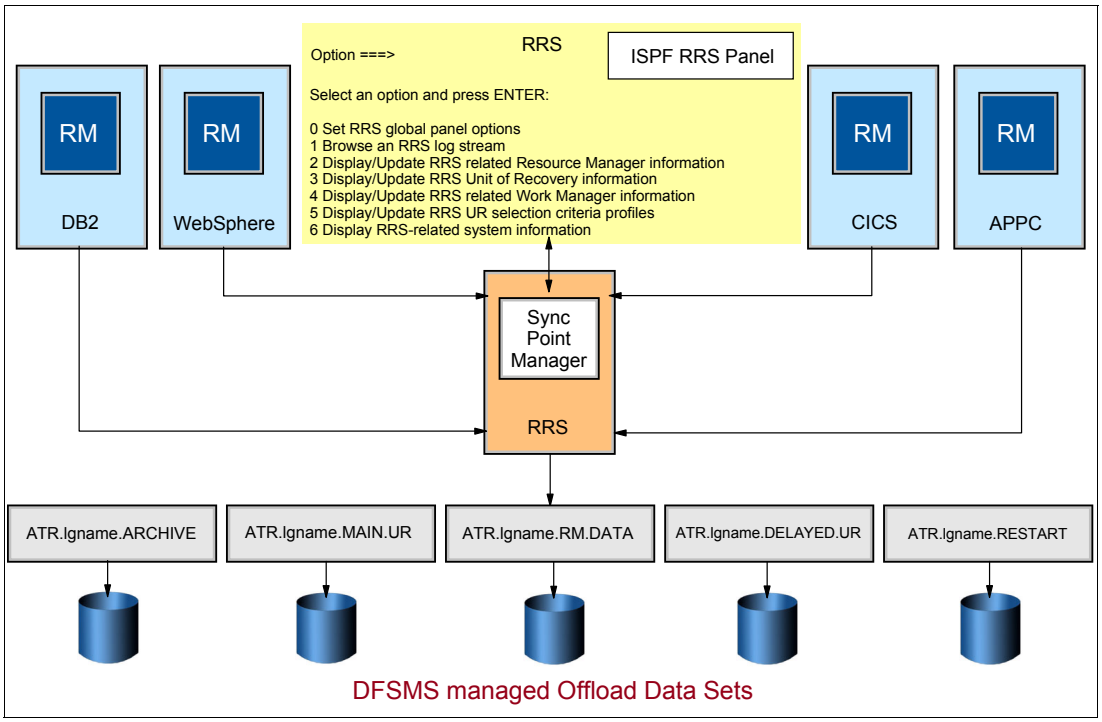


Figure 34-7 Introduction to Resource Recovery Services (RRS)

RRS functions

RRS coordinates the two-phase commit process, which is a protocol used by the RM exploiter. RRS runs in its own address space and normally is started at IPL time. However, if RRS suffers problems, it is restartable.

RRS is also responsible for the association between a unit of recovery (UR) and a unit of work (UOW). RRS builds a UR each time a resource manager wants to make a transaction. A *unit of recovery* is the base element of a transaction. A *unit of work* is a representation of a work request (transaction), and it can consist of a number of URs. RRS is an exploiter of the System Logger and it stores the logs with logger functions in the coupling facility.

The benefit to using RRS is that it ensures data integrity for all applications that exploit RRS because if a transaction fails, RRS notifies the resource managers.

RRS is a sync point manager that provides services to each resource manager, and informs the resource manager changes. An *exit manager* is an authorized program that gives control to an RM if there is an event that is to be triggered. RRS checks that each event is processed in the right sequence, and if there are problems, RRS informs the RM. This means that RRS itself does not perform any actions on a DB2 database, or back out changes to an IMS database. Instead, RRS triggers resource manager-supplied exits based on certain events.

An application can request RRS to commit a unit of recovery. RRS then invokes the commit exits for all the resource managers involved in the UR.

Over the years resource managers on z/OS have evolved to provide two-phase commit processing between them. When the application is ready to commit or back out its changes, the application invokes RRS to begin the two-phase commit protocol.

RRS supports the two-phase commit protocol as shown in Figure 3-2. The two-phase commit protocol is a set of actions used to make sure that an application program either makes all changes to the resources represented by a single unit of recovery, or makes no changes at

all. This protocol verifies that either all changes or no changes are applied even if one of the elements (such as the application, the system, or the resource manager) fails. The protocol allows for restart and recovery processing to take place after system or subsystem failure.

34.3.2 Java COBOL with DB2 interoperability

Java COBOL with DB2 interoperability support provided with z/OS V1R13 gives the ability to replace or add functions in current 3GL DB2 (for example, COBOL DB2) application inventory with new Java DB2 code. To do so, it requires local attach z/OS DB2 connection sharing for common DB2 access and UOW (Transactional) integrity among the application components.

This offers a generalized solution without requiring a specific run-time or middleware. That is, a pure batch environment and its implementation requires few or no changes to existing code; it only needs special callbacks for commit and rollback.

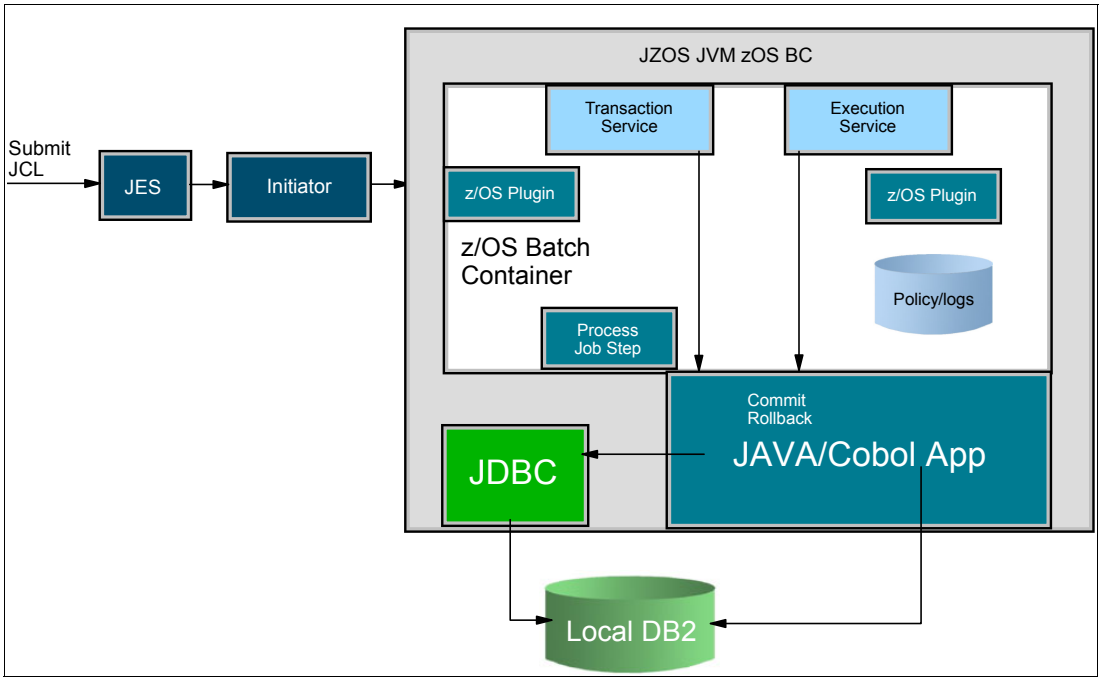


Figure 34-8 Java and COBOL with DB2 interoperability

34.3.3 Usage and invocation

Batch Runtime is invoked through batch JCL. A sample JCL procedure BCDPROC is shown in Example 34-1 on page 757.

Example 34-1 BCDPROC procedure

```
//BCDPROC PROC VERSION='61',      JVMLDM version: 61 (Java 6.0.1 31bit)
//            LOGLVL='+I',       Debug level: +I(info) +T(trc)
//            LEPARM=''          Language Environment parms
//*
//*******************************************************************
//*                                                                 *
//*  Proprietary Statement:                                         *
//*                                                                 *
//*    Licensed Materials - Property of IBM                         *
```

```
//*    5694-A01                                                  *
//*    Copyright IBM Corp. 2011.                                 *
//*                                                              *
//*    Status = HBB7780                                          *
//*                                                              *
//*  Component = z/OS Batch Runtime (SC1BC)                      *
//*                                                              *
//*  EXTERNAL CLASSIFICATION = OTHER                             *
//*  END OF EXTERNAL CLASSIFICATION:                             *
//*                                                              *
//*  Sample procedure JCL to invoke z/OS Batch Runtime           *
//*                                                              *
//*  Notes:                                                      *
//*                                                              *
//*  1. Override the VERSION symbolic parameter in your JCL      *
//*     to match the level of the Java SDK you are running.      *
//*                                                              *
//*       VERSION=61    Java SDK 6.0.1 (31 bit)                  *
//*                                                              *
//*  2. Override the LOGLVL symbolic parameter to control        *
//*     the messages issued by the jZOS Java launcher.           *
//*                                                              *
//*     Use the +T option when reporting problems to IBM or      *
//*     to diagnose problems in the STDENV script.               *
//*                                                              *
//*  3. Override the LEPARM symbolic parameter to add any        *
//*     application specific language environment options        *
//*     needed.                                                  *
//*                                                              *
//*  Change History =                                            *
//*                                                              *
//*    $L0=BATCH,HBB7780,100324,KDKJ:                            *
//*                                                              *
//*                                                              *
//**************************************************************
//JAVA EXEC PGM=JVMLDM&VERSION,REGION=0M,
//             PARM='&LEPARM/&LOGLVL'
//*
//SYSPRINT  DD  SYSOUT=*           System stdout
//SYSOUT    DD  SYSOUT=*           System stderr
//STDOUT    DD  SYSOUT=*           Java System.out
//STDERR    DD  SYSOUT=*           Java System.err
//BCDOUT    DD  SYSOUT=*           Batch container messages
//BCDTRACE  DD  SYSOUT=*           Batch container trace
//*
//CEEDUMP   DD  SYSOUT=*
//*
```

z/OS batch runtime is invoked through JCL that invokes the job BCDBATCH. BCDBATCH, in turn, invokes the JZOS launcher to initialize the Java environment.

Because BCDBATCH invokes JZOS, one level of the JZOS launcher exists for each Java SDK level and bit mode. You define the level with a symbolic, and your installation can update the symbolic as new levels of the Java SDK are added or made the default.

Table 34-1 summarizes the main JCL statements to consider for the job.

Table 34-1 JCL statements of interest for the BCDBATCH job

JCL statement	Explanation
//*STEPLIB DD DSN=hlq.yourapp.loadlib,DISP=SHR //* DSN=hlq.jzos.loadlib,DISP=SHR	Add any load libraries your application requires, such as the data set containing your COBOL application load modules to the STEPLIB. If the JZOS Java launcher is not installed in the lnklst, add a STEPLIB for it.
//STDENV DD *	This specifies the environment variables used for this run including CLASSPATH and LIBPATH.
//BCDIN DD *	This specifies a file containing the batch configuration options. Remember that various support elements obtain their options from Java system properties.

Customize the BCDBATCH job with the following:

- Application name and language type
- Application arguments (if any)
- CLASSPATH and LIBPATH needed by application

Batch runtime BCDBATCH can be used to launch a user COBOL or Java application, which itself can call another COBOL or Java application and so on in any call depth.

All RRS commit and rollback requests are handled by batch runtime:

- While applications cannot use SQL commit or rollback.
- Commit and rollback must be coordinated by batch runtime.
- Batch runtime provides a Java helper class used to invoke commit and rollback.

DB2 connection management

At startup, batch runtime calls JDBC to obtain connection to DB2:

- Subsequent COBOL SQL will share this connection.
- Java getConnection will return the same connection.

RRS calls

The batch runtime begins the initial transaction (all transactions are global). It calls RRS atrbeg.

For commit and rollback:

- Batch runtime helpers will call back to the batch runtime. Commit and rollback helpers are:
  - For Java, call method directly
  - For COBOL, use the INVOKE statement

```
com.ibm.batch.spi.UserControlledTransactionHelper.commit()
com.ibm.batch.spi.UserControlledTransactionHelper.rollback()
```

- Batch runtime will call RRS atrend (commit or rollback)
- New transaction started using atrbeg

34.3.4 Interactions and dependencies

Software dependencies are as follows:

- IBM 31-bit SDK for Java Technology Edition, V6.0.1, a z/OS launcher is required (which is distributed with SDK).
- Enterprise COBOL V4R2.
- DB2 V9 with PTFs UK62190 and UK62191.
- DB2 V10 with PTFs UK62141 and UK62145.

There are no hardware dependencies to this new functionality.

34.3.5 Installation considerations

Batch Container is implemented in Java. It has to be installed into existing path `/usr/lpp/bcp`. The BCDPROC is installed into SYS1.PROCLIB. Check for any modifications that are needed. Sample JCL BCDBATCH is installed in SYS1.SAMPLIB. It must be copied to your private JCL data set and customized as needed.

Required COBOL and Java applications must be re-instrumented to use batch runtime-provided commit and rollback helper methods.

An installation verification job is also provided.

34.3.6 Benefits of Java batch on z/OS

Starting with z/OS V1R13, running Java applications in batch provides the following benefits:

- Specific Java Batch APIs for z/OS are provided.
- MVS and VSAM data sets can be accessed.
- Condition codes are passed.
- DFSORT support is provided.
- Java program can write to logstreams.
- Triggering of jobs from Java is opened up.
- RACF APIs are made available for any security need.
- High throughput to local DB2 databases is provided.
- Access to many Java skills available in IT is provided.
- Effective and efficient development tools are available.
- IBM Rational® tools are available specific to Batch Container.
- Many classes, libraries, frameworks, and applications based on open source are available.
- Interoperability is offered with other programming languages on z/OS, such as C and COBOL.

34.4 JES2 batch modernization

Various enhancements have been required to fully support Java applications in batch environments.

34.4.1 JCL instream data sets

So far, there has been no support in JCL to allow instream data sets in proclibs or INCLUDEs. To provide batch modernization for Java:

- Support is added to allow instream data sets to be created when processing DD DATA or DD * JCL within proclibs or INCLUDEs.
- Support is added for both cataloged and instream procedures.

```
//ABC     JOB
//INCLUDE MEMBER=HELLO
//STEPA EXEC PGM=IEBGENER
//SYSIN DD DUMMY
//SYSPRINT DD SYSOUT=A
//SYSUT2 DD SYSOUT=A
//SYSUT1 DD DATA
HELLO WORLD
/*
```

Figure 34-9 Example of possible new procedure

This support is added to allow instream data in JCL proclibs and INCLUDE sections:

- It works like instream data in a normal JCL stream.
- It does not support generating //SYSIN DD *.

This support is based on where the job converts (z/OS V1R13). It can run on down-level systems.

New SYSIN data sets are included in the extended status DSLIST function. They are *not* included the spool data set browse of JCLIN (they are not part of the original JCL submitted).

The support works for batch jobs and started tasks. In z/OS V1R13, the support provides the ability for conceptual control data sets (where the application parameters reside) to not be separated from proclibs or INCLUDEs.

34.4.2 Support for JOBRC (job return code)

The highest completion code of a job is not necessarily the most meaningful. To provide you with a more useful way to take into account the most meaningful completion code, a new JCL keyword is added to control the reported completion code. This new JCL keyword allows you to select the last step, a specific step, or the highest step completion code.

A new JOBRC keyword on the JOB card is introduced. The possible values for JOBRC keyword are:

- MAXRC – Default, job return code is the maximum of any step.
- LASTRC – The job return code is the return code of the last step.

- (STEP,stepname.procstepname) – The job return code is the indicated step if it executes. Otherwise, it is the same as MAXRC.

Similarly, the $T JOBCLASS(x) command is enhanced with a new JOBRC operand:

- JOBRC=MAXRC | LASTRC
- The JOBRC keyword on the job card takes precedence.

Note the following related enhancements:

- Extended status STTRMXRC and STVBMXRC (traditional) fields
  - These now include two new conditions regarding how a job can end.
- Converter error – The converter returned a bad return code and failed the job.
- System failure – The job was executing at the time of a system failure and was not re-queued for execution.
- Old maximum return code in verbose job STVBMXRC field
- $DJQ,CC= value
- ENF 70 field ENF70_MAXCC
- A new step end SSI is used to extract the step return code.
- New JES2 exit 58 is called in step end SSI.
  - This can be used to influence the step return code.
- There is updated HASP165 message text:
  - `Jobname ENDED AT node reason`
  - Examples of reason:
    - MAXCC=code - JOBRC was not specified.
    - Code is now always 4 digits (MAXCC=0000).
    - JOBRC=code - JOBRC was specified and affected the return code.
    - MAXRC=code - JOBRC was specified but MAXRC was returned.
    - ABENDED Sxxx,Uyyy.
    - ABENDED abend_code, JOBRC=code.
    - JOBRC=(STEP,stepname), step executed, but later step ABENDed.

Usage and invocation

Note the use and invocation with this new option in z/OS V1R13:

- If JOBRC=LASTRC is specified, this specification indicates to use the return code of the last executed step as the completion code for the job.
- If JOBRC=(STEP,C.HLASM) is specified, the return code for the C step in the HLASM procstepname is used as the completion code for the job.

Benefit and value

By using JOBRC, you can better determine the success of a job without having to examine the job output.

34.4.3 Evict job on a step boundary

With z/OS V1R13 batch modernization, a more orderly method of getting jobs out of execution was needed. A new operand has been implemented on the $E J command that forces a job out of execution when the current step ends. The job resumes execution from next step. The following new JES2 command option is provided:

- $EJxxxx,STEP[,HOLD]

This option deals with cross-member requests in the following way:

- It forces a job out of execution when the current step ends.
- The job is requeued for execution (and held, if requested).
- The option utilizes existing restart logic (continue restart) to perform the function.
  - It requires JES journal to be active (JOBCLASS(x) JOURNAL=YES).
- It uses the new step end SSI to communicate with the initiator to requeue the job:
  - New JES2 exit 58 called in step end SSI.
  - This can be used to inhibit or to trigger the function.

Benefit and value

This provides more orderly shutdowns of systems, to enhance batch processing.

A

Setting up WebSphere OEM, z/OSMF, CIM, and Capacity Provisioning

This appendix describes the installation and setup of the following products:

- WebSphere Application Server OEM V7R0
- z/OSMF V1R13
- CIM
- Capacity Provisioning

These products are prerequisites for setting up z/OSMF V1R13, which is also described in this appendix.

A.1 Setting up WebSphere Application Server OEM

This section describes setting up WebSphere Application Server OEM V7RO after it has been received, applied, and accepted by SMP/E.

The steps to complete this setup are listed here:

1. Run the script `WASOEM-config -mode typical`.
2. Run four batch jobs to set up RACF.
3. Run the script `WASOEM.sh -create CONFIG1`, shown in Example A-2.
4. START BB7ACR, shown in Example A-3.

Example A-1 `WASOEM.sh -config -mode typical`

```
LAFITTE @ SC74:/SC74/etc>/usr/lpp/zWebSphereOEM/V7RO/bin/WASOEM.sh -config -mode
typical
/usr/lpp/zWebSphereOEM/V7RO/bin/WASOEM.sh -config -mode typical
+ /usr/lpp/zWebSphereOEM/V7RO/bin/WASOEM.sh -config -mode typical
BBNO400I:The current top-level WebSphere Application Server OEM configuration wo
rking directory is set to:
/etc

BBNO400I:The WebSphere Application Server OEM configuration working directories
will be located under:
/etc

BBNO400I:This location setting will be valid for this session only.

BBNO235I:Creating configuration directory /etc/zWebSphereOEM/V7RO/conf
.........................................................
the next 17 free ports from the provided base

Here are the port values currently defined for your configuration

 Location Service Daemon port ............ (32200)
 Location Service Daemon SSL port ........ (32201)
 JMX SOAP connector port ................. (32202)
 ORB port ................................ (32203)
 ORB SSL port ............................ (32204)
 Administrative console port ............. (32205)
 Administrative console secure port ...... (32206)
 HTTP transport port ..................... (32207)
 HTTPS transport port .................... (32208)
 Administrative interprocess communication port ....... (32209)
 High Availability Manager Communications port ........ (32210)
 Service Integration port ............................. (32211)
 Service Integration Secure port ...................... (32212)
 Service Integration MQ Interoperability port ......... (32213)
 Service Integration MQ Interoperability Secure port .. (32214)
 Session Initiation Protocol (SIP) port ............... (32215)
 Session Initiation Protocol (SIP) secure port ........ (32216)


Simply press return to accept the values from your response file.
Or, enter a base port to automatically assign the 17 ports. For example, if you
```

```
enter 32200, 32200 and the next 16 free ports will be assigned. If 32200 is in u
se then it will start at the next free port.

osgi> Customization definition successfully written to /etc/zWebSphereOEM/V7R0/c
onf/CONFIG1/zpmt
Copying CNTL files to BBN.V7R0.CONFIG1.ZPMTJOBS.CNTL...
Copy successful.
Copying DATA files to BBN.V7R0.CONFIG1.ZPMTJOBS.DATA...
Copy successful.
Success: customization jobs have been created successfully.
Submit the following jobs before running WASOEM.sh -create
First, submit BBN.V7R0.CONFIG1.ZPMTJOBS.CNTL(BBOSBRAK) - Make sure that you sele
ct BBOSBRAK.

After BBN.V7R0.CONFIG1.ZPMTJOBS.CNTL(BBOSBRAK) completes, submit the following.
     BBN.V7R0.CONFIG1.ZPMTJOBS.CNTL(BBOSBRAM)
     BBN.V7R0.CONFIG1.ZPMTJOBS.CNTL(BBOCBRAK)

Run /Z1DRB1/usr/lpp/zWebSphereOEM/V7R0/bin/WASOEM.sh -create /etc/zWebSphereOEM/
V7R0/conf/CONFIG1

WASOEM.sh has completed
Check log file /tmp/WASOEM_042011_092407.log for more information
LAFITTE @ SC74:/Z1DRB1/usr/lpp/zWebSphereOEM/V7R0/bin>
```

Next, three batch jobs must be submitted from `BBN.V7R0.CONFIG1.ZPMTJOBS.CNTL`. They mostly run RACF commands under the cover of a TSO TMP.

After all of the batch jobs complete with a RC=0000, the script shown in Example A-2 must be issued and run.

Example A-2 WASOEM.sh -create CONFIG1

```
LAFITTE @ SC74:/SC74/etc>/usr/lpp/zWebSphereOEM/V7R0/bin/WASOEM.sh -create CONFIG1
/usr/lpp/zWebSphereOEM/V7R0/bin/WASOEM.sh -create CONFIG1
+ /usr/lpp/zWebSphereOEM/V7R0/bin/WASOEM.sh -create CONFIG1
.............................
BBN0025I:Reading in response file /etc/zWebSphereOEM/V7R0/conf/CONFIG1/zpmt/resp
BBN0123I:Create WASOEM Configuration HFS
BBN0125I:Mount point /zWebSphereOEM/V7R0/config1 already exists.
issued tso mount filesystem('BBN.V7R0.CONFIG1.ZFS') TYPE(ZFS) MODE(RDWR) MOUNTPO
BPXF135E RETURN CODE 00000081, REASON CODE EF096055.  THE MOUNT FAILED FOR FILE
BBN0011W:Could not mount BBN.V7R0.CONFIG1.ZFS at /zWebSphereOEM/V7R0/config1.. A
BBN0008I:Setting mode of /zWebSphereOEM/V7R0/config1 to 775
BBN0127I:BBN.V7R0.CONFIG1.ZFS is not mounted at /zWebSphereOEM/V7R0/config1. It
-n BBN0128I:Has the configuration file system BBN.V7R0.CONFIG1.ZFS on volume BH5
BBN0229I:   * Your response was ÝN¨
BBN0130I:Invoking /Z1DRB1/usr/lpp/zWebSphereOEM/V7R0/bin/createWASOEMHFS.sh  -pa
BBN0007I:Wait ...
BBN0008I:Setting mode of /zWebSphereOEM/V7R0/config1 to 775
BBN3015I: Copying WebSphere procs into proclib data set ZOSMF113.V7R0.CONFIG1.PR

/etc/zWebSphereOEM/V7R0/conf/CONFIG1/zpmt/cntl/bbopasr -> ZOSMF113.V7R0.CONFIG1.
/etc/zWebSphereOEM/V7R0/conf/CONFIG1/zpmt/cntl/bbopacr -> ZOSMF113.V7R0.CONFIG1.
/etc/zWebSphereOEM/V7R0/conf/CONFIG1/zpmt/cntl/bbopdmn -> ZOSMF113.V7R0.CONFIG1.
/etc/zWebSphereOEM/V7R0/conf/CONFIG1/zpmt/cntl/bbopcra -> ZOSMF113.V7R0.CONFIG1.
```

```
/etc/zWebSphereOEM/V7R0/conf/CONFIG1/zpmt/cntl/bbowpsh -> ZOSMF113.V7R0.CONFIG1.
BBN3O16I: Finished copying WebSphere procs into proclib data set ZOSMF113.V7R0.C
BBN3O17I: Allocating configuration file system data set BBN.V7R0.CONFIG1.ZFS.
IOEZ00248I VSAM linear dataset BBN.V7R0.CONFIG1.ZFS successfully created.
IOEZ00077I HFS-compatibility aggregate BBN.V7R0.CONFIG1.ZFS has been successfull
BBN3O18I: Finished allocating configuration file system data set BBN.V7R0.CONFIG
BBN3O19I: Mounting data set BBN.V7R0.CONFIG1.ZFS at mount point /zWebSphereOEM/V
mount filesystem('BBN.V7R0.CONFIG1.ZFS') TYPE(ZFS) MODE(RDWR) MOUNTPOINT('/zWebS
BBN3O20I: Finished mounting data set BBN.V7R0.CONFIG1.ZFS at mount point /zWebSp
BBN3O21I: Invoking symlink script.
SMPEROOT = /usr/lpp/zWebSphereOEM/V7R0
WASHOME = /zWebSphereOEM/V7R0/config1/AppServer
OWNER = WOEMADM
GROUP = WSCFG1

Creating directory: /zWebSphereOEM/V7R0/config1/AppServer...
Creating directory: /zWebSphereOEM/V7R0/config1/AppServer/temp...
Creating directory: /zWebSphereOEM/V7R0/config1/AppServer/IBM...
Linking files in /zWebSphereOEM/V7R0/config1/AppServer/IBM to /usr/lpp/zWebSpher
Creating directory: /zWebSphereOEM/V7R0/config1/AppServer/MQSeries...
Linking files in /zWebSphereOEM/V7R0/config1/AppServer/MQSeries to /usr/lpp/zWeb
Creating directory: /zWebSphereOEM/V7R0/config1/AppServer/Scheduler...
Linking files in /zWebSphereOEM/V7R0/config1/AppServer/Scheduler to /usr/lpp/zWe
Creating directory: /zWebSphereOEM/V7R0/config1/AppServer/UDDIReg...
Linking files in /zWebSphereOEM/V7R0/config1/AppServer/UDDIReg to /usr/lpp/zWebS
Creating directory: /zWebSphereOEM/V7R0/config1/AppServer/bin...
..............................
Linking files in /zWebSphereOEM/V7R0/config1/AppServer/web to /usr/lpp/zWebSpher
Creating directory: /zWebSphereOEM/V7R0/config1/AppServer/zOS-config...
Linking files in /zWebSphereOEM/V7R0/config1/AppServer/zOS-config to /usr/lpp/zW
Configuring java directory...
Creating directory: /zWebSphereOEM/V7R0/config1/AppServer/java...
Creating directory: /zWebSphereOEM/V7R0/config1/AppServer/java/cache...
Linking /zWebSphereOEM/V7R0/config1/AppServer/java/HelloWorld.class to /usr/lpp/
Linking /zWebSphereOEM/V7R0/config1/AppServer/java/J6.0 to /usr/lpp/zWebSphereOE
Linking /zWebSphereOEM/V7R0/config1/AppServer/java/JDK_INSTALL_OK to /usr/lpp/zW
Linking /zWebSphereOEM/V7R0/config1/AppServer/java/bin to /usr/lpp/zWebSphereOEM
..............................
Linking /zWebSphereOEM/V7R0/config1/AppServer/java/src.zip to /usr/lpp/zWebSpher
Linking /zWebSphereOEM/V7R0/config1/AppServer/java/standard to /usr/lpp/zWebSphe
Configuring java64 directory...
Creating directory: /zWebSphereOEM/V7R0/config1/AppServer/java64/cache...
Linking /zWebSphereOEM/V7R0/config1/AppServer/java64/HelloWorld.class to /usr/lp
Linking /zWebSphereOEM/V7R0/config1/AppServer/java64/J6.0_64 to /usr/lpp/zWebSph
..............................
linking files in /zWebSphereOEM/V7R0/config1/AppServer/properties to /usr/lpp/zW
Setting up /zWebSphereOEM/V7R0/config1/AppServer/properties/service directory fo
linking: /zWebSphereOEM/V7R0/config1/AppServer/properties/service/postinstaller
linking: /zWebSphereOEM/V7R0/config1/AppServer/properties/service/product/WebSph
chmod -Rh 775 /zWebSphereOEM/V7R0/config1/AppServer
chown -Rh WOEMADM /zWebSphereOEM/V7R0/config1/AppServer
chgrp -Rh WSCFG1 /zWebSphereOEM/V7R0/config1/AppServer
rc=0
BBN3O22I: Invoking post installer setup script.
/usr/lpp/zWebSphereOEM/V7R0/bin/bbowpost.sh - BEGIN - Wed Apr 20 14:34:01 EDT 20
```

```
/usr/lpp/zWebSphereOEM/V7R0/bin/bbowpost.sh - END - Wed Apr 20 14:34:01 EDT 2011
BBN3023I: Invoking profile creation wizard.
INSTCONFSUCCESS: Success: Profile default now exists. Please consult /zWebSphere
BBN0239I:Creating a symbolic link to /zWebSphereOEM/V7R0/config1/AppServer/prope
BBN0131I:Success: Profile has been successfully created.
BBN0015I:Updating configuration.
BBN0007I:Wait ...
WASX7357I: By request, this scripting client is not connected to any server proc
*sys-package-mgr*: processing new jar, '/Z1DRB1/usr/lpp/zWebSphereOEM/V7R0/lib/s
*sys-package-mgr*: processing new jar, '/Z1DRB1/usr/lpp/zWebSphereOEM/V7R0/lib/b
*sys-package-mgr*: processing new jar, '/Z1DRB1/usr/lpp/zWebSphereOEM/V7R0/lib/l
*sys-package-mgr*: processing new jar, '/Z1DRB1/usr/lpp/zWebSphereOEM/V7R0/lib/u
*sys-package-mgr*: processing new jar, '/Z1DRB1/usr/lpp/zWebSphereOEM/V7R0/java/
*sys-package-mgr*: processing new jar, '/Z1DRB1/usr/lpp/zWebSphereOEM/V7R0/lib/b
*sys-package-mgr*: processing new jar, '/Z1DRB1/usr/lpp/zWebSphereOEM/V7R0/optio
*sys-package-mgr*: processing new jar, '/Z1DRB1/usr/lpp/zWebSphereOEM/V7R0/plugi
*sys-package-mgr*: processing new jar, '/Z1DRB1/usr/lpp/zWebSphereOEM/V7R0/plugi
.................................
*sys-package-mgr*: processing new jar, '/Z1DRB1/usr/lpp/zWebSphereOEM/V7R0/lib/w
*sys-package-mgr*: processing new jar, '/Z1DRB1/usr/lpp/zWebSphereOEM/V7R0/lib/z
WASX7303I: The following options are passed to the scripting environment and are
BBN2000I:updateConfigWASOEM.py is starting.
BBN2001I:Set garbage collection policy to gencon.
BBN2002I:Set pass by reference to noLocalCopies.
BBN2003I:Disabling PMI.
BBN2004I:Set startupTraceSpecification to warning.
BBN2005I:Enabling Application Security.
BBN2006I:Exiting updateConfigWASOEM.py
BBN0370I: ownership of files...updating
BBN0016I:Success: Update of configuration completed.
The following ports have been set, ensure that they are added to the reserved po
 zDaemonPort                          32200
zDaemonSslPort                       32201
 zSoapPort                            32202
 zOrbListenerPort                     32203
 zOrbListenerSslPort                  32204
 zAdminConsolePort                    32205
 zAdminConsoleSecurePort              32206
 zHttpTransportPort                   32207
 zHttpTransportSslPort                32208
 zAdminLocalPort                      32209
 zHighAvailManagerPort                32210
 zServiceIntegrationPort              32211
 zServiceIntegrationSecurePort        32212
 zServiceIntegrationMqPort            32213
 zServiceIntegrationSecureMqPort      32214
 zSessionInitiationPort               32215
 zSessionInitiationSecurePort         32216
BBN0152I:To start the application server, issue the MVS command:
BBN0153I:START BBN7ACR,JOBNAME=BBNS001,ENV=BBNBASE.BBNNODE.BBNS001
BBN0154I:To stop the application server, enter the MVS command:
BBN0153I:START BBN7ACR,JOBNAME=BBNS001,ENV=BBNBASE.BBNNODE.BBNS001
BBN0154I:To stop the application server, enter the MVS command:
BBN0155I:STOP BBN7ACRS
```

```
BBNO231I:The administrative console for your server can be accessed at
http://WTSC74.ITSO.IBM.COM:32205/ibm/console using user ID WOEMADM
BBNO237I:A password needs to be assigned to WOEMADM before it can be used.
BBNO148I:WASOEM.sh has completed
```

After the WebSphere Application Server OEM setup is complete, you can start WebSphere Application Server itself, as partially shown in Example A-3.

Example A-3 START BBN7ACR

```
S BBN7ACR,JOBNAME=BBNSO01,ENV=BBNBASE.BBNNODE.BBNSO01
IRR812I PROFILE BBN7ACR.* (G) IN THE STARTED CLASS WAS USED 475
        TO START BBN7ACR WITH JOBNAME BBNSO01.
$HASP100 BBNSO01  ON STCINRDR
IEF695I START BBN7ACR  WITH JOBNAME BBNSO01  IS ASSIGNED TO USER
WSCRU1  , GROUP WSCFG1
$HASP373 BBNSO01  STARTED
BBOOOOO1I WEBSPHERE FOR Z/OS CONTROL PROCESS 479
BBNBASE/BBNNODE/BBNCO01/BBNSO01 IS STARTING.
BBOOO238I WEBSPHERE FOR Z/OS CONTROL PROCESS bbnbase/bbnnode/server1
480
IS STARTING.
BBOOO309I CONTROL PROCESS BBNBASE/BBNNODE/BBNCO01/BBNSO01 IS
EXECUTING 481
IN 64-BIT ADDRESSING MODE.
BBOMO007I CURRENT CB SERVICE LEVEL IS build level 7.0.0.15 (AM35611)
482
release WAS70.ZNATV date 03/30/11 20:08:05.
BBOMO001I        adapter_max_conn: NOT SET, DEFAULT=100.
BBOMO001I  adapter_max_otma_names: NOT SET, DEFAULT=16.
BBOMO001I        adapter_max_serv: NOT SET, DEFAULT=100.
BBOMO001I      adapter_max_shrmem: NOT SET, DEFAULT=33554432.
BBOMO001I adjunct_jvm_direct_options: NOT SET.
BBOMO001I adjunct_region_dynapplenv_jclparms: 488
AMODE=64,ENV=BBNBASE.BBNNODE.&IWMSSNM.,JOBNAME=&IWMSSNM.A.
BBOMO001I adjunct_region_dynapplenv_jclproc: BBN7CRA.
BBOMO001I adjunct_region_jvm_properties_file: 490
/zWebSphereOEM/V7RO/config1/AppServer/profiles/default/config/cells/bb
nbase/nodes/bbnnode/servers/server1/adjunct.jvm.options.
BBOMO001I adjunct_region_server_configured_to_bus: NOT SET, DEFAULT=0.
BBOMO001I adjunct_region_start_default_adjunct: 0.
BBOMO001I  allow_large_SAF_groups: NOT SET, DEFAULT=0.
BBOMO001I               cell_name: bbnbase.
BBOMO001I         cell_short_name: BBNBASE.
BBOMO001I client_ras_logstreamname: NOT SET.
BBOMO001I com_ibm_userRegistries_type: security:LocalOSUserRegistry.
BBOMO001I com_ibm_ws_logging_zos_errorlog_format_cbe: NOT SET, 498
DEFAULT=0.
BBOMO001I com_ibm_CSI_claimTransportAssocSSLTLSRequired: 0.
BBOMO001I com_ibm_CSI_claimTransportAssocSSLTLSSupported: 1.
BBOMO001I com_ibm_DAEMON_claim_ssl_sys_v3_timeout: NOT SET, 501
DEFAULT=600.
BBOMO001I com_ibm_DAEMON_claimKeyringName: NOT SET.
BBOMO001I com_ibm_DAEMON_claimSecurityCipherSuiteList: NOT SET.
BBOMO001I com_ibm_Server_Security_Enabled: 1.
BBOMO001I             config_root: NOT SET.
```

```
BBOM0001I control_region_classpath: 506
/zWebSphereOEM/V7R0/config1/AppServer/profiles/default/properties:/zWe
bSphereOEM/V7R0/config1/AppServer/properties:/zWebSphereOEM/V7R0/confi
g1/AppServer/lib/bootstrap.jar:/zWebSphereOEM/V7R0/config1/AppServer/l
ib/bootstrapws390.jar:/zWebSphereOEM/V7R0/config1/AppServer/lib/lmprox
y.jar:/zWebSphereOEM/V7R0/config1/AppServer/lib/startup.jar:/zWebSpher
eOEM/V7R0/config1/AppServer/java64/lib/tools.jar.
BBOM0001I control_region_confirm_recovery_on_no_srs: NOT SET, 507
DEFAULT=0.
BBOM0001I control_region_dreg_on_no_srs: NOT SET, DEFAULT=0.
BBOM0001I control_region_http_queue_timeout_percent: NOT SET, 509
DEFAULT=99.
BBOM0001I control_region_https_queue_timeout_percent: NOT SET, 510
DEFAULT=99.
BBOM0001I control_region_iiop_queue_timeout_percent: NOT SET, 511
DEFAULT=99.
BBOM0001I control_region_jms_request_timeout: NOT SET, DEFAULT=60.
BBOM0001I control_region_jvm_localrefs: NOT SET, DEFAULT=128.
BBOM0001I control_region_jvm_logfile: NOT SET.
BBOM0001I control_region_jvm_properties_file: 515
/zWebSphereOEM/V7R0/config1/AppServer/profiles/default/config/cells/bb
nbase/nodes/bbnnode/servers/server1/control.jvm.options.
 BBOM0001I  control_region_libpath: 516
 /zWebSphereOEM/V7R0/config1/AppServer/java64/bin:/zWebSphereOEM/V7R0/c
 onfig1/AppServer/java64/bin/j9vm:/zWebSphereOEM/V7R0/config1/AppServer
 /java64/lib/s390/j9vm:/zWebSphereOEM/V7R0/config1/AppServer/java64/lib
 /s390:/zWebSphereOEM/V7R0/config1/AppServer/java64/lib/s390x/j9vm:/zWe
 bSphereOEM/V7R0/config1/AppServer/java64/lib/s390x:/zWebSphereOEM/V7R0
 /config1/AppServer/lib:/zWebSphereOEM/V7R0/config1/AppServer/java64/jr
 e/bin:/zWebSphereOEM/V7R0/config1/AppServer/java64/jre/lib/s390/j9vm:/
 zWebSphereOEM/V7R0/config1/AppServer/java64/jre/lib/s390x/j9vm:/zWebSp
 hereOEM/V7R0/config1/AppServer/java64/jre/lib/s390x.
 BBOM0001I control_region_mdb_queue_timeout_percent: NOT SET, 517
 DEFAULT=99.
 BBOM0001I control_region_mdb_request_timeout: NOT SET, DEFAULT=120.
 BBOM0001I control_region_sip_queue_timeout_percent: NOT SET, 519
 DEFAULT=99.
 BBOM0001I control_region_sips_queue_timeout_percent: NOT SET, 520
 DEFAULT=99.
 BBOM0001I control_region_start_cmd: NOT SET.
 BBOM0001I control_region_thread_pool_size: 25.
 BBOM0001I control_region_thread_stack_size: NOT SET, DEFAULT=0.
 BBOM0001I control_region_timeout_delay: NOT SET, DEFAULT=0.
 BBOM0001I control_region_timeout_dump_action: NOT SET.
 BBOM0001I control_region_timeout_dump_action_session: NOT SET.
BBOM0001I control_region_timeout_save_last_servant: NOT SET, 527
DEFAULT=0.
BBOM0001I control_region_use_java_g: NOT SET, DEFAULT=0.
BBOM0001I control_region_wlm_dispatch_timeout: 1200.
BBOM0001I controller_jvm_direct_options: NOT SET.
BBOM0001I       daemon_group_name: BBNBASE.
BBOM0001I    daemon_start_command: START BBN7DMNB.
BBOM0001I daemon_start_command_args: 533
JOBNAME=BBN7ACRS,ENV=BBNBASE.BBNBASE.BBN7ACRS,REUSASID=YES.
BBOM0001I          daemon_wlmable: NOT SET, DEFAULT=0.
```

```
BBOM0001I      daemonInstanceName: BBN7ACRS.
BBOM0001I              daemonName: BBNBASE.
BBOM0001I default_internal_work_transaction_class: NOT SET.
BBOM0001I          enable_adapter: NOT SET, DEFAULT=0.
BBOM0001I    iiop_max_msg_megsize: NOT SET, DEFAULT=0.
BBOM0001I iiop_max_send_queue_megsize: NOT SET, DEFAULT=0.
BBOM0001I local_comm_max_msg_megsize: NOT SET, DEFAULT=0.
BBOM0001I         multi_cell_mode: NOT SET, DEFAULT=0.
BBOM0001I            nls_language: NOT SET, DEFAULT=ENUS.
BBOM0001I               node_name: bbnnode.
BBOM0001I         node_short_name: BBNNODE.
BBOM0001I odr_tclass_propogation_enabled: NOT SET, DEFAULT=1.
BBOM0001I ola_cicsuser_identity_propagate: NOT SET, DEFAULT=0.
BBOM0001I ola_outbound_transaction_inactivity_timeout: NOT SET, 548
DEFAULT=120.
BBOM0001I ola_trace_settings_file: NOT SET.
BBOM0001I protocol_accept_http_work_after_min_srs: NOT SET, DEFAULT=1.
BBOM0001I protocol_accept_iiop_work_after_min_srs: NOT SET, DEFAULT=0.
BBOM0001I protocol_bboc_log_response_failure: NOT SET, DEFAULT=0.
BBOM0001I protocol_bboc_log_return_exception: NOT SET, DEFAULT=0.
BBOM0001I protocol_giop_level_highest: NOT SET, DEFAULT=1.2.
BBOM0001I protocol_http_large_data_inbound_buffer: NOT SET, DEFAULT=0.
BBOM0001I protocol_http_large_data_inbound_buffer_64bit: NOT SET, 556
DEFAULT=0.
BBOM0001I protocol_http_large_data_response_buffer: NOT SET, 557
DEFAULT=104857600.
BBOM0001I protocol_http_resolve_foreign_hostname: NOT SET, DEFAULT=0.
BBOM0001I protocol_http_timeout_output: NOT SET, DEFAULT=300.
BBOM0001I protocol_http_timeout_output_recovery: NOT SET, 560
DEFAULT=SERVANT.
BBOM0001I protocol_http_transactionClass: NOT SET.
BBOM0001I protocol_http_transport_class_mapping_file: NOT SET.
BBOM0001I protocol_https_cert_mapping_file: NOT SET.
BBOM0001I protocol_https_default_cert_label: NOT SET.
BBOM0001I protocol_https_mutual_auth_cbind_check: NOT SET, DEFAULT=0.
BBOM0001I protocol_https_timeout_output: NOT SET, DEFAULT=300.
BBOM0001I protocol_https_timeout_output_recovery: NOT SET, 567
DEFAULT=SERVANT.
BBOM0001I protocol_https_transactionClass: NOT SET.
BBOM0001I   protocol_iiop_backlog: NOT SET, DEFAULT=10.
BBOM0001I protocol_iiop_backlog_ssl: NOT SET, DEFAULT=10.
BBOM0001I protocol_iiop_daemon_listenIPAddress: WTSC74.ITSO.IBM.COM.
BBOM0001I protocol_iiop_daemon_maxopenconnections: NOT SET, DEFAULT=0.
BBOM0001I protocol_iiop_daemon_port: 32200.
BBOM0001I protocol_iiop_daemon_port_ssl: 32201.
BBOM0001I protocol_iiop_listenIPAddress: *.
BBOM0001I protocol_iiop_local_propagate_wlm_enclave: NOT SET, 576
DEFAULT=1.
BBOM0001I      protocol_iiop_port: 32203.
BBOM0001I protocol_iiop_resolve_foreign_hostname: NOT SET, DEFAULT=1.
BBOM0001I protocol_sip_timeout_output: NOT SET, DEFAULT=300.
BBOM0001I protocol_sip_timeout_output_recovery: NOT SET, 580
DEFAULT=SERVANT.
BBOM0001I protocol_sips_timeout_output: NOT SET, DEFAULT=300.
BBOM0001I protocol_sips_timeout_output_recovery: NOT SET, 582
```

```
DEFAULT=SERVANT.
BBOM0001I proxy_server_has_servants: NOT SET, DEFAULT=0.
BBOM0001I        ras_debugEnabled: NOT SET, DEFAULT=0.
BBOM0001I      ras_default_msg_dd: NOT SET.
BBOM0001I ras_dumpoptions_dumptype: NOT SET, DEFAULT=3.
BBOM0001I ras_dumpoptions_ledumpoptons: NOT SET, DEFAULT=THREAD(ALL)
587 BLOCKS.
BBOM0001I     ras_hardcopy_msg_dd: NOT SET.
BBOM0001I   ras_log_logstreamName: NOT SET.
BBOM0001I    ras_minorcode_action: NOT SET, DEFAULT=NODIAGNOSTICDATA.
BBOM0001I  ras_stderr_ff_interval: NOT SET, DEFAULT=0.
BBOM0001I ras_stderr_ff_line_interval: NOT SET, DEFAULT=0.
BBOM0001I  ras_stdout_ff_interval: NOT SET, DEFAULT=0.
BBOM0001I ras_stdout_ff_line_interval: NOT SET, DEFAULT=0.
BBOM0001I          ras_time_local: NOT SET, DEFAULT=0.
BBOM0001I         ras_trace_basic: NOT SET.
BBOM0001I   ras_trace_ctraceParms: NOT SET.
BBOM0001I ras_trace_defaultTracingLevel: 1.
BBOM0001I        ras_trace_detail: NOT SET.
BBOM0001I ras_trace_exclude_specific: NOT SET.
BBOM0001I ras_trace_minorCodeTraceBacks: NOT SET.
BBOM0001I ras_trace_outputLocation: SYSPRINT BUFFER.
BBOM0001I      ras_trace_specific: NOT SET.
BBOM0001I   ras_trace_BufferCount: 4.
BBOM0001I    ras_trace_BufferSize: 128K.
BBOM0001I  register_ifaedreg_also: NOT SET, DEFAULT=0.
BBOM0001I security_zOS_domainName: NOT SET.
BBOM0001I security_zOS_domainType: NOT SET, DEFAULT=2.
BBOM0001I security_zOS_overrideStartupAPPL: NOT SET.
BBOM0001I security_zOS_profilePrefix: BBNBASE.
BBOM0001I security_SMF_record_first_auth_user: NOT SET, DEFAULT=1.
BBOM0001I servant_jvm_direct_options: NOT SET.
BBOM0001I servant_region_custom_thread_count: NOT SET, DEFAULT=40.
BBOM0001I server_configured_system_name: SC74.
BBOM0001I server_generic_short_name: BBNC001.
BBOM0001I     server_generic_uuid: 616
C7A6E2E927AFAE21000001BC00000001090C0446.
BBOM0001I server_region_classpath: 617
/zWebSphereOEM/V7R0/config1/AppServer/profiles/default/properties:/zWe
bSphereOEM/V7R0/config1/AppServer/properties:/zWebSphereOEM/V7R0/confi
g1/AppServer/lib/bootstrap.jar:/zWebSphereOEM/V7R0/config1/AppServer/l
ib/urlprotocols.jar:/zWebSphereOEM/V7R0/config1/AppServer/lib/bootstra
pws390.jar:/zWebSphereOEM/V7R0/config1/AppServer/lib/lmproxy.jar:/zWeb
SphereOEM/V7R0/config1/AppServer/lib/startup.jar:/zWebSphereOEM/V7R0/c
onfig1/AppServer/java64/lib/tools.jar:/zWebSphereOEM/V7R0/config1/AppS
erver/deploytool/itp/batchboot.jar:/zWebSphereOEM/V7R0/config1/AppServ
er/deploytool/itp/batch2.jar.
BBOM0001I server_region_connect_to_wlm_early: NOT SET, DEFAULT=0.
BBOM0001I server_region_cputimeused_dump_action: NOT SET, 619
DEFAULT=TRACEBACK.
BBOM0001I server_region_dpm_dump_action: NOT SET, DEFAULT=TRACEBACK.
BBOM0001I server_region_dynapplenv_jclparms: 621
AMODE=64,ENV=BBNBASE.BBNNODE.&IWMSSNM.,JOBNAME=&IWMSSNM.S
BBOM0001I server_region_dynapplenv_jclproc: BBN7ASR .
BBOM0001I server_region_http_stalled_thread_dump_action: NOT SET, 623
```

```
DEFAULT=TRACEBACK.
BBOM0001I server_region_https_stalled_thread_dump_action: NOT SET, 624
DEFAULT=TRACEBACK.
BBOM0001I server_region_iiop_stalled_thread_dump_action: NOT SET, 625
DEFAULT=TRACEBACK.
BBOM0001I server_region_jvm_localrefs: NOT SET, DEFAULT=128.
BBOM0001I server_region_jvm_logfile: NOT SET.
BBOM0001I server_region_jvm_properties_file: 628
/zWebSphereOEM/V7R0/config1/AppServer/profiles/default/config/cells/bb
nbase/nodes/bbnnode/servers/server1/servant.jvm.options.
BBOM0001I   server_region_libpath: 629
/zWebSphereOEM/V7R0/config1/AppServer/java64/bin:/zWebSphereOEM/V7R0/c
onfig1/AppServer/java64/bin/j9vm:/zWebSphereOEM/V7R0/config1/AppServer
/java64/lib/s390/j9vm:/zWebSphereOEM/V7R0/config1/AppServer/java64/lib
/s390:/zWebSphereOEM/V7R0/config1/AppServer/java64/lib/s390x/j9vm:/zWe
bSphereOEM/V7R0/config1/AppServer/java64/lib/s390x:/zWebSphereOEM/V7R0
/config1/AppServer/lib:/zWebSphereOEM/V7R0/config1/AppServer/java64/jr
e/bin:/zWebSphereOEM/V7R0/config1/AppServer/java64/jre/lib/s390/j9vm:/
zWebSphereOEM/V7R0/config1/AppServer/java64/jre/lib/s390x/j9vm:/zWebSp
hereOEM/V7R0/config1/AppServer/java64/jre/lib/s390x.
BBOM0001I server_region_mdb_stalled_thread_dump_action: NOT SET, 630
DEFAULT=TRACEBACK.
BBOM0001I server_region_recycle_count: NOT SET, DEFAULT=0.
BBOM0001I server_region_request_cputimeused_limit: NOT SET, DEFAULT=0.
BBOM0001I server_region_sip_stalled_thread_dump_action: NOT SET, 633
DEFAULT=TRACEBACK.
BBOM0001I server_region_sips_stalled_thread_dump_action: NOT SET, 634
DEFAULT=TRACEBACK.
BBOM0001I server_region_stalled_thread_threshold_percent: NOT SET, 635
DEFAULT=0.
BBOM0001I server_region_thread_stack_size: NOT SET, DEFAULT=0.
BBOM0001I server_region_use_java_g: NOT SET, DEFAULT=0.
BBOM0001I server_region_workload_profile: IOBOUND.
BBOM0001I    server_specific_name: server1.
BBOM0001I server_specific_short_name: BBNS001.
BBOM0001I    server_specific_uuid: 641
C7A6E2E91DB54DA3000001BC00000001090C0446.
BBOM0001I server_start_wait_for_initialization_Timeout: NOT SET, 642
DEFAULT=0.
BBOM0001I             server_type: AppServer.
BBOM0001I server_use_wlm_to_queue_work: NOT SET, DEFAULT=1.
BBOM0001I server_work_distribution_algorithm: NOT SET, DEFAULT=0.
BBOM0001I server_SMF_container_activity_enabled: NOT SET, DEFAULT=0.
BBOM0001I server_SMF_container_interval_enabled: NOT SET, DEFAULT=0.
BBOM0001I server_SMF_interval_length: NOT SET, DEFAULT=0.
BBOM0001I server_SMF_request_activity_enabled: NOT SET, DEFAULT=0.
BBOM0001I server_SMF_request_activity_security: NOT SET, DEFAULT=0.
BBOM0001I server_SMF_request_activity_timestamps: NOT SET, DEFAULT=0.
BBOM0001I server_SMF_request_activity_CPU_detail: NOT SET, DEFAULT=0.
BBOM0001I server_SMF_server_activity_enabled: NOT SET, DEFAULT=0.
BBOM0001I server_SMF_server_interval_enabled: NOT SET, DEFAULT=0.
BBOM0001I shell_command_proc_name: BBN7ADM.
BBOM0001I suppress_hung_thread_abend: NOT SET, DEFAULT=0.
BBOM0001I suppress_hung_thread_dump: NOT SET, DEFAULT=0.
BBOM0001I transaction_defaultTimeout: 120.
```

```
BBOM0001I transaction_heuristicRetryLimit: NOT SET, DEFAULT=0.
BBOM0001I transaction_heuristicRetryWait: NOT SET, DEFAULT=0.
BBOM0001I transaction_maximumTimeout: 300.
BBOM0001I transaction_recoveryTimeout: NOT SET, DEFAULT=15.
BBOM0001I transaction_LPSHeuristicCompletion: NOT SET, 663
DEFAULT=ROLLBACK.
BBOM0001I            was_env_file: 664
/zWebSphereOEM/V7R0/config1/AppServer/profiles/default/config/cells/bb
nbase/nodes/bbnnode/servers/server1/was.env.
BBOM0001I wlm_classification_file: NOT SET.
BBOM0001I wlm_dynapplenv_single_server: 1.
BBOM0001I      wlm_maximumSRCount: 1.
BBOM0001I      wlm_minimumSRCount: 1.
BBOM0001I wlm_servant_start_parallel: NOT SET, DEFAULT=0.
BBOM0001I wlm_stateful_session_placement_on: 0.
BBOM0001I wsadmin_dumpthreads_enable_heapdump: NOT SET, DEFAULT=1.
BBOM0001I wsadmin_dumpthreads_enable_javatdump: NOT SET, DEFAULT=1.
START BBN7DMNB,JOBNAME=BBN7ACRS,ENV=BBNBASE.BBNBASE.BBN7ACRS,
REUSASID=YES
IRR812I PROFILE BBN7DMNB.* (G) IN THE STARTED CLASS WAS USED 674
        TO START BBN7DMNB WITH JOBNAME BBN7ACRS.
$HASP100 BBN7ACRS ON STCINRDR
IEF695I START BBN7DMNB WITH JOBNAME BBN7ACRS IS ASSIGNED TO USER
WSCRU1  , GROUP WSCFG1
$HASP373 BBN7ACRS STARTED
BBO000000I WEBSPHERE APPLICATION SERVER FOR Z/OS 678
LICENSED MATERIAL - PROPERTY OF IBM
5655-N02 (C) COPYRIGHT IBM CORP. 2000, 2008 ALL RIGHTS RESERVED.
U.S. GOVERNMENT USERS - RESTRICTED RIGHTS - USE, DUPLICATION, OR
DISCLOSURE RESTRICTED BY GSA-ADP SCHEDULE CONTRACT WITH IBM CORP.
IBM IS A REGISTERED TRADEMARK OF THE IBM CORP.
BBO000007I WEBSPHERE FOR Z/OS DAEMON BBNBASE/BBNNODE/BBNBASE/BBN7ACRS
679
IS STARTING.
BBO000237I WEBSPHERE FOR Z/OS DAEMON bbnbase/bbnnode/BBN7ACRS IS 680
STARTING.
BBO000307I DAEMON PROCESS BBNBASE/BBNNODE/BBNBASE/BBN7ACRS IS
EXECUTING 681
IN 64-BIT ADDRESSING MODE.
BBOM0007I CURRENT CB SERVICE LEVEL IS build level 7.0.0.15 (AM35611)
 682
 release WAS70.ZNATV date 03/30/11 20:08:05.
BBOM0001I        adapter_max_conn: NOT SET, DEFAULT=100.
BBOM0001I  adapter_max_otma_names: NOT SET, DEFAULT=16.
BBOM0001I        adapter_max_serv: NOT SET, DEFAULT=100.
BBOM0001I      adapter_max_shrmem: NOT SET, DEFAULT=33554432.
BBOM0001I adjunct_jvm_direct_options: NOT SET.
BBOM0001I adjunct_region_dynapplenv_jclparms: NOT SET.
BBOM0001I adjunct_region_dynapplenv_jclproc: NOT SET.
BBOM0001I adjunct_region_jvm_properties_file: NOT SET.
BBOM0001I adjunct_region_server_configured_to_bus: NOT SET, DEFAULT=0.
BOM0001I adjunct_region_start_default_adjunct: NOT SET, DEFAULT=0.
BBOM0001I  allow_large_SAF_groups: NOT SET, DEFAULT=0.
BBOM0001I               cell_name: bbnbase.
BBOM0001I         cell_short_name: BBNBASE.
```

```
BBOM0001I client_ras_logstreamname: NOT SET.
BBOM0001I com_ibm_userRegistries_type: security:LocalOSUserRegistry.
BBOM0001I com_ibm_ws_logging_zos_errorlog_format_cbe: NOT SET, 698
DEFAULT=0.
BBOM0001I com_ibm_CSI_claimTransportAssocSSLTLSRequired: 0.
BBOM0001I com_ibm_CSI_claimTransportAssocSSLTLSSupported: 1.
BBOM0001I com_ibm_DAEMON_claim_ssl_sys_v3_timeout: 600.
BBOM0001I com_ibm_DAEMON_claimKeyringName: WASKeyring.BBNBASE.
BBOM0001I com_ibm_DAEMON_claimSecurityCipherSuiteList: NOT SET.
BBOM0001I com_ibm_Server_Security_Enabled: 1.
BBOM0001I             config_root: NOT SET.
BBOM0001I control_region_classpath: NOT SET.
BBOM0001I control_region_confirm_recovery_on_no_srs: NOT SET, 707
DEFAULT=0.
BBOM0001I control_region_dreg_on_no_srs: NOT SET, DEFAULT=0.
BBOM0001I control_region_http_queue_timeout_percent: NOT SET, 709
DEFAULT=99.
BBOM0001I control_region_https_queue_timeout_percent: NOT SET, 710
DEFAULT=99.
BBOM0001I control_region_iiop_queue_timeout_percent: NOT SET, 711
DEFAULT=99.
BBOM0001I control_region_jms_request_timeout: NOT SET, DEFAULT=60.
BBOM0001I control_region_jvm_localrefs: NOT SET, DEFAULT=128.
BBOM0001I control_region_jvm_logfile: NOT SET.
BBOM0001I control_region_jvm_properties_file: NOT SET.
BBOM0001I  control_region_libpath: NOT SET.
BBOM0001I control_region_mdb_queue_timeout_percent: NOT SET, 717
DEFAULT=99.
BBOM0001I control_region_mdb_request_timeout: NOT SET, DEFAULT=120.
BBOM0001I control_region_sip_queue_timeout_percent: NOT SET, 719
DEFAULT=99.
BBOM0001I control_region_sips_queue_timeout_percent: NOT SET, 720
DEFAULT=99.
BBOM0001I control_region_start_cmd: NOT SET.
BBOM0001I control_region_thread_pool_size: NOT SET, DEFAULT=25.
BBOM0001I control_region_thread_stack_size: NOT SET, DEFAULT=0.
BBOM0001I control_region_timeout_delay: NOT SET, DEFAULT=0.
BBOM0001I control_region_timeout_dump_action: NOT SET.
BBOM0001I control_region_timeout_dump_action_session: NOT SET.
BBOM0001I control_region_timeout_save_last_servant: NOT SET, 727
DEFAULT=0.
BBOM0001I control_region_use_java_g: NOT SET, DEFAULT=0.
BBOM0001I control_region_wlm_dispatch_timeout: NOT SET, DEFAULT=1200.
BBOM0001I controller_jvm_direct_options: NOT SET.
BBOM0001I       daemon_group_name: BBNBASE.
BBOM0001I    daemon_start_command: NOT SET.
BBOM0001I daemon_start_command_args: 733
JOBNAME=BBN7ACRS,ENV=BBNBASE.BBNBASE.BBN7ACRS,REUSASID=YES.
BBOM0001I          daemon_wlmable: 0.
BBOM0001I      daemonInstanceName: BBN7ACRS.
BBOM0001I              daemonName: BBNBASE.
BBOM0001I default_internal_work_transaction_class: NOT SET.
BBOM0001I          enable_adapter: NOT SET, DEFAULT=0.
BBOM0001I    iiop_max_msg_megsize: NOT SET, DEFAULT=0.
BBOM0001I iiop_max_send_queue_megsize: NOT SET, DEFAULT=0.
```

```
BBOM0001I local_comm_max_msg_megsize: NOT SET, DEFAULT=0.
BBOM0001I local_comm_max_msg_megsize: NOT SET, DEFAULT=0.
BBOM0001I         multi_cell_mode: NOT SET, DEFAULT=0.
BBOM0001I            nls_language: NOT SET, DEFAULT=ENUS.
BBOM0001I               node_name: bbnnode.
BBOM0001I         node_short_name: BBNNODE.
BBOM0001I odr_tclass_propogation_enabled: NOT SET, DEFAULT=1.
BBOM0001I ola_cicsuser_identity_propagate: NOT SET, DEFAULT=0.
BBOM0001I ola_outbound_transaction_inactivity_timeout: NOT SET, 748
DEFAULT=120.
BBOM0001I ola_trace_settings_file: NOT SET.
BBOM0001I protocol_accept_http_work_after_min_srs: NOT SET, DEFAULT=1.
BBOM0001I protocol_accept_iiop_work_after_min_srs: NOT SET, DEFAULT=0.
BBOM0001I protocol_bboc_log_response_failure: NOT SET, DEFAULT=0.
BBOM0001I protocol_bboc_log_return_exception: NOT SET, DEFAULT=0.
BBOM0001I protocol_giop_level_highest: NOT SET, DEFAULT=1.2.
BBOM0001I protocol_http_large_data_inbound_buffer: NOT SET, DEFAULT=0.
BBOM0001I protocol_http_large_data_inbound_buffer_64bit: NOT SET, 756
DEFAULT=0.
BBOM0001I protocol_http_large_data_response_buffer: NOT SET, 757
DEFAULT=104857600.
BBOM0001I protocol_http_resolve_foreign_hostname: NOT SET, DEFAULT=0.
BBOM0001I protocol_http_timeout_output: NOT SET, DEFAULT=300.
BBOM0001I protocol_http_timeout_output_recovery: NOT SET, 760
DEFAULT=SERVANT.
BBOM0001I protocol_http_transactionClass: NOT SET.
BBOM0001I protocol_http_transport_class_mapping_file: NOT SET.
BBOM0001I protocol_https_cert_mapping_file: NOT SET.
BBOM0001I protocol_https_default_cert_label: NOT SET.
BBOM0001I protocol_https_mutual_auth_cbind_check: NOT SET, DEFAULT=0.
BBOM0001I protocol_https_timeout_output: NOT SET, DEFAULT=300.
BBOM0001I protocol_https_timeout_output_recovery: NOT SET, 767
DEFAULT=SERVANT.
BBOM0001I protocol_https_transactionClass: NOT SET.
BBOM0001I   protocol_iiop_backlog: NOT SET, DEFAULT=10.
BBOM0001I protocol_iiop_backlog_ssl: NOT SET, DEFAULT=10.
BBOM0001I protocol_iiop_daemon_listenIPAddress: WTSC74.ITSO.IBM.COM.
BBOM0001I protocol_iiop_daemon_maxopenconnections: NOT SET, DEFAULT=0.
BBOM0001I protocol_iiop_daemon_port: 32200.
BBOM0001I protocol_iiop_daemon_port_ssl: 32201.
BBOM0001I protocol_iiop_listenIPAddress: *.
BBOM0001I protocol_iiop_local_propagate_wlm_enclave: NOT SET, 776
DEFAULT=1.
BBOM0001I      protocol_iiop_port: NOT SET, DEFAULT=0.
BBOM0001I protocol_iiop_resolve_foreign_hostname: NOT SET, DEFAULT=1.
BBOM0001I protocol_sip_timeout_output: NOT SET, DEFAULT=300.
BBOM0001I protocol_sip_timeout_output_recovery: NOT SET, 780
DEFAULT=SERVANT.
BBOM0001I protocol_sips_timeout_output: NOT SET, DEFAULT=300.
BBOM0001I protocol_sips_timeout_output_recovery: NOT SET, 782
DEFAULT=SERVANT.
BBOM0001I proxy_server_has_servants: NOT SET, DEFAULT=0.
BBOM0001I        ras_debugEnabled: NOT SET, DEFAULT=0.
BBOM0001I      ras_default_msg_dd: NOT SET.
BBOM0001I ras_dumpoptions_dumptype: NOT SET, DEFAULT=3.
```

```
BBOM0001I ras_dumpoptions_ledumpoptons: NOT SET, DEFAULT=THREAD(ALL)
787
BLOCKS.
BBOM0001I     ras_hardcopy_msg_dd: NOT SET.
BBOM0001I   ras_log_logstreamName: NOT SET.
BBOM0001I    ras_minorcode_action: NOT SET, DEFAULT=NODIAGNOSTICDATA.
BBOM0001I  ras_stderr_ff_interval: NOT SET, DEFAULT=0.
BBOM0001I ras_stderr_ff_line_interval: NOT SET, DEFAULT=0.
BBOM0001I  ras_stdout_ff_interval: NOT SET, DEFAULT=0.
BBOM0001I ras_stdout_ff_line_interval: NOT SET, DEFAULT=0.
BBOM0001I          ras_time_local: NOT SET, DEFAULT=0.
BBOM0001I         ras_trace_basic: NOT SET.
BBOM0001I   ras_trace_ctraceParms: 60.
BBOM0001I ras_trace_defaultTracingLevel: 1.
BBOM0001I        ras_trace_detail: NOT SET.
BBOM0001I ras_trace_exclude_specific: NOT SET.
BBOM0001I ras_trace_minorCodeTraceBacks: NOT SET.
BBOM0001I ras_trace_outputLocation: SYSPRINT BUFFER.
BBOM0001I      ras_trace_specific: NOT SET.
BBOM0001I   ras_trace_BufferCount: 4.
BBOM0001I    ras_trace_BufferSize: 128K.
BBOM0001I  register_ifaedreg_also: NOT SET, DEFAULT=0.
BBOM0001I security_zOS_domainName: NOT SET.
BBOM0001I security_zOS_domainType: NOT SET, DEFAULT=2.
BBOM0001I security_zOS_overrideStartupAPPL: NOT SET.
BBOM0001I security_zOS_profilePrefix: NOT SET.
BBOM0001I security_SMF_record_first_auth_user: NOT SET, DEFAULT=1.
BBOM0001I servant_jvm_direct_options: NOT SET.
BBOM0001I servant_region_custom_thread_count: NOT SET, DEFAULT=40.
BBOM0001I server_configured_system_name: SC74.
BBOM0001I server_generic_short_name: BBNBASE.
BBOM0001I     server_generic_uuid: 816
C7A6E2E9227171A3000001BC000000001090C0446.
BBOM0001I server_region_classpath: NOT SET.
BBOM0001I server_region_connect_to_wlm_early: NOT SET, DEFAULT=0.
BBOM0001I server_region_cputimeused_dump_action: NOT SET, 819
DEFAULT=TRACEBACK.
BBOM0001I server_region_dpm_dump_action: NOT SET, DEFAULT=TRACEBACK.
BBOM0001I server_region_dynapplenv_jclparms: NOT SET.
BBOM0001I server_region_dynapplenv_jclproc: NOT SET.
BBOM0001I server_region_http_stalled_thread_dump_action: NOT SET, 823
DEFAULT=TRACEBACK.
BBOM0001I server_region_https_stalled_thread_dump_action: NOT SET, 824
DEFAULT=TRACEBACK.
BBOM0001I server_region_iiop_stalled_thread_dump_action: NOT SET, 825
DEFAULT=TRACEBACK.
BBOM0001I server_region_jvm_localrefs: NOT SET, DEFAULT=128.
BBOM0001I server_region_jvm_logfile: NOT SET.
BBOM0001I server_region_jvm_properties_file: NOT SET.
BBOM0001I   server_region_libpath: NOT SET.
BBOM0001I server_region_mdb_stalled_thread_dump_action: NOT SET, 830
DEFAULT=TRACEBACK.
BBOM0001I server_region_recycle_count: NOT SET, DEFAULT=0.
BBOM0001I server_region_request_cputimeused_limit: NOT SET, DEFAULT=0.
BBOM0001I server_region_sip_stalled_thread_dump_action: NOT SET, 833
```

```
DEFAULT=TRACEBACK.
BBOM0001I server_region_sips_stalled_thread_dump_action: NOT SET, 834
DEFAULT=TRACEBACK.
BBOM0001I server_region_stalled_thread_threshold_percent: NOT SET, 835
DEFAULT=0.
BBOM0001I server_region_thread_stack_size: NOT SET, DEFAULT=0.
BBOM0001I server_region_use_java_g: NOT SET, DEFAULT=0.
BBOM0001I server_region_workload_profile: NOT SET, DEFAULT=IOBOUND.
BBOM0001I    server_specific_name: NOT SET.
BBOM0001I server_specific_short_name: BBN7ACRS.
BBOM0001I    server_specific_uuid: 841
C7A6E2E9184AFF23000001BC00000001090C0446.
BBOM0001I server_start_wait_for_initialization_Timeout: NOT SET, 842
DEFAULT=0.
BBOM0001I             server_type: NOT SET, DEFAULT=AppServer.
BBOM0001I server_use_wlm_to_queue_work: NOT SET, DEFAULT=1.
BBOM0001I server_work_distribution_algorithm: NOT SET, DEFAULT=0.
BBOM0001I server_SMF_container_activity_enabled: NOT SET, DEFAULT=0.
BBOM0001I server_SMF_container_interval_enabled: NOT SET, DEFAULT=0.
BBOM0001I server_SMF_interval_length: NOT SET, DEFAULT=0.
BBOM0001I server_SMF_request_activity_enabled: NOT SET, DEFAULT=0.
BBOM0001I server_SMF_request_activity_security: NOT SET, DEFAULT=0.
BBOM0001I server_SMF_request_activity_timestamps: NOT SET, DEFAULT=0.
BBOM0001I server_SMF_request_activity_CPU_detail: NOT SET, DEFAULT=0.
BBOM0001I server_SMF_server_activity_enabled: NOT SET, DEFAULT=0.
BBOM0001I server_SMF_server_interval_enabled: NOT SET, DEFAULT=0.
BBOM0001I shell_command_proc_name: NOT SET.
BBOM0001I suppress_hung_thread_abend: NOT SET, DEFAULT=0.
BBOM0001I suppress_hung_thread_dump: NOT SET, DEFAULT=0.
BBOM0001I transaction_defaultTimeout: NOT SET, DEFAULT=120.
BBOM0001I transaction_heuristicRetryLimit: NOT SET, DEFAULT=0.
BBOM0001I transaction_heuristicRetryWait: NOT SET, DEFAULT=0.
BBOM0001I transaction_maximumTimeout: NOT SET, DEFAULT=300.
BBOM0001I transaction_recoveryTimeout: NOT SET, DEFAULT=15.
BBOM0001I transaction_LPSHeuristicCompletion: NOT SET, 863
BBOM0001I            was_env_file: NOT SET.
BBOM0001I wlm_classification_file: NOT SET.
BBOM0001I wlm_dynapplenv_single_server: NOT SET, DEFAULT=0.
BBOM0001I      wlm_maximumSRCount: NOT SET, DEFAULT=0.
BBOM0001I      wlm_minimumSRCount: NOT SET, DEFAULT=1.
BBOM0001I wlm_servant_start_parallel: NOT SET, DEFAULT=0.
BBOM0001I wlm_stateful_session_placement_on: NOT SET, DEFAULT=0.
BBOM0001I wsadmin_dumpthreads_enable_heapdump: NOT SET, DEFAULT=1.
BBOM0001I wsadmin_dumpthreads_enable_javatdump: NOT SET, DEFAULT=1.
IEE538I CTIBBO60 MEMBER NOT FOUND IN PARMLIB
IEE538I CTIBBOOO MEMBER NOT FOUND IN PARMLIB
BBOOO024I ERRORS WILL BE WRITTEN TO CERR FOR JOB BBN7ACRS.
BBOOO302I REGION REQUESTED = OK 876
ACTUAL BELOW/ABOVE LINE LIMIT = 7M / 1756M
ABOVE BAR FREE/ALLOC ADDR = 0 / 0
BBOOO331I MEMLIMIT=OOOOOfffffffOOO.  MEMLIMIT CONFIGURATION 877
SOURCE=REGO
BBOOO341I VARIOUS RESOURCE MONITORING DATA: 878
(64):():():():():():():():():()
BBOOO374I PROCESSING OLA TRACE FILE ''.
```

```
ICH408I USER(WSCRU1  ) GROUP(WSCFG1  ) NAME(WAS CR OWNER        ) 880
   CSFIQA CL(CSFSERV )
   INSUFFICIENT ACCESS AUTHORITY
   FROM ** (G)
ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
 BBOOO015I INITIALIZATION COMPLETE FOR DAEMON BBN7ACRS.
 BBOOO246I INITIALIZATION COMPLETE FOR DAEMON 882
 BBNBASE/BBNNODE/BBNBASE/BBN7ACRS.
 IEA631I  OPERATOR LAFITTE  NOW INACTIVE, SYSTEM=SC74    , LU=SC38TCAO
 BBOOO024I ERRORS WILL BE WRITTEN TO CERR FOR JOB BBNSOO1.
 BBOOO302I REGION REQUESTED = OK 885
 ACTUAL BELOW/ABOVE LINE LIMIT = 7M / 1756M
 ABOVE BAR FREE/ALLOC ADDR = 0 / 0
 BBOOO331I MEMLIMIT=OOOOOfffffffOOO.  MEMLIMIT CONFIGURATION 886
 SOURCE=JCL
 BBOOO341I VARIOUS RESOURCE MONITORING DATA: 887
 (64):():():():():():():():()
 BBOOO234I SERVANT PROCESS THREAD COUNT IS 6.
BBOOO201I JVM HEAP INFORMATION FOR SERVER BBNCOO1/BBNSOO1/STCO4414
 BBOOO202I (STCO4414) HEAP(NURSERY), COUNT(OOOOOOOO), FREE 901
 STORAGE(OOOOOOOOOOF7A340), TOTAL STORAGE(OOOOOOOOO1OOOOOO)
 BBOOO202I (STCO4414) HEAP(MATURE), COUNT(OOOOOOOO), FREE 902
 STORAGE(OOOOOOOOO5FD9F48), TOTAL STORAGE(OOOOOOOOO6OOOOOO)
 BBOOO204I JVM HEAP INFORMATION FOR SERVER BBNCOO1/BBNSOO1/STCO4414 903
 COMPLETE
 BBOOO215I PRODUCT 'IBM WebSphere Ap' SUCCESSFULLY REGISTERED WITH 904
 IFAED SERVICE.
 IWMO34I PROCEDURE BBN7ASR STARTED FOR SUBSYSTEM BBNSOO1 905
 APPLICATION ENVIRONMENT BBNCOO1
 PARAMETERS AMODE=64,ENV=BBNBASE.BBNNODE.BBNSOO1,JOBNAME=BBNSOO1S
 IRR812I PROFILE BBNSOO1S.* (G) IN THE STARTED CLASS WAS USED 906
         TO START BBNSOO1S WITH JOBNAME BBNSOO1S.
 $HASP1OO BBNSOO1S ON STCINRDR
 $HASP373 BBNSOO1S STARTED
 +BBOOOOO4I WEBSPHERE FOR Z/OS SERVANT PROCESS 909
  BBNBASE/BBNNODE/BBNCOO1/BBNSOO1 IS STARTING.
 +BBOOO239I WEBSPHERE FOR Z/OS SERVANT PROCESS bbnbase/bbnnode/server1
 910
  IS STARTING.
 +BBOOO308I SERVANT PROCESS BBNBASE/BBNNODE/BBNCOO1/BBNSOO1 IS
 EXECUTING 911
  IN 64-BIT ADDRESSING MODE.
 +BBOMOOO7I CURRENT CB SERVICE LEVEL IS build level 7.0.0.15 (AM35611)
912
  release WAS7O.ZNATV date O3/3O/11 2O:O8:O5.
 +BBOMOOO1I        adapter_max_conn: NOT SET, DEFAULT=1OO.
 +BBOMOOO1I  adapter_max_otma_names: NOT SET, DEFAULT=16.
 +BBOMOOO1I        adapter_max_serv: NOT SET, DEFAULT=1OO.
 +BBOMOOO1I      adapter_max_shrmem: NOT SET, DEFAULT=33554432.
 +BBOMOOO1I adjunct_jvm_direct_options: NOT SET.
 +BBOMOOO1I adjunct_region_dynapplenv_jclparms: 918
  AMODE=64,ENV=BBNBASE.BBNNODE.&IWMSSNM.,JOBNAME=&IWMSSNM.A.
 +BBOMOOO1I adjunct_region_dynapplenv_jclproc: BBN7CRA.
 +BBOMOOO1I adjunct_region_jvm_properties_file: 92O
  /zWebSphereOEM/V7RO/config1/AppServer/profiles/default/config/cells/b
```

```
 bnbase/nodes/bbnnode/servers/server1/adjunct.jvm.options.
 +BBOM0001I adjunct_region_server_configured_to_bus: NOT SET,
 DEFAULT=0.
 +BBOM0001I adjunct_region_start_default_adjunct: 0.
 +BBOM0001I  allow_large_SAF_groups: NOT SET, DEFAULT=0.
 +BBOM0001I               cell_name: bbnbase.
 +BBOM0001I         cell_short_name: BBNBASE.
 +BBOM0001I client_ras_logstreamname: NOT SET.
 +BBOM0001I com_ibm_userRegistries_type: security:LocalOSUserRegistry.
 +BBOM0001I com_ibm_ws_logging_zos_errorlog_format_cbe: NOT SET, 928
  DEFAULT=0.
+BBOM0001I com_ibm_CSI_claimTransportAssocSSLTLSRequired: 0.
+BBOM0001I com_ibm_CSI_claimTransportAssocSSLTLSSupported: 1.
+BBOM0001I com_ibm_DAEMON_claim_ssl_sys_v3_timeout: NOT SET, 931
 DEFAULT=600.
+BBOM0001I com_ibm_DAEMON_claimKeyringName: NOT SET.
+BBOM0001I com_ibm_DAEMON_claimSecurityCipherSuiteList: NOT SET.
+BBOM0001I com_ibm_Server_Security_Enabled: 1.
+BBOM0001I             config_root: NOT SET.
+BBOM0001I control_region_classpath: 936
 /zWebSphereOEM/V7R0/config1/AppServer/profiles/default/properties:/zW
ebSphereOEM/V7R0/config1/AppServer/properties:/zWebSphereOEM/V7R0/conf
ig1/AppServer/lib/bootstrap.jar:/zWebSphereOEM/V7R0/config1/AppServer/
lib/bootstrapws390.jar:/zWebSphereOEM/V7R0/config1/AppServer/lib/lmpro
xy.jar:/zWebSphereOEM/V7R0/config1/AppServer/lib/startup.jar:/zWebSphe
reOEM/V7R0/config1/AppServer/java64/lib/tools.jar.
+BBOM0001I control_region_confirm_recovery_on_no_srs: NOT SET, 937
 DEFAULT=0.
+BBOM0001I control_region_dreg_on_no_srs: NOT SET, DEFAULT=0.
+BBOM0001I control_region_http_queue_timeout_percent: NOT SET, 939
DEFAULT=99.
+BBOM0001I control_region_https_queue_timeout_percent: NOT SET, 940
 DEFAULT=99.
+BBOM0001I control_region_iiop_queue_timeout_percent: NOT SET, 941
 DEFAULT=99.
+BBOM0001I control_region_jms_request_timeout: NOT SET, DEFAULT=60.
+BBOM0001I control_region_jvm_localrefs: NOT SET, DEFAULT=128.
+BBOM0001I control_region_jvm_logfile: NOT SET.
+BBOM0001I control_region_jvm_properties_file: 945
 /zWebSphereOEM/V7R0/config1/AppServer/profiles/default/config/cells/b
b
 nbase/nodes/bbnnode/servers/server1/control.jvm.options.
+BBOM0001I  control_region_libpath: 946
 /zWebSphereOEM/V7R0/config1/AppServer/java64/bin:/zWebSphereOEM/V7R0/
config1/AppServer/java64/bin/j9vm:/zWebSphereOEM/V7R0/config1/AppServe
r/java64/lib/s390/j9vm:/zWebSphereOEM/V7R0/config1/AppServer/java64/li
b/s390:/zWebSphereOEM/V7R0/config1/AppServer/java64/lib/s390x/j9vm:/zW
ebSphereOEM/V7R0/config1/AppServer/java64/lib/s390x:/zWebSphereOEM/V7R
0/config1/AppServer/lib:/zWebSphereOEM/V7R0/config1/AppServer/java64/j
re/bin:/zWebSphereOEM/V7R0/config1/AppServer/java64/jre/lib/s390/j9vm:/
zWebSphereOEM/V7R0/config1/AppServer/java64/jre/lib/s390x/j9vm:/zWebS
 phereOEM/V7R0/config1/AppServer/java64/jre/lib/s390x.
 +BBOM0001I control_region_mdb_queue_timeout_percent: NOT SET, 947
  DEFAULT=99.
 +BBOM0001I control_region_mdb_request_timeout: NOT SET, DEFAULT=120.
```

```
 +BBOM0001I control_region_sip_queue_timeout_percent: NOT SET, 949
  DEFAULT=99.
 +BBOM0001I control_region_sips_queue_timeout_percent: NOT SET, 950
  DEFAULT=99.
 +BBOM0001I control_region_start_cmd: NOT SET.
 +BBOM0001I control_region_thread_pool_size: 25.
 +BBOM0001I control_region_thread_stack_size: NOT SET, DEFAULT=0.
 +BBOM0001I control_region_timeout_delay: NOT SET, DEFAULT=0.
 +BBOM0001I control_region_timeout_dump_action: NOT SET.
 +BBOM0001I control_region_timeout_dump_action_session: NOT SET.
 +BBOM0001I control_region_timeout_save_last_servant: NOT SET, 957
  DEFAULT=0.
 +BBOM0001I control_region_use_java_g: NOT SET, DEFAULT=0.
 +BBOM0001I control_region_wlm_dispatch_timeout: 1200.
 +BBOM0001I controller_jvm_direct_options: NOT SET.
+BBOM0001I       daemon_group_name: BBNBASE.
 +BBOM0001I    daemon_start_command: START BBN7DMNB.
 +BBOM0001I daemon_start_command_args: 963
  JOBNAME=BBN7ACRS,ENV=BBNBASE.BBNBASE.BBN7ACRS,REUSASID=YES.
+BBOM0001I          daemon_wlmable: NOT SET, DEFAULT=0.
 +BBOM0001I      daemonInstanceName: BBN7ACRS.
 +BBOM0001I              daemonName: BBNBASE.
 +BBOM0001I default_internal_work_transaction_class: NOT SET.
 +BBOM0001I          enable_adapter: NOT SET, DEFAULT=0.
 +BBOM0001I    iiop_max_msg_megsize: NOT SET, DEFAULT=0.
 +BBOM0001I iiop_max_send_queue_megsize: NOT SET, DEFAULT=0.
 +BBOM0001I local_comm_max_msg_megsize: NOT SET, DEFAULT=0.
 +BBOM0001I         multi_cell_mode: NOT SET, DEFAULT=0.
 +BBOM0001I            nls_language: NOT SET, DEFAULT=ENUS.
 +BBOM0001I               node_name: bbnnode.
 +BBOM0001I         node_short_name: BBNNODE.
 +BBOM0001I odr_tclass_propogation_enabled: NOT SET, DEFAULT=1.
B +BBOM0001I ola_cicsuser_identity_propagate: NOT SET, DEFAULT=0.
 +BBOM0001I ola_outbound_transaction_inactivity_timeout: NOT SET, 978
  DEFAULT=120.
 +BBOM0001I ola_trace_settings_file: NOT SET.
 +BBOM0001I protocol_accept_http_work_after_min_srs: NOT SET,
 DEFAULT=1.
 +BBOM0001I protocol_accept_iiop_work_after_min_srs: NOT SET,
DEFAULT=0.
+BBOM0001I protocol_bboc_log_response_failure: NOT SET, DEFAULT=0.
+BBOM0001I protocol_bboc_log_return_exception: NOT SET, DEFAULT=0.
+BBOM0001I protocol_giop_level_highest: NOT SET, DEFAULT=1.2.
+BBOM0001I protocol_http_large_data_inbound_buffer: NOT SET,
DEFAULT=0.
+BBOM0001I protocol_http_large_data_inbound_buffer_64bit: NOT SET, 986
 DEFAULT=0.
+BBOM0001I protocol_http_large_data_response_buffer: NOT SET, 987
 DEFAULT=104857600.
+BBOM0001I protocol_http_resolve_foreign_hostname: NOT SET, DEFAULT=0.
+BBOM0001I protocol_http_timeout_output: NOT SET, DEFAULT=300.
+BBOM0001I protocol_http_timeout_output_recovery: NOT SET, 990
DEFAULT=SERVANT.
+BBOM0001I protocol_http_transactionClass: NOT SET.
+BBOM0001I protocol_http_transport_class_mapping_file: NOT SET.
```

```
+BBOM0001I protocol_https_cert_mapping_file: NOT SET.
+BBOM0001I protocol_https_default_cert_label: NOT SET.
+BBOM0001I protocol_https_mutual_auth_cbind_check: NOT SET, DEFAULT=0.
+BBOM0001I protocol_https_timeout_output: NOT SET, DEFAULT=300.
+BBOM0001I protocol_https_timeout_output_recovery: NOT SET, 997
 DEFAULT=SERVANT.
+BBOM0001I protocol_https_transactionClass: NOT SET.
+BBOM0001I   protocol_iiop_backlog: NOT SET, DEFAULT=10.
+BBOM0001I   protocol_iiop_backlog: NOT SET, DEFAULT=10.
+BBOM0001I protocol_iiop_backlog_ssl: NOT SET, DEFAULT=10.
+BBOM0001I protocol_iiop_daemon_listenIPAddress: WTSC74.ITSO.IBM.COM.
+BBOM0001I protocol_iiop_daemon_maxopenconnections: NOT SET,
DEFAULT=0.
+BBOM0001I protocol_iiop_daemon_port: 32200.
+BBOM0001I protocol_iiop_daemon_port_ssl: 32201.
+BBOM0001I protocol_iiop_listenIPAddress: *.
+BBOM0001I protocol_iiop_local_propagate_wlm_enclave: NOT SET, 006
 DEFAULT=1.
+BBOM0001I      protocol_iiop_port: 32203.
+BBOM0001I protocol_iiop_resolve_foreign_hostname: NOT SET, DEFAULT=1.
+BBOM0001I protocol_sip_timeout_output: NOT SET, DEFAULT=300.
+BBOM0001I protocol_sip_timeout_output_recovery: NOT SET, 010
 DEFAULT=SERVANT.
+BBOM0001I protocol_sips_timeout_output: NOT SET, DEFAULT=300.
+BBOM0001I protocol_sips_timeout_output_recovery: NOT SET, 012
 DEFAULT=SERVANT.
+BBOM0001I proxy_server_has_servants: NOT SET, DEFAULT=0.
+BBOM0001I        ras_debugEnabled: NOT SET, DEFAULT=0.
+BBOM0001I      ras_default_msg_dd: NOT SET.
+BBOM0001I ras_dumpoptions_dumptype: NOT SET, DEFAULT=3.
+BBOM0001I ras_dumpoptions_ledumpoptons: NOT SET, DEFAULT=THREAD(ALL)
017 BLOCKS.
+BBOM0001I     ras_hardcopy_msg_dd: NOT SET.
 +BBOM0001I   ras_log_logstreamName: NOT SET.
 +BBOM0001I    ras_minorcode_action: NOT SET, DEFAULT=NODIAGNOSTICDATA.
 +BBOM0001I  ras_stderr_ff_interval: NOT SET, DEFAULT=0.
 +BBOM0001I ras_stderr_ff_line_interval: NOT SET, DEFAULT=0.
 +BBOM0001I  ras_stdout_ff_interval: NOT SET, DEFAULT=0.
 +BBOM0001I ras_stdout_ff_line_interval: NOT SET, DEFAULT=0.
 +BBOM0001I          ras_time_local: NOT SET, DEFAULT=0.
 +BBOM0001I         ras_trace_basic: NOT SET.
 +BBOM0001I   ras_trace_ctraceParms: NOT SET.
 +BBOM0001I ras_trace_defaultTracingLevel: 1.
 +BBOM0001I        ras_trace_detail: NOT SET.
 +BBOM0001I ras_trace_exclude_specific: NOT SET.
 +BBOM0001I ras_trace_minorCodeTraceBacks: NOT SET.
 +BBOM0001I ras_trace_outputLocation: SYSPRINT BUFFER.
 +BBOM0001I      ras_trace_specific: NOT SET.
 +BBOM0001I   ras_trace_BufferCount: 4.
 +BBOM0001I    ras_trace_BufferSize: 128K.
 +BBOM0001I  register_ifaedreg_also: NOT SET, DEFAULT=0.
 +BBOM0001I security_zOS_domainName: NOT SET.
+BBOM0001I security_zOS_domainType: NOT SET, DEFAULT=2.
 +BBOM0001I security_zOS_overrideStartupAPPL: NOT SET.
 +BBOM0001I security_zOS_profilePrefix: BBNBASE.
```

```
 +BBOM0001I security_SMF_record_first_auth_user: NOT SET, DEFAULT=1.
+BBOM0001I servant_jvm_direct_options: NOT SET.
+BBOM0001I servant_region_custom_thread_count: NOT SET, DEFAULT=40.
+BBOM0001I server_configured_system_name: SC74.
+BBOM0001I server_generic_short_name: BBNC001.
+BBOM0001I     server_generic_uuid: 046
 C7A6E2E927AFAE21000001BC00000001090C0446.
+BBOM0001I server_region_classpath: 047
 /zWebSphereOEM/V7R0/config1/AppServer/profiles/default/properties:/zW
ebSphereOEM/V7R0/config1/AppServer/properties:/zWebSphereOEM/V7R0/conf
ig1/AppServer/lib/bootstrap.jar:/zWebSphereOEM/V7R0/config1/AppServer/
lib/urlprotocols.jar:/zWebSphereOEM/V7R0/config1/AppServer/lib/bootstr
apws390.jar:/zWebSphereOEM/V7R0/config1/AppServer/lib/lmproxy.jar:/zWe
bSphereOEM/V7R0/config1/AppServer/lib/startup.jar:/zWebSphereOEM/V7R0/
config1/AppServer/java64/lib/tools.jar:/zWebSphereOEM/V7R0/config1/App
Server/deploytool/itp/batchboot.jar:/zWebSphereOEM/V7R0/config1/AppSer
ver/deploytool/itp/batch2.jar.
+BBOM0001I server_region_connect_to_wlm_early: NOT SET, DEFAULT=0.
+BBOM0001I server_region_cputimeused_dump_action: NOT SET, 049
 DEFAULT=TRACEBACK.
+BBOM0001I server_region_dpm_dump_action: NOT SET, DEFAULT=TRACEBACK.
+BBOM0001I server_region_dynapplenv_jclparms: 051
 AMODE=64,ENV=BBNBASE.BBNNODE.&IWMSSNM.,JOBNAME=&IWMSSNM.S
 .
+BBOM0001I server_region_dynapplenv_jclproc: BBN7ASR .
+BBOM0001I server_region_http_stalled_thread_dump_action: NOT SET, 053
 DEFAULT=TRACEBACK.
+BBOM0001I server_region_https_stalled_thread_dump_action: NOT SET,
054
 DEFAULT=TRACEBACK.
+BBOM0001I server_region_iiop_stalled_thread_dump_action: NOT SET, 055
 DEFAULT=TRACEBACK.
+BBOM0001I server_region_jvm_localrefs: NOT SET, DEFAULT=128.
+BBOM0001I server_region_jvm_logfile: NOT SET.
+BBOM0001I server_region_jvm_properties_file: 058
 /zWebSphereOEM/V7R0/config1/AppServer/profiles/default/config/cells/b
b
 nbase/nodes/bbnnode/servers/server1/servant.jvm.options.
+BBOM0001I   server_region_libpath: 059
 /zWebSphereOEM/V7R0/config1/AppServer/java64/bin:/zWebSphereOEM/V7R0/
.....................................
+BBOO0302I REGION REQUESTED = 0K 104
  ACTUAL BELOW/ABOVE LINE LIMIT = 7M / 1756M
  ABOVE BAR FREE/ALLOC ADDR = 0 / 0
 +BBOO0331I MEMLIMIT=00000fffffff000.  MEMLIMIT CONFIGURATION 105
  SOURCE=JCL
 +BBOO0341I VARIOUS RESOURCE MONITORING DATA: 106
  (64):():():():():():():():():()
 +BBOO0201I JVM HEAP INFORMATION FOR SERVER BBNC001/BBNS001/STC04417
 +BBOO0202I (STC04417) HEAP(NURSERY), COUNT(00000000), FREE 108
  STORAGE(0000000001F7AA80), TOTAL STORAGE(0000000002000000)
 +BBOO0202I (STC04417) HEAP(MATURE), COUNT(00000000), FREE 109
  STORAGE(000000000BFD9F48), TOTAL STORAGE(000000000C000000)
 +BBOO0204I JVM HEAP INFORMATION FOR SERVER BBNC001/BBNS001/STC04417
 110
```

```
  COMPLETE
 IAT7036 CONNECT NOT RECEIVED FOR WRITER FSS IPDSWAY IN ASID 0066 ON
 SC75
 +BBOO0020I INITIALIZATION COMPLETE FOR WEBSPHERE FOR Z/OS SERVANT 111
  PROCESS BBNS001.
 +BBOO0248I INITIALIZATION COMPLETE FOR WEBSPHERE FOR Z/OS SERVANT 112
  PROCESS BBNBASE/BBNNODE/BBNC001/BBNS001.
 BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR Z/OS CONTROL 113
PROCESS BBNS001.

BBOO0247I INITIALIZATION COMPLETE FOR WEBSPHERE FOR Z/OS CONTROL 114
PROCESS BBNBASE/BBNNODE/BBNC001/BBNS001.
```

A.2 Setting up z/OSMF V1R13

Then set up z/OSMF, as partially described in Example A-4 on page 785.

Note: To install the deployment manager plug-in z/OSMF V1R13, SMP/E V3R6 (which is new with z/OS V1R13) is required (FMID HMP1J00).

To set up z/OSMF, follow these steps:

1. Run script `izusetup.sh -config` (Example A-4)
2. Run script `izudflt.cfg.rexx` (Example A-5).
3. Run script `izudflt.cfg -prime` (Example A-6).
4. Run script `izudflt.cfg -finish` (Example A-7).
5. Run script `izuauthuser` for each potential user of z/OSMF (Example A-8).
6. Run `izudflt.userid.rexx` for each user ID potentially using z/OSMF (Example A-9).

Example A-4 izusetup

```
izusetup.sh -config
izusetup.sh -config
+ izusetup.sh -config
IZUG215I: Starting z/OSMF "setup" procedure.

Version: 1.159
Delta Date: 3/25/11
Delta Time: 15:57:03

IZUG091I: Environment variable "IZU_CODE_ROOT" is set to the value "/usr/lpp/zos
mf/V1R13".
IZUG091I: Environment variable "IZU_CONFIG_DIR" is set to the value "/etc/zosmf"
.

IZUG091I: Environment variable "IZU_LOGFILE_DIR" is set to the value "/var/zosmf
/configuration/logs".

IZUG213I: Log information will be written to file "/var/zosmf/configuration/logs
/izusetup.04.20.11.16.39.52.log".
```

```
IZUG215I: Starting z/OSMF "setup" procedure.

Wed Apr 20 13:39:52 EDT 2011


IZUG146I: Invoking script "/usr/lpp/zosmf/V1R13/bin/izuconfig.sh -config /usr/lp
p/zosmf/V1R13/defaults/izudflt.cfg".
IZUG215I: Starting z/OSMF "configuration" procedure.

Wed Apr 20 13:39:54 EDT 2011


IZUG237I: Enter the name of the file to save the configuration data (must be .cf
g extension), or press Enter to save as file "/usr/lpp/zosmf/V1R13/defaults/izud
flt.cfg":
Overwrite (Y|N)?
Y
IZUG243I: Accepted input: /usr/lpp/zosmf/V1R13/defaults/izudflt.cfg

IZUG145I: Enter the mount point for the z/OSMF data file system, or press Enter
to accept the default "/var/zosmf/data":


IZUG243I: Accepted input: /var/zosmf/data

IZUG229I: Enter the fully qualified name of the z/OSMF data file system, or pres
s Enter to accept the default data file system name "IZU.SIZUDATA":

IZUG232I: The specified z/OSMF data file system with name "IZU.SIZUDATA" was acc
epted.

IZUG157I: Enter the z/OSMF data file system type for the file system: "IZU.SIZUD
ATA", or press Enter to accept the default "ZFS":

IZUG232I: The specified z/OSMF data file system with type "ZFS" was accepted.

IZUG158I: Enter the name of the volume to use for creating the z/OSMF data file
system, enter an asterisk (*) to use SMS managed storage, or press Enter to acce
pt the default "*":
BH5ST3
IZUG247I: z/OSMF data file system will be created on volume: "BH5ST3"

IZUG159I: Enter the size (in cylinders) to allocate for the data file system, or
 press Enter to accept the default "200":

IZUG243I: Accepted input: 200

IZUG061I: What security mode do you want to use? To use SAF mode, enter S. To us
e Repository mode, enter R. Or press Enter to accept the default "S":

IZUG243I: Accepted input: SAF

IZUG072I: If you have AUTOUID enabled, RACF can assign unused UIDs for your user
 ids. Do you want RACF to automatically assign UIDs to user ids created by z/OSM
F? For yes, enter Y. For no, enter N:
```

```
Y
IZUG243I: Accepted input: Y

IZUG070I: If you have AUTOGID enabled, RACF can assign unused GIDs for your grou
p ids. Do you want RACF to automatically assign GIDs to groups created by z/OSMF
? For yes, enter Y. For no, enter N:
Y
IZUG243I: Accepted input: Y

IZUG274I: Enter the z/OSMF administrator group name, or press Enter to accept "I
ZUADMIN":

IZUG243I: Accepted input: IZUADMIN

IZUG069I: The configuration property "IZU_ADMIN_GROUP_GID" is set to the value "
AUTOGID".

IZUG274I: Enter the z/OSMF user group name, or press Enter to accept "IZUUSER":

IZUG243I: Accepted input: IZUUSER

IZUG069I: The configuration property "IZU_USERS_GROUP_GID" is set to the value "
AUTOGID".

IZUG256I: Enter the z/OSMF administrator user ID, or press Enter to accept the d
efault "ZOSMFAD":

IZUG257W: User "ZOSMFAD" already exists.
IZUG243I: Accepted input: ZOSMFAD

IZUG069I: The configuration property "IZU_ADMIN_UID" is set to the value "AUTOUI
D".

IZUG256I: Enter the z/OSMF administrator home directory, or press Enter to accep
t the default "/u/zosmfad":

IZUG243I: Accepted input: /u/zosmfad

IZUG256I: Enter the z/OSMF administrator shell program name, or press Enter to a
ccept the default "/bin/sh":

IZUG243I: Accepted input: /bin/sh

IZUG255I: Enter the z/OSMF administrator logon procedure name:
IKJACCNT
IZUG243I: Accepted input: IKJACCNT

IZUG255I: Enter the z/OSMF administrator account number:
MVS
IZUG243I: Accepted input: MVS

IZUG153I: Enter the directory path and name of the WebSphere Application Server
configuration file, or press enter to accept "/etc/zWebSphereOEM/V7R0/conf/CONFI
G1/CONFIG1.responseFile":
OEM
```

```
IZUG155E: The WebSphere Application Server configuration file "OEM" does not exi
st. Enter "1" to specify the WebSphere Application Server values, or press Enter
 to specify the file again:
cd /etc
IZUG195E: The value for variable "THERESPONSE" contains one or more spaces. Ente
r the value without spaces.

IZUG153I: Enter the directory path and name of the WebSphere Application Server
configuration file, or press enter to accept "/etc/zWebSphereOEM/V7R0/conf/CONFI
G1/CONFIG1.responseFile":

IZUG243I: Accepted input: /etc/zWebSphereOEM/V7R0/conf/CONFIG1/CONFIG1.responseF
ile
IZUG154I: The WebSphere Application Server configuration values have been read.
IZUG340I: Variable substitution entry "IZU_APPSERVER_ROOT" is being updated with
value "/zWebSphereOEM/V7R0/config1".
IZUG340I: Variable substitution entry "IZU_APPSERVER_GROUP" is being updated wit
h value "WSCFG1".
IZUG340I: Variable substitution entry "IZU_WAS_PROFILE_PREFIX" is being updated
with value "BBNBASE".
IZUG340I: Variable substitution entry "IZU_CLUSTER_TRANSITION_NAME" is being upd
ated with value "BBNC001".
IZUG340I: Variable substitution entry "IZU_CELL_SHORT_NAME" is being updated wit
h value "BBNBASE".
IZUG340I: Variable substitution entry "IZU_CONTROL_USERID" is being updated with
 value "WSCRU1".
IZUG340I: Variable substitution entry "IZU_SERVANT_USERID" is being updated with
 value "WSSRU1".
IZUG340I: Variable substitution entry "IZU_ORB_PORT" is being updated with value
 "32203".
IZUG340I: Variable substitution entry "IZU_HTTP_SSL_PORT" is being updated with
value "32208".
IZUG340I: Variable substitution entry "IZU_APPSERVER_HOSTNAME" is being updated
with value "WTSC74.ITSO.IBM.COM".

IZUG270I: Enter the path of the root WBEM directory, or press Enter to accept th
e default "/usr/lpp/wbem":

IZUG243I: Accepted input: /usr/lpp/wbem


IZUG162I: Select the plug-ins to be configured. Multiple plug-ins can be selecte
d by separating plug-ins with a comma.
IZUG163I: Select "0" to configure ALL plug-ins.
IZUG163I: Select "1" to configure Incident Log.
IZUG163I: Select "2" to configure Configuration Assistant.
IZUG163I: Select "3" to configure Workload Management.
IZUG163I: Select "4" to configure Resource Monitoring.
IZUG163I: Select "5" to configure Capacity Provisioning.
IZUG163I: Select "6" to configure Software Deployment.
IZUG163I: Select "7" to configure ISPF.
IZUG163I: Select "8" to configure DASD Management.
IZUG173I: Enter "N" to select none of these plug-ins.
IZUG164I: Which plug-ins do you want to configure?
0
```

```
IZUG165I: You have selected to configure Incident Log.
IZUG165I: You have selected to configure Configuration Assistant.
IZUG165I: You have selected to configure Workload Management.
IZUG165I: You have selected to configure Resource Monitoring.
IZUG165I: You have selected to configure Capacity Provisioning.
IZUG165I: You have selected to configure Software Deployment.
IZUG165I: You have selected to configure ISPF.
IZUG165I: You have selected to configure DASD Management.



IZUG272I: Do you want to enable the common event adapter (CEA) component and upd
ate related parmlib options for using the Incident Log task? For yes, enter Y. F
or no, enter N. Or press Enter to accept the default "Y":
Y
IZUG243I: Accepted input: Y

IZUG060I: Specify the CEA high level qualifier (HLQ) to use for log snapshot dat
a sets. The HLQ can be 1-4 characters. Or press Enter to accept the default "CEA
":

IZUG243I: Accepted input: CEA

IZUG278I: Enter the country code:
000
IZUG243I: Accepted input: 000

IZUG278I: Enter the branch code:
180
IZUG243I: Accepted input: 180

IZUG281I: What storage option do you want to use? Enter V for VOLSER or S for ST
ORCLAS.
V
IZUG243I: Accepted input: V


IZUG283I: Specify one or more of the non-SMS direct access volumes to use. When
you are finished entering the values, press Enter again without a value to compl
ete:
BH5ST4
IZUG243I: Accepted input: BH5ST4

IZUG283I: Specify one or more of the non-SMS direct access volumes to use. When
you are finished entering the values, press Enter again without a value to compl
ete:

IZUG243I: Accepted input: "VOLSER(BH5ST4)"

IZUG284I: Enter the name of the source data set for your existing CEAPRM00 parml
ib member. Specify the fully qualified data set name, or press Enter to accept t
he default "SYS1.PARMLIB":

IZUG243I: Accepted input: SYS1.PARMLIB
```

```
IZUG064I: Enter the name of the target data set to be used for saving the update
d "CEAPRM00" parmlib member. Specify the fully qualified data set name, or press
 Enter to accept the default: "SYS1.PARMLIB":

IZUG243I: Accepted input: SYS1.PARMLIB

IZUG066I: Enter the name of the source data set for the IEADMCZM parmlib member.
 Specify the fully qualified data set name, or press Enter to accept the default
 "SYS1.SAMPLIB":

IZUG243I: Accepted input: SYS1.SAMPLIB

IZUG064I: Enter the name of the target data set to be used for saving the update
d "IEADMCZM" parmlib member. Specify the fully qualified data set name, or press
Enter to accept the default: "SYS1.PARMLIB":

IZUG243I: Accepted input: SYS1.PARMLIB

IZUG275I: Enter the member name suffix to use for the "CEAPRM" parmlib member, o
r press Enter to accept the default "01":

IZUG243I: Accepted input: 01

IZUG199W: File "SYS1.PARMLIB(CEAPRM01)" already exists.


IZUG275I: Enter the member name suffix to use for the "IEADMC" parmlib member, o
r press Enter to accept the default "ZM":

IZUG243I: Accepted input: ZM

IZUG199W: File "SYS1.PARMLIB(IEADMCZM)" already exists.


IZUG166I: No configuration prompts are required for the plug-in Configuration As
sistant.

IZUG047I: Enter the existing Workload Management group name that is used to auth
orize users to the Workload Management resources. Press Enter to accept the defa
ult "WLMGRP", or enter "NO.KNOWN.VALUE" if no group exists.
NO.KNOWN.VALUE
IZUG243I: Accepted input: NO.KNOWN.VALUE

IZUG166I: No configuration prompts are required for the plug-in Resource Monitor
ing.

IZUG047I: Enter the existing Capacity Provisioning query group name that is used
 to authorize users to the Capacity Provisioning resources. Press Enter to accep
t the default "CPOQUERY", or enter "NO.KNOWN.VALUE" if no group exists.
NO.KNOWN.VALUE
IZUG243I: Accepted input: NO.KNOWN.VALUE

IZUG047I: Enter the existing Capacity Provisioning control group name that is us
ed to authorize users to the Capacity Provisioning resources. Press Enter to acc
ept the default "CPOCTRL", or enter "NO.KNOWN.VALUE" if no group exists.
```

```
NO.KNOWN.VALUE
IZUG243I: Accepted input: NO.KNOWN.VALUE

IZUG166I: No configuration prompts are required for the plug-in Software Deploym
ent.

IZUG166I: No configuration prompts are required for the plug-in ISPF.


IZUG274I: Enter the z/OSMF storage group name, or press Enter to accept "IZUSTGA
":

IZUG243I: Accepted input: IZUSTGA

IZUG069I: The configuration property "IZU_STORAGE_GROUP_GID" is set to the value
 "AUTOGID".

IZUG047I: Enter the existing Common Information Model admin group name that is u
sed to authorize users to the Common Information Model resources. Press Enter to
 accept the default "CFZADMGP", or enter "NO.KNOWN.VALUE" if no group exists.

IZUG243I: Accepted input: CFZADMGP

IZUG047I: Enter the existing Common Information Model user group name that is us
ed to authorize users to the Common Information Model resources. Press Enter to
accept the default "CFZUSRGP", or enter "NO.KNOWN.VALUE" if no group exists.

IZUG243I: Accepted input: CFZUSRGP

IZUG224I: The configuration data was saved in file "/usr/lpp/zosmf/V1R13/default
s/izudflt.cfg".

IZUG146I: Invoking script "/usr/lpp/zosmf/V1R13/bin/izuracf.sh".

IZUG215I: Starting z/OSMF "RACF processing" procedure.

Wed Apr 20 16:02:15 EDT 2011

IZUG287I: z/OSMF RACF "exec generation" processing complete. Review and run "/et
c/zosmf/izudflt.cfg.rexx" before proceeding with configuration.

IZUG095I: The Common Information Model (CIM) server must be configured and start
ed before proceeding with configuration.

IZUG210I: The script "/usr/lpp/zosmf/V1R13/bin/izuracf.sh" has completed.
IZUG210I: The script "./izusetup.sh" has completed.
LAFITTE @ SC74:/Z1DRB1/usr/lpp/zosmf/V1R13/bin>
```

New with z/OSMF V1R13, the common event adapter (CEA) component of z/OS has security profiles for protecting portions of its processing. If your installation has selected to configure the Incident Log plug-in, be aware that this exec will provide CEA group access to CEA.CEAPDWB* in the SERVAUTH class.

Also regarding CIM and z/OSMF V1R13, verify that the z/OSMF administrator profile is properly set up for the z/OS UNIX shell environment. By default, the file `profile.add`, which is

shipped with the CIM server, provides the environment variables that you need to define for the administrator; see /usr/lpp/wbem/install/ profile.add. If you are not using the defaults, modify the appropriate settings. Copy the contents of the profile.add file to the.profile file in the home directory of the z/OSMF administrator user ID. The z/OSMF administrator is to own the .profile file and have read-write-execute access.

After checking how much of the /etc/zosmf/izudflt.cfg.rexx script is suitable to your environment, you can run it, as partially shown in Example A-5 on page 792.

Example A-5 /etc/zosmf/izudflt.cfg.rexx

```
/etc/zosmf/izudflt.cfg.rexx
Invoking: ADDGROUP IZUADMIN OMVS(AUTOGID)
Invoking: ADDGROUP IZUUSER OMVS(AUTOGID)
..........................
Invoking: PERMIT BBNBASE.izuUsers CLASS(EJBROLE) ID(IZUUSER) ACCESS(READ)
RACLISTED PROFILES FOR EJBROLE  WILL NOT REFLECT THE UPDATE(S) UNTIL A SETROPTS
REFRESH IS ISSUED
Invoking: SETROPTS RACLIST(EJBROLE) REFRESH
SETROPTS command complete.
Invoking: PERMIT BBO.SYNC.BBNBASE.BBNCOO1 CLASS(FACILITY) ID(WSCRU1) ACC(CONTROL
)
RACLISTED PROFILES FOR FACILITY WILL NOT REFLECT THE UPDATE(S) UNTIL A SETROPTS
REFRESH IS ISSUED
Invoking: SETROPTS RACLIST(FACILITY) REFRESH
SETROPTS command complete.
Invoking: SETROPTS CLASSACT(ZMFAPLA) GENERIC(ZMFAPLA) RACLIST(ZMFAPLA)
SETROPTS command complete.
Invoking: RDEFINE ZMFAPLA BBNBASE.ZOSMF.** UACC(NONE)
RACLISTED PROFILES FOR ZMFAPLA WILL NOT REFLECT THE ADDITION(S) UNTIL A SETROPTS
REFRESH IS ISSUED.
Invoking: PERMIT BBNBASE.ZOSMF.** CLASS(ZMFAPLA) ID(IZUADMIN) ACCESS(READ)
RACLISTED PROFILES FOR ZMFAPLA  WILL NOT REFLECT THE UPDATE(S) UNTIL A SETROPTS
REFRESH IS ISSUED
Invoking: PERMIT BBNBASE.ZOSMF.** CLASS(ZMFAPLA) ID(IZUUSER) ACCESS(READ)
RACLISTED PROFILES FOR ZMFAPLA  WILL NOT REFLECT THE UPDATE(S) UNTIL A SETROPTS
REFRESH IS ISSUED
Invoking: RDEFINE ZMFAPLA BBNBASE.ZOSMF.ADMINTASKS.** UACC(NONE)
RACLISTED PROFILES FOR ZMFAPLA WILL NOT REFLECT THE ADDITION(S) UNTIL A SETROPTS
 REFRESH IS ISSUED.
Invoking: PERMIT BBNBASE.ZOSMF.ADMINTASKS.** CLASS(ZMFAPLA) ID(IZUADMIN) ACCESS(
RACLIST REFRESH of class APPL ignored. The class is not active yet.
SETROPTS command complete.
Invoking: PERMIT BBNBASE.izuUsers CLASS(EJBROLE) ID(IZUSTGA) ACCESS(READ)
RACLISTED PROFILES FOR EJBROLE  WILL NOT REFLECT THE UPDATE(S) UNTIL A SETROPTS
REFRESH IS ISSUED
Invoking: SETROPTS RACLIST(EJBROLE) REFRESH
SETROPTS command complete.
Invoking: PERMIT BBNBASE.ZOSMF.** CLASS(ZMFAPLA) ID(IZUSTGA) ACCESS(READ)
RACLISTED PROFILES FOR ZMFAPLA  WILL NOT REFLECT THE UPDATE(S) UNTIL A SETROPTS
REFRESH IS ISSUED
Invoking: RDEFINE ZMFAPLA BBNBASE.ZOSMF.DASD_MANAGEMENT.** UACC(NONE)
RACLISTED PROFILES FOR ZMFAPLA WILL NOT REFLECT THE ADDITION(S) UNTIL A SETROPTS
 REFRESH IS ISSUED.
Invoking: PERMIT BBNBASE.ZOSMF.DASD_MANAGEMENT.** CLASS(ZMFAPLA) ID(IZUSTGA) ACC
ESS(READ)
```

```
RACLISTED PROFILES FOR ZMFAPLA  WILL NOT REFLECT THE UPDATE(S) UNTIL A SETROPTS
REFRESH IS ISSUED
Invoking: SETROPTS RACLIST(ZMFAPLA) REFRESH
SETROPTS command complete.
```

Next, run the `izudflt.cfg -prime`, shown in Example A-6, to set up the prime environment.

Example A-6 `izudflt.cfg -prime`

```
LAFITTE @ SC74:/u/lafitte>izusetup.sh -file /Z1DRB1/usr/lpp/zosmf/V1R13/defaults
/izudflt.cfg -prime
izusetup.sh -file /Z1DRB1/usr/lpp/zosmf/V1R13/defaults/izudflt.cfg -prime
+ izusetup.sh -file /Z1DRB1/usr/lpp/zosmf/V1R13/defaults/izudflt.cfg -prime

IZUG091I: Environment variable "IZU_CODE_ROOT" is set to the value "/Z1DRB1/usr/
lpp/zosmf/V1R13".

IZUG091I: Environment variable "IZU_CONFIG_DIR" is set to the value "/etc/zosmf"
.

IZUG091I: Environment variable "IZU_LOGFILE_DIR" is set to the value "/var/zosmf
/configuration/logs".

IZUG213I: Log information will be written to file "/var/zosmf/configuration/logs
/izusetup_prime.04.26.11.08.20.01.log".

IZUG215I: Starting z/OSMF "setup" procedure.

Tue Apr 26 08:20:01 EDT 2011



IZUG099W: File "/zWebSphereOEM/V7RO/config1/AppServer/profiles/default/propertie
s/izu.config.properties" does not exist.
IZUG227I: Creating file "/zWebSphereOEM/V7RO/config1/AppServer/profiles/default/
properties/izu.config.properties".

IZUG031I: The "izuadmin.env" file will be used from the following location: "/Z1
DRB1/usr/lpp/zosmf/V1R13/defaults/izuadmin.env"


IZUG227I: Creating directory "/var/zosmf/data".
IZUG250I: The z/OSMF data file system "IZU.SIZUDATA" has a "primary" allocation
size of "180" cylinders.
IZUG250I: The z/OSMF data file system "IZU.SIZUDATA" has a "secondary" allocatio
n size of "20" cylinders.

IZUG251I: Allocating z/OSMF data file system "IZU.SIZUDATA".

IZUG252I: Mounting "IZU.SIZUDATA" at "/var/zosmf/data".
IZUG146I: Invoking script "/Z1DRB1/usr/lpp/zosmf/V1R13/bin/izuprime.sh -prime ZO
SMFAD -datadir /var/zosmf/data".
IZUG300I: Processing of script "izuprime.sh" has started at "Tue Apr 26 08:20:10
 EDT 2011".
IZUG306I: Script "izuprime.sh" was invoked with options "-prime ZOSMFAD -datadir
 /var/zosmf/data".
```

```
IZUG308I: Processing of script "izuprime.sh" has completed at "Tue Apr 26 08:20:
10 EDT 2011".
IZUG053W: The owner assigned to directory "/var/zosmf/data" will be changed to "
WSSRU1".
IZUG052W: The group assigned to directory "/var/zosmf/data" will be changed to "
WSCFG1".
IZUG051W: The permissions assigned to directory "/var/zosmf/data" will be change
d to "rwxr-x---".
IZUG227I: Creating directory "/var/zosmf/data/logs".
IZUG053W: The owner assigned to directory "/var/zosmf/configuration/logs" will b
e changed to "ZOSMFAD".
IZUG052W: The group assigned to directory "/var/zosmf/configuration/logs" will b
e changed to "IZUADMIN".
IZUG053W: The owner assigned to directory "/etc/zosmf" will be changed to "ZOSMF
IZUG053W: The owner assigned to directory "/etc/zosmf" will be changed to "ZOSMF
AD".
IZUG052W: The group assigned to directory "/etc/zosmf" will be changed to "IZUAD
MIN".
IZUG051W: The permissions assigned to directory "/etc/zosmf" will be changed to
"rwxrwxr-x".
IZUG052W: The group assigned to directory "/u/zosmfad" will be changed to "IZUAD
MIN".
IZUG227I: Creating directory "/var/zosmf/data/importutil".
IZUG227I: Creating directory "/etc/zosmf/helps/eclipse/plugins".
IZUG146I: Invoking script "/Z1DRB1/usr/lpp/zosmf/V1R13/bin/izuincidentlogconfig.
sh".

IZUG215I: Starting z/OSMF "Incident Log configuration" procedure.

Tue Apr 26 08:20:11 EDT 2011




IZUG239W: File name "SYS1.PARMLIB(CEAPRM01)" already exists: Overwrite (Y|N)?
Y

IZUG239W: File name "SYS1.PARMLIB(IEADMCZM)" already exists: Overwrite (Y|N)?
Y
IZUG210I: The script "/Z1DRB1/usr/lpp/zosmf/V1R13/bin/izusetup.sh" has completed
.
```

Then run the `izudflt.cfg -finish` script, shown in Example A-7, to finish the setup.

Example A-7 `izudflt.cfg -finish`

```
ZOSMFAD @ SC74:/u/zosmfad>izusetup.sh -file /Z1DRB1/usr/lpp/zosmf/V1R13/defaults
/izudflt.cfg -finish
izusetup.sh -file /Z1DRB1/usr/lpp/zosmf/V1R13/defaults/izudflt.cfg -finish
+ izusetup.sh -file /Z1DRB1/usr/lpp/zosmf/V1R13/defaults/izudflt.cfg -finish

IZUG091I: Environment variable "IZU_CODE_ROOT" is set to the value "/Z1DRB1/usr/
lpp/zosmf/V1R13".

IZUG091I: Environment variable "IZU_CONFIG_DIR" is set to the value "/etc/zosmf"
.
```

```
IZUG091I: Environment variable "IZU_LOGFILE_DIR" is set to the value "/var/zosmf
/configuration/logs".
IZUG213I: Log information will be written to file "/var/zosmf/configuration/logs
/izusetup_finish.04.26.11.08.45.21.log".

IZUG215I: Starting z/OSMF "setup" procedure.

Tue Apr 26 08:45:21 EDT 2011



IZUG146I: Invoking script "/Z1DRB1/usr/lpp/zosmf/V1R13/bin/izuincidentlogconfig.
sh".

IZUG215I: Starting z/OSMF "Incident Log configuration" procedure.

Tue Apr 26 08:45:23 EDT 2011



IZUG124I: The Common Event Adapter (CEA) parmlib member "CEAPRM01" is being acti
vated.
IZUG146I: Invoking script "/usr/lpp/zosmf/V1R13/bin/izuadmin.sh -deployBase".
IZUG300I: Processing of script "izuadmin.sh" has started at "Tue Apr 26 11:35:00
 EDT 2011".
IZUG306I: Script "izuadmin.sh" was invoked with options "-deployBase".
IZUG348I: Processing of your request has started. This process might require sev
eral minutes or more to complete.
IZUG308I: Processing of script "izuadmin.sh" has completed at "Tue Apr 26 11:43:
53 EDT 2011".
IZUG146I: Invoking script "/usr/lpp/zosmf/V1R13/bin/izuadmin.sh -verifyBase".
IZUG300I: Processing of script "izuadmin.sh" has started at "Tue Apr 26 11:43:53
 EDT 2011".
IZUG306I: Script "izuadmin.sh" was invoked with options "-verifyBase".
IZUG348I: Processing of your request has started. This process might require sev
eral minutes or more to complete.
IZUG308I: Processing of script "izuadmin.sh" has completed at "Tue Apr 26 11:46:
19 EDT 2011".
IZUG083I: The verification of "core" has completed successfully.
IZUG146I: Invoking script "/usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh".

IZUG215I: Starting z/OSMF "Verify incident log processing" procedure.

Tue Apr 26 11:46:19 EDT 2011



IZUG296I: Verification process "/usr/lpp/wbem/bin/cimivp" has completed.

IZUG297I: Provider "IBMzOS_PDW_IVP" is already registered with Common Informatio
n Model (CIM).

IZUG104I: Provider "IBMzOS_PDW_IVP" module has already been registered with the
```

```
Common Information Model (CIM) server.

IZUG297I: Provider "IBMzOS_PDWLogstream" is already registered with Common Infor
mation Model (CIM).
the Common Information Model (CIM) server.

IZUG297I: Provider "IBMzOS_SysplexDumpDirectory" is already registered with Comm
on Information Model (CIM).

IZUG104I: Provider "IBMzOS_SysplexDumpDirectory" module has already been registe
red with the Common Information Model (CIM) server.

IZUG117I: A creation of the test incident for the Incident Log has occurred.

IZUG118I: Checking Incident Log dependencies.

IZUG117I: A deletion of the test incident for the Incident Log has occurred.

IZUG119I: Obtaining data for dependency sysplex dump directory.

IZUG119I: Obtaining data for dependency log stream for the Incident Log.

IZUG120I: Creating Incident Log report "/var/zosmf/configuration/logs/izuinciden
tlogverify.report".
sed: /usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh 621: FSUM9209 cannot execu
te: reason code = 0b250012: EDC5112I Resource temporarily unavailable.
sed: /usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh 622: FSUM9209 cannot execu
te: reason code = 0b250012: EDC5112I Resource temporarily unavailable.
sed: /usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh 623: FSUM9209 cannot execu
te: reason code = 0b250012: EDC5112I Resource temporarily unavailable.
sed: /usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh 624: FSUM9209 cannot execu
te: reason code = 0b250012: EDC5112I Resource temporarily unavailable.
sed: /usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh 625: FSUM9209 cannot execu
te: reason code = 0b250012: EDC5112I Resource temporarily unavailable.
sed: /usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh 626: FSUM9209 cannot execu
te: reason code = 0b250012: EDC5112I Resource temporarily unavailable.
sed: /usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh 627: FSUM9209 cannot execu
te: reason code = 0b250012: EDC5112I Resource temporarily unavailable.
sed: /usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh 628: FSUM9209 cannot execu
te: reason code = 0b250012: EDC5112I Resource temporarily unavailable.
sed: /usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh 629: FSUM9209 cannot execu
te: reason code = 0b250012: EDC5112I Resource temporarily unavailable.
sed: /usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh 630: FSUM9209 cannot execu
te: reason code = 0b250012: EDC5112I Resource temporarily unavailable.
sed: /usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh 631: FSUM9209 cannot execu
te: reason code = 0b250012: EDC5112I Resource temporarily unavailable.
sed: /usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh 632: FSUM9209 cannot execu
te: reason code = 0b250012: EDC5112I Resource temporarily unavailable.
sed: /usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh 633: FSUM9209 cannot execu
te: reason code = 0b250012: EDC5112I Resource temporarily unavailable.
sed: /usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh 634: FSUM9209 cannot execu
te: reason code = 0b250012: EDC5112I Resource temporarily unavailable.
sed: /usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh 635: FSUM9209 cannot execu
te: reason code = 0b250012: EDC5112I Resource temporarily unavailable.
sed: /usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh 636: FSUM9209 cannot execu
```

```
te: reason code = 0b250012: EDC5112I Resource temporarily unavailable.
sed: /usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh 637: FSUM9209 cannot execu
te: reason code = 0b250012: EDC5112I Resource temporarily unavailable.
sed: /usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh 638: FSUM9209 cannot execu
te: reason code = 0b250012: EDC5112I Resource temporarily unavailable.
sed: /usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh 639: FSUM9209 cannot execu
te: reason code = 0b250012: EDC5112I Resource temporarily unavailable.
sed: /usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh 640: FSUM9209 cannot execu
te: reason code = 0b250012: EDC5112I Resource temporarily unavailable.
sed: /usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh 641: FSUM9209 cannot execu
te: reason code = 0b250012: EDC5112I Resource temporarily unavailable.
sed: /usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh 642: FSUM9209 cannot execu
te: reason code = 0b250012: EDC5112I Resource temporarily unavailable.
sed: /usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh 643: FSUM9209 cannot execu
te: reason code = 0b250012: EDC5112I Resource temporarily unavailable.
sed: /usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh 643: FSUM9209 cannot execu
te: reason code = 0b250012: EDC5112I Resource temporarily unavailable.

IZUG121I: To obtain the results of the Incident Log verification, review report
"/var/zosmf/configuration/logs/izuincidentlogverify.report".
IZUG083I: The verification of "Incident Log" has completed successfully.

IZUG210I: The script "/usr/lpp/zosmf/V1R13/bin/izuincidentlogverify.sh" has comp
leted.
IZUG349I: The "z/OSMF Welcome panel" can be accessed at "https://WTSC74.ITSO.IBM
.COM:32208/zosmf" after the application server is started on your system.
IZUG210I: The script "/usr/lpp/zosmf/V1R13/bin/izusetup.sh" has completed.
ZOSMFAD @ SC74:/u/zosmfad>
```

Run the `izudflt.cfg -userid` script, shown in Example A-8, to define every RACF user to z/OSMF.

Example A-8 `izudflt.cfg -userid`

```
LAFITTE @ SC74:/Z1DRC1/usr/lpp/zosmf/V1R13/bin>izuauthuser.sh -file /Z1DRC1/usr/
lpp/zosmf/V1R13/defaults/izudflt.cfg -userid LAFITTE
izuauthuser.sh -file /Z1DRC1/usr/lpp/zosmf/V1R13/defaults/izudflt.cfg -userid LA
FITTE
+ izuauthuser.sh -file /Z1DRC1/usr/lpp/zosmf/V1R13/defaults/izudflt.cfg -userid
LAFITTE

IZUG091I: Environment variable "IZU_CODE_ROOT" is set to the value "/Z1DRC1/usr/
lpp/zosmf/V1R13".

IZUG091I: Environment variable "IZU_CONFIG_DIR" is set to the value "/etc/zosmf"
.

IZUG091I: Environment variable "IZU_LOGFILE_DIR" is set to the value "/var/zosmf
/configuration/logs".

IZUG213I: Log information will be written to file "/var/zosmf/configuration/logs
/izuauthuser.04.27.11.11.13.35.log".

IZUG215I: Starting z/OSMF "RACF authorize user" procedure.

Wed Apr 27 11:13:35 EDT 2011
```

```
IZUG287I: z/OSMF RACF "authorize user" processing complete. Review and run "/etc
/zosmf/izudflt.cfg.LAFITTE.rexx" before proceeding with configuration.
IZUG210I: The script "./izuauthuser.sh" has completed.
LAFITTE @ SC74:/Z1DRC1/usr/lpp/zosmf/V1R13/bin>
```

Finally, run the `izudflt.cfg.userid.rexx` script, shown in Example A-9, to polish the security definition of every RACF user ID already defined to z/OSMF.

Example A-9 `izudflt.cfg.userid.rexx`

```
LAFITTE @ SC74:/Z1DRC1/usr/lpp/zosmf/V1R13/bin>/etc/zosmf/izudflt.cfg.LAFITTE.re
xx
/etc/zosmf/izudflt.cfg.LAFITTE.rexx
+ /etc/zosmf/izudflt.cfg.LAFITTE.rexx
Invoking: CONNECT LAFITTE GROUP(IZUUSER)
Invoking: SETROPTS RACLIST(APPL) REFRESH
RACLIST REFRESH of class APPL ignored. The class is not active yet.
SETROPTS command complete.
Invoking: SETROPTS RACLIST(EJBROLE) REFRESH
SETROPTS command complete.
Invoking: CONNECT (LAFITTE) GROUP(CFZUSRGP)
LAFITTE @ SC74:/Z1DRC1/usr/lpp/zosmf/V1R13/bin>
```

A.3 Setting up CIM

To use the Incident Log task and the Workload Management task of z/OSMF, CIM has to be set up and running.

Follow these steps to set up and run CIM:

1. Run script `CFZRCUST` (Example A-10).
2. Run script `cimserv.env` to set up the environment variables for CIM (Example A-12).
3. Start CIM (Example A-16).
4. Run minor and final setup scripts for z/OS V1R13 (Example A-17 and Example A-18).

Script CFZRCUST should be properly run; Example A-10 shows its output for our environment.

Example A-10 CFZRCUST

```
Start script processing...
Check for space on file system for /var/wbem is enabled.
** Warning, the data directory /var/wbem will not be placed on a seperate file s
The time stamp used for temporary files and directories is: 20110422122339
Use existing /etc/wbem.
Nothing to do for /etc/wbem/cimserver.env.
Back up old repository /var/wbem/repository to file /var/wbem/PreviousRepos20110
Rename /var/wbem/repository to /var/wbem/repository.20110422122339.
Copy new repository to target system path /var/wbem.
Running repository migration from /var/wbem/repository.20110422122339 to /var/wb
Repository successfully migrated.
Previous repository is stored in /var/wbem/PreviousRepos20110422122339.pax.z.
Copying repository_status file.
```

```
Script processing ended.
```

To doublecheck the validity of the CIM setup you can try to run the ivp, which uses ASCII characters to convey its messages; see Example A-11.

Example A-11 Output of doublecheck

```
LAFITTE @ SC74:/u/lafitte>/usr/lpp/wbem/bin/cimivp
/usr/lpp/wbem/bin/cimivp
+ /usr/lpp/wbem/bin/cimivp
äáá----ë-èÇÁ-_?ÀÍ%Á-%ÑÂøÁÅÄ?__?>-Ë?-Ï/Ë->?È-Ã?Í>À-----------èÇÁ-ÈÊ/ÄÁÂ/Ä,-Ñ>Ã?Ê_
/ÈÑ?>-Ä?Í%À->?È-ÂÁ-ÀÁÈÁÊ_Ñ>ÁÀ--Ý1¨ + Done(137) /usr/lpp/wbem/bin/cimivp
  83951721      Killed  /usr/lpp/wbem/bin/cimivp
```

Run the `cimserver.env` script to set up the CIM environment variables; see Example A-12.

Example A-12 cimserver.env

```
PEGASUS_HOME=/usr/lpp/wbem
LIBPATH=/usr/lpp/wbem/lib:/usr/lpp/wbem/provider:/usr/lib
_CEE_RUNOPTS=FILETAG(AUTOCVT,AUTOTAG) STACK(32K,32K,ANYWHERE,KEEP,96K,32K) THREA
_BPX_SHAREAS=NO
_BPXK_AUTOCVT=ON
_TAG_REDIR_ERR=TXT
_TAG_REDIR_IN=TXT
_TAG_REDIR_OUT=TXT
#OSBASE_TRACE=0
#OSBASE_TRACE_FILE=/tmp/wbemosbase.trc
#RMF_CIM_HOST=127.0.0.1
#RMF_CIM_PORT=8803
#RMF_CIM_TRACE=0
#RMF_CIM_TRACE_FILE=/tmp/wbemosmonitoring.trc
```

Because of its `rwx` settings, as shown in Example A-14, you cannot execute or easily chgmod the variables (see Example A-13).

Example A-13 Permission denied message

```
LAFITTE @ SC74:/u/lafitte> /usr/lpp/wbem/cimserver.env
 /usr/lpp/wbem/cimserver.env
+ /usr/lpp/wbem/cimserver.env
/usr/lpp/wbem/cimserver.env: FSUM9209 cannot execute: reason code = ef076015: ED
C5111I Permission denied.
```

Example A-14 rwx settings

```
-rw-r--r--   1 SYSPROG  OMVSGRP     4930 Dec  2 02:12 cimserver.env
```

For this reason, the ASCII-to-EBCDIC conversion must be performed manually, as shown in Example A-15, to switch on the proper conversion.

Example A-15 ASCII-to-EBCDIC conversion

```
LAFITTE @ SC74:/u/lafitte>_BPXK_AUTOCVT=ON
_BPXK_AUTOCVT=ON
```

```
+ _BPXK_AUTOCVT=ON
```

At this point you can start CIM as shown; see Example A-16.

Example A-16 Starting CIM

```
S CFZCIM
 $HASP100 CFZCIM   ON STCINRDR
 IEF695I START CFZCIM   WITH JOBNAME CFZCIM   IS ASSIGNED TO USER
 CFZSRV  , GROUP CFZSRVGP
 $HASP373 CFZCIM   STARTED
 IRR812I PROFILE ** (G) IN THE STARTED CLASS WAS USED 861
         TO START BPXAS WITH JOBNAME BPXAS.
 $HASP100 BPXAS    ON STCINRDR
 $HASP373 BPXAS    STARTED
 BPXPO24I BPXAS INITIATOR STARTED ON BEHALF OF JOB CFZCIM1 RUNNING IN
 ASID 004F
CFZ12580I: CIM server running eligible for zIIP.
CFZ10025I: The CIM server is listening on HTTP port 5988.
CFZ10028I: The CIM server is listening on the local connection socket.
CFZ10030I: Started CIM Server version 2.11.0 Development.
CFZ12532I: The CIM server successfully registered to ARM using
element name CFZ_SRV_SC74    .
```

With z/OS V1R13, further setup is needed, as explained in this section. The security definitions shown in Example A-17 are required for the CIM Server started task user (the default is CFZSRV) to configure Indication support for the SNIA storage HBA and multipath management profiles.

Example A-17 Permit commands for HBA and multipath management

```
PERMIT CEA.CONNECT CLASS(SERVAUTH) ID(CFZSRV) ACCESS(READ)
 PERMIT CEA.SUBSCRIBE.ENF_0009* CLASS(SERVAUTH) ID(CFZSRV) ACCESS(READ)
 PERMIT CEA.SUBSCRIBE.ENF_0027* CLASS(SERVAUTH) ID(CFZSRV) ACCESS(READ)
 PERMIT CEA.SUBSCRIBE.ENF_0033* CLASS(SERVAUTH) ID(CFZSRV) ACCESS(READ)
 PERMIT IOSCDR CLASS(FACILITY) ID(CFZSRV) ACCESS(UPDATE)
```

If this is not done, the error messages shown in Example A-18 are issued by the CIM server.

Example A-18 Error messages

```
CEZ03010E User CFZSRV not authorized to connect to Common Event
 Adapter (CEA)

CEZ03011E User CFZSRV not authorized for subscription to Common Event
 Adapter (CEA)

CEZ03000E Request user ID CFZSRV requires UPDATE permission on profile
 IOSCDR CL(FACILITY)
```

A.4 Setting up Capacity Provisioning

Before you can successfully start the Provisioning Manager, prerequisites must be satisfied. These prerequisites include the runtime system and the systems that are observed by the

Provisioning Manager. The runtime system may also be one of the observed systems. On the runtime system, follow these steps:

1. Define data sets used for the runtime data.
2. Set the configuration parameters to your chosen values.
3. Create a started task procedure.
4. Provide APF authorization.
5. Define the security.
6. Define a restart policy.

A.4.1 Defining runtime data sets

The Provisioning Manager stores permanent and temporary data in data sets. You simply need to define these data sets one time for a domain. The data sets must be accessible on all runtime systems.

After you have created the data sets, copy two sample files from the Capacity Provisioning installation file system into the data sets for the Provisioning Manager parameters. These are the files `env` and `parm` from directory `/usr/lpp/cpo/samples`. Copy these as members ENV and PARM, respectively.

Capacity Provisioning provides a sample job for defining these data sets and copying the files. The sample job is available as member CPOMKDSN in library SYS1.SAMPLIB.

Important: This job will delete existing data sets with the same names as those to be defined.

A.4.2 Adapting Provisioning Manager parameters

Certain parameters of the Provisioning Manager might need to be adapted to your environment. These parameters are held in the Provisioning Manager parameters data set, `prefix.PARM`, in the members ENV for the Provisioning Manager runtime environment data and PARM for the Provisioning Manager configuration information.

The ENV member contains information about the runtime processing environment for your Provisioning Manager. Modify the following paths to match your installation settings

- LIBPATH: This entry must contain all of the following:
  - The path /usr/lib for SAF libraries
  - The Java installation paths `/usr/lpp/java/J6.0/bin` and `usr/lpp/java/J6.0/bin/classic`
  - The Capacity Provisioning installation path `/usr/lpp/cpo/lib`
- CLASSPATH: This entry must contain the following:
  - The Capacity Provisioning JAR file `cpom.jar` from the installation directory/usr/lpp/cpo/classes
  - The SAF JAR file `/usr/include/java_classes/IRRRacf.jar`
  - Optionally, if your Capacity Provisioning installation directory or your CIM installation directory are not at the default locations, you also need to add both the CIM Clients for Java:

- /usr/lpp/wbem/jclient/sblim-cim-client2.jar
- /usr/lpp/wbem/jclient/sblimCIMClient.jar

The PARM member contains configuration information for the Provisioning Manager. It has the structure of a Java property-file with keyword-value pairs. Both keywords and values are case-sensitive. Comment lines may be included, starting with a hash character (#).

In the PARM member you can enter keywords for the following:

- Hardware access

  See “Capacity Provisioning” on page 611 for more information.

- Automatic Restart Manager (ARM) setting

  To use ARM to monitor availability of the Provisioning Manager you must set the value ARM.Register to Yes. You can do so by removing the comment symbol from this statement in the sample member provided.

  If this key is not specified, or if it is given a value other than Yes, the Provisioning Manager will not be registered with ARM. The value is not case-sensitive.

  If you use ARM to monitor the Provisioning Manager, then you must define an ARM policy. This policy specifies an ARM element type and an ARM element name. If you have chosen the default element type and name, SYSCPM and SYSCPO, no changes are needed. However, if you have changed these values, you must replace the values of the keys ARM.ElementType and ARM.ElementName with those you have chosen. The defaults are:

  – # ARM settings
  – ARM.Register = No
  – ARM.ElementType = SYSCPM
  – ARM.ElementName = SYSCPO

- Security groups for Control Center commands authorization

  To allow the Control Center user to communicate with the Provisioning Manager, you must define the Provisioning Manager query security group and the Provisioning Manager control security group in the configuration keys CIM.ReadGroup and CIM.ModifyGroup.

  The defaults for these are:

  – # Command authorization definitions
  – CIM.ReadGroup=CPOQUERY
  – CIM.ModifyGroup=CPOCTRL

- Trace and Log data

  If you have chosen directories other than the defaults for the trace and log data, then you must set configuration keys Trace.Path and Log.Path accordingly.

  The directories must already exist and the Provisioning Manager must have write access to them. The default entries are:

  – # Service data location
  – Trace.Path = /tmp
  – Log.Path = /tmp

- Additional parameters to control provisioning management

In addition to the configuration parameters described here, the PARM member can also contain optional directives that influence the operation of the

Provisioning Manager. When these values are specified, they override the default values of the Provisioning Manager. It is best to specify a value only if you need to override the default.

A.4.3 Creating the started task procedure

You can create the started task procedure by copying the member CPOSERV from data set SYS1.SAMPLIB to the started task procedure data set. This is normally SYS1.PROCLIB. Example A-19 displays the CPOSERV procedure.

If you have not chosen to use the default name for the started task, CPOSERV, you must rename the member appropriately. In the header section of the procedure, change the following values to those you have chosen:

HLQ The high-level qualifier of the runtime data sets

DOMAIN The name of the domain

CPODIR The home directory of the Provisioning Manager user

OUTCLS A suitable output class

RGNSIZE A region size allowed by the server

Example A-19 CPOSERV procedure

```
//*****************************************************
//* Licensed Materials - Property of IBM
//* 5694-A01
//* Copyright IBM Corp. 2007
//* Status = HPV7740
//*****************************************************
//CPOSERV  PROC PMODE='*',
//              POLICY='*'
//* This section of variables may require customization
//HLQ      SET HLQ=CPO               HLQ of runtime datasets
//DOMAIN   SET DOMAIN=DOMAIN1        provisioning domain name
//CPODIR   SET CPODIR='/u/cposrv'    home directory of cposrv
//OUTCLS   SET OUTCLS=T              output class
//RGNSIZE  SET RGNSIZE=175M          server region size
//*
//CPOPOL   SET CPOPOL=&HLQ..&DOMAIN..POLICIES
//CPOCFG   SET CPOCFG=&HLQ..&DOMAIN..DOMCFG
//CPOPARM  SET CPOPARM=&HLQ..&DOMAIN..PARM
//CPORES   SET CPORES=&HLQ..&DOMAIN..RESTART
//DUMPDSN  SET DUMPDSN=&HLQ..&DOMAIN..CPOSERV.DUMP.D&YYMMDD..T&HHMMSS
//*
//*****************************************************
//* Copy environment to SYSOUT
//*****************************************************
//COPYENV   EXEC PGM=IKJEFT01,REGION=2M,
//          PARM='OCOPY INDD(INPUTENV) OUTDD(ENVFILE)'
//SYSTSIN   DD DUMMY
//SYSTSPRT  DD DUMMY
//INPUTENV  DD DISP=SHR,DSN=&CPOPARM(ENV)
//ENVFILE   DD SYSOUT=&OUTCLS.,DCB=(RECFM=V,LRECL=512)
//*****************************************************
//* Run Capacity Provisioning Manager
```

```
//*****************************************************
//CPOSERVR EXEC PGM=CPOJLNCH,REGION=&RGNSIZE.,TIME=NOLIMIT,
//             PARM='&DOMAIN &POLICY &PMODE'
//*
//STDOUT   DD  PATH='&CPODIR/stdout',PATHOPTS=(OCREAT,OWRONLY,OTRUNC),
//           PATHMODE=(SIRWXU,SIRWXG)
//*
//STDERR   DD  PATH='&CPODIR/stderr',PATHOPTS=(OCREAT,OWRONLY,OTRUNC),
//           PATHMODE=(SIRWXU,SIRWXG)
//*
//STDENV   DD DISP=SHR,DSN=&CPOPARM(ENV)
//CPOPARM  DD DISP=SHR,DSN=&CPOPARM
//CPOPOL   DD DISP=SHR,DSN=&CPOPOL
//CPOCFG   DD DISP=SHR,DSN=&CPOCFG
//CPORES   DD DISP=OLD,DSN=&CPORES
//SYSDUMP  DD SYSOUT=&OUTCLS.
//SYSOUT   DD SYSOUT=&OUTCLS.
//SYSPRINT DD SYSOUT=&OUTCLS.
//SYSMDUMP DD DSN=&DUMPDSN,DISP=(NEW,DELETE,CATLG),
//           SPACE=(CYL,(100,300),RLSE),UNIT=SYSALLDA,
//           DCB=(DSORG=PS,RECFM=FBS,LRECL=4160,BLKSIZE=24960)
//*CEEDUMP  DD SYSOUT=&OUTCLS.
//*****************************************************
//* Copy stdout to SYSOUT
//*****************************************************
//COPYOUT   EXEC PGM=IKJEFT01,COND=EVEN,
//          PARM='OCOPY INDD(INPUTOUT) OUTDD(STDOUT)'
//SYSTSPRT  DD DUMMY
//INPUTOUT  DD PATH='&CPODIR/stdout',
//             PATHOPTS=(ORDONLY)
//STDOUT    DD SYSOUT=&OUTCLS.,DCB=(RECFM=V,LRECL=512)
//SYSTSIN   DD DUMMY
//*****************************************************
//* Copy stderr to SYSOUT
//*****************************************************
//*
//COPYERR   EXEC PGM=IKJEFT01,COND=EVEN,
//          PARM='OCOPY INDD(INPUTERR) OUTDD(STDERR)'
//SYSTSPRT  DD DUMMY
//INPUTERR  DD PATH='&CPODIR/stderr',
//             PATHOPTS=(ORDONLY)
//STDERR    DD SYSOUT=&OUTCLS,DCB=(RECFM=V,LRECL=512)
//SYSTSIN   DD DUMMY
//*
//*-----------------------------------------------------
//         PEND
```

A.4.4 Providing APF authorization

To provide APR authorization, first ensure that data set SYS1.SIEALNKE is in the link list. The system automatically places this data set at the beginning of the link list, unless this is overridden by a SYSLIB statement in PROGxx. In addition, the default IEASYSxxvalue LNKAUTH=LNKLST must be in effect, or SYS1.SIEALNKE must be APF authorized.

The next steps in this process are optional. They are only required if you copied the Capacity Provisioning files or the Java files on your runtime system.

On any runtime system, the Provisioning Manager must run with APFauthorization. The code must be authorized for this; the main program is located in a program library data set (PDSE) and the libraries are located in the file system.

If you have copied the Capacity Provisioning library files from the UNIX fleshiest, you must ensure that all Provisioning Manager libraries and the Java libraries are sufficiently authorized. The Provisioning Manager libraries are located in `/usr/lpp/cpo/lib`. You can check the authorization by entering the command at a UNIX shell prompt:

```
extattr /usr/lpp/cpo/lib/*
```

The following libraries require APF authorized = YES:

- libcpoarm.so
- libcpoconsole.so
- libcposocket.so
- libcpostream.so
- libcpoii.so

The Java library is in `/usr/lpp/java/J6.0/bin/classic`, and it is named `libjvm.so`. If you have copied the Java SDK code, you must ensure that this has APF authorization in the same way as for the Provisioning Manager libraries.

A.4.5 Securing the runtime system

Set up security on the runtime system and on the observed systems. Because an observed system can also be a runtime system, you may have to perform both definitions on these. On the runtime system, perform these tasks:

- Define the started task
- Define ARM access
- Define access for the Provisioning Manager user
- Define the secured signon function
- Define access for the Control Center user
- Define access to the hardware

Ensure that the Provisioning Manager user and the Control Center user are already defined to the security manager, and that an OMVS segment has been defined for both users.

All RACF security definitions that are required for a runtime system that is also an observed system are contained in member CPOSEC1 in SYS1.SAMPLIB. You can copy and change this sample job to match your needs. All security definitions required for additional observed systems that are not sharing the same security database are listed in sample member CPOSEC2.

The RACF security definitions listed in this document and contained in the members CPOSEC1 and CPOSEC2 assume a prior CIM setup. Detailed information about CIM setup can be found in “Setting up CIM” on page 798.

A.4.6 Defining security for hardware access

This step is only required if your communication to the hardware console is based on BCPii. If you are using this communication, you need to have the Common Event Adapter (CEA) running in full function mode and you need to authorize the Provisioning Manager user to various Common Event Adapter (CEA) services and to the CPCs that need to be managed. For information about how to set CEA into full function mode, see *z/OS Planning for Installation*.

For CEA services, the Provisioning Manager user needs READ authority to the following profiles in the SERVAUTH class:

- CEA.CONNECT
- CEA.SUBSCRIBE.ENF_0068*

If you have previously defined access through more generic profiles, such as CEA.*, you might want to use those profiles to also permit the Capacity Provisioning user.

The following list illustrates a sample definition:

- SETROPTS CLASSACT(SERVAUTH)
- RDEFINE SERVAUTH CEA.CONNECT UACC(NONE)
- RDEFINE SERVAUTH CEA.SUBSCRIBE.ENF_0068* UACC(NONE)
- PERMIT CEA.CONNECT CLASS(SERVAUTH) ID(CPOSRV) ACCESS(READ)
- PERMIT CEA.SUBSCRIBE.ENF_0068* CLASS(SERVAUTH) ID(CPOSRV) ACCESS(READ)
- SETROPTS RACLIST(SERVAUTH) REFRESH

To allow Provisioning Manager users to access information about the hardware and perform activation and deactivation requests for temporary capacity on a CPC, they need the following authorizations:

- READ access to profile HWI.APPLNAME.HWISERV in the FACILITY class
- CONTROL access to profile HWI.TARGET.name in the FACILITY class

  The netid and name represent the SNA name of the CPC as defined at the SE. The APPLDATA of the security definition needs to contain the uppercase community name for the Provisioning Manager. The profiles need to cover all CPCs to be managed by Capacity Provisioning, and all CPCs on which the Provisioning Manager may run.
- v READ access to profile HWI.CAPREC.netid.nau.* in the FACILITY class

  The net id and name represent the SNA name of the CPC as defined at the SE. The profiles need to cover all capacity records.

For example, if you have a CPC with SNA Name IBMNET.CPC1, the definitions look as follows:

- RDEFINE FACILITY HWI.APPLNAME.HWISERV UACC(NONE)
- RDEFINE FACILITY HWI.TARGET.IBMNET.CPC1 APPLDATA(...) UACC(NONE)
- RDEFINE FACILITY HWI.CAPREC.IBMNET.CPC1.* UACC(NONE)
- PERMIT HWI.APPLNAME.HWISERV CLASS(FACILITY) ID(CPOSRV) ACCESS(READ)
- PERMIT HWI.TARGET.IBMNET.CPC1 CLASS(FACILITY) ID(CPOSRV) ACCESS(CONTROL)

- PERMIT HWI.CAPREC.IBMNET.CPC1.* CLASS(FACILITY) ID(CPOSRV) ACCESS(READ)
- SETROPTS RACLIST(FACILITY) REFRESH

For more information about BCPii setup, refer to 25.3, “Installing BCPii” on page 592.

A.4.7 Defining security for the Control Center user

The Control Center user must be authorized to connect the Control Center to the Provisioning Manager. Define this user on the runtime system with an OMVS segment, and add the user to the appropriate Provisioning Manager security group, depending on which administration and operation commands the user is allowed to use.

A.4.8 Securing the observed systems

When a system is observed, the Provisioning Manager connects to the CIM server on that system and retrieves configuration information and performance information about the workload.

To enable this communication, establish a connection in the name of the Provisioning Manager user and authorized by a PassTicket by performing these steps:

- Define the Provisioning Manager user on the observed systems with the same password as on the runtime system.
- Enable the secured signon function.
- Authorize the Provisioning Manager user to access the CIM server.

A.4.9 Setting up Automatic Restart Manager

This step is only needed if you use the Automatic Restart Manager (ARM) to restart the Provisioning Manager.

The Provisioning Manager requires the following items to use ARM capability:

- The element name is SYSCPO, unless you chose another value in Provisioning Manager runtime environment.
- The element type is SYSCPM, unless you chose another value in Provisioning Manager runtime environment.
- The Provisioning Manager is to normally be restarted with the policy and processing mode that were in use the last time it ran, together with any modifications to the policy that were triggered by console commands. To achieve this, the policy name and the processing mode on the restart command are to be specified as an asterisk (*).
- The Provisioning Manager may be restarted on another system of the sysplex if this system has access to the runtime data sets and the required file systems used by the previous system.

An example setup is supplied in member CPOARMPO of SYS1.SAMPLIB.

A.4.10 Preparing the connection to the CIM server

The Provisioning Manager and the Control Center communicate through a CIM server.You can establish connections through the server using either HTTP or HTTPS, if the CIM server

is configured to support the chosen protocol. For details about how to configure the CIM server, see *z/OS Common Information Model User's Guide*. For HTTPS, use an AT-TLS configuration. The Provisioning Manager and the Control Center do not support authentication based on SSL certificates.

Note: Ensure that the configured port of the CIM server matches the definitions that you make in your domain configuration and the Control Center.

A.4.11 Preparing the connection to the Provisioning Manager

To prepare the connection to the Provisioning Manager, first logon to a z/OS UNIX session as a CIM administrator user, then follow these steps:

1. Copy the Capacity Provisioning CIM provider properties file to the `/etc` directory:

   ```
   cp /usr/lpp/cpo/provider/cpoprovider.properties /etc
   ```

2. If your domain name is not the default (DOMAIN1), you must edit the file you copied (for example, using `oedit /etc/cpoprovider.properties`) to change the DomainNames = DOMAIN1 line to reflect the name of your domain.
3. Ensure that the file is readable: `chmod a+r /etc/cpoprovider.properties`
4. Verify that the program-controlled flag is set in the extended file attributes for the Capacity Provisioning CIM provider library:

   ```
   ls -E /usr/lpp/cpo/lib/libcpoprovider.so
   ```

   If the attribute is not set, then use the following command to set the program control flag manually:

   ```
   extattr +p /usr/lpp/cpo/lib/libcpoprovider.so
   ```

5. Verify that a link to the Capacity Provisioning CIM provider library has been created in the CIM server provider directory:

   ```
   ls -l /usr/lpp/wbem/provider/libcpoprovider.so
   ```

   If the link does not exist, use the following command to create the link manually:

   ```
   ln -s /usr/lpp/cpo/lib/libcpoprovider.so
   /usr/lpp/wbem/provider/libcpoprovider.so
   ```

6. If the `cpoprovider.properties` file was changed, restart the CIM server.

B

zFS commands

You can monitor zFS performance by using the **modify** command. The output from the **modify zfs,query** command is written to the system log.

This appendix illustrates the syntax of this command and provides an explanation of the report and the option values. It also contains an example of output from the **f zfs,query,all** command.

B.1 Using zFS commands

Example B-1 shows sample output from zFS QUERY reports and describes the relevant fields of each report. Several fields are used mainly by IBM service, but we include them here for completeness.

Note: You can reset the performance monitoring statistics for any given zFS report, or reset all of the internal zFS statistics. The syntax of this command is shown here, where <report> is KN, VM, LFS, LOG, LOCK, STOR, FILE, STKM, CTKC, DATASET, SVI, or ALL.

```
f zfs,reset,<report>
```

Example B-1 Display of f zfs,query,all command output

```
IOEZ00438I Starting Query Command KN. 163
          PFS Calls on Client
          -------------------
Operation              Count  XCF req.    Avg Time           Bytes
---------         ----------  ----------  ----------    ----------
zfs_opens              39115       18231      18.906
zfs_closes             37319       16923       4.222
zfs_reads               2008           2       0.255       36.736M
zfs_writes            455898           2       0.005       18.097M
zfs_ioctls                 0           0       0.000
zfs_getattrs           35874           1       0.004
zfs_setattrs              10          10      18.650
zfs_accesses              79           0       0.007
zfs_lookups           262460        4624       1.147
zfs_creates             2692        2692      16.444
zfs_removes                7           7      16.479
zfs_links                  0           0       0.000
zfs_renames                0           0       0.000
zfs_mkdirs                 5           5      12.330
zfs_rmdirs                 3           3      18.078
zfs_readdirs            1697           0       0.020
zfs_symlinks               0           0       0.000
zfs_readlinks           9617           7       0.022
zfs_fsyncs                 0           0       0.000
zfs_truncs                 0           0       0.000
zfs_lockctls               0           0       0.000
zfs_audits                90           0       0.013
zfs_inactives          25222           0       0.009
zfs_recoveries             0           0       0.000
zfs_vgets                 14           8       3.725
zfs_pfsctls                0           0       0.000
zfs_statfss                2           0       0.012
zfs_mounts                19           0      91.010
zfs_unmounts               0           0       0.000
zfs_vinacts                0           0       0.000
---------         ----------  ----------  ----------
*TOTALS*              872131       42515       1.431

IOEZ00438I Starting Query Command VM. 164
                  User File (VM) Caching System Statistics
```

```
                ---------------------------------------

External Requests:
------------------
Reads           2023     Fsyncs           225     Schedules        264
Writes        459122     Setattrs          88     Unmaps          2188
Asy Reads         90     Getattrs        1768     Flushes            0

File System Reads:
------------------
Reads Faulted          31     (Fault Ratio    1.532%)
Writes Faulted          5     (Fault Ratio    0.001%)
Read Waits              4     (Wait Ratio    0.198%)
           PAGE      2
Total Reads            71

File System Writes:
-------------------
Scheduled Writes         36372     Sync  Waits            0
Error Writes                 0     Error Waits            0
Scheduled deletes            3
Page Reclaim Writes          0     Reclaim Waits          0
Write Waits                  0     (Wait Ratio    0.000%)

Page Management (Segment Size = 64K) ) (Page Size = 4K)
-------------------------------------------------
Total Pages      393216     Free             384300
Segments          16384
Steal Invocations          0     Waits for Reclaim           0

Number of dataspaces used:    24  Pages per dataspace:       16384

Dataspace   Allocated     Free
Name        Segments      Pages
--------    ----------    ----------
ZFSUCD00           108         16014
ZFSUCD01           132         16014
ZFSUCD02           129         16013
ZFSUCD03           121         16014
ZFSUCD04           129         16014
ZFSUCD05           130         16011
ZFSUCD06           115         16014
ZFSUCD07           118         16015
ZFSUCD08           130         16011
ZFSUCD09           121         16010
ZFSUCD0A           106         16011
ZFSUCD0B           147         16009
ZFSUCD0C           115         16014
ZFSUCD0D           128         16013
ZFSUCD0E           117         16010
ZFSUCD0F           115         16010
ZFSUCD10           122         16012
ZFSUCD11           116         16012
ZFSUCD12           118         16015
ZFSUCD13           137         16012
```

```
ZFSUCD14           120         16012
ZFSUCD15           122         16014
ZFSUCD16           116         16015
ZFSUCD17           130         16011
IOEZ00438I Starting Query Command VM. 165
                   Sysplex Client Caching System Statistics
                   ----------------------------------------

External Requests:
------------------
Reads              0     Fsyncs             0     Schedules          0
Writes             0     Setattrs           0     Unmaps             0
           PAGE      3
Asy Reads          0     Getattrs           0     Flushes            0

File System Reads:
------------------
Reads Faulted           0     (Fault Ratio    0.000%)
Writes Faulted          0     (Fault Ratio    0.000%)
Read Waits              0     (Wait Ratio    0.000%)
Total Reads             0

File System Writes:
-------------------
Scheduled Writes             0     Sync  Waits            0
Error Writes                 0     Error Waits            0
Scheduled deletes            0
Page Reclaim Writes          0     Reclaim Waits          0
Write Waits                  0     (Wait Ratio    0.000%)

Page Management (Segment Size = 32K) ) (Page Size = 4K)
-----------------------------------------------------
Total Pages        8192     Free               8192
Segments          16384
Steal Invocations          0     Waits for Reclaim           0

Number of dataspaces used:     1  Pages per dataspace:        8192

Dataspace   Allocated     Free
Name        Segments      Pages
--------    ----------    ----------
ZFSCCD00             0          8192
IOEZ00438I Starting Query Command LFS. 166
                              zFS Vnode Op Counts

Vnode Op                      Count   Vnode Op
Count
-----------------------  ----------   -----------------------
----------
efs_hold                          0   efs_readdir
420
efs_rele                          0   efs_create
241
efs_inactive                      0   efs_remove
1632
```

```
efsvn_getattr                  26401   efs_rename
0
efs_setattr                      265   efs_mkdir
1
efs_access                      1684   efs_rmdir
1
efs_lookup                     22498   efs_link
0
efs_getvolume                      0   efs_symlink
0
efs_getlength                      0   efs_readlink
           PAGE      4
21
efs_afsfid                         0   efs_rdwr
0
efs_fid                            0   efs_fsync
0
efs_vmread                         0   efs_waitIO
16514
efs_vmwrite                        0   efs_cancelIO
1060
efs_clrsetid                       0   efs_audit
8301
efs_getanode                    5772   efs_vmblkinfo
0
efs_readdir_raw                    0

Total zFS Vnode Ops      84811

                         zFS Vnode Cache Statistics

Vnodes      Requests     Hits    Ratio  Allocates   Deletes
---------- ---------- ---------- ----- ---------- ----------
     32768    6942293    6936894  99.922%          0       4658

zFS Vnode structure size: 224 bytes
zFS extended vnodes: 32768, extension size 724 bytes (minimum)
Held zFS vnodes:       1923 (high       1958) Open zFS vnodes:
1842 (high       1846) Reusable: 29907

Total osi_getvnode Calls:      27418 (high resp          0) Avg. Call
Time:         0.007 (msecs)
Total SAF Calls:              328565 (high resp          0) Avg. Call
Time:         0.030 (msecs)

                         zFS Fast Lookup Statistics

Buffers     Lookups      Hits    Ratio  Neg. Hits   Updates
---------- ---------- ---------- ----- ---------- ----------
      4096      13279      10652  80.217%         67      12936

                         Metadata Caching Statistics

Buffers    (K bytes)  Requests     Hits    Ratio  Updates
---------- --------- ---------- ---------- ----- ----------
```

```
     32768    262144    1865450    1466989  78.6%      29992


                      Transaction Cache Statistics

Transactions started:       8163    Lookups on tran:      11813    EC
Merges:          0
Allocated Transactions:       2069  (Act=         0, Pend=         0,
Comp=        11, Free=      2058)

           PAGE      5

                 I/O Summary By Type
                 -------------------

Count       Waits       Cancels     Merges      Type
----------  ----------  ----------  ----------  ----------
    602975      463999           0           0  File System Metadata
       588         230           0           0  Log File
     19473          53           0           0  User File Data

                 I/O Summary By Circumstance
                 ---------------------------

Count       Waits       Cancels     Merges      Circumstance
----------  ----------  ----------  ----------  ------------
    602739      464002           0           0  Metadata cache read
        68          44           0           0  User file cache direct
read
        17          17           0           0  Log file read
         0           0           0           0  Metadata cache async
delete write
         0           0           0           0  Metadata cache async
write
         0           0           0           0  Metadata cache lazy
write
         0           0           0           0  Metadata cache sync
delete write
         0           0           0           0  Metadata cache sync
write
     19396           0           0           0  User File cache direct
write
         0           0           0           0  Metadata cache file
sync write
       240           1           0           0  Metadata cache sync
daemon write
         0           0           0           0  Metadata cache
aggregate detach write
         0           0           0           0  Metadata cache buffer
block reclaim write
         0           0           0           0  Metadata cache buffer
allocation write
         5           5           0           0  Metadata cache file
system quiesce write
         0           0           0           0  Metadata cache log
```

```
file full write
       571         213           0           0  Log file write
         0           0           0           0  Metadata cache
shutdown write
         0           0           0           0  Format, grow write

                    zFS I/O by Currently Attached Aggregate

DASD   PAV
           PAGE      6
VOLSER IOs Mode  Reads       K bytes     Writes      K bytes
Dataset Name
------ --- ----  ----------  ----------  ----------  ----------
------------
BH5ST1   2  R/W           8          56           0           0
PLEX75.SYSPLEX.ROOT.ZFS
BH5ST1   2  R/W           9          64           0           0
PLEX75.SC74.SYSTEM.ZFS
BH5HF4   2  R/O         153        3776           0           0
OMVS.ZOSR1D.Z1DRC1.ROOT
BH5HF4   2  R/O           9          60           0           0
OMVS.ZOSR1D.Z1DRC1.JAVA31V5
BH5HF4   2  R/O           9          60           0           0
OMVS.ZOSR1D.Z1DRC1.JAVA64V5
BH5HF4   2  R/O           9          60           0           0
OMVS.ZOSR1D.Z1DRC1.JAVA31V6
BH5HF4   2  R/O           9          60           0           0
OMVS.ZOSR1D.Z1DRC1.JAVA31M1
BH5HF4   2  R/O           9          60           0           0
OMVS.ZOSR1D.Z1DRC1.JAVA64V6
BH5HF4   2  R/O          11          76           0           0
OMVS.ZOSR1D.Z1DRC1.XML
BH5ST1   2  R/W          16         116       27645      110860
PLEX75.SC75.SYSTEM.ZFS
BH5HF3   3  R/W          27         196       27644      110756
OMVS.PP.HFS
BH5OE2  72  R/O       27605      110444           0           0
OMVS.DB2V9.DO80325.HFS
BH5ST2   1  R/W           9          60       27598      110392
SYSPROG.ZFS
BH5ST3  72  R/O       27607      110460           0           0
ZOSMF113.SBBN7ZFS
BH5ST3  72  R/O       27605      110444           0           0
ZOSMF113.SIZUZFS
BH5OE2  72  R/W           9          60       27598      110392
CEA.HFS
BH5ST2   1  R/W          21         148       27603      110432
CFZSRV.ZFS
BH5OE2  72  R/W      380929     3047368       20855      133880
PFA.HFS
BH5HF2   3  R/W           2           8           0           0
OMVS.SC74.VAR.WBEM
BH5HF2   3  R/W           2           8           0           0
OMVS.SC74.TDSSRV1.HFS
BH5HF1   3  R/W           2           8           0           0
```

```
OMVS.SC74.WEB.PRINTWAY.ZFS
BH5ST1   2  R/W           2           8           0           0
JES2.ZFS
BH5ST4  72  R/W           2           8           0           0
RC74.HFS
BH5ST3  72  R/W           2           8           0           0
BBN.V7RO.CONFIG1.ZFS
BH5OE2  72  R/W          10          68          11          88
          PAGE      7
RODOLFI.HFS
BH5ST2   1  R/W           2           8           0           0
HERING.ZFS
BH5ST2   1  R/W           2           8           0           0
BIN.ZFS
BH5ST1   2  R/W           2           8           0           0
HARJANS.ZFS
------           ----------  ----------  ----------  ----------
    28               464082     3383708      158954      686800
*TOTALS*


Total number of waits for I/O:     464282
Average I/O wait time:                0.314 (msecs)
IOEZ00438I Starting Query Command LOG. 167
                        Log File Caching Statistics

Buffers    (K bytes)  Requests     Hits    Ratio   Written
---------- --------- ---------- ---------- ------ ----------
      4112     32896        588         99  16.8%        571

New buffer: log full waits          0  NBS IO waits          0
IOEZ00438I Starting Query Command LOCKING. 168
                                   Locking Statistics


Untimed sleeps:      13878   Timed Sleeps:          0   Wakeups:
14099

Total waits for locks:                              27
Average lock wait time:            24.683 (msecs)

Total monitored sleeps:                         400410
Average monitored sleep time:              7.723 (msecs)

Total starved waiters:              0
Total task priority boosts:         0

               Top 15 Most Highly Contended Locks
  Thread      Async       Spin
   Wait       Disp.       Resol.     Pct.    Description
----------  ----------  ----------   -----   --------------
        25           0          16    27.891%   Anode bitmap
allocation handle lock
        13           0           8    14.286%   Log system map lock
        15           0           0    10.204%   Aggregate syscall lock
```

```
        15           0           0    10.204%   Transaction-cache main
lock
         0           0          11     7.483%   XCF storage pool size
lock
         2           0           7     6.122%   Cache Services item
refcount lock
         0           0           7     4.762%   Sysplex connection
           PAGE      8
item lock
         6           0           0     4.82%   Vnode-cache access lock
         5           0           0     3.401%   CDIO bitmap lock
         4           0           0     2.721%   Metadata-cache buffer
lock
         3           0           0     2.41%   Vnode lock
         0           0           3     2.41%   Fileset registry hash
row lock
         0           1           0     0.680%   Volser I/O queue lock
         1           0           0     0.680%   Transaction lock
         1           0           0     0.680%   Vnode-cache main lock

Total lock contention of all kinds:        147

               Top 15 Most Common Thread Sleeps
Thread Wait     Pct.      Description
-----------     -----     -----------
     340205     84.964%     Request completion on server thread
      60137     15.19%     XCF waiting for message replies
         65      0.16%     STKM token revoke reply wait
          1      0.0%     XAGGR first pass takeover processing
          1      0.0%     CTKC pending token obtain
          1      0.0%     CTKC user file pending IO wait
          0      0.0%     XCF reply buffer reservation wait
          0      0.0%     STKC token obtain wait
          0      0.0%     STKC token revoke wait
          0      0.0%     STKM garbage collection in progress
          0      0.0%     STKM pending token grant
          0      0.0%     Transaction allocation wait
          0      0.0%     OSI cache item init wait
          0      0.0%     OSI cache empty wait
          0      0.0%     OSI cache item cleanup wait
IOEZ00438I Starting Query Command STORAGE. 169
                zFS Primary Address Space Storage Usage
                ---------------------------------------

Total Storage Available to zFS: 1825570816 (1782784K) (1741M)
Non-critical Storage Limit: 1804599296 (1762304K) (1721M)
USS/External Storage Access Limit: 1762656256 (1721344K) (1681M)
Total Bytes Allocated (Stack+Heap+OS): 502427648 (490652K) (479M)
Heap Bytes Allocated: 452730004 (442119K) (431M)
Heap Pieces Allocated: 933987
Heap Allocation Requests: 952545
Heap Free Requests: 18558

             Heap Usage By Component
```

```
            Storage Usage By Component
            --------------------------
Bytes               No. of  No. of
Allocated   Pieces  Allocs  Frees   Component
----------  ------  ------  ------  ---------

           PAGE      9
    133072      18      18       0  Interface
     37608      90     158      68  Media Manager I/O driver
      1828       5       5       0  Trace Facility
    473052       7       7       0  Message Service
    304940     705     706       1  Miscellaneous
     47144     108     112       4  Aggregate Management
    116780      35      37       2  Filesystem Management
     33952      24    8359    8335  Administration Command Handling
  16568832   65960   65961       1  Vnode Management
  16596432   39107   39916     809  Anode Management
         0       0       0       0  Directory Management
    916800    8250    8703     453  Log File Management
 275669296   65541   65547       6  Metadata Cache
    425008    2074    2074       0  Transaction Management
  25840052   98357   98359       2  Asynchronous I/O Component
     66532     193     195       2  Lock Facility
   1264796     671     671       0  Threading Services
  10880468  214968  215007      39  Cache Services
     43148       3    8286    8283  Configuration parameters
processing
  15243512  401426  401434       8  User File Cache
     62988     131     178      47  Storage Management
  63261924    1543    1545       2  XCF Services
     70880      20      48      28  Cross system attach validation
   2488448   20857   21315     458  Server Token Manager (STKM)
     25348     170     170       0  Server Token Cache (STKC)
  21106756   13713   13714       1  Client Token Cache (CTKC)
         0       0       0       0  Server Vnode Interface (SVI)
      1836       9      18       9  Name Space (NS)
   1048572       2       2       0  Directory storage

IOEZ00438I Starting Query Command FILESETS. 170
File System Name                                Aggr #  Flg  Operations
----------------------------------------------  ------  ---  ----------
PFA.HFS                                             18  AMS
750292

PLEX75.SYSPLEX.ROOT.ZFS                              1  AMS
104734

PLEX75.SC75.SYSTEM.ZFS                              10  AMS
28276

OMVS.ZOSR1D.Z1DRC1.ROOT                              3  AM
19234

PLEX75.SC74.SYSTEM.ZFS                               2  AMS
2172
```

```
RODOLFI.HFS                                      25  AMS
464

OMVS.PP.HFS                                      11  AMS
          PAGE     10
237

CFZSRV.ZFS                                       17  AMS
8

SYSPROG.ZFS                                      13  AMS
6

CEA.HFS                                          16  AMS
4

OMVS.DB2V9.DO80325.HFS                           12  AM
2

OMVS.ZOSR1D.Z1DRC1.JAVA31V6                       6  AM
2

OMVS.ZOSR1D.Z1DRC1.JAVA64V5                       5  AM
2

OMVS.ZOSR1D.Z1DRC1.JAVA31V5                       4  AM
2

OMVS.SC74.VAR.WBEM                               19  AMS
2

OMVS.SC74.TDSSRV1.HFS                            20  AMS
2

BIN.ZFS                                          27  AMS
2

HARJANS.ZFS                                      28  AMS
2

ZOSMF113.SIZUZFS                                 15  AM
2

RC74.HFS                                         23  AMS
2

ZOSMF113.SBBN7ZFS                                14  AM
2

OMVS.ZOSR1D.Z1DRC1.XML                            9  AM
2

OMVS.ZOSR1D.Z1DRC1.JAVA64V6                       8  AM
2
```

```
OMVS.ZOSR1D.Z1DRC1.JAVA31M1                          7  AM
2

           PAGE     11
OMVS.SC74.WEB.PRINTWAY.ZFS                          21  AMS
2

JES2.ZFS                                            22  AMS
2

BBN.V7RO.CONFIG1.ZFS                                24  AMS
2

HERING.ZFS                                          26  AMS
2

IOEZ00438I Starting Query Command IOBYDASD. 171
                    zFS I/O by Currently Attached DASD/VOLs

DASD   PAV
VOLSER IOs   Reads       K bytes     Writes           K bytes
Waits         Average Wait
------ ---   ----------  ----------  ----------       ----------
----------    ------------
BH5ST4  72            2           8           0                0
2           0.224
BH5HF2   3            4          16           0                0
4           0.462
BH5ST2   1           34         224       55201           220824
40           3.150
BH5OE2  72       408553     3157940       48464           244360
408723          0.316
BH5HF4   2          209        4152           0                0
185           4.692
BH5HF1   3            2           8           0                0
2           0.486
BH5ST3  72        55214      220912           0                0
55213           0.294
BH5HF3   3           27         196       27644           110756
52           2.490
BH5ST1   2           37         252       27645           110860
61           2.961

Total number of waits for I/O:      464282
Average wait time per I/O:           0.316
IOEZ00438I Starting Query Command DATASET. 172
   Printing Dataset Allocation Stats
        Allocates           8
        Allocates failed    0
        Unallocates         8
        Unallocates failed  0
        Opens               8
        Open failures       0
        Closes              8
```

```
IOEZ00438I Starting Query Command STKM. 173
                 Server Token Manager (STKM) Statistics
                 --------------------------------------
           PAGE     12
Maximum tokens:        65536        Allocated tokens:        20480
Tokens In Use:            45        File structures:           362
Token obtains:         10002        Token returns:            9668
Token revokes:           224        Async Grants:                0
Garbage Collects:          0        TKM Establishes:             0
Thrashing Files:           0        Thrash Resolutions:          0

                         Usage Per System:
System   Tokens     Obtains   Returns    Revokes   Async Grt Establish
-------- --------- ---------- ---------- --------- --------- ---------
    SC74        15       5404       5149        65         0         0
ZEROLINK         0       4324       4324         0         0         0
LOCALUSR        30        274        195       159         0         0
IOEZ00438I Starting Query Command CS. 174
          SVI Calls to System SC74
          ----------------------------
SVI Call                      Count      Avg. Time
--------------------       ----------   ----------
GetToken                         6917       21.967
GetMultTokens                       0        0.000
ReturnTokens                        6        0.032
ReturnFileTokens                    0        0.000
FetchData                           2       12.831
StoreData                       35047       25.370
Setattr                            10       17.837
FetchDir                            0        0.000
Lookup                              0        0.000
GetTokensDirSearch                  0        0.000
Create                           2697       16.068
Remove                             10       16.892
Rename                              0        0.000
Link                                0        0.000
ReadLink                            7        1.915
SetACL                              0        0.000
Statfs                              0        0.000
TSR                                 1       15.169
FilesysSyncTable                    0        0.000
FileSyncMeta                     2019       12.694
BitmapReserve                       2       16.766
BitmapUnreserve                     0        0.000
BitmapReclaim                       0        0.000
FileUpdateIB                    19121        0.031
FileCreateIB                        1        0.032
FwdReaddir                          0        0.000
LkupInvalidate                      0        0.000
FileDebug                           0        0.000
--------------------       ----------   ----------
*TOTALS*                        65840       16.876

IOEZ00438I Starting Query Command SV. 175
          SVI Calls from System SC74
```

```
          ----------------------------
SVI Call                    Count      Qwait    XCF Req.  Avg. Time
           PAGE     13
--------------------     ----------  --------  --------  ----------
GetToken                      1866         0         0       0.296
GetMultTokens                    0         0         0       0.000
ReturnTokens                     1         0         0       0.045
ReturnFileTokens                 0         0         0       0.000
FetchData                        0         0         0       0.000
StoreData                      320         0         0       0.113
Setattr                          0         0         0       0.000
FetchDir                         0         0         0       0.000
Lookup                           0         0         0       0.000
GetTokensDirSearch               0         0         0       0.000
Create                         165         0         0       0.567
Remove                        1628         0         0       0.208
Rename                           0         0         0       0.000
Link                             0         0         0       0.000
ReadLink                         0         0         0       0.000
SetACL                           0         0         0       0.000
Statfs                           0         0         0       0.000
TSR                              0         0         0       0.000
FilesysSyncTable                 0         0         0       0.000
FileSyncMeta                    48         0         0       0.016
BitmapReserve                    1         0         0       9.709
BitmapUnreserve                  0         0         0       0.000
BitmapReclaim                    0         0         0       0.000
FileUpdateIB                   265         0         0       0.117
FileCreateIB                     6         0         0       0.080
FwdReaddir                       0         0         0       0.000
LkupInvalidate                   0         0         0       0.000
FileDebug                        0         0         0       0.000
--------------------     ----------  --------  --------  ----------
*TOTALS*                      4300         0         0       0.247

IOEZ00025I zFS kernel: MODIFY command - QUERY,ALL completed successful
ly.
```

C

BCPii Metal C example

This appendix contains an example, written in C code, that is used to call the BCP internal interface. It also contains the JCL to compile, assemble, and link edit this proposed code.

The appendix further provides sample System REXX that allows you to call such an example from the console as if it were a z/OS command. This gives you an opportunity to insert such new z/OS commands into classical automation tools that are used for taking advantage of and managing zSeries systems.

C.1 C code

Example C-1 shows the C program proposed as an example for programming the BCP internal interface (BCPii).

Example C-1 C program calling BCPii

```
*--------------------------------------------------------------------*
*                                                                    *
*    MODULE NAME= HWIOO1                                             *
*                                                                    *
*    DESCRIPTIVE NAME= Sample C code calls to BCPii services.        *
*                                                                    *
*    FUNCTION                                                        *
*     This module provides sample calls to these BCPii services:    *
*       HWILIST  - List CPCs; list the images on a CPC.              *
*     This sample calls sprintf to format output data and calls      *
*       WTO to make the output immediately available.  You may       *
*       choose to direct the output elsewhere.                       *
*                                                                    *
*    DEPENDENCIES                                                    *
*                                                                    *
*     1. Include the header file HWICIC.H.                           *
*                                                                    *
*     2. Compile with a C or a C++ compiler.                         *
*        If the Metal C compiler is used, use the "metal" compiler   *
*        option; include the metal header files.                     *
*                                                                    *
*     3. Assemble using the "-mgoff" and "-mrent" options.           *
*                                                                    *
*     4. Link the resultant object file with the SYS1.CSSLIB to build *
*        an executable file that can use BCPii services.             *
*        Make the load module APF-authorized using the AC=1 link     *
*        option and place it in an APF-authorized dataset.           *
*                                                                    *
* Change Activity                                                    *
*   $LO  OA31731  HBB7750 100916 PDRH:  BCPii C Sample code.         *
*                                                                    *
*************************END OF SPECIFICATIONS************************/
#pragma filetag ("IBM-1047")  /* Define the codepage as EBCDIC.      */
#pragma longName              /* Allow names longer than 8 chars.    */
#include <stdio.h>
#include <string.h>
#include <hwicic.h>
typedef struct CPC_elements {
   char elementÝ18¨;
   char null_char;
} CPC_elements;
typedef struct IMAGE_elements {
   char elementÝ9¨;
   char null_char;
} IMAGE_elements;
/* ---------------------------------------------------------------- */
/* Function prototypes                                              */
/* ---------------------------------------------------------------- */
```

```
void print_diagarea(HWI_DIAGAREA_TYPE DA);
void CPC_AnswerArea_into_Array(char Ý¨,CPC_elements Ý¨,int);
 void Image_AnswerArea_into_Array(char Ý¨,IMAGE_elements Ý¨,int);
 void Call_HWISET(int  *                returncodePtr,      /* Output */
                  HWI_CONNTOKEN_TYPE   connectToken,       /* Input  */
                  HWI_DIAGAREA_TYPE *  diagareaPtr);       /* Output */
 void Call_HWICMD(int  *                returncodePtr,      /* Output */
                  HWI_CONNTOKEN_TYPE   connectToken,       /* Input  */
                  HWI_DIAGAREA_TYPE *  diagareaPtr);       /* Output */
 void Call_HWIEVENT(int  *              returncodePtr,      /* Output */
                  HWI_CONNTOKEN_TYPE   connectToken,       /* Input  */
                  HWI_DIAGAREA_TYPE *  diagareaPtr);       /* Output */
 void Try_EventAndCommand(void);
 void Try_Set(void);
 void ConnectToCPC(int *                returncodePtr,      /* Output */
                  HWI_CONNTOKEN_TYPE * CPCConnectToken);   /* Output */
 void ConnectToImage(int *              returncodePtr,      /* Output */
                  HWI_CONNTOKEN_TYPE   CPCConnectToken,    /* Input  */
                  HWI_CONNTOKEN_TYPE * ImageConnectToken); /* Output */
 void DisconnectFromCPC(
                  HWI_CONNTOKEN_TYPE   CPCConnectToken);   /* Input  */
 void DisconnectFromImage(
                  HWI_CONNTOKEN_TYPE   ImageConnectToken); /* Input  */
 /* Associate variable eventExitEP with register 0                    */
 register int * eventExitEP  __asm("r0");
 main()
 {
  int rc, listtype, numofCPCs, numofImages, answerarealen,
     numofattributes;
 int CPCconnecttype, IMAGEconnecttype;
 int i,j,k;
 char CPC_targetÝ17¨;
 char IMAGE_targetÝ8¨;
 char *CPCconnecttypevaluePtr, *IMAGEconnecttypevaluePtr;
 char answerareaÝ9000¨;
 char *answerarea_ptr;
 char HWI_MMODEL_valueÝ20¨;
 char HWI_OSTYPE_valueÝ20¨;
 char * query_ptr;
 HWI_DIAGAREA_TYPE diagarea;
 HWI_CONNTOKEN_TYPE CPCinconnecttoken, CPCoutconnecttoken;
 HWI_CONNTOKEN_TYPE IMAGEoutconnecttoken;
 HWI_QUERYPARM_TYPE queryparmÝ1¨;
 CPC_elements List_of_CPCsÝ500¨;
 IMAGE_elements List_of_ImagesÝ100¨;
 struct WTO_PARM {      /* mapped by IEZWPL macro.                    */
   unsigned short len;  /* total length of structure                  */
   unsigned short code; /* routing code flag                          */
   char textÝ80¨;
   } wto_buff;
 int  textLen;
 int  wto_prefixLen;
 /* Initialize the target object names with blanks.                   */
 memset(CPC_target,' ',sizeof(CPC_target));
 memset(IMAGE_target,' ',sizeof(IMAGE_target));
```

```
textLen = 0;
wto_prefixLen = sizeof(wto_buff.len) + sizeof(wto_buff.code);
wto_buff.code = 0;                        /* Primary Console Action */
textLen = sprintf(wto_buff.text,"%s",
                             " =>> BCPii C Sample starting ... <<=");
wto_buff.len  = wto_prefixLen + textLen;
__asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
strcpy(diagarea.Diag_Text,"Diag Area");
/* ------------------------------------------------------------- */
/* Call HWILIST to list all CPCs in an HMC network to which you  */
/* have authority.                                               */
/* ------------------------------------------------------------- */
numofCPCs = 0;
listtype = HWI_LIST_CPCS;
memset(List_of_CPCs, 0xOO, sizeof(List_of_CPCs));
answerarealen = sizeof(List_of_CPCs);
answerarea_ptr = &answerareaŸO¨;
/* ------- */
/* HWILIST */
/* ------- */
hwilist(&rc,
        CPCoutconnecttoken,
        listtype,
        &numofCPCs,
        &answerarea_ptr,
        answerarealen,
        &diagarea);
/* ------------------------------------------------------------- */
/* Write to operator to report the HWILIST results.              */
/* ------------------------------------------------------------- */
memset(wto_buff.text,0,sizeof(wto_buff.text));   /* clear wto buff  */
textLen  = sprintf(wto_buff.text,"%s"," =>> LIST all CPCs. ");
textLen += sprintf(wto_buff.text+textLen,"%s",
                                     "   HWILIST Retcode = ");
textLen += sprintf(wto_buff.text+textLen, "%X\n",rc);
wto_buff.len = wto_prefixLen + textLen;
__asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
if ( rc == 0 )
 {
  /* ------------------------------------------------------------- */
  /* Write to operator to report the number of CPCs.               */
  /* ------------------------------------------------------------- */
  memset(wto_buff.text,0,sizeof(wto_buff.text));
  textLen = sprintf(wto_buff.text,"%s",
                                     " =>> Number of CPCs found = ");
  textLen += sprintf(wto_buff.text+textLen, "%d\n",numofCPCs);
  wto_buff.len = wto_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
  /* ------------------------------------------------------------- */
  /* Save the list of CPCs in the answerarea.                      */
   /* ------------------------------------------------------------ */
   CPC_AnswerArea_into_Array(answerarea,
                         List_of_CPCs,
                         numofCPCs);
   i = 0;
```

```
while( i < numofCPCs )
 {
  /* ------------------------------------------------------------ */
  /* Call HWICONN to connect to each CPC.                         */
  /* No CPCinconnecttoken is provided in this case.               */
  /* A CPCoutconnecttoken is returned, which is used as the       */
  /*   input connect token on subsequent requests for connections */
  /*   to images, caprecs, etc. defined for that CPC.             */
  /* ------------------------------------------------------------ */
  memset(CPCinconnecttoken, 0x00, sizeof(CPCinconnecttoken));
  strcpy(CPC_target,List_of_CPCsÝi¨.element);
  CPCconnecttypevaluePtr = &CPC_targetÝO¨;
  CPCconnecttype = HWI_CPC;
  /* ------- */
  /* HWICONN */
  /* ------- */
  hwiconn(&rc,
          CPCinconnecttoken,
          &CPCoutconnecttoken,
          CPCconnecttype,
          &CPCconnecttypevaluePtr,
          &diagarea);
  /* ------------------------------------------------------------ */
  /* Write to operator to report the HWICONN results.             */
  /* ------------------------------------------------------------ */
  memset(wto_buff.text,0,sizeof(wto_buff.text));
  textLen  = sprintf(wto_buff.text,"%s"," =>> CONNECT to CPC ");
  textLen += sprintf(wto_buff.text+textLen, "%d\n", i+1 );
  textLen += sprintf(wto_buff.text+textLen, "%s\n",
                                       &List_of_CPCsÝi¨.element );
  textLen += sprintf(wto_buff.text+textLen,"%s",
                                      "HWICONN Retcode = ");
  textLen += sprintf(wto_buff.text+textLen, "%X\n",rc);
  wto_buff.len = wto_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
  if ( rc == 0 )
   {
    /* ------------------------------------------------------------ */
    /* Call HWIQUERY to query the Model attribute of a CPC using  */
    /* the returned output CPC connect token as the input CPC     */
    /* connect token.                                             */
    /* ------------------------------------------------------------ */
    numofattributes = 1;
    queryparmÝO¨.AttributeIdentifier = HWI_MMODEL;
    queryparmÝO¨.AttributeValue_Ptr  = &HWI_MMODEL_valueÝO¨;
    queryparmÝO¨.AttributeValueLen   = sizeof(HWI_MMODEL_value);
    queryparmÝO¨.AttributeValueLenReturned = -1;
    query_ptr = (char *)&queryparmÝO¨;
    /* -------- */
    /* HWIQUERY */
    /* -------- */
    hwiquery(&rc,
             CPCoutconnecttoken,
             (void **)&query_ptr,
             numofattributes,
```

```
         &diagarea);
 /* ------------------------------------------------------------ */
 /* Write to operator to report the HWIQUERY results.            */
 /* ------------------------------------------------------------ */
 memset(wto_buff.text,0,sizeof(wto_buff.text));
 textLen  = sprintf(wto_buff.text,"%s",
                          "        QUERY CPC Model Number.");
 textLen += sprintf(wto_buff.text+textLen,"%s",
                                "   HWIQUERY Retcode = ");
 textLen += sprintf(wto_buff.text+textLen, "%X\n",rc);
 wto_buff.len = wto_prefixLen + textLen;
 __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
 if ( rc == 0 )
  {
   /* ------------------------------------------------------------ */
   /* Write to operator to report the Model Number                 */
   /* ------------------------------------------------------------ */
   memset(wto_buff.text,0,sizeof(wto_buff.text));
   textLen  = sprintf(wto_buff.text,"%s",
                                     "        Model Number is ");
   textLen += sprintf(wto_buff.text+textLen,"%s",
                                     HWI_MMODEL_value);
   wto_buff.len = wto_prefixLen + textLen;
   __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
  }
 else /* else HWIQUERY failed.                                    */
  {
   print_diagarea(diagarea);
  }
 /* ------------------------------------------------------------ */
 /* Call HWILIST List to list the images on a CPC using the      */
 /* CPC connect token as input.                                  */
 /* ------------------------------------------------------------ */
 numofImages = 0;
 listtype = HWI_LIST_IMAGES;
 memset(List_of_Images, 0x00, sizeof(List_of_Images));
 answerarealen = sizeof(List_of_Images);
 answerarea_ptr = &answerareaŸ0¨;
 /* ------- */
 /* HWILIST */
 /* ------- */
 hwilist(&rc,
         CPCoutconnecttoken,
         listtype,
        &numofImages,
        &answerarea_ptr,
        answerarealen,
        &diagarea);
/* ------------------------------------------------------------ */
/* Write to operator to report the HWILIST results.             */
/* ------------------------------------------------------------ */
memset(wto_buff.text,0,sizeof(wto_buff.text));
textLen  = sprintf(wto_buff.text,"%s",
                               "        LIST images on a CPC. ");
textLen += sprintf(wto_buff.text+textLen,"%s",
```

```
                                  "   HWILIST Retcode = ");
textLen += sprintf(wto_buff.text+textLen, "%X\n",rc);
wto_buff.len = wto_prefixLen + textLen;
__asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
if ( rc == 0 )
 {
  /* ---------------------------------------------------- */
  /* Write to operator to report the number of images.    */
  /* ---------------------------------------------------- */
  memset(wto_buff.text,0,sizeof(wto_buff.text));
  textLen  = sprintf(wto_buff.text,"%s",
                  "            Number of Images found = ");
  textLen += sprintf(wto_buff.text+textLen, "%d\n",numofImages);
  wto_buff.len = wto_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
  /* ---------------------------------------------------- */
  /* Save the list of images in the answerarea.           */
  /* ---------------------------------------------------- */
  Image_AnswerArea_into_Array(answerarea,
                              List_of_Images,
                              numofImages);
  k = 0;
  while( k < numofImages)
   {
    /* ---------------------------------------------------- */
    /* Call HWICONN to connect to an image using the CPC    */
    /* connect token as input.                              */
    /* ---------------------------------------------------- */
     memset(IMAGEoutconnecttoken, 0xOO,
                                 sizeof(IMAGEoutconnecttoken));
     strcpy(IMAGE_target,List_of_ImagesŸk¨.element);
     IMAGEconnecttypevaluePtr = &IMAGE_targetŸO¨;
     IMAGEconnecttype = HWI_IMAGE;
     /* ------- */
     /* HWICONN */
     /* ------- */
     hwiconn(&rc,
             CPCoutconnecttoken,
             &IMAGEoutconnecttoken,
             IMAGEconnecttype,
             &IMAGEconnecttypevaluePtr,
              &diagarea);
      /* ---------------------------------------------------- */
      /* Write to operator to report the HWICONN results.     */
      /* ---------------------------------------------------- */
      memset(wto_buff.text,0,sizeof(wto_buff.text));
      textLen  = sprintf(wto_buff.text,"%s",
                      "            CONNECT to image ");
      textLen += sprintf(wto_buff.text+textLen, "%d\n", k+1 );
      textLen += sprintf(wto_buff.text+textLen, "%s\n",
                                    &List_of_ImagesŸk¨.element );
      textLen += sprintf(wto_buff.text+textLen,"%s",
                                  "HWICONN Retcode = ");
      textLen += sprintf(wto_buff.text+textLen, "%X\n",rc);
      wto_buff.len = wto_prefixLen + textLen;
```

```
__asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
if ( rc == 0 )
 {
  /* ------------------------------------------------- */
  /* Query the OS Type attribute for an image.         */
  /* Call HWIQUERY using the image connect token.      */
  /* ------------------------------------------------- */
  numofattributes = 1;
  queryparmŸO¨.AttributeIdentifier = HWI_OSTYPE;
  queryparmŸO¨.AttributeValue_Ptr  = &HWI_OSTYPE_valueŸO¨;
  queryparmŸO¨.AttributeValueLen=sizeof(HWI_OSTYPE_value);
  queryparmŸO¨.AttributeValueLenReturned = -1;
  query_ptr = (char *)&queryparmŸO¨;
  /* -------- */
  /* HWIQUERY */
  /* -------- */
  hwiquery(&rc,
           IMAGEoutconnecttoken,
           (void **)&query_ptr,
           numofattributes,
           &diagarea);
  /* ------------------------------------------------------*/
  /* Write to operator to report the results from        */
  /* HWIQUERY.                                           */
  /* ------------------------------------------------------*/
  memset(wto_buff.text,0,sizeof(wto_buff.text));
  textLen  = sprintf(wto_buff.text,"%s",
                "                QUERY Image OS Type.");
  textLen += sprintf(wto_buff.text+textLen,"%s",
                             "   HWIQUERY Retcode = ");
  textLen += sprintf(wto_buff.text+textLen, "%X\n",rc);
  wto_buff.len = wto_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
  if ( rc == 0 )
    {
     /* ------------------------------------------------ */
     /* Write to operator to report the OS Type.         */
     /* ------------------------------------------------ */
   memset(wto_buff.text,0,sizeof(wto_buff.text));
   textLen  = sprintf(wto_buff.text,"%s",
                      "                OS Type is ");
   textLen += sprintf(wto_buff.text+textLen,"%s",
                                  HWI_OSTYPE_value);
   wto_buff.len = wto_prefixLen + textLen;
   __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
  }
else /* else HWIQUERY of an image attribute failed.    */
  print_diagarea(diagarea);
/* end HWIQUERY an image.                              */
/* ------------------------------------------------- */
/* Call HWIDISC to disconnect from an image using the  */
/* image connect token as input.                       */
/* ------------------------------------------------- */
hwidisc(&rc,
        IMAGEoutconnecttoken,
```

```
                &diagarea);
            /* ---------------------------------------------- */
            /* Write to operator to report HWIDISC results.   */
            /* ---------------------------------------------- */
            memset(wto_buff.text,0,sizeof(wto_buff.text));
            textLen = sprintf(wto_buff.text,"%s",
                        "           DISCONNECT from image ");
            textLen += sprintf(wto_buff.text+textLen, "%s\n",
                                  &List_of_ImagesÝk¨.element );
             textLen += sprintf(wto_buff.text+textLen,"%s",
                                       "    HWIDISC Retcode = ");
             textLen += sprintf(wto_buff.text+textLen,"%X",rc);
             wto_buff.len = wto_prefixLen + textLen;
             __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
             if(rc]=0)
                 print_diagarea(diagarea);
             }   /* end if HWICONN to an image succeeds.          */
         /* else     HWICONN to an image failed.
              print_diagarea(diagarea);
         */
        /* Advance to the next image in the list.                 */
        k++;
       }  /* end while loop for images                           */
      }
    else  /* else HWILIST of Images on a CPC failed.             */
      print_diagarea(diagarea);
     /* ---------------------------------------------------------- */
     /* Call HWIDISC to disconnect from a CPC using the CPC connect */
     /* token as input.                                           */
     /* ---------------------------------------------------------- */
     hwidisc(&rc,
             CPCoutconnecttoken,
             &diagarea);
     /* ---------------------------------------------------------- */
     /* Write to operator to report HWIDISC results.              */
     /* ---------------------------------------------------------- */
     memset(wto_buff.text,0,sizeof(wto_buff.text));
     textLen = sprintf(wto_buff.text,"%s",
                              "        DISCONNECT from CPC ");
     textLen += sprintf(wto_buff.text+textLen, "%s\n",
                                    &List_of_CPCsÝi¨.element );
     textLen += sprintf(wto_buff.text+textLen,"%s",
                                      "   HWIDISC Retcode = ");
     textLen += sprintf(wto_buff.text+textLen,"%X",rc);
     wto_buff.len = wto_prefixLen + textLen;
     __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
     if ( rc ]= 0 )
       print_diagarea(diagarea);
    }
    else /* else HWICONN to a CPC failed.                         */
      print_diagarea(diagarea);
    /* Advance to the next CPC in the list.                       */
    i++;
   } /* end do while CPCs                                        */
}   /* end if HWILIST to list CPCs succeeded.                    */
```

```
 else /* else HWILIST failed.                                          */
   print_diagarea(diagarea);
 /* ============================================================== */
 /*                                                                */
 /* SAMPLE Routines.                                               */
 /*                                                                */
 /* To try BCPii services HWIEVENT and HWICMD, uncomment the call  */
 /*   to Try_EventAndCommand, recompile, and relink load module    */
 /*   HWIXMCS1.                                                    */
 /* To try the BCPii HWISET service, uncomment the call to Try_Set,*/
 /*   recompile and relink load module HWXMCS1.  The HWISET service*/
 /*   is available only in z/OS 1.11 and higher.                   */
 /*                                                                */
 /* ============================================================== */
 /*
  Try_EventAndCommand();
  Try_Set();
 */
  return 0; /* end of mainline code                              */
}
 /* ============================================================== */
 /*  Try_EventAndCommand will invoke a series of subroutines:      */
 /*    ConnectToCPC to get a CPC connect token.                    */
 /*    ConnectToImage to get an image connect token.               */
 /*    Call_HWIEVENT to call the BCPii HWIEVENT service.           */
 /*    Call_HWICMD to call the BCPii HWICMD service.               */
 /*    DisconnectFromImage to release an image connect token.      */
 /*    DisconnectFromCPC to release a CPC connect token.           */
 /* ============================================================== */
 void  Try_EventAndCommand(void)
 {
 int rc;
 HWI_CONNTOKEN_TYPE CPCConnectToken;
 HWI_CONNTOKEN_TYPE ImageConnectToken;
 HWI_DIAGAREA_TYPE diagarea;
 struct WTO_PARM {       /* mapped by IEZWPL macro.                */
    unsigned short len;  /* total length of structure              */
    unsigned short code; /* routing code flag                      */
    char textŸ80¨;
    } tryec_buff;
 int textLen;
 int tryec_prefixLen;
 textLen = 0;
 tryec_prefixLen = sizeof(tryec_buff.len) + sizeof(tryec_buff.code);
 tryec_buff.code = 0;
 /* ------------------------------------------------------------- */
 /* Write to operator to say that Try_EventAndCommand was called  */
 /* ------------------------------------------------------------- */
 memset(tryec_buff.text,0,sizeof(tryec_buff.text));
 textLen = sprintf(tryec_buff.text,"%s",
          " =>> Routine Try_EventAndCommand has been called.");
 tryec_buff.len = tryec_prefixLen + textLen;
 __asm( " WTO MF=(E,(%0)) " : : "r"(&tryec_buff));
 /* ------------------------------------------------------------- */
 /* Get a CPC connect token and an image connect token            */
```

```
/* ------------------------------------------------------------- */
rc = 0;
ConnectToCPC(&rc, &CPCConnectToken);
if(rc==0)
   ConnectToImage(&rc, CPCConnectToken, &ImageConnectToken);
if(rc==0)
  {
  /* ------------------------------------------------------------- */
  /* Invoke "Call_HwiEvent", which is a sample routine that calls   */
  /* HWIEVENT to register for a command completion event, and to    */
  /* provide the address of an event exit routine.                  */
  /*                                                               */
  /* The BCPii sample exit, HWIXMCX1, is available for use with     */
  /* this C sample program.  The exit must have been previously     */
  /* compiled and linked into a load module which is available in   */
  /* common storage.                                               */
  /* ------------------------------------------------------------- */
   Call_HWIEVENT(&rc, ImageConnectToken, &diagarea);
  /* ------------------------------------------------------------- */
  /* Write to operator to report results from HWIEVENT.             */
  /* ------------------------------------------------------------- */
   memset(tryec_buff.text,0,sizeof(tryec_buff.text));
   textLen = sprintf(tryec_buff.text,"%s",
                     " =>> Register for a command response EVENT.");
   textLen += sprintf(tryec_buff.text+textLen,"%s",
                                       " HWIEVENT Retcode = ");
   textLen += sprintf(tryec_buff.text+textLen,"%X",rc);
   tryec_buff.len = tryec_prefixLen + textLen;
   __asm( " WTO MF=(E,(%0)) " : : "r"(&tryec_buff));
   if(rc) print_diagarea(diagarea);
  /* ------------------------------------------------------------- */
  /* Invoke "Call_HwiCmd", which is a sample routine that calls     */
  /* HWICMD to issue a simple system console command.               */
  /* The return code from HWICMD indicates that a command has been  */
  /* accepted.                                                     */
  /*                                                               */
  /* Your user must have Control authority to issue a command.      */
  /*                                                               */
  /* When a command completes, a command completion event will      */
  /* occur.  If you have registered to receive notification of      */
  /* command completion events, and you have provided the address   */
  /* of an event exit routine (through HWIEVENT), then the Event    */
  /* Notification Facility (ENF) will call the event exit upon      */
  /* command completion.                                           */
  /* -------------------------------------------------------------*/
   Call_HWICMD(&rc, ImageConnectToken, &diagarea);
  /* ------------------------------------------------------------- */
  /* Write to operator to report results from HWICMD.               */
  /* ------------------------------------------------------------- */
   memset(tryec_buff.text,0,sizeof(tryec_buff.text));
   textLen = sprintf(tryec_buff.text,"%s",
                           " =>> Issue a COMMAND to display GRS.");
   textLen += sprintf(tryec_buff.text+textLen,"%s",
                                       " HWICMD Retcode = ");
   textLen += sprintf(tryec_buff.text+textLen,"%X",rc);
```

```
    tryec_buff.len = tryec_prefixLen + textLen;
    __asm( " WTO MF=(E,(%0)) " : : "r"(&tryec_buff));
    if(rc) print_diagarea(diagarea);
    DisconnectFromImage(ImageConnectToken);
    DisconnectFromCPC(CPCConnectToken);
   }
  else
   {
   /* ---------------------------------------------------------- *
   /* Write to operator to report results from HWICONN           *
   /* ---------------------------------------------------------- *
    memset(tryec_buff.text,0,sizeof(tryec_buff.text));
    textLen = sprintf(tryec_buff.text,"%s",
                             " =>> Connect to CPC or image failed.");
    textLen += sprintf(tryec_buff.text+textLen,"%s",
                                       " HWICONN Retcode = ");
    textLen += sprintf(tryec_buff.text+textLen,"%X",rc);
    tryec_buff.len = tryec_prefixLen + textLen;
    __asm( " WTO MF=(E,(%0)) " : : "r"(&tryec_buff));
   }
 }
/* ============================================================== */
/*  Try_Set will invoke a series of subroutines:                  */
/*    ConnectToCPC to get a CPC connect token.                    */
/*    Call_HWISET to call the BCPii HWISET service (available in  */
/*                z/OS 1.11 and higher.)                          */
/*    DisconnectFromCPC.                                          */
/* ============================================================== */
void  Try_Set(void)
{
 int rc;
 HWI_CONNTOKEN_TYPE CPCConnectToken;
 HWI_DIAGAREA_TYPE  diagarea;
 struct WTO_PARM {      /* mapped by IEZWPL macro.               */
    unsigned short len;  /* total length of structure            */
    unsigned short code; /* routing code flag                    */
    char textÝ8O¨;
    } tryset_buff;
 int textLen;
 int tryset_prefixLen;
 textLen = 0;
 tryset_prefixLen = sizeof(tryset_buff.len) + sizeof(tryset_buff.code);
 tryset_buff.code = 0;
 /* ---------------------------------------------------------- */
 /* Write to operator to say that Try_EventAndCommand was called */
 /* ---------------------------------------------------------- */
 memset(tryset_buff.text,0,sizeof(tryset_buff.text));
 textLen = sprintf(tryset_buff.text,"%s",
          " =>> Routine Try_Set has been called.");
 tryset_buff.len = tryset_prefixLen + textLen;
 __asm( " WTO MF=(E,(%0)) " : : "r"(&tryset_buff));
 /* ---------------------------------------------------------- */
/* Get a CPC connect token.                                    */
/* ---------------------------------------------------------- */
rc = 0;
```

```
 ConnectToCPC(&rc, &CPCConnectToken);
 if(rc==0)
   {
   /* ------------------------------------------------------------ */
   /* Invoke "Call_HwiSet", which is a sample routine that calls   */
   /* HWISET to set the attribute of a system object.   In this case,*/
   /* set the value of the Acceptable Status attribute for a CPC.  */
   /*                                                              */
   /* Your user must have Update authority to set an attribute.    */
   /* ------------------------------------------------------------ */
   Call_HWISET(&rc, CPCConnectToken, &diagarea);
   /* ------------------------------------------------------------ */
   /* Write to operator to report results from HWISET.             */
   /* ------------------------------------------------------------ */
   memset(tryset_buff.text,0,sizeof(tryset_buff.text));
   textLen = sprintf(tryset_buff.text,"%s",
             " =>> SET a value for the acceptable status attribute.");
   textLen += sprintf(tryset_buff.text+textLen,"%s",
                                       "  HWISET Retcode = ");
   textLen += sprintf(tryset_buff.text+textLen,"%X",rc);
   tryset_buff.len = tryset_prefixLen + textLen;
   __asm( " WTO MF=(E,(%0)) " : : "r"(&tryset_buff));
   if(rc) print_diagarea(diagarea);
   DisconnectFromCPC(CPCConnectToken);
   }
 else
   {
   /* ------------------------------------------------------------ */
   /* Write to operator to report results from HWICONN             */
   /* ------------------------------------------------------------ */
    memset(tryset_buff.text,0,sizeof(tryset_buff.text));
    textLen = sprintf(tryset_buff.text,"%s",
                              " =>> Connect to CPC failed.");
    textLen += sprintf(tryset_buff.text+textLen,"%s",
                                       " HWICONN Retcode = ");
    textLen += sprintf(tryset_buff.text+textLen,"%X",rc);
    tryset_buff.len = tryset_prefixLen + textLen;
    __asm( " WTO MF=(E,(%0)) " : : "r"(&tryset_buff));
  }
}
/* ============================================================== */
/* Connect to a CPC to obtain a connect token.                    */
/* ============================================================== */
void ConnectToCPC(int  *              returncodePtr,      /* Output */
                  HWI_CONNTOKEN_TYPE * CPCConnectTokenPtr) /* Output */
 {
 HWI_CONNTOKEN_TYPE CPCinconnecttoken;
 char CPC_targetÝ17¨;
 char * CPCConnectTypeValuePtr;
 HWI_DIAGAREA_TYPE diagarea;
 /* ------------------------------------------------------------ */
 /* The CPC SNA Address must be padded with trailing blanks.     */
 /* You can specify a specific CPC SNA address.    e.g.          */
 /*        strncpy(CPC_target,"XYZ390PS.R123"),sizeof(CPC_target));  */
 /* -or-                                                         */
```

```
/* An asterisk can be used to denote the local CPC on which you     */
/* are running.      e.g.                                           */
/*        strncpy(CPC_target,"*",strlen("*"))                        */
/* ---------------------------------------------------------------- */
memset(&CPCinconnecttoken,0x00,sizeof(CPCinconnecttoken));
memset(CPC_target,' ',sizeof(CPC_target));
strncpy(CPC_target,"*",strlen("*"));
CPCConnectTypeValuePtr = &CPC_targetÝO¨;
hwiconn(returncodePtr,
        CPCinconnecttoken,
        CPCConnectTokenPtr,
        HWI_CPC,
        (char **)(&CPCConnectTypeValuePtr),
        &diagarea);
if(*returncodePtr) print_diagarea(diagarea);
}
/* ================================================================ */
/* Connect to an image.                                             */
/* Change the image name from LPC to match an image name on your    */
/* system.                                                          */
 /* ============================================================== */
 void ConnectToImage(int  *      returncodePtr,           /* Output */
            HWI_CONNTOKEN_TYPE   CPCConnectToken,         /* Input  */
            HWI_CONNTOKEN_TYPE * ImageConnectTokenPtr)    /* Output */
 {
 char   Image_targetÝ9¨;
 char * ImageConnectTypeValuePtr;
 HWI_DIAGAREA_TYPE diagarea;
 memset(Image_target,' ',sizeof(Image_target));
 strncpy(Image_target,"LPC",strlen("LPC"));
 ImageConnectTypeValuePtr = &Image_targetÝO¨;
 hwiconn(returncodePtr,
         CPCConnectToken,
         ImageConnectTokenPtr,
         HWI_IMAGE,
         (char **)(&ImageConnectTypeValuePtr),
         &diagarea);
 if(*returncodePtr) print_diagarea(diagarea);
 }
 /* ============================================================== */
 /* Disconnect from an image.                                       */
 /* ============================================================== */
 void DisconnectFromImage(
                     HWI_CONNTOKEN_TYPE  ImageConnectToken) /* Input */
 {
 int    rc;
HWI_DIAGAREA_TYPE diagarea;
hwidisc(&rc, ImageConnectToken, &diagarea);
if(rc) print_diagarea(diagarea);
}
/* ================================================================ */
/* Disconnect from a CPC.                                           */
/* ================================================================ */
void DisconnectFromCPC(HWI_CONNTOKEN_TYPE CPCConnectToken) /* Input */
{
```

```
int    rc;
HWI_DIAGAREA_TYPE diagarea;
hwidisc(&rc, CPCConnectToken, &diagarea);
if(rc) print_diagarea(diagarea);
}
/* ============================================================== */
/*                                                                */
/* This is a SAMPLE routine which illustrates a call to HWIEVENT.   */
/*                                                                */
/*   The purpose of HwiEvent is                                   */
/*     - to register or unregister a user for notification about    */
/*       events of a particular type which might occur on a CPC or  */
/*       or on an image, and                                      */
/*     - to point to a user-defined exit routine which will be      */
/*       called whenever that event occurs.                       */
/*       (A sample exit is provided as HWIXMCX1.)                 */
/*     For example, a user might register to be alerted for changes */
 /*     to the status of a processor by passing event type          */
 /*     HWI_EVENT_STATUS_CHANGE.                                    */
 /*                                                                 */
 /*   Pass these arguments to HWIEVENT:                             */
 /*   = the address of a return code variable.                      */
 /*   = a CPC or an image connect token.                            */
 /*   = an indication whether to register or unregister for alerts   */
 /*       for an event, that is, HWI_EVENT_ADD or HWI_EVENT_DELETE.  */
 /*   = a variable specifying the event(s) in which a user is       */
 /*       interested.  More than one event can be set at a time.    */
 /*       (See the HWI_EVENTIDS_TYPE in HWICIC.H, and the           */
 /*       MVS Callable Services Guide for more information.)        */
 /*   = a value indicating the event exit routine mode.             */
 /*       (HWI_EVENT_TASK)                                          */
 /*   = a four-byte address of the event exit routine.              */
 /*   = a four-byte optional user-defined parameter which is passed  */
 /*       to HWIEVENT, and forwarded to an exit routine by ENF.     */
 /*       (Pass 0 if there is no user data.)                        */
 /*   = a pointer to the caller's diagnostic area.                  */
 /*                                                                 */
 /* ============================================================== */
 void Call_HWIEVENT(int  *              returncodePtr,     /* Output */
                    HWI_CONNTOKEN_TYPE  connectToken,      /* Input  */
                    HWI_DIAGAREA_TYPE * diagareaPtr)       /* Output */
 {
   *returncodePtr = 0;                       /* init the return code */
 int               eventAction;
 HWI_EVENTIDS_TYPE eventIDs;
 int               eventExitMode;
 int               eventExitAddr;
 int               eventExitParm;
 struct WTO_PARM {      /* mapped by IEZWPL macro.                */
   unsigned short len;  /* total length of structure              */
   unsigned short code; /* routing code flag                      */
   char textÝ80¨;
   } evt_buff;
 int textLen;
 int evt_prefixLen;
```

```
 textLen = 0;
 evt_prefixLen = sizeof(evt_buff.len) + sizeof(evt_buff.code);
 evt_buff.code = 0;
 /* ------------------------------------------------------------- */
 /* Initialize HWIEVENT parameters                                */
 /* ------------------------------------------------------------- */
 memset(&eventIDs,0,sizeof(eventIDs));
 memset(&eventExitParm,0,sizeof(eventExitParm));
 eventAction   = HWI_EVENT_ADD;
 strcpy(eventIDs.Hwi_EventID_EyeCatcher,HWI_EVENTID_TEXT);
 eventIDs.Hwi_Event_CmdResp = 1;
 eventExitMode = HWI_EVENT_TASK;
 eventExitAddr = 0;
 /* ------------------------------------------------------------- */
  /* Retrieve the address of the event exit which must be provided   */
  /* on the call to HWIEVENT, which LOAD returns in register 0.      */
  /* An ENF event exit routine must reside in common storage.        */
  /* ------------------------------------------------------------- */
  __asm ( " LOAD EP=HWIXMCX1  " : "=r"(eventExitEP) :  );
  eventExitAddr = (int)eventExitEP;
  /* ------------------------------------------------------------- */
  /* Report the address of the event exit.                         */
  /* ------------------------------------------------------------- */
  textLen = sprintf(evt_buff.text,"%s",
                              " =>> Event Exit found at address ");
  textLen += sprintf(evt_buff.text+textLen,"%X",eventExitAddr);
  evt_buff.len  = evt_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&evt_buff));
  /* ------------------------------------------------------------- */
  /* Call HWIEVENT.                                                */
  /* ------------------------------------------------------------- */
  hwievent(returncodePtr,
           connectToken,
           eventAction,
           eventIDs,
           eventExitMode,
           eventExitAddr,
           &eventExitParm,
           diagareaPtr);
}
/* ============================================================= */
/*                                                               */
/* This is a SAMPLE routine which illustrates a call to HWICMD.  */
/*                                                               */
/*   Pass these arguments to HWICMD:                             */
/*   = the address of a return code variable.                    */
/*   = a connect token for a CPC, or an image connection.        */
/*   = the type of command being issued.                         */
/*       For example, a cmd type of HWI_OSCMD would indicate that */
/*       a system operating command is being issued.             */
/*   = the address of a pointer to the value of the operating    */
/*     system command.                                           */
/*       For HWI_OSCMD, a user might provide the address of an   */
/*       input command string, such as "d grs".                  */
/*   = a pointer to the caller's diagnostic area.                */
```

```
/*                                                                   */
/*   Remember:  HWICMD responds in two ways.  An immediate response */
/*              indicates whether a command was accepted for         */
/*              processing.  At command completion, an               */
/*              asynchronous result is provided through event        */
/*              processing.                                          */
/*                                                                   */
/* ================================================================= */
void Call_HWICMD(int  *              returncodePtr,    /* Output    */
                 HWI_CONNTOKEN_TYPE  connectToken,     /* Input     */
                 HWI_DIAGAREA_TYPE * diagareaPtr)      /* Output    */
 {
   *returncodePtr = 0;
   int cmdType;
   HWI_CMD_OSCMD_PARM_TYPE   cmdParm;
   HWI_CMD_OSCMD_PARM_TYPE * cmdParmPtr;
   cmdType    = HWI_CMD_OSCMD;
   cmdParmPtr = &cmdParm;
   /* ------------------------------------------------------------- */
   /* Initialize HWICMD parameters                                  */
   /* ------------------------------------------------------------- */
   memset(&cmdParm,0,sizeof(cmdParm));
   cmdParm.PriorityType = HWI_CMD_NONPRIORITY;
   /* An OS command string must be null-terminated.                 */
   strcpy(cmdParm.OSCMDString,"D GRS");
   /* ------------------------------------------------------------- */
   /* Call HWICMD.                                                  */
   /* ------------------------------------------------------------- */
   hwicmd(returncodePtr,
          connectToken,
          cmdType,
          (void **)&cmdParmPtr,
          diagareaPtr);
 }
 /* ================================================================ */
 /*                                                                  */
 /* This is a SAMPLE routine which illustrates a call to HWISET.     */
/*                                                                   */
/*   Pass these arguments to HWISET:                                 */
/*   = the address of a return code variable.                        */
/*   = a connect token for a CPC, image, caprec, or an activation    */
/*     profile connection, depending on the attribute being set.     */
/*   = the name of the attribute whose value will be changed.        */
/*       For example, the acceptable status attribute, HWI_ACCSTAT, */
/*       can be set on a CPC or on an image.                         */
/*   = the address of a pointer to a value to which an attribute     */
/*     will be set.                                                  */
/*       For example, the HWI_ACCSTAT attribute this could be set    */
/*       to Hwmca_Status_Operating or Hwmca_Status_Exceptions, etc.  */
/*       (See the MVS Callable Services publication.)                */
/*   = the length of the input value.                                */
/*       The length of the input value for HWI_ACCSTAT, for example,*/
/*       would be 4 bytes, the length of an unsigned integer.        */
/*   = a pointer to the caller's diagnostic area.                    */
/*                                                                   */
```

```
/* ============================================================== */
void Call_HWISET(int  *              returncodePtr,     /* Output   */
                 HWI_CONNTOKEN_TYPE  connectToken,      /* Input    */
                 HWI_DIAGAREA_TYPE * diagareaPtr)       /* Output   */
{
  *returncodePtr = 0;                       /* init the return code */
  int                    setType;
  HWI_SETTYPEVALUE_PARM setTypeValue;
   void *                setTypeValue_Ptr;
   int                   setTypeValue_Len;
   /* Initialize HWISET parameters                                  */
   setTypeValue_Ptr             = &setTypeValue;
   setType                      = HWI_ACCSTAT;  /* Acceptable Status */
   setTypeValue.Data.IntegerData= HWI_STATUS_EXCEPTIONS;
   setTypeValue_Len             = sizeof(setTypeValue.Data.IntegerData);
   /* ------------------------------------------------------------- */
   /* Call HWISET.                                                  */
   /* ------------------------------------------------------------- */
   hwiset(returncodePtr,
          connectToken,
          setType,
          (void **)&setTypeValue_Ptr,
          setTypeValue_Len,
          diagareaPtr);
 }
 /* ============================================================== */
 /* Utility Routines                                               */
 /* -------------------------------------------------------------- */
 /* Write the contents of the Diagarea to the operator.            */
 /* -------------------------------------------------------------- */
 void print_diagarea(HWI_DIAGAREA_TYPE DA)
 {
   struct WTO_PARM {      /* mapped by IEZWPL macro.               */
     unsigned short len;  /* total length of structure             */
    unsigned short code; /* routing code flag                      */
    char textÝ100¨;
    } wto_buff;
 int textLen = 0;
 int wto_prefixLen = sizeof(wto_buff.len) + sizeof(wto_buff.code);
 wto_buff.code = 0;
 /* ------------------------------------------------------------- */
 /* Write to operator to report the contents of the diagarea.     */
 /* ------------------------------------------------------------- */
 memset(wto_buff.text,0,sizeof(wto_buff.text));
 textLen  = sprintf(wto_buff.text,"%s",
                           "                    *>>DIAGAREA:");
 wto_buff.len = wto_prefixLen + textLen;
 __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
 /* ------------------------------------------------------------- */
 textLen  = sprintf(wto_buff.text,"%s",
                           "                    *>>Diag_Index:");
 textLen += sprintf(wto_buff.text+textLen,"%d",DA.Diag_Index);
 wto_buff.len = wto_prefixLen + textLen;
 __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
 /* ------------------------------------------------------------- */
```

```
  textLen  = sprintf(wto_buff.text,"%s",
                              "                    *>>Diag_Key:");
  textLen += sprintf(wto_buff.text+textLen,"%X",DA.Diag_Key);
  wto_buff.len = wto_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
  /* ------------------------------------------------------------- */
  textLen  = sprintf(wto_buff.text,"%s",
                              "                    *>>Diag_Actual:");
  textLen += sprintf(wto_buff.text+textLen,"%d",DA.Diag_Actual);
  wto_buff.len = wto_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
  /* ------------------------------------------------------------- */
  textLen  = sprintf(wto_buff.text,"%s",
                              "                    *>>Diag_Expected:");
  textLen += sprintf(wto_buff.text+textLen,"%d",DA.Diag_Expected);
  wto_buff.len = wto_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
  /* ------------------------------------------------------------- */
  textLen  = sprintf(wto_buff.text,"%s",
                              "                    *>>Diag_CommErr:");
  textLen += sprintf(wto_buff.text+textLen,"%X",DA.Diag_CommErr);
  wto_buff.len = wto_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
  /* ------------------------------------------------------------- */
  textLen  = sprintf(wto_buff.text,"%s",
                              "                    *>>Diag_Text:");
  textLen += sprintf(wto_buff.text+textLen,"%s",DA.Diag_Text);
  wto_buff.len = wto_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
  /* ------------------------------------------------------------- */
}
 /* ------------------------------------------------------------- */
 /* Save the returned list of CPCs to the answer area.            */
 /* ------------------------------------------------------------- */
 void CPC_AnswerArea_into_Array(char answerareaÝ¨,
                            CPC_elements ListÝ¨,
                            int number)
 {
     int i,j;
     i = 0; j = 0;
     while( i < number)
     {
        ListÝi¨.elementÝ0¨ = answerareaÝj¨;
        ListÝi¨.elementÝ1¨ = answerareaÝj+1¨;
        ListÝi¨.elementÝ2¨ = answerareaÝj+2¨;
        ListÝi¨.elementÝ3¨ = answerareaÝj+3¨;
        ListÝi¨.elementÝ4¨ = answerareaÝj+4¨;
        ListÝi¨.elementÝ5¨ = answerareaÝj+5¨;
        ListÝi¨.elementÝ6¨ = answerareaÝj+6¨;
        ListÝi¨.elementÝ7¨ = answerareaÝj+7¨;
        ListÝi¨.elementÝ8¨ = answerareaÝj+8¨;
        ListÝi¨.elementÝ9¨ = answerareaÝj+9¨;
        ListÝi¨.elementÝ10¨ = answerareaÝj+10¨;
        ListÝi¨.elementÝ11¨ = answerareaÝj+11¨;
        ListÝi¨.elementÝ12¨ = answerareaÝj+12¨;
```

```
        ListÝi¨.elementÝ13¨ = answerareaÝj+13¨;
        ListÝi¨.elementÝ14¨ = answerareaÝj+14¨;
       ListÝi¨.elementÝ15¨ = answerareaÝj+15¨;
       ListÝi¨.elementÝ16¨ = answerareaÝj+16¨;
       ListÝi¨.elementÝ17¨ = answerareaÝj+17¨;
       ListÝi¨.null_char = '\0';
       i++;
       j=j+18;
    }
}
/* ---------------------------------------------------------------- */
/* Save the returned list of images to the answer area.             */
/* ---------------------------------------------------------------- */
void Image_AnswerArea_into_Array(char answerareaÝ¨,
                           IMAGE_elements ListÝ¨,
                           int number)
{
    int i,j;
    i = 0; j = 0;
    while( i < number)
    {
       ListÝi¨.elementÝ0¨ = answerareaÝj¨;
       ListÝi¨.elementÝ1¨ = answerareaÝj+1¨;
       ListÝi¨.elementÝ2¨ = answerareaÝj+2¨;
       ListÝi¨.elementÝ3¨ = answerareaÝj+3¨;
       ListÝi¨.elementÝ4¨ = answerareaÝj+4¨;
       ListÝi¨.elementÝ5¨ = answerareaÝj+5¨;
       ListÝi¨.elementÝ6¨ = answerareaÝj+6¨;
       ListÝi¨.elementÝ7¨ = answerareaÝj+7¨;
       ListÝi¨.elementÝ8¨ = answerareaÝj+8¨;
       ListÝi¨.null_char = '\0';
        i++;
        j=j+9;
     }
 }
```

C.2 JCL to compile, assemble, and link edit

Example C-2, shows the JCL used to compile, assemble, and link edit the C code provided in Example C-1.

Example C-2 JCL to compile and link edit

```
//COMPC2   JOB    (999,POK),CONWAY,MSGCLASS=H,REGION=0M,
//         NOTIFY=&SYSUID
/*JOBPARM S=*,L=9999
//*-------------------------------------------------------------------
//*  C COMPILE PROCEDURE FOR METAL C
//*-------------------------------------------------------------------
//EDCCB PROC INFILE=,                    < INPUT ... REQUIRED
//   CREGSIZ='0M',                       < COMPILER REGION SIZE
//   CRUN=,                              < COMPILER RUNTIME OPTIONS
//   CPARM='LO SO OPTFILE(DD:OPTFILE)',
//   LIBPRFX='CEE',                      < PREFIX FOR LIBRARY DSN
```

```
//   LNGPRFX='CBC',                       < PREFIX FOR LANGUAGE DSN
//   OPTFILE='NULLFILE',                  < PREFIX FOR LANGUAGE DSN
//   OUTFILE=                             < LOAD MODULE LOCATION
//*--------------------------------------------------------------------
//*  COMPILE STEP:
//*--------------------------------------------------------------------
//COMPILE EXEC PGM=CCNDRVR,REGION=&CREGSIZ,
//    PARM=('&CRUN/&CPARM')
//STEPLIB  DD  DSNAME=&LIBPRFX..SCEERUN2,DISP=SHR
//         DD  DSNAME=&LIBPRFX..SCEERUN,DISP=SHR
//         DD  DSNAME=&LNGPRFX..SCCNCMP,DISP=SHR
//SYSIN    DD  DSNAME=&INFILE,DISP=SHR
//SYSLIB   DD  DSNAME=SYS1.SIEAHDRV.H,DISP=SHR
//         DD  PATH='/usr/include/metal/',
//         PATHOPTS=(OWRONLY)
//OPTFILE  DD  DSNAME=&OPTFILE,DISP=SHR
//SYSLIN   DD  DSNAME=&OUTFILE,DISP=SHR
//SYSPRINT DD  SYSOUT=*
//SYSOUT   DD  SYSOUT=*
//SYSCPRT  DD  SYSOUT=*
//        PEND
//*--------------------------------------------------------------------
//*  CALL OF INSTREAM PROCEDURE EDCCB
//*--------------------------------------------------------------------
//CL      EXEC EDCCB,
//        OPTFILE=LAFITTE.CNTL(COPTS),
//        INFILE=LAFITTE.CNTL(HWIQO1),
//        OUTFILE=LAFITTE.SOURCE.ASM(HWIQO1)
//*--------------------------------------------------------------------
//*  ASSEMBLE OUTPUT OF C COMPILE
//*--------------------------------------------------------------------
//ASM1  EXEC ASMACL,PARM.C='GOFF,RENT,LIST(133)',PARM.L='LIST'
//C.SYSLIB DD  DSN=SYS1.MACLIB,DISP=SHR
//         DD  DSN=SYS1.MODGEN,DISP=SHR
//C.SYSIN  DD  DSN=LAFITTE.SOURCE.ASM(HWIQO1),DISP=SHR
//*--------------------------------------------------------------------
//*  LINK METAL MODULE
//*--------------------------------------------------------------------
//L.CSSLIB   DD DSN=SYS1.CSSLIB,DISP=SHR
//L.SYSLMOD  DD DSN=LAFITTE.APF.LOAD(HWIQO1),DISP=SHR
//L.SYSLIN   DD DISP=(OLD,DELETE),DSN=&&OBJ
//           DD *
         INCLUDE CSSLIB(HWISET)
         INCLUDE CSSLIB(HWIEVENT)
         INCLUDE CSSLIB(HWILIST)
         INCLUDE CSSLIB(HWIDISC)
         INCLUDE CSSLIB(HWICONN)
         INCLUDE CSSLIB(HWICMD)
         SETCODE  AC(1)
         NAME HWIQO1(R)
/*
//
//
```

C.2.1 C compiler options

Example C-3 shows the options specified in step CL for compiling the C code shown in Example C-2.

Example C-3 Compiler options

```
SERVICE(BUILD DATE: 10/18/2010)
      METAL
      NOSEARCH
      search(/usr/include/metal/)
```

C.3 BCPii example as a System REXX

Example C-4 shows a REXX procedure calling the BCPii-calling program.

Example C-4 REXX procedure calling BCPii program

```
/*        Rexx                                */
Say 'Query zHybrid'
/*                                            */
Parse upper arg obj1
/*                                            */
"ALLOCATE FILE(LOADLIB) DSN('LAFITTE.APF.LOAD') SHR "
/*                                            */
select
  when obj1 = "CPC" then do
/*                                            */
/* logg-in the event into proper logstream */
/*                                            */
/*  check with RACF if the user running    */
/*            is able to issue this command*/
/*                                            */
   "CALL 'LAFITTE.APF.LOAD(HWIQ01)'"
  end
  otherwise do
     say "zHybrid interface: unknown object:" obj1
     exit 1
  end
end
exit
```

The REXX procedure shown in Example C-4, when inserted into SYS1.ARX, can then be called as a new z/OS command; see Example C-5.

Example C-5 A new z/OS command

```
 SC74 2010347 09:18:18.91 TSU02712 IEA630I  OPERATOR LAFITTE  NOW ACTIVE,   SYSTEM
 SC74 2010347 09:18:25.68 LAFITTE   @Q CPC
 SC74 2010347 09:18:25.69 IRR812I PROFILE ** (G) IN THE STARTED CLASS WAS USED
                                865 TO START AXR03 WITH JOBNAME AXR03.
 SC74 2010347 09:18:25.71 STC02716 $HASP100 AXR03    ON STCINRDR
 SC74 2010347 09:18:25.76 STC02716 $HASP373 AXR03    STARTED
 SC74 2010347 09:18:25.81 STC02716 IEA630I  OPERATOR *AXT0374 NOW ACTIVE, SYSTEM
```

```
 SC74 2010347 09:18:25.83 STC02716 =>> BCPii C Sample starting ... <<=
 SC74 2010347 09:18:26.33 STC02716 =>> LIST all CPCs.    HWILIST Retcode = 0
 SC74 2010347 09:18:26.33 STC02716 =>> Number of CPCs found = 6
 SC74 2010347 09:18:25.82 STC02716 AXR0500I AXREXX OUTPUT DISPLAY 869
                                869 EXECNAME=Q
REQTOKEN=00004000000000000C705BA95383ED8A6
                                869 Query zHybrid
 SC74 2010347 09:18:30.68 TSU02712 IEA631I  OPERATOR LAFITTE  NOW INACTIVE, SYST
```

C.4 Sample C code for BCPii services

Following are two examples of C code provided as part of APAR OA36043 (PTF UA59994). The first example calls BCPii services, and the second example provides a routing for receiving ENF 68 events.

Example C-6 provides a general C code to call BCPii services.

Example C-6 General C code to call BCPii services

```
/* START OF SPECIFICATIONS ******************************************
*                                                                   *
*-------------------------------------------------------------------*
*   PROPRIETARY STATEMENT                                           *
*                                                                   *
*   LICENSED MATERIALS - PROPERTY OF IBM                            *
*   5694-A01 COPYRIGHT IBM CORP. 2011                               *
*   STATUS= HBB7780                                                 *
*                                                                   *
*   END_OF_PROPRIETARY_STATEMENT                                    *
*-------------------------------------------------------------------*
*                                                                   *
*    This SAMPLIB member is an example only.                        *
*    Modify this sample code as needed.                             *
*                                                                   *
*-------------------------------------------------------------------*
*                                                                   *
*    MODULE NAME= HWIXMCS1                                          *
*                                                                   *
*    DESCRIPTIVE NAME= Sample C code calls to BCPii services.       *
*                                                                   *
*    FUNCTION                                                       *
*     This module provides sample calls to these BCPii services:    *
*       HWICMD   - Issue a command.                                 *
*       HWICONN  - Connect to a CPC; connect to an image.           *
*       HWIDISC  - Disconnect from an image; disconnect from a CPC. *
*       HWIEVENT - Register for an Event.                           *
*                - Specify an ENF (Event Notification Facility) exit. *
*       HWILIST  - List CPCs; list the images on a CPC.             *
*       HWIQUERY - Query a CPC attribute; query an Image attribute. *
*       HWISET   - Set an image attribute. Available in z/OS 1.11   *
*                  and higher.                                      *
*                                                                   *
*     CODE FLOW in this sample:                                     *
*       Call HWILIST to list all CPCs in an HMC network.            *
```

```
*         For each CPC returned:                                       *
*           Call HWICONN to connect to the CPC.                        *
*            Call HWIQUERY to query the Model and SNAAddr of the CPC.  *
*            Call HWILIST to list all of the images on the CPC.        *
*               For each image returned:                               *
*                 Call HWICONN to connect to the image on a CPC.       *
*                   Call HWIQUERY to query the OS Type of the image.   *
*                 Call HWIDISC to disconnect from the image on a CPC.  *
*           Call HWIDISC to disconnect from the CPC.                   *
*                                                                      *
*     Optional routines are included which provide sample code for:    *
*       o HWIEVENT to register/unregister for a command-completion     *
*           event. (A sample ENF (Event Notification Facility)         *
*           exit routine is supplied as HWIXMCX1 in samplib)           *
*       o HWICMD to issue a command against an image.                  *
*       o HWISET to set an attribute for an image.                     *
*                                                                      *
*     This sample calls sprintf to format output data and calls        *
*       WTO to make the output immediately available.  You may         *
*       choose to direct the output elsewhere.                         *
*                                                                      *
*    DEPENDENCIES                                                      *
*                                                                      *
*     1. Include the header file HWICIC.H.                             *
*                                                                      *
*     2. To compile this sample without modification, you must         *
*        compile with a C compiler with the "metal" compiler option.   *
*        Include the metal header files.                               *
*                                                                      *
*        Should you choose to compile with a C++ compiler, all         *
*        __asm statements will be ignored in this sample.              *
*                                                                      *
*        Should you choose to compile with a C compiler without        *
*        the metal compile option, any global variables with the       *
*        register storage class specification will not be declared.    *
*                                                                      *
*        See "Notes" section below.                                    *
*                                                                      *
*     3. Assemble using the GOFF and RENT options.                     *
*        See "Notes" section below.                                    *
*                                                                      *
*     4. Link the resultant object file with the SYS1.CSSLIB to build  *
*        an executable file that can use BCPii services.               *
*        Make the load module APF-authorized using the AC=1 link       *
*        option and place it in an APF-authorized dataset.             *
*        See "Notes" section below.                                    *
*                                                                      *
*                                                                      *
*    NOTES:                                                            *
*                                                                      *
* The imbedded __asm WTO statements are included in this sample        *
* to provide immediate output and to demonstrate the code flow.        *
* It is not expected that this mechanism will be employed in a         *
* production environment.  A C compiler is required to generate        *
* inline assembly statements using __asm.                              *
```

```
*                                                                     *
* This BCPii sample provides two methods for compiling, assembling    *
* and linking this Metal C code:                                      *
* 1) JCL method with source code residing in a PDS, or                *
* 2) z/OS UNIX shell script method with source code residing in a     *
*    z/OS UNIX file system.                                           *
*                                                                     *
* JCL Method:                                                         *
* -----------                                                         *
*                                                                     *
* Job to compile and assemble the program.                            *
*                                                                     *
* //CMPXMCS1 JOB '?',' ',                                            *
* //  CLASS=J,MSGCLASS=H,REGION=0M,                                   *
* //  MSGLEVEL=(1,1),NOTIFY=????????                                  *
* //*****************************************************************  *
* //*  This JCL is written to compile the BCPii C Sample,             *
* //*  using the Metal C Compiler                                     *
* //*****************************************************************  *
* //MAMAM PROC INFILE=,                                               *
* // CRUN=,                                                           *
* // CPARM=,                                                          *
* // OUTFILE='&&LOADSET,DISP=(MOD,PASS),SPACE=(TRK,(7,7)),UNIT=SYSDA' *
* //*-------------------------------------------------------------- *
* //*  Compile Step : generate the ASM file                           *
* //*-------------------------------------------------------------- *
* //COMPILE EXEC  PGM=CCNDRVR,REGION=0M,                              *
* // PARM=('&CRUN/&CPARM')                                           *
* //*                                                                *
* //* STEPLIB specifies the location of the SCEERUN2 run-time library *
* //* and C compiler                                                 *
* //STEPLIB  DD DISP=SHR,DSN=CEE.SCEERUN2                            *
* //         DD DISP=SHR,DSN=CBC.SCCNCMP                             *
* //*                                                                *
* //SYSIN    DD DISP=SHR,DSN=&INFILE                                 *
* //*                                                                *
* //* SYSLIB respresnts the location of the IBM-supplied BCPii C     *
* //* header (HWICIC)                                                *
* //SYSLIB   DD DISP=SHR,DSN=SYS1.SIEAHDRV.H                         *
* //*                                                                *
* //SYSLIN   DD DSN=&OUTFILE,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200)    *
* //SYSPRINT DD SYSOUT=*                                             *
* //SYSOUT   DD SYSOUT=*                                             *
* //SYSCPRT  DD SYSOUT=*                                             *
* // PEND                                                           *
* //*-------------------------------------------------------------- *
* //* Specify the source and output files                            *
* //*-------------------------------------------------------------- *
* //*                                                                *
* //* SEARCH argument points to the Metal C compiler include files   *
* //METALC  EXEC MAMAM,                                              *
* // CPARM='NOSEARCH,METAL,SEARCH(/usr/include/metal/,DD:SYSLIB)',   *
* //      INFILE='BCPII.SAMPLE.C(HWIXMCS1)',                         *
* //     OUTFILE='BCPII.SAMPLE.ASM(HWIXMCS1),DISP=SHR'               *
* //*                                                                *
```

```
* //*-------------------------------------------------------------*
* //* Assemble step : generate the listing and OBJ                *
* //*-------------------------------------------------------------*
* //HWIXMCS1   EXEC PGM=ASMA90,REGION=0M,                          *
* //    PARM='ADATA,DECK,RENT,GOFF,LIST(MAX),TERM(WIDE)'           *
* //SYSIN DD DSN=BCPII.SAMPLE.ASM(HWIXMCS1),                       *
* //         DISP=SHR                                              *
* //SYSPUNCH DD DSN=BCPII.SAMPLE.OBJ(HWIXMCS1),DISP=SHR            *
* //SYSLIN   DD  DUMMY                                             *
* //SYSTERM  DD SYSOUT=H                                           *
* //SYSPRINT DD SYSOUT=*                                           *
* //SYSUT1   DD  UNIT=SYSVIO,SPACE=(CYL,(5,5))                     *
* //SYSLIB   DD  DSN=SYS1.MACLIB,DISP=SHR                          *
* //SYSADATA DD  SYSOUT=*                                          *
*                                                                  *
*        Job to link the object code into a PDSE load module       *
*                                                                  *
* //LNKXMCS1 JOB NOTIFY=????????,                                  *
* // MSGLEVEL=1,MSGCLASS=H                                         *
* //*************************************************************  *
* //*  THIS JCL IS WRITTEN TO LINK THE BCPII C SAMPLE              *
* //*                                                             *
* //*  IMPORTANT NOTE:                                            *
* //*     THE LOAD LIBRARY MUST HAVE THE FOLLOWING ATTRIBUTES:    *
* //*     RECFM=U,LRECL=80,BLKSIZE=27998                          *
* //*************************************************************  *
* //SG1 EXEC LINKS,                                                *
* //    PARM='MAP,LET,LIST,RENT,REUS,AMODE=31,AC=1',               *
* //    UNIT='',SER=,N='BCPII.SAMPLE.PDSE',NAME=LOAD,P1=' ',       *
* //    MOD=,P2=' ',OBJ=,CLASS=*                                   *
* //SYSPUNCH DD DUMMY                                              *
* //SYSPRINT DD SYSOUT=*                                           *
* //OBJDSN DD DISP=SHR,DSN=BCPII.SAMPLE.OBJ                        *
* //CSSLIB DD DSN=SYS1.CSSLIB,DISP=SHR                             *
* //*                                                             *
* //SYSLIN DD *                                                    *
*  INCLUDE OBJDSN(HWIXMCS1)                                       *
*  INCLUDE CSSLIB(HWICMD)                                         *
*  INCLUDE CSSLIB(HWICONN)                                        *
*  INCLUDE CSSLIB(HWIDISC)                                        *
*  INCLUDE CSSLIB(HWIEVENT)                                       *
*  INCLUDE CSSLIB(HWILIST)                                        *
*  INCLUDE CSSLIB(HWIQUERY)                                       *
*  INCLUDE CSSLIB(HWISET)                                         *
*  ENTRY   HWIXMCS1                                               *
*  NAME    HWIXMCS1(R)                                            *
*                                                                 *
* z/OS UNIX Shell Script Method                                   *
* -----------------------------                                   *
*  This is an example of a shell script which compiles this C     *
*  sample residing in a z/OS UNIX file system using the Metal C   *
*  compiler option. It then assembles the resultant Assembler     *
*  source code:                                                   *
*                                                                 *
*     xlc -o hwixmcs1.s -S -Wc,metal,                             *
```

```
*     nosearch -I /usr/include/metal,                                  *
*     "//'XXX.XXX.XXX'" hwixmcs1.c                                     *
*                                                                      *
*     as -o hwixmcs1.o -mrent -mgoff -a=xmcs1.txt hwixmcs1.s           *
*                                                                      *
*  where XXX.XXX.XXX is the library where the required header          *
*  files reside, such as SYS1.SIEAHDRV.H.                              *
*                                                                      *
*  This is an example of a shell script which links the object         *
*  code into a load module and writes it to a load library             *
*                                                                      *
*    /usr/bin/linkedit -d -V -wRENT,REUS,MAP,AMODE=31,AC=1 -l          *
*    //XXX.XXX.XXX -i //sys1.csslib -o hwixmcs1                        *
*    ./hwixmcs1.o  "ENTRY HWIXMCS1"                                    *
*                                                                      *
*  where XXX.XXX.XXX is the target load library where the              *
*  load module will reside.                                            *
*                                                                      *
*    REFERENCE:                                                        *
*        See the z/OS MVS Programming: Callable Services for           *
*        High-Level Languages publication for more information         *
*        regarding the usage of BCPii APIs.                            *
*                                                                      *
* Change Activity                                                      *
*   $L0  OA31731  HBB7750 101130 PDRH:  BCPii C Sample code.           *
*   $L1  OA36043  HBB7750 110430 PDLH:  Add JCL methods in prolog to   *
*        compile, assemble, and link sample                            *
*                                                                      *
************************END OF SPECIFICATIONS***********************/
#pragma filetag ("IBM-1047")  /* Define the codepage as EBCDIC.      */
#pragma longName              /* Allow names longer than 8 chars.    */

#include <stdio.h>
#include <string.h>
#include <hwicic.h>

/* ------------------------------------------------------------------ */
/* Type definitions for use of saving the returned list of objects    */
/* from HWIList.                                                      */
/* ------------------------------------------------------------------ */
typedef struct CPC_ELEMENTS {
   char elementÝHWI_CPCNETADDR_LENGTH¨;
   char null_char;
} CPC_ELEMENTS;

typedef struct IMAGE_ELEMENTS {
   char elementÝHWI_IMAGENAME_LENGTH¨;
   char null_char;
} IMAGE_ELEMENTS;

/* ------------------------------------------------------------------ */
/* Function prototypes                                                */
/* ------------------------------------------------------------------ */
void ConnectToCPC(
           int *               returncodePtr,            /* Output */
```

```
          HWI_CONNTOKEN_TYPE * CPCconnecttokenPtr,        /* Output */
          char *               CPCconnecttypevaluePtr);   /* Input  */

void ConnectToImage(
          int *                returncodePtr,             /* Output */
          HWI_CONNTOKEN_TYPE   CPCconnecttoken,           /* Input  */
          HWI_CONNTOKEN_TYPE * IMAGEconnecttokenPtr,      /* Output */
          char *               IMAGEconnecttypevaluePtr); /* Input  */

void IssueBCPiiCommand(
            int  *             returncodePtr,             /* Output */
            HWI_CONNTOKEN_TYPE connectToken,              /* Input  */
            int                cmdType,                   /* Input  */
            void *             cmdParm_Ptr);              /* Input  */

void IssueBCPiiSet(
         int  *             returncodePtr,                /* Output */
         HWI_CONNTOKEN_TYPE connectToken,                 /* Input  */
         int                setType,                      /* Input  */
         void *             setTypeValue_Ptr,             /* Input  */
         int                setTypeValue_Len);            /* Input  */

void ListCPCs(
          int *                returncodePtr,             /* Output */
          int *                numofCPCs,                 /* Output */
          CPC_ELEMENTS *       List_of_CPCs);             /* Output */

void ListImages(
          int *                returncodePtr,             /* Output */
          HWI_CONNTOKEN_TYPE   CPCconnecttoken,           /* Input  */
          int *                numofImages,               /* Output */
          IMAGE_ELEMENTS *     List_of_Images);           /* Output */

void ManageEventRegistration(
          int  *             returncodePtr,               /* Output */
          HWI_CONNTOKEN_TYPE connectToken,                /* Input  */
          int                eventAction,                 /* Input  */
          HWI_EVENTIDS_TYPE  eventIDs,                    /* Input  */
          int                eventExitParm);              /* Input  */

void QueryCPCAttr(
          int *                returncodePtr,             /* Input  */
          HWI_CONNTOKEN_TYPE   connecttoken);             /* Input  */

void QueryIMAGEAttr(
          int *                returncodePtr,             /* Input  */
          HWI_CONNTOKEN_TYPE   connecttoken);             /* Input  */

void ReleaseConnection(
          int *                returncodePtr,             /* Output */
          HWI_CONNTOKEN_TYPE   connecttoken);             /* Input  */

void Try_EventAndCommand(void);

void Try_Set(void);
```

```
void print_diagarea(HWI_DIAGAREA_TYPE DA);
void CPC_AnswerArea_into_Array(char Ý¨,CPC_ELEMENTS Ý¨,int);
void Image_AnswerArea_into_Array(char Ý¨,IMAGE_ELEMENTS Ý¨,int);

/* ------------------------------------------------------------------ */
/* Associate variable eventExitEP with register 0 so that the        */
/* address of an event exit can be retrieved using the LOAD          */
/* instruction in this sample.                                        */
/*                                                                    */
/* The address of an event exit is passed as a parameter on a call    */
/* to HWIEVENT.  It can be hard-coded, or determined by some other    */
/* method.                                                            */
/* Note:  A standard C compiler (without the metal compiler option)   */
/*        will ignore variables declared using the register storage   */
/*        class in a global scope.  This means that the global        */
/*        variable eventExitEP will not be declared.                  */
/* ------------------------------------------------------------------ */
register int * eventExitEP  __asm("r0");

/* ------------------------------------------------------------------ */
/* Main                                                               */
/* ------------------------------------------------------------------ */
main()
{
 int rc, numofCPCs, numofImages;
 int i,j,k;
 char CPC_targetÝHWI_CPCNETADDR_LENGTH¨;
 char IMAGE_targetÝHWI_IMAGENAME_LENGTH¨;
 char *CPCconnecttypevaluePtr, *IMAGEconnecttypevaluePtr;

 HWI_CONNTOKEN_TYPE CPCoutconnecttoken, IMAGEoutconnecttoken;

 CPC_ELEMENTS   List_of_CPCsÝ100¨;
 IMAGE_ELEMENTS List_of_ImagesÝ100¨;

 struct WTO_PARM {      /* mapped by IEZWPL macro.                  */
   unsigned short len;  /* total length of structure                */
   unsigned short code; /* routing code flag                        */
   char textÝ80¨;
   } wto_buff;
 int  textLen;
 int  wto_prefixLen;

 textLen       = 0;
 wto_prefixLen = sizeof(wto_buff.len) + sizeof(wto_buff.code);
 wto_buff.code = 0;                        /* Primary Console Action */

 /* ------------------------------------------------------------------ */
 /* Write to operator to report this sample begins running.           */
 /* ------------------------------------------------------------------ */
 textLen = sprintf(wto_buff.text,"%s",
                              " =>> BCPii C Sample starting ... <<=");
 wto_buff.len  = wto_prefixLen + textLen;
 __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
```

```
/* ------------------------------------------------------------- */
/* Call ListCPCs to list all CPCs in an HMC network to which you    */
/* have authority and then save the list in List_of_CPCs.           */
/* ------------------------------------------------------------- */
ListCPCs(&rc, &numofCPCs, &List_of_CPCsÝO¨);

if ( rc == 0 )  /* if List CPCs succeeds */
 {
  i = 0;
  while( i < numofCPCs )
   {
    /* ------------------------------------------------------------- */
    /* Call to connect to each CPC that was returned from ListCPCs. */
    /* A CPCoutconnecttoken is returned, which is used as the       */
    /*   input connect token on subsequent requests for connections */
    /*   to images, caprecs, etc.                                   */
    /* ------------------------------------------------------------- */
    memset(CPC_target,' ',HWI_CPCNETADDR_LENGTH);
    strcpy(CPC_target,List_of_CPCsÝi¨.element);
    CPCconnecttypevaluePtr = &CPC_targetÝO¨;

    ConnectToCPC(&rc, &CPCoutconnecttoken, CPCconnecttypevaluePtr);

    /* ------------------------------------------------------------- */
    /* Write to operator to report the CPC target for the connect   */
    /* request.                                                     */
    /* ------------------------------------------------------------- */
    memset(wto_buff.text,0,sizeof(wto_buff.text));
    textLen  = sprintf(wto_buff.text,"%s","       > Connect to CPC ");
    textLen += sprintf(wto_buff.text+textLen, "%d\n", i+1 );
    textLen += sprintf(wto_buff.text+textLen, "%s\n",
                                        &List_of_CPCsÝi¨.element );
    wto_buff.len = wto_prefixLen + textLen;
    __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));

    if ( rc == 0 ) /* if connect to CPC succeeds */
     {
      /* ------------------------------------------------------------- */
      /* Call to retrieve some CPC attributes.                        */
      /* Note: additional attributes can be added or changed in       */
      /*       the proc QueryCPCAttr                                  */
      /* ------------------------------------------------------------- */
      QueryCPCAttr(&rc, CPCoutconnecttoken);

      /* ------------------------------------------------------------- */
      /* Call to list images on the CPC, to which you have            */
      /* authority and then save the returned list in                 */
      /* List_of_Images.                                              */
      /* ------------------------------------------------------------- */
      ListImages(&rc, CPCoutconnecttoken, &numofImages,
                 &List_of_ImagesÝO¨);

      if ( rc == 0 )  /* If ListImages succeeds */
       {
```

```
k = 0;
while( k < numofImages)
 {
  /* ---------------------------------------------------- */
  /* Call HWICONN to connect to each image that was       */
  /* returned, using the output CPC connect token as input. */
  /* ---------------------------------------------------- */
   memset(IMAGEoutconnecttoken, 0x00,HWI_CONNTOKEN_LENGTH);
   memset(IMAGE_target,' ',HWI_IMAGENAME_LENGTH);
   strcpy(IMAGE_target,List_of_ImagesŸk¨.element);
   IMAGEconnecttypevaluePtr = &IMAGE_targetŸO¨;

   ConnectToImage(&rc, CPCoutconnecttoken,
       &IMAGEoutconnecttoken, IMAGEconnecttypevaluePtr);

   /* ---------------------------------------------------- */
   /* Write to operator to report the image name for the   */
   /* connect request.                                     */
   /* ---------------------------------------------------- */
   memset(wto_buff.text,0,sizeof(wto_buff.text));
   textLen  = sprintf(wto_buff.text,"%s",
                    "       > Connect to image ");
   textLen += sprintf(wto_buff.text+textLen, "%d\n", k+1 );
   textLen += sprintf(wto_buff.text+textLen, "%s\n",
                                &List_of_ImagesŸk¨.element );
   wto_buff.len = wto_prefixLen + textLen;
   __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));

   if ( rc == 0 )  /* if connect to image succeeds */
    {
     /* ---------------------------------------------------- */
     /* Call to retrieve image attributes for the returned  */
     /* image.                                               */
     /* Note: additional attributes can be added or changed */
     /*       in the proc QueryIMAGEAttr                     */
     /* ---------------------------------------------------- */
     QueryIMAGEAttr(&rc, IMAGEoutconnecttoken);

     /* ---------------------------------------------------- */
     /* Call to disconnect from the image that we           */
     /* previously connected to.                             */
     /* ---------------------------------------------------- */
     ReleaseConnection(&rc,IMAGEoutconnecttoken);

     /* ---------------------------------------------------- */
     /* Write to operator to report the image name for   */
     /* the disconnect request.                              */
     /* ---------------------------------------------------- */
     memset(wto_buff.text,0,sizeof(wto_buff.text));
     textLen = sprintf(wto_buff.text,"%s",
                      "       > Disconnect from image ");
     textLen += sprintf(wto_buff.text+textLen, "%s\n",
                              &List_of_ImagesŸk¨.element );
     wto_buff.len = wto_prefixLen + textLen;
     __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
```

```
             }   /* end if connect to an image succeeds.            */
            /* Advance to the next image in the list.               */
            k++;
          } /* end while loop for images                           */

        }

       /* ------------------------------------------------------------ */
       /* Call to disconnect the CPC that we previously connected to.*/
       /* ------------------------------------------------------------ */
       ReleaseConnection(&rc,CPCoutconnecttoken);

       /* ------------------------------------------------------------ */
       /* Write to operator to report the CPC name for the             */
       /* disconnect request.                                          */
       /* ------------------------------------------------------------ */
       memset(wto_buff.text,0,sizeof(wto_buff.text));
       textLen = sprintf(wto_buff.text,"%s",
                                "       > Disconnect from CPC ");
       textLen += sprintf(wto_buff.text+textLen, "%s\n",
                                      &List_of_CPCsÝi¨.element );
       wto_buff.len = wto_prefixLen + textLen;
       __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));

      }
     /* Advance to the next CPC in the list.                          */
     i++;
    } /* end do while CPCs                                           */
  }   /* end if list CPCs succeeded.                                 */

 /* ============================================================== */
 /*                                                                */
 /* SAMPLE Routines.                                               */
 /*                                                                */
 /* To try BCPii services HWIEVENT and HWICMD, uncomment the call  */
 /*   to Try_EventAndCommand, recompile, and relink load module    */
 /*   HWIXMCS1.                                                    */
 /* To try the BCPii HWISET service, uncomment the call to Try_Set,*/
 /*   recompile and relink load module HWIXMCS1. The HWISET service*/
 /*   is available only in z/OS 1.11 and higher.                   */
 /*                                                                */
 /* ============================================================== */
 /* Try_EventAndCommand(); */
 /* Try_Set();             */

  return 0; /* end of mainline code                                 */
}

 /* ============================================================== */
 /* Try_EventAndCommand()                                          */
 /*                                                                */
 /*  Try_EventAndCommand will invoke a series of subroutines:      */
 /*    ConnectToCPC to get a CPC connect token.                    */
 /*    ConnectToImage to get an image connect token.               */
```

```
/*    Event to call the BCPii HWIEVENT service.                    */
/*    Command to call the BCPii HWICMD service.                    */
/*    ReleaseConnection to release an image connect token.         */
/*    ReleaseConnection to release a CPC connect token.            */
/*                                                                 */
/*  NOTE:                                                          */
/*    BCPii event registration remains active until one of         */
/*    the following occurs:                                        */
/*    1. The connection associated with the event registration     */
/*       is disconnected.                                          */
/*    2. For image connections, the CPC connection associated      */
/*       with the image is disconnected.                           */
/*    3. The address space of the caller is terminated or the job  */
/*       in an initiator has completed.                            */
/*                                                                 */
/*    In order for the BCPii ENF exit to be driven for the event(s) */
/*    previously registered, it is important that the event        */
/*    registration remains active.  Therefore this program should  */
/*    not disconnect or end until event notification is no longer  */
/*    required.                                                    */
/*                                                                 */
/* =============================================================== */
void  Try_EventAndCommand(void)
{
 int    rc, eventAction, eventExitParm, cmdType;
 char   CPC_targetÝHWI_CPCNETADDR_LENGTH¨;
 char   IMAGE_targetÝHWI_IMAGENAME_LENGTH¨;
 char * CPCconnecttypevaluePtr;
 char * IMAGEconnecttypevaluePtr;

 HWI_CONNTOKEN_TYPE        CPCconnecttoken;
 HWI_CONNTOKEN_TYPE        IMAGEconnecttoken;
 HWI_EVENTIDS_TYPE         eventIDs;
 HWI_CMD_OSCMD_PARM_TYPE   cmdParm;
 HWI_CMD_OSCMD_PARM_TYPE * cmdParm_Ptr;

 struct WTO_PARM {       /* mapped by IEZWPL macro.               */
    unsigned short len;  /* total length of structure             */
    unsigned short code; /* routing code flag                     */
    char textÝ80¨;
 } tryec_buff;

 int textLen;
 int tryec_prefixLen;
 int userdata = 0x0B0E0C0A; /* Sample user defined data to be passed
                               to the ENF exit                     */

 textLen = 0;
 tryec_prefixLen = sizeof(tryec_buff.len) + sizeof(tryec_buff.code);
 tryec_buff.code = 0;

 /* ------------------------------------------------------------- */
 /* Write to operator to say that Try_EventAndCommand was called  */
 /* ------------------------------------------------------------- */
 memset(tryec_buff.text,0,sizeof(tryec_buff.text));
```

```
textLen = sprintf(tryec_buff.text,"%s",
         " =>> Routine Try_EventAndCommand has been called.");
tryec_buff.len = tryec_prefixLen + textLen;
__asm( " WTO MF=(E,(%0)) " : : "r"(&tryec_buff));

rc = 0;

/* ------------------------------------------------------------- */
/* Set the CPC target and use it to establish a connection       */
/* Note: '*' is specified here to represent connecting to the local*/
/*       CPC in this case                                        */
/* ------------------------------------------------------------- */
memset(CPC_target,' ',HWI_CPCNETADDR_LENGTH);
CPCconnecttypevaluePtr = &CPC_targetŸO¨;
strncpy(CPC_target,"*",strlen("*"));
ConnectToCPC(&rc, &CPCconnecttoken, CPCconnecttypevaluePtr);

if(rc==0)  /* if connect to CPC succeeds */
  {
   /* ------------------------------------------------------------ */
   /* Set the image target and use it to establish a connection    */
   /* Note: Change 'LPC' to an image name on the target CPC.       */
   /* ------------------------------------------------------------ */
   memset(IMAGE_target,' ',HWI_IMAGENAME_LENGTH);
   IMAGEconnecttypevaluePtr = &IMAGE_targetŸO¨;
   strncpy(IMAGE_target,"LPC",strlen("LPC"));

   ConnectToImage(&rc, CPCconnecttoken,&IMAGEconnecttoken,
                  IMAGEconnecttypevaluePtr);

   if(rc==0)  /* if connect to image succeeds */
     {
      /* ---------------------------------------------------------- */
      /* Initialize and set the parameters to call                  */
      /* ManageEventRegistration.                                   */
      /*   Event IDs are set to register for a command response     */
      /*   event and disabled wait event in this case               */
      /* ---------------------------------------------------------- */
      memcpy(&eventExitParm,&userdata,sizeof(eventExitParm));
      memset(&eventIDs,0,sizeof(eventIDs));
      strcpy(eventIDs.Hwi_EventID_EyeCatcher,HWI_EVENTID_TEXT);
      eventIDs.Hwi_Event_CmdResp = 1;
      eventIDs.Hwi_Event_DisabledWait = 1;
      eventAction = HWI_EVENT_ADD;

      ManageEventRegistration(&rc, IMAGEconnecttoken, eventAction,
                              eventIDs, eventExitParm);

      /* ----------------------------------------------------------*/
      /* Call IssueBCPiiCommand to issue a simple system console   */
      /* command to display GRS in this case.                      */
      /*                                                           */
      /* The return code from HWICMD indicates that a command has  */
      /* been accepted.                                            */
      /*                                                           */
```

```
      /* Your user must have Control authority to issue a command. */
      /*                                                           */
      /* When a command completes, a command response event will   */
      /* occur.  If you have registered to receive notification of */
      /* command response events, and you have provided the        */
      /* address of an event exit routine (through HWIEVENT), then */
      /* the Event Notification Facility (ENF) will call the event */
      /* exit upon command completion.                             */
      /* ----------------------------------------------------------*/

      /* ----------------------------------------------------------*/
      /* Initialize the cmdParm to null.                           */
      /* Note: An OS command string must be null-terminated.       */
      /* ----------------------------------------------------------*/

      memset(&cmdParm,0,sizeof(cmdParm));
      cmdParm_Ptr = &cmdParm;
      cmdParm.PriorityType = HWI_CMD_NONPRIORITY;
      strcpy(cmdParm.OSCMDString,"D GRS");
      cmdType = HWI_CMD_OSCMD;

      IssueBCPiiCommand(
                    &rc, IMAGEconnecttoken, cmdType, cmdParm_Ptr);

      /* ----------------------------------------------------------*/
      /* Write to operator to report a display GRS command was     */
      /* issued                                                    */
      /* ----------------------------------------------------------*/
      memset(tryec_buff.text,0,sizeof(tryec_buff.text));
      textLen = sprintf(tryec_buff.text,"%s",
                      "       > Issued a COMMAND to display GRS.");
      tryec_buff.len = tryec_prefixLen + textLen;
      __asm( " WTO MF=(E,(%0)) " : : "r"(&tryec_buff));

      /* ----------------------------------------------------------*/
      /* Calls to disconnect the image and CPC connections         */
      /*                                                           */
      /* WARNING:                                                  */
      /*  Before releasing the connections, you may wish to        */
      /*  consider adding programming logic here to keep the       */
      /*  program from proceeding.  If the connections are         */
      /*  released, your ENF exit may not receive control.         */
      /*  See NOTE above regarding event notification              */
      /*  considerations.                                          */
      /* ----------------------------------------------------------*/
      ReleaseConnection(&rc,IMAGEconnecttoken);
      ReleaseConnection(&rc,CPCconnecttoken);
     } /* end if Connect to Image succeeds. */
   } /* end if Connect to CPC succeeds. */
} /* end Try_EventAndCommand                                      */

/* ============================================================= */
/* Try_Set()                                                     */
/*                                                               */
/*  Try_Set will invoke a series of subroutines:                 */
```

```
/*    ConnectToCPC to create a CPC connection.                     */
/*    IssueBCPiiSet to set a value to an attribute for an image.   */
/*    ReleaseConnection to release the CPC connection.             */
/*                                                                 */
/*  Note: The BCPii HWISET service is only available in z/OS 1.11  */
/*        and higher.                                              */
/*                                                                 */
/* =============================================================== */
void  Try_Set(void)
{
 struct WTO_PARM {       /* mapped by IEZWPL macro.               */
    unsigned short len;  /* total length of structure             */
    unsigned short code; /* routing code flag                     */
    char textÝ80¨;
 } tryset_buff;

 int textLen;
 int tryset_prefixLen;

 int rc, setType, setTypeValue, setTypeValue_Len;
 char   CPC_targetÝHWI_CPCNETADDR_LENGTH¨;
 char * CPCconnecttypevaluePtr;
 void * setTypeValue_Ptr;

 HWI_CONNTOKEN_TYPE CPCconnecttoken;
 HWI_DIAGAREA_TYPE  diagarea;

 rc = 0;
 textLen = 0;
 tryset_prefixLen = sizeof(tryset_buff.len) + sizeof(tryset_buff.code);
 tryset_buff.code = 0;

 /* ------------------------------------------------------------- */
 /* Write to operator to say that Try_Set was called.             */
 /* ------------------------------------------------------------- */
 memset(tryset_buff.text,0,sizeof(tryset_buff.text));
 textLen = sprintf(tryset_buff.text,"%s",
          " =>> Routine Try_Set has been called.");
 tryset_buff.len = tryset_prefixLen + textLen;
 __asm( " WTO MF=(E,(%0)) " : : "r"(&tryset_buff));

 /* ------------------------------------------------------------- */
 /* Set the CPC target and use it to establish a connection       */
 /* Note: '*' is specified here to represent connecting to the host */
 /*       CPC in this case                                        */
 /* ------------------------------------------------------------- */
 memset(CPC_target,' ',HWI_CPCNETADDR_LENGTH);
 strncpy(CPC_target,"*",strlen("*"));
 CPCconnecttypevaluePtr = &CPC_targetÝ0¨;

 ConnectToCPC(&rc, &CPCconnecttoken, CPCconnecttypevaluePtr);

 if(rc==0)  /* if connect to CPC succeeds */
   {
   /* ---------------------------------------------------------- */
```

```
   /* Call IssueBCPiiSet to invoke HWISET to set a value to an      */
   /* attribute for an object.  In this case, set the value of the  */
   /* Acceptable Status attribute for a CPC.                        */
   /*                                                               */
   /* The user must have Update authority to set an attribute.      */
   /* ------------------------------------------------------------- */

    setType           = HWI_ACCSTAT;/* Acceptable Status */
    setTypeValue      = HWI_STATUS_EXCEPTIONS;
    setTypeValue_Ptr  = &setTypeValue;
    setTypeValue_Len  = sizeof(setTypeValue);

    IssueBCPiiSet(&rc, CPCconnecttoken, setType, setTypeValue_Ptr,
                  setTypeValue_Len);

   /* ------------------------------------------------------------- */
   /* Write to operator to report the attribute to set.             */
   /* ------------------------------------------------------------- */

   memset(tryset_buff.text,0,sizeof(tryset_buff.text));
   textLen = sprintf(tryset_buff.text,"%s",
       "       > SET a value for the acceptable status attribute.");
   tryset_buff.len = tryset_prefixLen + textLen;
   __asm( " WTO MF=(E,(%0)) " : : "r"(&tryset_buff));

   ReleaseConnection(&rc,CPCconnecttoken);
   } /* end if Connect to CPC succeeds */
} /* end Try_Set */

/* ============================================================== */
/* ConnectToCPC()                                                 */
/*                                                                */
/*  Call HWICONN to connect to a CPC to obtain a connect token.   */
/*                                                                */
/* ============================================================== */
void ConnectToCPC(
             int *                returncodePtr,          /* Output */
             HWI_CONNTOKEN_TYPE * CPCconnecttokenPtr,     /* Output */
             char *               CPCconnecttypevaluePtr) /* Input  */
{
 struct WTO_PARM {          /* mapped by IEZWPL macro.             */
  unsigned short len;       /* total length of structure           */
  unsigned short code;      /* routing code flag                   */
   char textÝ80¨;
 } conc_buff;

 int textLen;
 int conc_prefixLen;

 HWI_CONNTOKEN_TYPE localCPCinconnecttoken;
 HWI_DIAGAREA_TYPE  diagarea;

 textLen = 0;
 conc_prefixLen = sizeof(conc_buff.len) + sizeof(conc_buff.code);
 conc_buff.code = 0;                     /* Primary Console Action */
```

```
 /* ------------------------------------------------------------- */
 /* Write to operator to report HWICONN is starting.              */
 /* ------------------------------------------------------------- */
 memset(conc_buff.text,0,sizeof(conc_buff.text));
 textLen  = sprintf(conc_buff.text,"%s\n"," =>> HWICONN CPC starts:");
 conc_buff.len = conc_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&conc_buff));

 /* ------------------------------------------------------------- */
 /* localCPCinconnecttoken is ignored when connecting to a CPC.   */
 /* ------------------------------------------------------------- */
 memset(&localCPCinconnecttoken,0x00,HWI_CPCNETADDR_LENGTH);

 /* ------------- */
 /* Call HWICONN  */
 /* ------------- */
 hwiconn(returncodePtr,
         localCPCinconnecttoken,
         CPCconnecttokenPtr,
         HWI_CPC,
         &CPCconnecttypevaluePtr,
         &diagarea);

 /* ------------------------------------------------------------- */
 /* If HWICONN fails, call to get diagnostic information.         */
 /* ------------------------------------------------------------- */
 if (*returncodePtr) print_diagarea(diagarea);

 /* ------------------------------------------------------------- */
 /* Write to operator to report HWICONN results.                  */
 /* ------------------------------------------------------------- */
 memset(conc_buff.text,0,sizeof(conc_buff.text));
 textLen = sprintf(conc_buff.text,"%s",
                                   " =>> HWICONN Retcode = ");
 textLen += sprintf(conc_buff.text+textLen,"%X",*returncodePtr);
 conc_buff.len = conc_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&conc_buff));
} /* end ConnectToCPC */

/* ============================================================= */
/* ConnectToImage()                                              */
/*                                                               */
/*  Call HWICONN to connect to an image                          */
/*                                                               */
/* ============================================================= */
void ConnectToImage(int *       returncodePtr,            /* Output */
           HWI_CONNTOKEN_TYPE   CPCconnecttoken,          /* Input  */
           HWI_CONNTOKEN_TYPE * IMAGEconnecttokenPtr,     /* Output */
           char *               IMAGEconnecttypevaluePtr) /* Input  */
{
 struct WTO_PARM {          /* mapped by IEZWPL macro.             */
  unsigned short len;       /* total length of structure           */
  unsigned short code;      /* routing code flag                   */
   char textŸ80¨;
```

```
} coni_buff;

int  textLen;
int  coni_prefixLen;

HWI_DIAGAREA_TYPE diagarea;

textLen = 0;
coni_prefixLen = sizeof(coni_buff.len) + sizeof(coni_buff.code);
coni_buff.code = 0;                      /* Primary Console Action */

/* ------------------------------------------------------------- */
/* Write to operator to report HWICONN is starting.              */
/* ------------------------------------------------------------- */
memset(coni_buff.text,0,sizeof(coni_buff.text));
textLen  = sprintf(coni_buff.text,"%s\n",
                                    " =>> HWICONN IMAGE starts:");
coni_buff.len = coni_prefixLen + textLen;
__asm( " WTO MF=(E,(%0)) " : : "r"(&coni_buff));

/* -------------*/
/* Call HWICONN */
/* -------------*/
hwiconn(returncodePtr,
        CPCconnecttoken,
        IMAGEconnecttokenPtr,
        HWI_IMAGE,
        &IMAGEconnecttypevaluePtr,
        &diagarea);

/* ------------------------------------------------------------- */
/* If HWICONN fails, call to get diagnostic information.         */
/* ------------------------------------------------------------- */
if (*returncodePtr) print_diagarea(diagarea);

/* ------------------------------------------------------------- */
/* Write to operator to report HWICONN results.                  */
/* ------------------------------------------------------------- */
memset(coni_buff.text,0,sizeof(coni_buff.text));
textLen = sprintf(coni_buff.text,"%s",
                                   " =>> HWICONN Retcode = ");
textLen += sprintf(coni_buff.text+textLen,"%X",*returncodePtr);
coni_buff.len = coni_prefixLen + textLen;
 __asm( " WTO MF=(E,(%0)) " : : "r"(&coni_buff));
} /* end ConnectToImage */

/* ============================================================= */
/* IssueBCPiiCommand()                                           */
/*                                                               */
/*   Call HWICMD passing these arguments:                        */
/*   = the address of a return code variable.                    */
/*   = a connect token for a CPC, or an image connection.        */
/*   = the type of command being issued.                         */
/*       For example, a cmd type of HWI_OSCMD would indicate that */
/*       a system operating command is being issued.             */
```

```
/*   = the address of a pointer to the value of the operating      */
/*     system command.                                             */
/*       For HWI_OSCMD, a user might provide the address of an     */
/*       input command string, such as "d grs".                    */
/*   = a pointer to the caller's diagnostic area.                  */
/*                                                                 */
/*   Remember:  HWICMD responds in two ways.  An immediate response */
/*              indicates whether a command was accepted for       */
/*              processing.  At command completion, an             */
/*              asynchronous result is provided through event      */
/*              processing.                                        */
/*                                                                 */
/* =============================================================== */
void IssueBCPiiCommand(int * returncodePtr,            /* Output    */
            HWI_CONNTOKEN_TYPE  connecttoken,          /* Input     */
            int                 cmdType,               /* Input     */
            void *              cmdParm_Ptr)           /* Input     */
{
  struct WTO_PARM {      /* mapped by IEZWPL macro.                */
    unsigned short len;  /* total length of structure              */
    unsigned short code; /* routing code flag                      */
    char textÝ80¨;
  } cmd_buff;

  int textLen;
  int cmd_prefixLen;

  HWI_DIAGAREA_TYPE  diagarea;

  *returncodePtr = 0;
  textLen = 0;
  cmd_prefixLen = sizeof(cmd_buff.len) + sizeof(cmd_buff.code);
  cmd_buff.code = 0;                     /* Primary Console Action */

  /* ------------------------------------------------------------- */
  /* Write to operator to report HWICMD is starting                */
  /* ------------------------------------------------------------- */
  memset(cmd_buff.text,0,sizeof(cmd_buff.text));
  textLen  = sprintf(cmd_buff.text,"%s\n"," =>> HWICMD starts:");
  cmd_buff.len = cmd_prefixLen + textLen;
    __asm( " WTO MF=(E,(%0)) " : : "r"(&cmd_buff));

  /* ------------ */
  /* Call HWICMD  */
  /* ------------ */
  hwicmd(returncodePtr,
         connecttoken,
         cmdType,
         &cmdParm_Ptr,
         &diagarea);

  if(*returncodePtr) print_diagarea(diagarea);

  /* ------------------------------------------------------------- */
  /* Write to operator to report results from HWICMD.              */
```

```
  /* ------------------------------------------------------------ */
  memset(cmd_buff.text,0,sizeof(cmd_buff.text));
  textLen = sprintf(cmd_buff.text,"%s",
                                    " =>> HWICMD Retcode = ");
  textLen += sprintf(cmd_buff.text+textLen,"%X",*returncodePtr);
  cmd_buff.len = cmd_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&cmd_buff));

} /* end IssueBCPiiCommand */

/* ============================================================== */
/* IssueBCPiiSet()                                                */
/*                                                                */
/*   Pass these arguments to HWISET:                              */
/*   = the address of a return code variable.                     */
/*   = a connect token for a CPC, image, caprec, or an activation   */
/*     profile connection, depending on the attribute being set.    */
/*   = the name of the attribute whose value will be changed.       */
/*       For example, the acceptable status attribute, HWI_ACCSTAT, */
/*       can be set on a CPC or on an image.                        */
/*   = the address of a pointer to a value to which an attribute    */
/*     will be set.                                                 */
/*       For example, the HWI_ACCSTAT attribute could be set        */
/*       to Hwmca_Status_Operating or Hwmca_Status_Exceptions, etc. */
/*   = the length of the input value.                               */
/*       The length of the input value for HWI_ACCSTAT, for example,*/
/*       would be 4 bytes, the length of an unsigned integer.       */
/*   = a pointer to the caller's diagnostic area.                   */
/*                                                                  */
/* ============================================================== */
void IssueBCPiiSet(int *               returncodePtr,   /* Output   */
                   HWI_CONNTOKEN_TYPE  connecttoken,    /* Input    */
                   int                 setType,         /* Input    */
                   void *              setTypeValue_Ptr,/* Input    */
                   int                 setTypeValue_Len)/* Input    */
{
  struct WTO_PARM {          /* mapped by IEZWPL macro.            */
   unsigned short len;       /* total length of structure          */
   unsigned short code;      /* routing code flag                  */
   char textÝ8Ø¨;
  } set_buff;

 int  textLen;
 int  set_prefixLen;

 HWI_DIAGAREA_TYPE  diagarea;

 textLen = 0;
 set_prefixLen = sizeof(set_buff.len) + sizeof(set_buff.code);
 set_buff.code = 0;                     /* Primary Console Action */
 /* ------------------------------------------------------------ */
 /* Write to operator to report HWISET is starting               */
 /* ------------------------------------------------------------ */
 memset(set_buff.text,0,sizeof(set_buff.text));
 textLen  = sprintf(set_buff.text,"%s\n",
```

```
                                        " =>> HWISET starts:");
 set_buff.len = set_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&set_buff));

 /* ------------ */
 /* Call HWISET  */
 /* ------------ */
 hwiset(returncodePtr,
        connecttoken,
        setType,
        &setTypeValue_Ptr,
        setTypeValue_Len,
        &diagarea);

 if(*returncodePtr) print_diagarea(diagarea);

 /* ------------------------------------------------------------- */
 /* Write to operator to report the HWISET results                */
 /* ------------------------------------------------------------- */
 memset(set_buff.text,0,sizeof(set_buff.text));
 textLen = sprintf(set_buff.text,"%s", "  HWISET Retcode = ");
 textLen += sprintf(set_buff.text+textLen,"%X",*returncodePtr);
 set_buff.len = set_prefixLen + textLen;
 __asm( " WTO MF=(E,(%0)) " : : "r"(&set_buff));

} /* end IssueBCPiiSet */

/* ============================================================== */
/* ListCPCs()                                                     */
/*                                                                */
/*  Call HWILIST to list all CPCs in an HMC network to which you  */
/*  have authority. Save the returned list in List_of_CPCs array. */
/*                                                                */
/* ============================================================== */
void ListCPCs(int *          returncodePtr,              /* Output */
              int *          numofCPCs,                  /* Output */
              CPC_ELEMENTS * List_of_CPCs)               /* Output */
{

 struct WTO_PARM {          /* mapped by IEZWPL macro.            */
  unsigned short len;       /* total length of structure          */
  unsigned short code;      /* routing code flag                  */
   char textÝ80¨;
 }   lstc_buff;

 int  textLen;
 int  lstc_prefixLen;

 int  answerarealen;
 char answerareaÝ9000¨;
 char *answerarea_ptr;

 HWI_CONNTOKEN_TYPE localCPCconnecttoken;
 HWI_DIAGAREA_TYPE  diagarea;
```

```
answerarealen   = sizeof(answerarea);
answerarea_ptr = &answerareaÝO¨;

/* ------------------------------------------------------------- */
/* localCPCConnecttoken is ignored in the call to HWILIST to     */
/* list CPCs                                                     */
/* ------------------------------------------------------------- */
memset(&localCPCconnecttoken,0xOO,HWI_CPCNETADDR_LENGTH);

textLen = 0;
lstc_prefixLen = sizeof(lstc_buff.len) + sizeof(lstc_buff.code);
lstc_buff.code = 0;                      /* Primary Console Action */

/* ------------------------------------------------------------- */
/* Write to operator to report HWILIST is starting               */
/* ------------------------------------------------------------- */
memset(lstc_buff.text,0,sizeof(lstc_buff.text)); /* clear wto buff */
textLen  = sprintf(lstc_buff.text,"%s\n",
                                    " =>> HWILIST CPCs starts:");
lstc_buff.len = lstc_prefixLen + textLen;
__asm( " WTO MF=(E,(%0)) " : : "r"(&lstc_buff));

/* ------------ */
/* Call HWILIST */
/* ------------ */
hwilist(returncodePtr,
        localCPCconnecttoken,
        HWI_LIST_CPCS,
        numofCPCs,
        &answerarea_ptr,
        answerarealen,
        &diagarea);

if (*returncodePtr == 0 )
 {
  /* ------------------------------------------------------------- */
  /* Write to operator to report the number of CPCs returned.     */
  /* ------------------------------------------------------------- */
  memset(lstc_buff.text,0,sizeof(lstc_buff.text));
  textLen = sprintf(lstc_buff.text,"%s",
                                "       > Number of CPCs found = ");
  textLen += sprintf(lstc_buff.text+textLen, "%d\n",*numofCPCs);
  lstc_buff.len = lstc_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&lstc_buff));

  /* ------------------------------------------------------------- */
  /* Call to save the returned list of CPCs from the answer area.  */
  /* ------------------------------------------------------------- */
  memset(List_of_CPCs, 0xOO, sizeof(List_of_CPCs)); /* initialize */
  CPC_AnswerArea_into_Array(answerarea, List_of_CPCs, *numofCPCs);
 }
else
  /* ------------------------------------------------------------- */
  /* If HWILIST fails, call to get diagnostic information.         */
  /* ------------------------------------------------------------- */
```

```
  print_diagarea(diagarea);

 /* ------------------------------------------------------------ */
 /* Write to operator to report the HWILIST results.             */
 /* ------------------------------------------------------------ */
 memset(lstc_buff.text,0,sizeof(lstc_buff.text)); /* clear wto buff */

 textLen = sprintf(lstc_buff.text,"%s",
                                    " =>> HWILIST Retcode = ");
 textLen += sprintf(lstc_buff.text+textLen, "%X\n",*returncodePtr);
 lstc_buff.len = lstc_prefixLen + textLen;
 __asm( " WTO MF=(E,(%0)) " : : "r"(&lstc_buff));

} /* end ListCPCs */

/* ============================================================== */
/* ListImages()                                                   */
/*                                                                */
/*  Call HWILIST to list the images on the CPC that is represented  */
/*  by the input connect token.  Save the returned list in          */
/*  List_of_Images array.                                         */
/*                                                                */
/* ============================================================== */
void ListImages(int *              returncodePtr,         /* Output */
                HWI_CONNTOKEN_TYPE CPCconnecttoken,       /* Input  */
                int *              numofImages,           /* Output */
                IMAGE_ELEMENTS *   List_of_Images)        /* Output */
{

 struct WTO_PARM {          /* mapped by IEZWPL macro.             */
  unsigned short len;       /* total length of structure           */
  unsigned short code;      /* routing code flag                   */
   char textÝ80¨;
 } lsti_buff;

 int  textLen;
 int  lsti_prefixLen;
 int  answerarealen;
 char answerareaÝ9000¨;
 char *answerarea_ptr;

 HWI_DIAGAREA_TYPE  diagarea;

 answerarealen   = sizeof(answerarea);
 answerarea_ptr = &answerareaÝ0¨;

 textLen = 0;
 lsti_prefixLen = sizeof(lsti_buff.len) + sizeof(lsti_buff.code);
 lsti_buff.code = 0;                      /* Primary Console Action */

 /* ------------------------------------------------------------ */
 /* Write to operator to report HWILIST is starting              */
 /* ------------------------------------------------------------ */
 memset(lsti_buff.text,0,sizeof(lsti_buff.text));
 textLen  = sprintf(lsti_buff.text,"%s\n",
```

```
                                        " =>> HWILIST IMAGES starts:");
 lsti_buff.len = lsti_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&lsti_buff));

 /* -------------*/
 /* Call HWILIST */
 /* -------------*/
 hwilist(returncodePtr,
         CPCconnecttoken,
         HWI_LIST_IMAGES,
         numofImages,
         &answerarea_ptr,
         answerarealen,
         &diagarea);

 if ( *returncodePtr == 0 )
  {
   /* ------------------------------------------------------------ */
   /* Write to operator to report the number of images returned.   */
   /* ------------------------------------------------------------ */
   memset(lsti_buff.text,0,sizeof(lsti_buff.text));
   textLen  = sprintf(lsti_buff.text,"%s",
                    "            Number of Images found = ");
   textLen += sprintf(lsti_buff.text+textLen, "%d\n",*numofImages);
   lsti_buff.len = lsti_prefixLen + textLen;
   __asm( " WTO MF=(E,(%0)) " : : "r"(&lsti_buff));

   /* ------------------------------------------------------------ */
   /* Call to save the returned list of images from the answer area.*/
   /* ------------------------------------------------------------ */
   memset(List_of_Images, 0x00, sizeof(List_of_Images));
   Image_AnswerArea_into_Array(answerarea, List_of_Images,
                               *numofImages);
  }
 else
   /* ------------------------------------------------------------ */
   /* If HWILIST fails, call to get diagnostic information.        */
   /* ------------------------------------------------------------ */
   print_diagarea(diagarea);

 /* ------------------------------------------------------------ */
 /* Write to operator to report the HWILIST results.             */
 /* ------------------------------------------------------------ */
 memset(lsti_buff.text,0,sizeof(lsti_buff.text));
 textLen = sprintf(lsti_buff.text,"%s",
                                  " =>> HWILIST Retcode = ");
 textLen += sprintf(lsti_buff.text+textLen, "%X\n",*returncodePtr);
 lsti_buff.len = lsti_prefixLen + textLen;
 __asm( " WTO MF=(E,(%0)) " : : "r"(&lsti_buff));

} /* end ListImages */

/* ============================================================== */
/* ManageEventRegistration()                                      */
/*                                                                */
```

```
/*   The purpose of HwiEvent is                                        */
/*     - to register or unregister a user for notification about       */
/*       events of a particular type which might occur on a CPC or     */
/*       on an image.                                                  */
/*         For example, a user might register to be alerted for        */
/*         changes to the status of a processor by passing event       */
/*         type HWI_EVENT_STATUS_CHANGE.                               */
/*                                                                     */
/*     - to point to a user-defined exit routine which will be         */
/*       called whenever that event occurs.                            */
/*         A sample exit is provided as HWIXMCX1, which is available*/
/*         for use with this C sample program.  The exit must have     */
/*         been previously compiled and linked into a load module      */
/*         that is available in common storage.                        */
/*                                                                     */
/*   Pass these arguments to HWIEVENT:                                 */
/*   = the address of a return code variable.                          */
/*   = a CPC or an image connect token.                                */
/*   = an indication whether to register or unregister for event       */
/*       notification, that is, HWI_EVENT_ADD or HWI_EVENT_DELETE.     */
/*   = a variable specifying the event(s) in which a user is           */
/*       interested.  More than one event can be set at a time.        */
/*       (See the HWI_EVENTIDS_TYPE in HWICIC.H, or BCPii              */
/*       publication for more information.)                            */
/*   = a value indicating the event exit routine mode.                 */
/*       (HWI_EVENT_TASK)                                              */
/*   = a four-byte address of the event exit routine.                  */
/*   = a four-byte optional user-defined parameter which is passed     */
/*       to HWIEVENT, and forwarded to an exit routine by ENF.         */
/*       (Pass 0 if there is no user data.)                            */
/*   = a pointer to the caller's diagnostic area.                      */
/*                                                                     */
/* ================================================================= */
void ManageEventRegistration(
           int * returncodePtr,                          /* Output */
           HWI_CONNTOKEN_TYPE connecttoken,              /* Input  */
           int                eventAction,               /* Input  */
           HWI_EVENTIDS_TYPE  eventIDs,                  /* Input  */
           int                eventExitParm)             /* Input  */
{
 struct WTO_PARM {      /* mapped by IEZWPL macro.                  */
   unsigned short len;  /* total length of structure                */
   unsigned short code; /* routing code flag                        */
   char textÝ80¨;
 } evt_buff;

 int textLen;
 int evt_prefixLen;
 int eventExitMode, eventExitAddr;
 int userdata = 0x0B0E0C0A;

 HWI_DIAGAREA_TYPE diagarea;

 *returncodePtr = 0;                       /* init the return code */
 textLen = 0;
```

```
 evt_prefixLen = sizeof(evt_buff.len) + sizeof(evt_buff.code);
 evt_buff.code = 0;

 /* ------------------------------------------------------------- */
 /* Initialize HWIEVENT parameters                                */
 /* ------------------------------------------------------------- */
 eventExitMode = HWI_EVENT_TASK;
 eventExitAddr = 0;

 /* ------------------------------------------------------------- */
 /* Retrieve the address of the event exit which must be provided */
 /* on the call to HWIEVENT, which LOAD returns in register 0.    */
 /* BCPii ENF event exit routines must reside in common storage.  */
 /* ------------------------------------------------------------- */
 __asm ( " LOAD EP=HWIXMCX1  " : "=r"(eventExitEP) :  );
 eventExitAddr = (int)eventExitEP;

 /* ------------------------------------------------------------- */
 /* Report the address of the event exit.                         */
 /* ------------------------------------------------------------- */
 memset(evt_buff.text,0,sizeof(evt_buff.text));
 textLen = sprintf(evt_buff.text,"%s",
              " =>> HWIEVENT starts: Event Exit found at address ");
 textLen += sprintf(evt_buff.text+textLen,"%X",eventExitAddr);
 evt_buff.len  = evt_prefixLen + textLen;
 __asm( " WTO MF=(E,(%0)) " : : "r"(&evt_buff));

 /* -------------- */
 /* Call HWIEVENT  */
 /* ---------------*/
 hwievent(returncodePtr,
          connecttoken,
          eventAction,
          eventIDs,
          eventExitMode,
          eventExitAddr,
          &eventExitParm,
          &diagarea);

 if(*returncodePtr) print_diagarea(diagarea);

 /* ------------------------------------------------------------- */
 /* Write to operator to report results from HWIEVENT.            */
 /* ------------------------------------------------------------- */
 memset(evt_buff.text,0,sizeof(evt_buff.text));
 textLen = sprintf(evt_buff.text,"%s",
                                    " =>> HWIEVENT Retcode = ");
 textLen += sprintf(evt_buff.text+textLen,"%X",*returncodePtr);
 evt_buff.len = evt_prefixLen + textLen;
 __asm( " WTO MF=(E,(%0)) " : : "r"(&evt_buff));

} /* end ManageEventRegistration */

/* ============================================================= */
/* QueryCPCAttr()                                                */
```

```
/*                                                                  */
/*  Call HWIQUERY to retrieve SE/HMC managed data for a CPC         */
/*  which is represented by the input connect token                 */
/*                                                                  */
/*  By changing the query parm, you may change or add any           */
/*  attributes to retrieve.                                         */
/*                                                                  */
/*  The following steps are necessary to change the query           */
/*  parameters:                                                     */
/*  1. Declare the attribute values                                 */
/*  2. Set the number of the attributes for the request             */
/*  3. Set the query parm                                           */
/*                                                                  */
/* ================================================================ */
void QueryCPCAttr(int *              returncodePtr,       /* Input  */
                  HWI_CONNTOKEN_TYPE connecttoken)        /* Input  */
{

 struct WTO_PARM {          /* mapped by IEZWPL macro.              */
  unsigned short len;       /* total length of structure            */
  unsigned short code;      /* routing code flag                    */
   char textÝ80¨;
 } qryc_buff;

 int  textLen;
 int  qryc_prefixLen;

 HWI_DIAGAREA_TYPE  diagarea;
 void *queryparm_Ptr;

 /* ---------------------------------------------------------------- */
 /* Specify the desired number of attributes.                        */
 /* ---------------------------------------------------------------- */
 #define NUMOFCPCATTRIBUTES 2
 HWI_QUERYPARM_TYPE queryparmÝNUMOFCPCATTRIBUTES¨;

 /* ---------------------------------------------------------------- */
 /* Declare attribute values.                                        */
 /* ---------------------------------------------------------------- */
 char HWI_MMODEL_valueÝ20¨;
 char HWI_SNAADDR_valueÝ17¨;

 textLen = 0;
 qryc_prefixLen = sizeof(qryc_buff.len) + sizeof(qryc_buff.code);
 qryc_buff.code = 0;                      /* Primary Console Action */

 /* ---------------------------------------------------------------- */
 /* Write to operator to report HWIQUERY is starting.                */
 /* ---------------------------------------------------------------- */
 memset(qryc_buff.text,0,sizeof(qryc_buff.text));
 textLen  = sprintf(qryc_buff.text,"%s\n",
                                   " =>> HWIQUERY CPC starts:");
 qryc_buff.len = qryc_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&qryc_buff));
```

```
/* ------------------------------------------------------------ */
/* Set the query parm.                                          */
/* ------------------------------------------------------------ */
queryparmÝ0¨.AttributeIdentifier = HWI_MMODEL;
queryparmÝ0¨.AttributeValue_Ptr  = &HWI_MMODEL_valueÝ0¨;
queryparmÝ0¨.AttributeValueLen   = sizeof(HWI_MMODEL_value);
queryparmÝ0¨.AttributeValueLenReturned = 0;

queryparmÝ1¨.AttributeIdentifier = HWI_SNAADDR;
queryparmÝ1¨.AttributeValue_Ptr  = &HWI_SNAADDR_valueÝ0¨;
queryparmÝ1¨.AttributeValueLen   = sizeof(HWI_SNAADDR_value);
queryparmÝ1¨.AttributeValueLenReturned = 0;

queryparm_Ptr = (char *)&queryparmÝ0¨;

/* ------------- */
/* Call HWIQUERY */
/* ------------- */
hwiquery(returncodePtr,
         connecttoken,
         &queryparm_Ptr,
         NUMOFCPCATTRIBUTES,
         &diagarea);

if ( *returncodePtr == 0 )
 {
  /* ------------------------------------------------------------ */
  /* Write to operator to report the returned attributes.         */
  /* ------------------------------------------------------------ */
  memset(qryc_buff.text,0,sizeof(qryc_buff.text));
  textLen  = sprintf(qryc_buff.text,"%s",
                                   "       > Model Number is ");
  textLen += sprintf(qryc_buff.text+textLen,"%s",
                                     HWI_MMODEL_value);
  qryc_buff.len = qryc_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&qryc_buff));

  memset(qryc_buff.text,0,sizeof(qryc_buff.text));
  textLen  = sprintf(qryc_buff.text,"%s",
                                   "       > SNA Addr is     ");
  textLen += sprintf(qryc_buff.text+textLen,"%s",
                                     HWI_SNAADDR_value);
  qryc_buff.len = qryc_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&qryc_buff));

 }
else
 /* ------------------------------------------------------------ */
 /* If HWIQUERY fails, call to get diagnostic information.       */
 /* ------------------------------------------------------------ */
 print_diagarea(diagarea);

/* ------------------------------------------------------------ */
/* Write to operator to report HWIQUERY results.                */
/* ------------------------------------------------------------ */
```

```
 memset(qryc_buff.text,0,sizeof(qryc_buff.text));
 textLen = sprintf(qryc_buff.text,"%s",
                                  " =>> HWIQUERY Retcode = ");
 textLen += sprintf(qryc_buff.text+textLen, "%X\n",*returncodePtr);
 qryc_buff.len = qryc_prefixLen + textLen;
 __asm( " WTO MF=(E,(%0)) " : : "r"(&qryc_buff));

} /* end QueryCPCAttr */

/* ============================================================== */
/* QueryIMAGEAttr()                                               */
/*                                                                */
/*  Call HWIQUERY to retrieve SE/HMC managed data for an image    */
/*  which is represented by the input connect token               */
/*                                                                */
/*  By changing the query parm, you may change or add any         */
/*  attributes to retrieve.                                       */
/*                                                                */
/*  The following steps are necessary to change the query         */
/*  parameters                                                    */
/*  1. Declare the attribute values                               */
/*  2. Set the number of the attributes for the request           */
/*  3. Set the query parm                                         */
/*                                                                */
/* ============================================================== */
void QueryIMAGEAttr(int *              returncodePtr,     /* Input  */
                    HWI_CONNTOKEN_TYPE connecttoken)      /* Input  */
{

 struct WTO_PARM {          /* mapped by IEZWPL macro.             */
  unsigned short len;       /* total length of structure           */
  unsigned short code;      /* routing code flag                   */
   char textÝ80¨;
 } qryi_buff;

 int  textLen;
 int  qryi_prefixLen;

 void *queryparm_Ptr;

 /* ------------------------------------------------------------- */
 /* Declare attribute value(s).                                   */
 /* ------------------------------------------------------------- */
 char HWI_OSTYPE_valueÝ20¨;

 HWI_DIAGAREA_TYPE  diagarea;

 /* ------------------------------------------------------------- */
 /* Specify the desired number of attributes.                     */
 /* ------------------------------------------------------------- */
 #define NUMOFIMAGEATTRIBUTES 1
 HWI_QUERYPARM_TYPE queryparmÝNUMOFIMAGEATTRIBUTES¨;

 textLen = 0;
 qryi_prefixLen = sizeof(qryi_buff.len) + sizeof(qryi_buff.code);
```

```
qryi_buff.code = 0;                    /* Primary Console Action */

/* ------------------------------------------------------------- */
/* Write to operator to report HWIQUERY is starting.             */
/* ------------------------------------------------------------- */
memset(qryi_buff.text,0,sizeof(qryi_buff.text));
textLen  = sprintf(qryi_buff.text,"%s\n",
                                    " =>> HWIQUERY IMAGE starts:");
qryi_buff.len = qryi_prefixLen + textLen;
 __asm( " WTO MF=(E,(%0)) " : : "r"(&qryi_buff));

/* ------------------------------------------------------------- */
/* Set the query parm.                                           */
/* ------------------------------------------------------------- */
queryparmÝ0¨.AttributeIdentifier = HWI_OSTYPE;
queryparmÝ0¨.AttributeValue_Ptr  = &HWI_OSTYPE_valueÝ0¨;
queryparmÝ0¨.AttributeValueLen=sizeof(HWI_OSTYPE_value);
queryparmÝ0¨.AttributeValueLenReturned = 0;

/* ------------- */
/* Call HWIQUERY */
/* ------------- */
queryparm_Ptr = (char *)&queryparmÝ0¨;
hwiquery(returncodePtr,
         connecttoken,
         &queryparm_Ptr,
         NUMOFIMAGEATTRIBUTES,
         &diagarea);

if ( *returncodePtr == 0 )
 {
  /* ------------------------------------------------------------- */
  /* Write to operator to report the returned attribute(s).        */
  /* ------------------------------------------------------------- */
  memset(qryi_buff.text,0,sizeof(qryi_buff.text));
  textLen  = sprintf(qryi_buff.text,"%s",
                                   "       > OS Type is ");
  textLen += sprintf(qryi_buff.text+textLen,"%s",
                                        HWI_OSTYPE_value);
  qryi_buff.len = qryi_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&qryi_buff));

 }
else
 /* ------------------------------------------------------------- */
 /* If HWIQUERY fails, call to get diagnostic information.        */
 /* ------------------------------------------------------------- */
 print_diagarea(diagarea);

/* ------------------------------------------------------------- */
/* Write to operator to report HWIQUERY results.                 */
/* ------------------------------------------------------------- */
memset(qryi_buff.text,0,sizeof(qryi_buff.text));
textLen = sprintf(qryi_buff.text,"%s",
                                  " =>> HWIQUERY Retcode = ");
```

```
 textLen += sprintf(qryi_buff.text+textLen, "%X\n",*returncodePtr);
 qryi_buff.len = qryi_prefixLen + textLen;
 __asm( " WTO MF=(E,(%0)) " : : "r"(&qryi_buff));

} /* end QueryIMAGEAttr */

/* ================================================================ */
/* ReleaseConnection()                                              */
/*                                                                  */
/*  Call HWIDISC to disconnect a connection which is represented    */
/*  by the input connect token                                      */
/*                                                                  */
/* ================================================================ */
void ReleaseConnection(int *       returncodePtr,         /* Output */
                       HWI_CONNTOKEN_TYPE connecttoken)   /* Input  */
{
 struct WTO_PARM {          /* mapped by IEZWPL macro.              */
  unsigned short len;       /* total length of structure            */
  unsigned short code;      /* routing code flag                    */
   char textŸ80¨;
 } disc_buff;

 int  textLen;
 int  disc_prefixLen;

 HWI_DIAGAREA_TYPE diagarea;

 textLen = 0;
 disc_prefixLen = sizeof(disc_buff.len) + sizeof(disc_buff.code);
 disc_buff.code = 0;                      /* Primary Console Action */

 /* ---------------------------------------------------------------- */
 /* Write to operator to report HWIDISC is starting.                 */
 /* ---------------------------------------------------------------- */
 memset(disc_buff.text,0,sizeof(disc_buff.text));
 textLen  = sprintf(disc_buff.text,"%s\n"," =>> HWIDISC starts:");
 disc_buff.len = disc_prefixLen + textLen;
   __asm( " WTO MF=(E,(%0)) " : : "r"(&disc_buff));

 /* -------------*/
 /* Call HWIDISC */
 /* -------------*/
 hwidisc(returncodePtr, connecttoken, &diagarea);

 /* ---------------------------------------------------------------- */
 /* If HWIDISC fails, call to get diagnostic information             */
 /* ---------------------------------------------------------------- */
 if (*returncodePtr) print_diagarea(diagarea);

 /* ---------------------------------------------------------------- */
 /* Write to operator to report HWIDISC results.                     */
 /* ---------------------------------------------------------------- */
 memset(disc_buff.text,0,sizeof(disc_buff.text));
 textLen = sprintf(disc_buff.text,"%s",
                                   " =>> HWIDISC Retcode = ");
```

```
 textLen += sprintf(disc_buff.text+textLen,"%X",*returncodePtr);
 disc_buff.len = disc_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&disc_buff));
} /* end ReleaseConnection */


/* ============================================================== */
/* Utility Routines                                               */
/* ============================================================== */

/* ============================================================== */
/* print_diagarea()                                               */
/*                                                                */
/*  Write the contents of the Diagarea to the operator.           */
/*                                                                */
/* ============================================================== */
void print_diagarea(HWI_DIAGAREA_TYPE DA) /* Input    */
{
  struct WTO_PARM {      /* mapped by IEZWPL macro.               */
    unsigned short len;  /* total length of structure             */
    unsigned short code; /* routing code flag                     */
    char textÝ100¨;
    } wto_buff;
  int textLen = 0;
  int wto_prefixLen = sizeof(wto_buff.len) + sizeof(wto_buff.code);
  wto_buff.code = 0;
  /* ------------------------------------------------------------ */
  /* Write to operator to report the contents of the diagarea.    */
  /* ------------------------------------------------------------ */
  memset(wto_buff.text,0,sizeof(wto_buff.text));
  textLen  = sprintf(wto_buff.text,"%s",
                             "                    *>>DIAGAREA:");
  wto_buff.len = wto_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
  /* ------------------------------------------------------------ */
  textLen  = sprintf(wto_buff.text,"%s",
                             "                    *>>Diag_Index:");
  textLen += sprintf(wto_buff.text+textLen,"%X",DA.Diag_Index);
  wto_buff.len = wto_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
  /* ------------------------------------------------------------ */
  textLen  = sprintf(wto_buff.text,"%s",
                             "                    *>>Diag_Key:");
  textLen += sprintf(wto_buff.text+textLen,"%X",DA.Diag_Key);
  wto_buff.len = wto_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
  /* ------------------------------------------------------------ */
  textLen  = sprintf(wto_buff.text,"%s",
                             "                    *>>Diag_Actual:");
  textLen += sprintf(wto_buff.text+textLen,"%X",DA.Diag_Actual);
  wto_buff.len = wto_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
  /* ------------------------------------------------------------ */
  textLen  = sprintf(wto_buff.text,"%s",
                           "                  *>>Diag_Expected:");
```

```
  textLen += sprintf(wto_buff.text+textLen,"%X",DA.Diag_Expected);
  wto_buff.len = wto_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
  /* ------------------------------------------------------------ */
  textLen  = sprintf(wto_buff.text,"%s",
                          "                    *>>Diag_CommErr:"); ;
  textLen += sprintf(wto_buff.text+textLen,"%X",DA.Diag_CommErr);
  wto_buff.len = wto_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
  /* ------------------------------------------------------------ */
  textLen  = sprintf(wto_buff.text,"%s",
                             "                     *>>Diag_Text:");
  textLen += sprintf(wto_buff.text+textLen,"%s\n",DA.Diag_Text);
  wto_buff.len = wto_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
  /* ------------------------------------------------------------ */

} /* end print_diagarea */

/* ============================================================== */
/* CPC_AnswerArea_into_Array()                                    */
/*                                                                */
/*  Save and return CPC names from the answerarea into an array of  */
/*  CPCs.                                                         */
/*  The answerarea contains a collection of blank-separated 17      */
/*  character network addresses with a null terminating character.  */
/* ============================================================== */
void CPC_AnswerArea_into_Array(
                         char         answerareaÝ¨,    /* Input    */
                         CPC_ELEMENTS ListÝ¨,          /* Output   */
                         int          number)          /* Output   */
{
    /*-------------------------------------------------------------*/
    /* i is the index in the CPC List array                        */
    /* j is the position of the current CPC in the answer area     */
    /*-------------------------------------------------------------*/
    int i=0,j=0;

    while( i < number)
    {
       memset(ListÝi¨.element,' ',HWI_CPCNETADDR_LENGTH);
       strncpy(ListÝi¨.element
              ,&answerareaÝj¨
              ,HWI_CPCNETADDR_LENGTH);
       i++;
       j+=18;

    }
} /* end CPC_AnswerArea_into_Array */

/* ============================================================== */
/* Image_AnswerArea_into_Array()                                  */
/*                                                                */
/*  Save and return image names from the answerarea into an array   */
/*  of images.                                                    */
```

```
/*  The answerarea contains a collection of blank-separated 8       */
/*  character names with a null terminating character.              */
/* ================================================================ */
void Image_AnswerArea_into_Array(
                         char answerareaÝ¨,            /* Input    */
                         IMAGE_ELEMENTS ListÝ¨,       /* Output   */
                         int number)                  /* Output   */
{
    /*--------------------------------------------------------------*/
    /* i is the index in the Image List array                       */
    /* j is the position of the current Image in the answer area    */
    /*--------------------------------------------------------------*/
    int i=0,j=0;
    while( i < number)
    {
       memset(ListÝi¨.element,' ',HWI_IMAGENAME_LENGTH);
       strncpy(ListÝi¨.element
              ,&answerareaÝj¨
              ,HWI_IMAGENAME_LENGTH);
    i++;
    j+=9;
    }
} /* end Image_AnswerArea_into_Array */
```

As a complement to Example C-6, we provide Example C-7 showing C code for a routine for receiving ENF 68 events.

Example C-7 C code for a routine for receiving ENF 68 events

```
/* START OF SPECIFICATIONS *****************************************
*                                                                   *
*-------------------------------------------------------------------*
*   PROPRIETARY STATEMENT                                           *
*                                                                   *
*   LICENSED MATERIALS - PROPERTY OF IBM                            *
*   5694-A01 COPYRIGHT IBM CORP. 2011                               *
*   STATUS= HBB7780                                                 *
*                                                                   *
*   END_OF_PROPRIETARY_STATEMENT                                    *
*-------------------------------------------------------------------*
*                                                                   *
*    This SAMPLIB member is an example only.                        *
*    Modify this sample code as needed.                             *
*                                                                   *
*-------------------------------------------------------------------*
*                                                                   *
*    MODULE NAME= HWIXMCX1                                          *
*                                                                   *
*    Description = Sample BCPii Event Exit Routine                  *
*                                                                   *
*    Function    = This routine receives control from the Event     *
*                  Notification Facility (ENF) when BCPii events    *
*                  occur for which a user has registered.           *
*                  An ENF Signal 68 represents a BCPii event.       *
*                                                                   *
*                  To use this sample exit, provide the address of  *
```

```
*                  the exit on a call to HWIEVENT.                    *
*                                                                     *
*                  This sample was written to handle only BCPii       *
*                  events; it demonstrates how to handle the data     *
*                  provided by ENF for a Command Response event and   *
*                  and a Disabled Wait event.                         *
*                                                                     *
*    Input       = Register 0 contains the ENF event code.            *
*                  Register 1 points to a word which contains the     *
*                           address of the ENF parameter list,        *
*                           a six-word structure.                     *
*                    ENF Parameter List pointed to by register 1:     *
*                    Word   Contains                                  *
*                     1     Address of the parameter list supplied by *
*                             the system for this event code.         *
*                     2     Fullword of zeroes                        *
*                     3     The address of the optional data supplied *
*                             by the user.                            *
*                     4     Fullword of zeroes                        *
*                     5     Not applicable.                           *
*                     6     Fullword of zeroes                        *
*                                                                     *
*    Dependencies = Event Notification Facility (ENF) is active.      *
*                                                                     *
*    Restrictions = BCPii ENF exit load modules must reside in Common *
*                   storage.                                          *
*                                                                     *
*    Module Type  = Exit                                              *
*                                                                     *
*    Processor    = C.  Use the METAL compiler option.                *
*                   The Metal compiler option generates Assembler     *
*                   code, which must be assembled and linked into a   *
*                   load module.                                      *
*                  Alternately, write an Exit in Assembler following  *
*                  the logic in this sample C routine.                *
*                                                                     *
*    Notes:                                                           *
*                                                                     *
*     The imbedded __asm WTO statements are included in this sample   *
*     to provide immediate output and to demonstrate the code flow.   *
*     It is not expected that this mechanism will be employed in a    *
*     production environment.  A C compiler is required to generate   *
*     inline assembly statement using __asm.                          *
*                                                                     *
*     BCPii provides sample compile, assemble, and link methods       *
*     for the BCPii samples in the HWIXMCS1 prolog.                   *
*                                                                     *
* Change Activity                                                     *
*  $L0 = OA31731 HBB7750 101130 PDRH:  Sample BCPii ENF Exit.         *
*  $L1 = OA36043 HBB7750 110430 PDLH:  Add a reference to methods in  *
*        prolog to compile, assemble, and link sample                 *
*                                                                     *
************************END OF SPECIFICATIONS***********************/
#include <stdio.h>
#include <string.h>
```

```
#include <hwicic.h>

typedef struct {
  HWIENF68 * ENFEventDataPtr;   /* Data for a specific BCPii event   */
  int    reserved1;
  int    ENFUserData;           /* Optional user-supplied data       */
  int    reserved2;
  int    reserved4;
  int    reserved5;
} ENFDATA_TYPE;

/* ------------------------------------------------------------------ */
/* String constants for the write-to-operator messages.               */
/* ------------------------------------------------------------------ */
char * XIT_BEGIN = " *XIT =>> ";
char * XIT_END   = " *XIT <<= ";
char * EVENT_EXIT_CALLED =
            " *XIT =>> An ENF Event EXIT routine has been called.  ";
char * COMMAND_COMPLETION_RESPONSE_RECEIVED =
                    " *XIT =>> Command Completion Response Received ";
char * DISABLED_WAIT_OCCURRED = " *XIT =>> Disabled Wait Occurred ";
char * FROM_CPC               = "from CPC ";
char * FROM_IMAGE             = "from image ";
char * ON_CPC                 = "on CPC ";
char * ON_IMAGE               = "on image ";
char * ENF_USER_DATA          = " *XIT =>> ENF User-Specified Data ";

int main(ENFDATA_TYPE ENFData)
{
 HWI_CONNTOKEN_TYPE savedConnectToken;

 int     offs, len;
 void  * addr = 0;
 char    objectNameÝHWI_CPCNETADDR_LENGTH¨;
 char    CPCSerialÝ13¨;
 int     PSWValÝ4¨;
 int     userData;
 /* ------------------------------------------------------------------ */
 /* Declare a buffer for the write-to-operator text.                   */
 /* ------------------------------------------------------------------ */
 struct WTO_PARM {      /* mapped by IEZWPL macro.                     */
   unsigned short len;  /* total length of structure                   */
   unsigned short code; /* routing code flag                           */
   char textÝ80¨;
   } wto_buff;

 int textLen = 0;
 int wto_prefixLen = sizeof(wto_buff.len) + sizeof(wto_buff.code);
 wto_buff.code = 0;                        /* Primary Console Action */
 memset(&wto_buff,0,sizeof(wto_buff));     /* Clear wto_buff         */

 memset(objectName,0,sizeof(objectName));
 memset(CPCSerial,0,sizeof(CPCSerial));
 /* ------------------------------------------------------------------ */
 /* Report that the exit has been called.                              */
```

```
/* ---------------------------------------------------------------- */
textLen = sprintf(wto_buff.text,"%s",XIT_BEGIN);
wto_buff.len = wto_prefixLen + textLen;
 __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
memset(&wto_buff,0,sizeof(wto_buff));     /* Clear wto_buff         */

/* ---------------------------------------------------------------- */
/* Handle each of the BCPii event types.                            */
/* ---------------------------------------------------------------- */
switch (ENFData.ENFEventDataPtr->eventType)
  {
   /* ---------------------------------------------------------- */
   /* Event Type Hardware Event                                  */
   /* ---------------------------------------------------------- */
   case HWIENF68_EVENTTYPE_HWEVENT:
     /* -------------------------------------------------------- */
     /* Check for a specific event subtype                       */
     /* -------------------------------------------------------- */
     switch (ENFData.ENFEventDataPtr->eventSubType)
       {
       /* -------------------------------------------------------- */
       /* Handle Command Response event subtype.                   */
       /* -------------------------------------------------------- */
       case HWIENF68_HWEVENT_CMDRESP:
         /* ------------------------------------------------------ */
         /* Look at the eventdata fields for this eventSubType     */
         /* which are mapped by the HWIENF68_CMDRESP_T             */
         /* structure.                                             */
         /* ------------------------------------------------------ */
         /* Check whether this command response is one for         */
         /* which you are waiting by comparing the connect         */
         /* token to a saved connect token.                        */
         /* Consider saving the connect token using Name Token     */
         /* services when the command is issued. Name Token        */
         /* Retrieve services could then be used to extract a      */
         /* saved connect token.                                   */
         /* ------------------------------------------------------ */
         /* Validate a saved connect token.                        */
         /* ------------------------------------------------------ */
         /*  savedConnectToken = xxxx;
         if(savedConnectToken == ENFData.ENFEventDataPtr->
                                 eventData.CmdResp.connectToken)
         */
         if(1)                          /* If connect token matches */
            {
            textLen = sprintf(wto_buff.text,"%s",
                         COMMAND_COMPLETION_RESPONSE_RECEIVED);

            /* ------------------------------------------------- */
            /* Check whether the event source is a CPC or an     */
            /* image.                                            */
            /* ------------------------------------------------- */
            if(ENFData.ENFEventDataPtr->eventSource ==
                                       HWIENF68_EVENTSOURCE_CPC)
              {
```

```
      textLen += sprintf(wto_buff.text+textLen,"%s",
                                                FROM_CPC);
      textLen += sprintf(wto_buff.text+textLen,"%s",
                     (ENFData.ENFEventDataPtr->cpcName));
      }
     else
      if(ENFData.ENFEventDataPtr->eventSource ==
                           HWIENF68_EVENTSOURCE_CPCIMAGE)
      {
      textLen += sprintf(wto_buff.text+textLen,"%s",
                                                FROM_IMAGE);
      textLen += sprintf(wto_buff.text+textLen,"%s",
                    (ENFData.ENFEventDataPtr->imageName));
      }

     wto_buff.len = wto_prefixLen + textLen;
     __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));

     /* ---------------------------------------------------- */
     /* Report the cmdRetCode, cmdType and the               */
     /* eventObjName.                                        */
     /* The eventObjName is an HWIENF68_STRING_T type        */
     /* which consists of an offset into the returned data   */
     /* and a length.                                        */
     /* ---------------------------------------------------- */
     offs =  ENFData.ENFEventDataPtr->
             eventData.CmdResp.eventObjName.offStr;
     len  =  ENFData.ENFEventDataPtr->
             eventData.CmdResp.eventObjName.lenStr;
     addr = (void *)(ENFData.ENFEventDataPtr) + offs;
     memcpy(objectName, addr, len);

     memset(&wto_buff,0,sizeof(wto_buff));
     textLen  = sprintf(wto_buff.text,"%s",
                      " *XIT =>> Command return code is ");
     textLen += sprintf(wto_buff.text+textLen,"%d",
                       ENFData.ENFEventDataPtr->
                       eventData.CmdResp.cmdRetCode);
     textLen += sprintf(wto_buff.text+textLen,"%s",
                                    " for command type ");
     textLen += sprintf(wto_buff.text+textLen,"%d",
                       ENFData.ENFEventDataPtr->
                       eventData.CmdResp.cmdType);
     textLen += sprintf(wto_buff.text+textLen,"%s",
                                                " on ");
     textLen += sprintf(wto_buff.text+textLen,"%s",
                       objectName);

     wto_buff.len = wto_prefixLen + textLen;
     __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
     }

   break;
 /* ---------------------------------------------------- */
 /* Handle Disabled Wait event subtype.                  */
```

```
/* ---------------------------------------------------------- */
case HWIENF68_HWEVENT_DISABLEDWAIT:
  /* ---------------------------------------------------------- */
  /* Look at the eventdata fields for this eventSubType       */
  /* which are mapped by the HWIENF68_DISABLEDWAIT_T          */
  /* structure.                                               */
  /* ---------------------------------------------------------- */
  textLen = sprintf(wto_buff.text,"%s",
                               DISABLED_WAIT_OCCURRED);
  /* ---------------------------------------------------------- */
  /* Check whether the event source is a CPC or an image      */
  /* ---------------------------------------------------------- */
  if(ENFData.ENFEventDataPtr->eventSource ==
                                HWIENF68_EVENTSOURCE_CPC)
    {
     textLen += sprintf(wto_buff.text+textLen,"%s",ON_CPC);
     textLen += sprintf(wto_buff.text+textLen,"%s",
                     (ENFData.ENFEventDataPtr->cpcName));
    }
  else
    if(ENFData.ENFEventDataPtr->eventSource ==
                           HWIENF68_EVENTSOURCE_CPCIMAGE)
      {
      textLen += sprintf(wto_buff.text+textLen,"%s",
                                              ON_IMAGE);
      textLen += sprintf(wto_buff.text+textLen,"%s",
                   (ENFData.ENFEventDataPtr->imageName));
      }

  /* ---------------------------------------------------------- */
  /* Write the completed string to the operator               */
  /* ---------------------------------------------------------- */
  wto_buff.len = wto_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));

  /* ---------------------------------------------------------- */
  /* Look at the eventdata fields specific to this            */
  /* eventSubType, which are mapped by the                    */
  /* HWIENF68_DISABLEDWAIT structure.                         */
  /* ---------------------------------------------------------- */
  /* Report the PartitionId, and the ProcessorNum.            */
  /* ---------------------------------------------------------- */
  memset(&wto_buff,0,sizeof(wto_buff));
  textLen  = sprintf(wto_buff.text,"%s",
                  " *XIT =>> This occurred on partition ");
  textLen += sprintf(wto_buff.text+textLen,"%d",
                    ENFData.ENFEventDataPtr->
                    eventData.DisabledWait.PartitionId);
  textLen += sprintf(wto_buff.text+textLen,"%s",
                                       " for processor ");
  textLen += sprintf(wto_buff.text+textLen,"%d",
                    ENFData.ENFEventDataPtr->
                    eventData.DisabledWait.ProcessorNum);
  wto_buff.len = wto_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
```

```
  /* ---------------------------------------------------- */
  /* Report the CPC Serial number                         */
  /* ---------------------------------------------------- */
  offs =  ENFData.ENFEventDataPtr->
          eventData.DisabledWait.CPCSerialNum.offStr;
  len  =  ENFData.ENFEventDataPtr->
          eventData.DisabledWait.CPCSerialNum.lenStr;
  addr = (void *)(ENFData.ENFEventDataPtr) + offs;
  memcpy(CPCSerial, addr, len);

  memset(&wto_buff,0,sizeof(wto_buff));
  textLen  = sprintf(wto_buff.text,"%s",
                  " *XIT =>> The CPC serial number is ");
  textLen += sprintf(wto_buff.text+textLen,"%s",
                    CPCSerial);

  wto_buff.len = wto_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));

  /* ---------------------------------------------------- */
  /* Report the PSWValue which is 16 bytes long.          */
  /* It is hex data, so save it in an array of integers for  */
  /* correct formatting into the wto_buff.                */
  /* ---------------------------------------------------- */
  offs =  ENFData.ENFEventDataPtr->
          eventData.DisabledWait.PSWValue.offStr;
  len  =  ENFData.ENFEventDataPtr->
          eventData.DisabledWait.PSWValue.lenStr;
  addr = (void *)(ENFData.ENFEventDataPtr) + offs;
  memcpy(PSWVal,    addr,    sizeof(int));
  memcpy(PSWVal+1,  addr+4,  sizeof(int));
  memcpy(PSWVal+2,  addr+8,  sizeof(int));
  memcpy(PSWVal+3,  addr+12, sizeof(int));

  memset(&wto_buff,0,sizeof(wto_buff));
  textLen  = sprintf(wto_buff.text,"%s",
                          " *XIT =>> The PSW value was ");
  /* ---------------------------------------------------- */
  /* Do not drop leading zeroes                           */
  /* Expect a value like 00020000 80000000 00000000 00000CCC */
  /* ---------------------------------------------------- */
  textLen += sprintf(wto_buff.text+textLen,"%08X",PSWValÝ0¨);
  textLen += sprintf(wto_buff.text+textLen,"%08X",PSWValÝ1¨);
  textLen += sprintf(wto_buff.text+textLen,"%08X",PSWValÝ2¨);
  textLen += sprintf(wto_buff.text+textLen,"%08X",PSWValÝ3¨);
  wto_buff.len = wto_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));

  break;
/* ...      other event subtypes
case HWIENF68_HWEVENT_STATUSCHG:
case HWIENF68_HWEVENT_NAMECHG:
case HWIENF68_HWEVENT_ACTPROFCHG:
case HWIENF68_HWEVENT_OBJCREATE:
```

```
    case HWIENF68_HWEVENT_OBJDESTROY:
    case HWIENF68_HWEVENT_OBJEXCEPTION:
    case HWIENF68_HWEVENT_APPLSTARTED:
    case HWIENF68_HWEVENT_APPLENDED:
    case HWIENF68_HWEVENT_OPSYSMSG:
    case HWIENF68_HWEVENT_HWMSG:
    case HWIENF68_HWEVENT_HWMSGDEL:
    case HWIENF68_HWEVENT_CAPACITYCHG:
    case HWIENF68_HWEVENT_CAPACITYRECORD:
    case HWIENF68_HWEVENT_SECURITYEVENT:
    case HWIENF68_HWEVENT_POWERCHANGE:
    */
    default:
      /* ---------------------------------------------------- */
      /* Unknown BCPii Hardware Event value returned.         */
      /* Take appropriate actions.                            */
      /* ---------------------------------------------------- */
      break;
   } /* end switch on the value of the BCPii Hardware event.  */
  break; /* end case Event Type Hardware Event                */

/* ---------------------------------------------------------- */
/* Event Type BCPii Status                                    */
/* ---------------------------------------------------------- */
case HWIENF68_EVENTTYPE_BCPIISTATUS:
  /* ---------------------------------------------------------- */
  /* Check for Status value                                     */
  /* ---------------------------------------------------------- */
  switch (ENFData.ENFEventDataPtr->eventSubType)
    {
     case HWIENF68_BCPIISTATUS_AVAIL:
       /* ---------------------------------------------------- */
       /* Take appropriate actions.                            */
       /* ---------------------------------------------------- */
       break;
     case HWIENF68_BCPIISTATUS_UNAVAIL:
       /* ---------------------------------------------------- */
       /* Take appropriate actions.                            */
       /* ---------------------------------------------------- */
       break;
     default:
       /* ---------------------------------------------------- */
       /* Unknown BCPii Status value returned.                 */
       /* Take appropriate actions.                            */
       /* ---------------------------------------------------- */
       break;
    } /* end switch on the value of the BCPii Status event.   */
  break;      /* end case Event Type BCPii Status              */

/* ---------------------------------------------------------- */
/* Event Type Hardware Communication Error                    */
/* ---------------------------------------------------------- */
case HWIENF68_EVENTTYPE_HWCOMMERROR:
  /* ---------------------------------------------------------- */
  /* Check for Error state.                                     */
```

```
      /* ------------------------------------------------------------ */
      switch (ENFData.ENFEventDataPtr->eventSubType)
        {
         case HWIENF68_HWCOMMERROR_TEMP:
           /* ------------------------------------------------------ */
           /* Take appropriate actions.                              */
           /* ------------------------------------------------------ */
           break;
         case HWIENF68_HWCOMMERROR_PERM:
           /* ------------------------------------------------------ */
           /* Take appropriate actions.                              */
           /* ------------------------------------------------------ */
           break;
         default:
           /* ------------------------------------------------------ */
           /* Unknown BCPii Communication Error value returned.      */
           /* Take appropriate actions.                              */
           /* ------------------------------------------------------ */
             break;
        } /* end switch on the value of the BCPii Communication
             Error event.                                           */
      break; /* end case Event Type Communication Error             */

    default:
      /* ------------------------------------------------------------ */
      /* Unknown BCPii Event Type returned.                         */
      /* Take appropriate actions.                                  */
      /* ------------------------------------------------------------ */
      break;
   }      /* end switch on the BCPii Event Type.                    */

 /* ------------------------------------------------------------------ */
 /* Report the user-supplied data, which can be supplied on a call   */
 /* to HWIEVENT during event registration.                           */
 /* This could be a pointer to some kind of global data or a simple  */
 /* four-byte value.                                                 */
 /* ------------------------------------------------------------------ */
 userData     = (int)ENFData.ENFUserData;
 memset(&wto_buff,0,sizeof(wto_buff));
 textLen      = sprintf(wto_buff.text,"%s",ENF_USER_DATA);
 textLen     += sprintf(wto_buff.text+textLen,"%X", userData);
 wto_buff.len = wto_prefixLen + textLen;
 __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));

 /* ------------------------------------------------------------------ */
 /* Report that the exit has ended.                                  */
 /* ------------------------------------------------------------------ */
 memset(&wto_buff,0,sizeof(wto_buff));     /* Clear wto_buff         */
 textLen      = sprintf(wto_buff.text,"%s",XIT_END);
 wto_buff.len = wto_prefixLen + textLen;
  __asm( " WTO MF=(E,(%0)) " : : "r"(&wto_buff));
 memset(&wto_buff,0,sizeof(wto_buff));     /* Clear wto_buff         */
 return (0);
}
```

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks publications

The following IBM Redbooks publications provide additional information about the topic in this document. Note that some publications referenced in this list might be available in softcopy only.

- *z/OS Intelligent Resource Director*, SG24-5952
- *DFSMS Release 10 Technical Update*, SG24-6120
- *z/OS Distributed File Service zSeries File System Implementation z/OS V1R11,* SG24-6580
- *z/OS Version 1 Release 7 Implementation*, SG24-6755
- *z/OS V1R7 DFSMS Technical Update*, SG24-7225
- *z/OS Version 1 Release 8 Implementation*, SG24-7265
- *z/OS Planned Outage Avoidance Checklist*, SG24-7328
- *z/OS Version 1 Release 9 Implementation*, SG24-7427
- *z/OS V1R8 DFSMS Technical Update*, SG24-7435
- *z/OS Version 1 Release 10 Implementation*, SG24-7605
- *DFSMS V1.10 and EAV Technical Guide*, SG24-7617
- *z/OS Version 1 Release 11 Implementation*, SG24-7729
- *z/OS Version 1 Release 12 Implementation*, SG24-7853
- *z/OS V1R12 Communication Server TCP/IP Implementation Volume 4: Security and Policy-Based Networking*, SG24-7899
- *Using SDSF in a JES3 Environment*, REDP-4531

You can search for, view, download, or order these documents and other Redbooks, Redpapers, Web Docs, draft and additional materials, at the following website:

ibm.com/redbooks

Other publications

These publications are also relevant as further information sources:

- *z/OS Problem Management*, G325-2564
- *z/OS V1R13 Migration*, GA22-7499
- *z/OS Introduction and Release Guide*, GA22-7502
- *z/OS Licensed Program Specifications*, GA22-7503
- *z/OS V1R13 Planning for Installation*, GA22-7504

- *z/OS MVS Diagnosis: Reference*, GA22-7588
- *z/OS UNIX System Services Planning*, GA22-7800
- *z/OS Program Directory*, GI10-0670
- *Program Directory for z/OS Management Facility*, GI11-2886
- *z/OS ICSF System Programmer's Guide*, SA22-7520
- *z/OS ICSF Administrator's Guide*, SA22-7521
- *z/OS ICSF Messages*, SA22-7523
- z/OS *MVS Initialization and Tuning Reference*, SA22-7592
- *z/OS MVS Planning: Global Resource Serialization*, SA22-7600
- *z/OS MVS Planning: Operations*, SA22-7601
- *z/OS MVS Authorized Assembler Services Guide*, SA22-7608
- *z/OS MVS Authorized Assembler Services Reference EDT-IXG*, SA22-7610
- *z/OS MVS Programming: Sysplex Services Reference*, SA22-7618
- *z/OS MVS Setting Up a Sysplex*, SA22-7625
- *z/OS MVS System Commands*, SA22-7627
- *z/OS MVS System Messages Volume 1 (ABA - AOM)*, SA22-7631
- *z/OS MVS System Messages, Volume 7 (IEB - IEE)*, SA22-7637
- *z/OS MVS Using the Subsystem Interface*, SA22-7642
- *z/OS Security Server RACF System Programmer's Guide*, SA22-7681
- *z/OS Security Server RACF Security Administrator's Guide*, SA22-7683
- *z/OS Security Server RACF Command Language Reference*, SA22-7687
- *SMP/E for z/OS Commands*, SA22-7771
- *ServerPac: Using the Installation Dialog*, SA22-7815
- *z/Architecture Principles of Operation*, SA22-7832
- *z/OS Metal C Programming Guide and Reference*, SA23-2225
- *z/OS ICSF Writing PKCS #11 Applications*, SA23-2231
- *z/OS IBM Ported Tools for z/OS: Supplementary Toolkit for z/OS Feature User's Guide and Reference*, SA23-2234
- *z/OS MVS Capacity Provisioning User's Guide*, SA33-8299
- *z/OS IBM z/OS Management Facility Configuration Guide*, SA38-0652
- *z/OS XL C/C++ User's Guide*, SC09-4767
- *Enterprise COBOL for z/OS, V4R2, Programming Guide*, SC23-8529
- *z/OS System SSL Programming*, SC24-5901
- *z/OS Distributed File Service SMB Administration*, SC24-5918
- *z/OS Distributed File Service zSeries File System Administration*, SC24-5989
- *z/OS DFSMS Installation Exits*, SC26-7396
- *z/OS DFSMSdfp Advanced Services*, SC26-7400
- *z/OS DFSMSrmm Managing and Using Removable Media*, SC26-7404
- *z/OS DFSMSrmm Implementation and Customization Guide*, SC26-7405

- *z/OS DFSMS Managing Catalogs*, SC26-7409
- *z/OS Communications Server: IP Configuration Guide*, SC31-8775
- *z/OS Communications Server: IP Configuration Reference*, SC31-8776
- *z/OS Resource Management Facility User's Guide*, SC33-7990
- *z/OS Resource Measurement Facility Report Analysis*, SC33-7991
- *z/OS Resource Measurement Facility Programmer's Guide*, SC33-7994
- *z/OS Common Information Model User's Guide*, SC33-7998
- *z/OS ISPF Services Guide*, SC34-4819
- *z/OS ISPF Edit and Edit Macros*, SC34-4820
- *z/OS DFSMShsm Implementation and Customization Guide*, SC35-0418
- *z/OS DFSMShsm Storage Administration*, SC35-0421
- *z/OS DFSMS OAM Planning, Installation, and Storage Administration Guide for Object Support*, SC35-0426

Online resources

These websites are also relevant as further information sources:

- For the RFC that describes the implementation of ECC for TLS (RC 4492):

  `http://www.ietf.org/rfc.html`
- PSP buckets are on the IBM RETAIN system:

  `http://www14.software.ibm.com/webapp/set2/psp/srchBroker`
- Find the appropriate PSP bucket in the Technical Help Database for Mainframe Preventive Service Planning Buckets:

  `http://www14.software.ibm.com/webapp/set2/psp/srchBroker`
- Download HOLDDATA:

  `http://service.software.ibm.com/holdata/390holddata.html`
- Find further information about CustomPac fee offerings:

  `http://www.ibm.com/services/custompac`
- Find the zfsspace tool, available in text mode:

  `ftp://public.dhe.ibm.com/s390/zos/tools/zfsspace/zfsspace.txt`
- For a REXX utility that allows you to run commands interactively (available from the ITSO tools disk):

  `ftp://www.redbooks.ibm.com/redbooks/SG247035/`
- Find details about the open source version of sudo:

  `http://www.sudo.ws/`
- Find the z/OS version of sudo:

  `http://www-03.ibm.com/systems/z/os/zos/features/unix/ported/suptlk/index.html`
- Find API Java documentation (available in `sblim-cim-client-doc.zip`):

  `http://sourceforge.net/projects/sblim/`

- For the annex at the Unicode Consortium's website:
  http://www.unicode.org/reports/tr9
- To search the RETAIN database, visit the IBMLink/ServiceLink website:
  http://www.ibm.com/ibmlink/servicelink)

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

Symbols

*MODIFY CONFIG command
  add spool volume 714
$D SPOOLDEF,TGSPACE command 695
$DCKPTSPACE command 695
$DSPL command 695–696, 701
$dspl command 698
  new parameters 697
$HASP720 message 709
$MSPL(volser,) command 692, 695–696
$T JOBCLASS(x) command
  JOBRC operand 706
$T JQ(xxx),SPIN,DDNAME=ddname command
  spin a data set 703
$T SPOOLDEF command
  bind spool volumes 701
$TJnnn,SPIN command 703
$TSPL command 700
$TSPOOL RESERVED=NO command 695

Numerics

3390 Model A 540

A

ACL support
  ISPF Option 3.17 148
ADDVOL sequence 138
ADDVOLUME command
  assign default retention method 174
AF_INET UDP request 328
Aggregate movement 289
AL line command 151–152
ALLOWAUTALTER specification
  CFRM policy 85
ALTER function
  messages 87
AMAPDUPL
  PDDU utility program 261
AMATERSE 261
AMS DEFINE command 60
APAR OA29619 284, 288, 298
APAR OA31116
  large page support 468
APAR OA31737 279
APAR OA31806 710
  JES2 spool for down-level systems 702
APAR OA32807 77
APAR OA32925 57, 298
APAR OA33022 123
APAR OA34061 77
APAR OA34579 85
APAR OA34589 278
APAR OA34940 566
APAR OA35929
  WARNUND parameter 51
APAR PM40869 382
APF authorization
  IEBCOPY 109
application Linking Manager interface 390
Application Linking Manager interface services 392
ARC0570I message 143
ARCCMDxx parmlib member
  fast replication 139
  SETSYS PDA(YES|NO)) 139
ARCENF15
  ENF signal 136
ARCMMEXT exit 137–138
ARGPARSE option 450
ARM element name
  CFZ_SRV_ 477
ASCII files 329
ATR247E message 346
ATR274I message 346
AUTHQLVL parameter 225
Auto Migrate 135
automatic direction RRSF node synchronization 210
automatic flush
  JES3 716

B

backup control data set (BCDS) 129
base addressing space 544–545
batch activities 751
batch launcher and toolkit 752
BCDBATCH 758
BCPii
  CPC attributes with V1R13 589
BCPii address space 583–584, 602
  startup and shutdown 603
BCPii APIs 590, 599
BCPii application 599
BCPii community name 595, 597–598
  in the security product 602
BCPii diagnostics tools 586
BCPii logical connection 588
BCPii services 587, 599
  security authorization 591
Bidi transformation 659
BLSCDDIR CLIST 406
BPXEKDA macro 317
BPXMKDIR REXX exec 309
  SYS1.SAMPLIB 309
BPXPRMxx parmlib member
  LOSTMSG (ON | OFF) 312
breakpoint value 542
BSG (Branch in Subspace Group) instruction 461

C

capacity increments 622
Capacity Provisioning Control Center (CPCC) 386
Capacity Provisioning Manager
  CIM connections 385, 388
Capacity Provisioning policy 624
CATMAX parameter 558
CBPDO 34
  z/OSMF delivery options 354
CBRCTI00 parmlib member 128
CBROAMxx parmlib member 124, 127
  SETDISK statement 124
  SETOSMC statement 125
CCCCcccH notation 544
CDILVL option
  LISTCAT LVL command 117
CDS backup 130
CEA 474
CEA address space 353, 474
  BCPii use 603
CEA ENF controls 603
CEEPIPI function 666
CF structure allocation placement 79
CFLEVEL 17 72, 74
CFRM couple data set 74
CFRM policy
  ALLOWAUTALTER processing 85
  ALLOWAUTALTER specification 85
CFZRCUS job 486
CFZSEC job 398, 486
CHAIND command 203
CHAINV command 203–205
CHANGEDATASET COPYFROM subcommand 185, 193
CHANGEDATASET subcommand 193–194
CIB connections 422
CIM 353
CIM client application 472
CIM client for Java API 476
CIM connections
  using z/OSMF 387
CIM instrumentation client 740
CIM server runtime 476
CIM/XML over HTTP 472
CIMSERV RACF profile 480
CIM-XML over HTTP 472
cipher suites 634
CIU enablement feature 626
client_cache_size 305
CLNOS390 34
CLNOS390 job
  DELZOSB job 61
CMDS ABEND command 90
CMDS command 90
  FORCE option 90–91
CMDS FORCE command 92
COLLECTINACTIVE parameter
  PFA support 497
COMMNDxx parmlib member
  configure Runtime Diagnostic 509
  Runtime Diagnostics 508
Common event adapter (CEA)
  z/OSMF 355
Common Information Model 353
  overview 472
Common Information Model (CIM) 737
  z/OSMF 355
common recall queue (CRQ) 140
Common Storage Usage Prediction Repo 528
Configuration Assistant task 409
CONSOLxx INIT statement
  Message Flood Automation 96
CPACF 260
CPACF feature 3863 261
Cross-Zone Requisite SYSMOD report 44
Cryptographic support 39
CSRSI Service 243
CSVDYNEX REQUEST=QUERY 244
CTGCDI option
  LISTCAT command 117
CTIXCFxx parmlib member 72
CTIXESxx parmlib member 72
CTRACE buffer size
  4MB for XCF 72
Customized Offerings Driver 37
cylinder-managed space
  EAV volume 543

D

D GRS,AN,WAITER command 516
D GRS,ANALYZE,WAITER command 222
D IOS,RECOVERY command 254
D IPLINFO,CATALOG command 555
D MSGFLD,MSGRATE 99
D MSGFLD,MSGRATE command 98
D MSGFLD,STATUS command 98–99
D OMVS, O command 321
D OMVS,USERMOUNTS command 323
D OMVS,W command 314, 317
D OMVS,W,A command 514
D OMVS,WAITERS command 314
D OPDATA,MODE command 52
D XCF,COUPLE command
  JES2 spool migration 693
D XCF,S,ALL command 77
D XCF,SYSPLEX command 77
DASD management task 382
Dashboards 407
DB2 database
  OAM configuration 125
DCBE macro
  MULTSDN 119
DEFAULTCMD statement
  Message Flood Automation 97
DELETE UCAT RECOVERY command 114
DELRECOVWNG parameter 114–115
DELZOSB 61
Deployment Checklist 373
deployment objectives 373
Deployment task 368, 371

deployment topology 374
detected system failures 490
DEVSUPxx parmlib member 65
  OCE_ABEND_DESCRIP option 118
  REFUCB function 64
DFS client (DFSCM) 61
DFSMSdss ADRDSSU service
  GIMADR service routine 40
DFSMSrmm parmlib option
  TVEXTPURGE 171
disabled aggregate 296
DISPLAY GRS,CONTENTION command 220
DISPLAY IPLINFO,CATALOG command 109
DISPLAY IPLINFO,WARNUND command 242
DISPLAY OPDATA,TRACKING command
  Tracking information 626
Distributed Data Server (DDS)
  z/OSMF 408
Distributed Management Task Force 353
Distributed Management Task Force (DMTF) 737
DISTRIBUTED mode
  console support 52
distributed mode
  console support 95
  z/OS console support 51
distributed mode console support 95
DMTF 353
DOCPU 268
DOCPU subcommand 268
driver 93 425
driving system 34
DS8000 4.0 version 540
DSCBs 550
DSINFO service 156
DSTORE processing 197

E

EADSCB=OK 552–553
EADSCB=OK keyword 552
  extended attribute DSCBs 552
EAS 544
EATTR value 554
EAV volume 540
ECC certificates 634
ECVT field 234
ECVTSRBJ field 233
ECVTSRBL field 233
EDG_EXIT100 exit 190
EDG_EXIT100 installation 174
EDG_EXIT100 installation exit 171
  volume retention method 166
EDGAUD audit reports 198
EDGPL100 macro 195
EDGRMMxx parmlib member
  LOCDEF-AUTOMOVE(YES|NO) 197
  OPTION operand 186
  retention method specification 166
  RETPD and MAXRETPD 196
EDGRT18 report 191
EDGTVEXT program 170
EDGUX100 exit module 190
Elliptic Curve Diffie-Hellman 634
EMC storage 84
enclave SRBs 232
ENF 15 listening exit
  ARCENF15 136
ENF 68
  BCPii support 586
ENF 68 broadcast
  BCPii issues 585
ENF35 signal
  CF structure alter 88
ENF72
  on demand migration 134
Enhanced DAT architecture 733
Enhanced PSP Tool (EPSPT) 40
EQDQ monitor 226
ERBRMF00 parmlib member 745
ERBRMF02 parmlib member 745
eServer OS monitoring profile 737
EXPDROP report 192
EXPDT retention method 166
extended addressing space 544, 546
EXTENDEDALIAS parameter 114
extended-format sequential data sets 548

F

F CATALOG command 561
F CATALOG,REPORT catalog command 112
F DEVMAN command 66
  refresh UCB function 64
F DEVMAN, REPORT command 66
F HZR,ANALYZE command 509
F HZR,ANALYZE,SYSNAME=SYSB command 508
F OAM,S,OSMC command 126
F OAM,START,RECYCLE command 129
F OAM,START,STORGRP,group-name 126
F OAM,START,STORGRP,group-name command 129
F OAM,UPDATE,SETOAM operator command 127
F OAM,UPDATE,SETOAM,scope,SGMAXTPR 129
F OAM,UPDATE,SETOAM,scope,SGMAXTPS command 129
F PFA,DISPLAY,CHECKS,DETAIL command 523
F PFA,UPDATE command 492
F ZFS,HANGBREAK command 292
f zfs,hangbreak command 293
F ZFS,QUERY,LEVEL command 58
F ZFS,QUERY,STORAGE command 58
FICON CHPIDs 421
filecodeset 330
FORCE option
  CMDS command 92
format-3 DSCB 550
format-4 DSCB 551
format-9 DSCB 550
FREEVOL parameter 121
FREEVOL=EOV
  DD statement keyword 120
FRRECOV DSNAME command 144
FRTV record 143

FSACCESS class 327
fseek() function 668
FTP support
  large-format data sets to EAVs 569
FTP.DATA
  EAS-eligible data se 567
  EATTR parameter 571
FTP.DATA data set
  STAtus subcommand 574
FTPCMDS data set 263
function shipping 285

G

getting started with z/OSMF 414
GIMADR service routine 40
GIMDDALC data set 41
global zone
  ZONESET entry 43
GPM4CIM 739
gpm4cim_setup.sh script 739
GPMSERVE 740
GRS_AUTHQLVL_SETTING health check 227
GRSCNFxx parmlib member
  AUTHQLVL parameter 225
gskkyman 639
gskkyman command 646
gskkyman utility 642–643
GSL_V3_CIPHER_SPECS
  environment variable 634

H

HASP720 message 708
HBA inserts 485
health check
  ZOSMIGV1R13_ZFS_FILESYS 57
health check exception
  PFA 490
HElp subcommand 578
HOT compiler option
  METAL C 443
HOT option 442
HSMHOST keyword 142
HWIBCPII address space 593
HWISTART procedure 584

I

i line command 162
IAR007I message 469
IARSUBSP macro
  create a subspace 461
IBM DS8000
  multicylinder units 544
IBM Health Checker for z/OS
  PFA checks 489–490
IBM JZOS Batch Toolkit for z/OS SDKs 752
IBM WebSphere Application Server OEM Edition for z/OS
  z/OSMF 354
IBM WebSphere Application Server OEM Edition for z/OS
Version 7 354
ICKDSF INIT command
  assign volumes to a reserve storage pool 383
IDCAMS LISTCAT command 60
IEAOPTxx parmlib member
  HiperDispatch=yes 241
  OPT settings 734
IEASYSxx parmlib member
  CATALOG=xx 112
  CATALOG=xx parameter 555
  large page support 467
  WARNUND 51
  WARNUND parameter 242
IEBCOPY utility 60
IEBDG utility
  JES3 format of spool 713
IEBDSCPY alias name 60
IEBPDSE program 105
IECIOS parmlib member
  PATH_SCOPE option 250
IECIOSxx parmlib member
  path recovery parameters 253
  PATH_INTERVAL option 250
  PATH_THRESHOLD option 250
  RECOVERY statement 249
IEDN 424
IFA832I message 275
IGGCATxx parmlib member 109, 112–114, 555
  TASKMAX parameter 114, 558
IGGPOST0_EXIT
  dynamic exit 115
IGGPRE00_EXIT
  dynamic exit 115–116
Incident Log
  CEA in z/OSMF processing 402
Incident Log task 362
  SYSLOG support 402
ini 531
ini file 531–532
Initial Access Response (IARS) parameter 124
instream data sets
  JES2 support 705
intraensemble data network (IEDN) 424
ioeagslv utility
  disabled aggregates 297
IOEZADM 301
IP_RECVPKTINFO option 328
IPA optimizations 444
IPA option 442
IPCS subcommand
  DOCPU 268
IPCS SYSTRACE subcommand
  new keywords 266
IPSec 410
IRASRMST macro 235
IRROPTxx parmlib member 211
ISGQUERY macro 221
  REQINFO=LATCHECA 222
  REQINFO=QSCAN 222
  REQINFO=USAGESTATS 222–223

ISGQUERY service 222
ISGYQUAC macro 223
ISPF Data Set Information 154
ISPF logon screen
  z/OSMF 376
ISPF Option 3.17
  ACL support 148
ISPF Postprocessor interface 721
ISPF task
  SDSF 380
  z/OSMF 376
IWM4ECRE macro 232
IWM4STBG SUBTASKS={NO | YES} 235
IWMEJOIN macro 232
IWMEJOIN SUBTASKS={NO | YES} 235
IWMELEAV macro 232
IXGCNFxx parmlib member 272
IXLALTER macro 85, 88
IXLCONN interface 73
izuconfig1.cfg
  new variables with V1R13 361

J

Java 1.4
  PFA requirement 490
Java 6.0 SR1
  large page support 466
javac command 753
JES levels supported
  SDSF for JES 38
JES2 exit 58
  step end SSI 708
JES2 spool migration 692
JES2 spool usage
  PFA check 504
JESLOG spin only 702
JOBRC keyword 706
JOBRC parameter 706
jobs instrumentation 474
JSON object 392, 396
JZOS 752
JZOS Batch Toolkit for z/OS 752
JZOS set of Java class libraries 752

K

Kanji version
  RMFJPN UTIL 726

L

large page coalesce 468
large page support 465
LATCHID 218
LFAREA parameter
  large page support 467
LibraryCenter 351
like parameter
  MKdir command 575
limited recovery time interval 249
Linux on System x
  z/OSMF 408
LISTCAT LEVEL command 16
LMDLIST service 156
LMkdir subcommand 577
LOCALSYSAREA
  REQUEST=GETSTOR 459
LOCALSYSAREA keyword 459
LOCALSYSAREA=NO|YES
  IARV64 macro 459
lock table entry size 73
LOCSIte subcommand 570–572
  AUTOMOUNT setting 576
LOCSTat subcommand 573
  EATTR parameter 573
LOCStat subcommand 573
LOSTMSG(ON|OFF) 312

M

MA line command 148
machine type 2818 421
MAPTARGET=Yes 697
MAXCONN keyword 73–74
MAXCONN(xmaxconn) keyword 73
MAXDORM option 278
MAXUSERMOUNTSYS 320
MAXUSERMOUNTUSE 320
MCU 543
MCUs 543
MEMLIMIT
  LOCALSYSAREA=YES 460
message arrival rate check 494
Message BLW004A 238
Message CNZ6002I 93
Message IDC3009I 566
Message IEA500A 238
Message IEA660I 242
Message IXC347I 83
Message IXC522I 76
Message IXC574I 84
Message IXL015I 82
METAL C environment
  HOT and IPA optimization techniques 442
METAL option 442
METAL programming 442
migration control data set (MCDS) 129
migration health check
  zFS R13 299
MKdir subcommand 575
ML line command 148
MODIFY CATALOG command 114, 557–558
  TASKMAX value 111
Monitor II OPT Settings report 734
Monitor III report format definition utility
  SYS1.SERBCLS 726
Monitor III Resource Group Data report 726
Monitor III Sysplex Summary report 720, 726
Monitoring Desktops task 407
MOVEVOL command 127
MPF parmlib member 96

MSGFLDxx parmlib member 96–98, 100
msys for Setup 62
MTFTPS 260
multicylinder unit
  EAV volume 543
multicylinder unit (MCU) 562
multicylinder units 543
multiple ISPF sessions
  z/OSMF 379
MULTSDN parameter 119

N

new system management tasks
  z/OSMF V1R13 362
NOARGPARSE option 449
non-intrusive journal backup 130
NORWSHARE 288
NOTIFYEXTENT(percent) 110
NUMUSERS keyword 74

O

Object storage groups
  OAM environment 123
OCE_ABEND_DESCRIP option 118
ODMNOTIFICATIONLIMIT 134
offline control data set (OCDS) 129
Omegamon XE 507
OMEGAMON z/OS Management Console 350
on-demand migration (ODM) 134
ONDEMANDMIGRATION 134
Online transaction processing (OLTP) 749
ONLYIF command 141
operations log (OPERLOG)
  Runtime Diagnostics 508
OPERCMDS class profile
  CMDS FORCE command 95
OSREQ keywords 126

P

password phrases
  using FTP from Configuration Assistant 409
password synchronization
  RRSF nodes 210
path-related errors 248
PAV-alias UCB 541
PDA facility 139
PDDU 260
PDUU utility program name
  AMAPDUPL 261
PFA 490
PFA install script
  AIRSHREP.sh 531
pfauser 530, 533
pgmcodeset 330
PKCS#11 tokens 634
PL100_SET_VRSELEXCLUDE bit 190
Policy Agent
  z/OSMF 408
Policy report
  Capacity Provisioning 622
pool storage group definition
  Auto Migrate 135
POSIX standard
  ACL support 148
Postprocessor Serialization Delay report 726
Predictive Failure Analysis (PFA) 490
Preinitialization compatibility interface (PICI) 666
preventive service planning (PSP) 39
Problem Documentation Upload Utility 260
profile sharing
  z/OSMF ISPF task 376
PROGxx parmlib member
  REXX alternate library 487
Provisioning Manager
  CIM connections 387
PSP buckets 39
PURGEDQ macro 242

Q

QUERY COMMONQUEUE(RECALL) command 141
QUERY SETSYS command 135
QUOTE subcommand 579

R

RACF class
  WBEM 476
RACF commands
  setting up security (z/OSMF) 357
RACF remote sharing facility (RRSF) 208
RECALL(DASD) 140
RECOVERY statement
  IECIOSxx parmlib member 249
RECOVERY statement parameters 250
Redbooks website 887
  Contact us xxi
REFORMAT NEWVTOC command 64
refreshing the UCB 64
REFUCB service 64
RELATIVEDATE parameter 275, 278
RELEASE RECALL(DASD) command 140
REPORT CONFIGURATION command 388
REPORT CROSSZONE command 42–46
REPORT DOMAIN command 388
REPORT MISSINGFIX command 47
REPORT POLICY command 388
Repository Authorization Mode 356–357
REQUEST=GETSTOR 459
reserve storage pool 383
REST interface 395
  z/OSMF 390
RETENTIONMETHOD operand 172
REXX exec
  izuconfig1.cfg.rexx 357
RFC 2246 634
RFC 4492 635
RMF Distributed Data Server
  CIM client/server 472

z/OSMF 740
RMF Distributed Data Server for z/OS (GPMSERVE) 739
RMF ensemble client task 739
RMF metrics
CIM client/server 472
RMF Postprocessor Workload Activity report 724
RMFJPN UTIL
Kanji version 726
rolling IPLs 48
RRSF network 208
RRSF nodes 209
RRSFDATA class 209, 211
RRSFLIST user data set 209
Runtime Diagnostics 492, 507
PFA check 504
PFA integration 504
RWSHARE 288

S

SAF Authorization Mode 356–357, 360, 390, 399
sample job CLNOS390 34
SYSMOD DELZOSA 62
script command 312–313
SDELAY report 726
SDSF REXX 352
SDSP selection algorithm 138
SEARCHDATASET operand 185
SEARCHDATASET subcommand 174
SEARCHVRS subcommand 200
LASTREFDATE operand 200
Serialization Delay report 726
ServerPac 34
z/OSMF delivery options 354
SET DEVSUP=XX command 65
SET DEVSUP=xx command 65
SET MSGFLD= command 100
SET MSGFLD=xx command 96
SET PROG=xx command 116
SETCON MODE=SHARED command 52
SETCON TRACKING=ON command 626
SETDISK statement 124
setfacl shell command 148
SETGRS AUTHQLVL=1 225
SETIOS command 252
SETIOS RECOVERY command 252
SETMF command 100
Message Flood Automation 100
SETMF ON command 96
SETOMVS command
LOSTMSG=ON | OFF 312
SETOMVS LOSTMSG=ON|OFF 312
SETOSMC statement
CBROAMxx parmlib member 125
SETPROG EXIT command 116
SETPROG EXIT,ADD command 116
SETRRS SHUTDOWN command 346
SETSMS USEEAV(YES) 579
SETSYS JOURNAL(RECOVERY) command 130
CDS backup function 133
SETSYS ODMNOTIFICATIONLIMIT command 134
SETSYS ONDEMANDMIGRATION command 134
SETXCF START,ALTER command 85
SHARED mode
console support 52
shared mode
z/OS console support 52
SHE control block 346
SITE command 570
SIte command 578
FTP support 576
SIte subcommand 572
SMA attributes 384
SMARTENDPOINT keyword 279
SMARTEPOVER parameter 279–280
SMF 72 subtype record 3 724
SMF arrival rate check 502
SMF record 72
subtype 5 726
SMF record 72-3 723
SMFPRMxx parmlib member
LSNAME parameter 277
SMP/E 36
SMP/E DDDEF entries
software deployment 374
SMP/E V3R6 40
HOLDDATA 42
SMS retention period 126
SNA name of the CPC
BCPii 601
SNMP community name 602
BCPii 595
software deployment 368
software instance 369
configure instance 375
software upgrades 40
SPIN= DD JCL enhanced 703
SPIN=UNALLOC 704
SPOOL initialization statement
JES2 701
spool volume prefix 702
SSRB
above 2 GB 232
STAtus subcommand 574–575
subspace 460
subspace mode 244–245, 461–462
sudo command 337
sudo utility 334
sudoedit command 336
SUPERUSER.FILESYS.USERMOUNT 318
Support Element (SE)
configure for BCPii 593
SYS1.NUCLEUS
SYSCATxx member 114
SYS1.NUCLEUS(SYSCATxx) 109
SYS1.PARMLIB ARCCMDxx member
PATCH command 137
SYS1.PARMLIB(LOADxx) 109
SYS1.PROCLIB.INSTALL 739
SYS1.SAMPLIB(AXR00)

SYSREXX address space 364
SYSBCPII CTRACE component
BCP11 cuts trace records 586
SYSCATxx parmlib member 555
SYSEVENT REQSRMST 235
sysplex dump directory 406
Sysplex Status task 407
sysplex=filesys 53, 287, 298–299
sysplex=off 285
sysplex=on 286
sysplex-unaware 285
SYSREXX facility 353
system area
new 64-bit storage area 459
System authorization facility (SAF)
z/OSMF 355
system logger monitoring intervals 272
System REXX 352
System REXX (SYSREXX)
z/OSMF 355
System SSL 634

T

TARGET LIST command 211
TASKMAX parameter 114, 558
TASKMAX(tasks) 111
TCP/IP stacks per image 408
TPATH keyword 424
tracing options
system logger 272
track-managed space 543
translation lookaside buffer (TLB) 465
triple DES encryption (CPACF)
PDDU 260
TRSMAIN 261
TSO HOMETEST command 567
typescript 312

U

Unicode Services Info API 658
UNIXPRIV class 317
SUPERUSER.FILESYS.USERMOUNT 317
UNIXPRIV profile
SUPERUSER.FILESYS.USERMOUNT 320
user_cache_size 304

V

View Diagnostic Details page 403
virtual storage constraint relief
subspaces 461
vital record specification
SEARCHVRS subcommand 200
VRSEL processing 184–185
VRSEL retention 187
VRSEL retention method 166, 186
VRSELEXCLUDE attribute 185
VRSRETN report 191
VSAM CA reclaim feature
SDSP selection 138
VVDS expansion 562
VVDSSPACE(primary,secondary) 110

W

WARNUND parameter 51, 242
IEASYSxx parmlib member 51
WBEM 353
WBEM class 480
RACF class for CIM 480
WithPIonly 625
WTOTYPE parameter 499

X

XL C Metal compiler 353

Z

z/OS lock table 73
z/OS Management Facility (z/OSMF) 350
z/OS SMB server 334
z/OSMF
log in a user 414
plug-ins 367
z/OSMF Capacity Provisioning task 386
z/OSMF ISPF panel
color settings 377
z/OSMF plug-ins 367
z/OSMF Welcome panel
customization 413
z/OSMF Welcome screen
customization 413
z196 server
HiperDispatch=yes 241
zEnterprise 196 with driver 93 425
zFS 8K block 298
zFS Direct I/O (DIO) 284
zFS IOEFSPRM file 298
zFS R13 DASD space 300
zFS R13 migration health check 299
zFS sysplex-aware 288
zfsadm aggrinfo command 297
zfsadm clonesys command 56
zfsadm command
help text 301
zfsadm config -aggrgrow off command 55
zfsadm configquery -aggrgrow command 55
zfsspace tool 300
zfsspace utility 55
ZMFAPLA class 360
ZMFAPLA resource class 357, 359
ZMFAPLA resource class profile 356
ZONESET entry 43
ZOSGEN 40
ZOSMIGV1R13_RO_SYMLINKS
health check 311

Redbooks

z/OS Version 1 Release 13 Implementation

(1.5” spine)
1.5”<-> 1.998”
789 <->1051 pages

z/OS Version 1 Release 13 Implementation

JES2, JES3, SDSF, RMF, z/OSMF, EAV, BCPii, Service aids, System Logger

Consoles, RSM, z/OS UNIX, zFS, CIM, PFA, HCD, HCM, RRS

XCF, DFSMS, ISPF, RRSF, GRS

This IBM Redbooks publication provides information about installation and migration changes to be aware of if you are responsible for migrating systems from z/OS V1R10, z/OS V1R11, and z/OS V1R12 to z/OS V1R13.

It also highlights actions that are needed to prepare for the installation of z/OS V1R12, including ensuring driving system and target system requirements are met and coexistence requirements are satisfied.

There is a special focus on identifying new migration actions that must be performed for selected elements when migrating to z/OS V1R13.

The book addresses the following topics:

- z/OS V1R13 overview, z/OS V1R13 installation, managing volume backups with fast replication, XCF enhancements, console services enhancements
- DFSMSdfp, DFSMSoam, DFSMShsm, ISPF enhancements, DFSMSrmm enhancements, establishing RACF security for RRSF TCP/IP connections
- GRS enhancements, BCP supervisor, contents supervisor and RSM updates, improved channel recovery, Service aids enhancements, System Logger - SMF
- z/OS UNIX System Services, z/OS UNIX-related applications, RRS, z/OS Management Facility, z/OS HCD and HCM, C language
- Storage management enhancements, Common Information Model, Predictive Failure Analysis, Extended Address Volume, BCPii, Capacity Provisioning
- System SSL enhancements, UNICODE, Language Environment, SDSF enhancements, JES2 enhancements, JES3 enhancements, RMF enhancements
- WebSphere Application Server OEM, z/OSMF, CIM and Capacity Provisioning setups
- BCPii Metal C example

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

SG24-7946-00 ISBN 0738436224