

# z/OS Version 1 Release 12 Implementation

z/OS base, DFSMS, HiperDispatch,  
z/OSMF, XCF, GRS, SMF, PSA, XML

LE, XML, z/OS UNIX, zFS, EAV,  
HyperSwap, RRS, TSO/E, Unicode

BCPii, PFA, JES2, JES3, SDSF,  
Auto-reply, RSM



Paul Rogers  
Robert Hering  
Paulo Heuser  
George Kozakos  
Lutz Kühner  
Jean-Louis Lafitte  
Diana Nakajima  
Paulo Cesar Nascimento  
Gil Peleg

**Redbooks**





International Technical Support Organization

**z/OS Version 1 Release 12 Implementation**

April 2011

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

**First Edition (April 2011)**

This edition applies to version 1 release 12 modification 0 of IBM z/OS (product number 5694-A01) and to all subsequent releases and modifications until otherwise indicated in new editions.

© Copyright International Business Machines Corporation 2011. All rights reserved.

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.



# Contents

<b>Notices</b> .....	xix
Trademarks .....	xx
<b>Preface</b> .....	xxi
The team who wrote this book .....	xxi
Now you can become a published author, too! .....	xxii
Comments welcome .....	xxiii
Stay connected to IBM Redbooks .....	xxiii
<b>Chapter 1. z/OS V1R12 enhancements</b> .....	1
1.1 Base control program (BCP) enhancements .....	3
1.1.1 HCD/HCM .....	5
1.1.2 Binder support .....	6
1.1.3 Global resource serialization (GRS) .....	7
1.1.4 Sysplex enhancements .....	7
1.1.5 Workload Manager (WLM) .....	8
1.1.6 z/OS Management Facility (zOSMF) .....	8
1.1.7 Cross-system coupling (XCF) .....	9
1.1.8 System Logger .....	10
1.1.9 Capacity Provisioning .....	10
1.1.10 Predictive failure analysis (PFA) .....	10
1.1.11 XML System Services .....	11
1.1.12 C/C++ enhancements .....	11
1.1.13 Common Information Model (CIM) .....	12
1.2 z/OS and hardware .....	12
1.3 z/OS UNIX System Services (z/OS UNIX) .....	13
1.4 zFS support .....	14
1.5 JES components .....	14
1.6 TSO/E .....	15
1.7 Library Server .....	15
1.8 z/OS V1R12 new components .....	16
1.9 Data Facility Storage Management Subsystem (DFSMS) .....	16
1.9.1 DFSORT .....	20
1.9.2 Constraint relief for XTIOIOT .....	21
1.9.3 NFS Server .....	21
1.10 DFSMSShsm .....	22
1.11 DFSMSrmm .....	23
1.12 DFSMSdss .....	24
1.13 DFSMSoam .....	24
1.14 SDSF .....	25
1.15 z/OS Health Checker .....	25
1.16 Communications Server .....	27
1.17 Serviceability items .....	33
1.18 Language Environment .....	34
1.19 Security Server .....	35
1.19.1 PKI extensions .....	36
1.19.2 Cryptography .....	37
1.20 Service aids .....	40
1.21 Infoprint Server for z/OS .....	40

<b>Chapter 2. z/OS V1R12 installation</b> . . . . .	43
2.1 z/OS V1R12 installation considerations . . . . .	44
2.1.1 Installation consideration for z/OS V1R12 CBPDO . . . . .	44
2.2 Driving system requirements for z/OS V1R12. . . . .	44
2.3 Methods of installing z/OS . . . . .	45
2.3.1 Customized Offerings Driver V3 (5751-COD). . . . .	46
2.3.2 Driving system requirements using ServerPac (entitled) . . . . .	47
2.3.3 Custom-Built Product Delivery Option (CBPDO) . . . . .	47
2.4 Coexistence, migration, and fallback considerations . . . . .	48
2.4.1 Installation considerations for coexistence PTFs . . . . .	49
2.5 Migration actions for z/OS V1R10, z/OS V1R11 systems. . . . .	50
2.5.1 Trace options with the CTIGRSxx member. . . . .	50
2.5.2 Programs that use system-generated temporary data sets . . . . .	50
2.5.3 Track CSVRTLS services . . . . .	51
2.5.4 Remove classification rules with the EWLM transaction class . . . . .	53
2.5.5 Update the SFM policy . . . . .	53
2.5.6 Accommodate new REUSASID default . . . . .	54
2.5.7 Review the list of WTORs in parmlib member AUTOR00. . . . .	54
2.5.8 Removing some references to Unicode services . . . . .	55
2.6 Migration considerations before first IPL of z/OS V1R12 . . . . .	56
2.6.1 Infoprint Server . . . . .	56
2.6.2 Language Environment (LE). . . . .	58
2.6.3 z/OS UNIX . . . . .	59
2.6.4 Shell and Utilities version of the tsocmd command. . . . .	59
2.7 BCP migration actions after the first IPL . . . . .	60
2.7.1 Hardware instrumentation services (HIS) . . . . .	60
2.7.2 BCP program management Binder. . . . .	61
2.7.3 Cryptographic Services. . . . .	62
2.7.4 Unicode Services . . . . .	62
2.8 JES2, JES3, and SDSF installation considerations . . . . .	63
2.9 Elements withdrawn from z/OS V1R12. . . . .	64
2.10 IBM zEnterprise 196 (z196) . . . . .	64
2.10.1 IBM System z Discovery and Auto-Configuration (zDAC) . . . . .	65
2.10.2 Three subchannel sets for zEnterprise 196 servers . . . . .	65
2.10.3 IBM zEnterprise 196 server performance . . . . .	66
2.10.4 Parallel Sysplex environments . . . . .	66
2.10.5 Networking performance. . . . .	67
2.10.6 IBM zEnterprise Unified Resource Manager. . . . .	68
<b>Chapter 3. HiperDispatch</b> . . . . .	71
3.1 HiperDispatch overview . . . . .	72
3.1.1 Without HiperDispatch . . . . .	72
3.1.2 With HiperDispatch . . . . .	72
3.2 Activating HiperDispatch . . . . .	74
3.3 Monitoring HiperDispatch . . . . .	75
3.3.1 WLM considerations . . . . .	75
3.3.2 RMF reports . . . . .	75
3.4 Help processing . . . . .	76
3.4.1 Alternate wait management . . . . .	77
3.5 HiperDispatch enhancements to z/OS V1R10 . . . . .	78
3.6 HiperDispatch enhancements with z/OS V1R11. . . . .	81
3.7 HiperDispatch enhancements with z/OS V1R12. . . . .	82
3.7.1 z196 Server features. . . . .	83

3.7.2 HiperDispatch enhancements in z/OS V1R12 for the IBM zEnterprise 196 . . . . .	83
<b>Chapter 4. System Logger</b> . . . . .	87
4.1 System Logger enhancements with z/OS V1R11 . . . . .	88
4.2 Logger SHAREOPTIONS . . . . .	90
4.2.1 System Logger enhancement . . . . .	91
4.2.2 Using SHAREOPTIONS(3,3) . . . . .	91
4.3 Log stream data set support . . . . .	93
4.3.1 Maximum log stream data set size to 4 GB (OA30548) . . . . .	93
4.3.2 Coexistence and migration considerations . . . . .	94
4.3.3 Testing log data set parameter modifications . . . . .	95
<b>Chapter 5. Auto-reply to WTORs</b> . . . . .	97
5.1 Auto-reply policy for WTORs . . . . .	98
5.1.1 Auto-reply policy . . . . .	98
5.2 Activating an auto-reply policy for WTORs . . . . .	99
5.2.1 AUTORxx parmlib member . . . . .	99
5.3 IBM-supplied default AUTOR00 parmlib member . . . . .	103
5.4 Changing the auto-reply policy . . . . .	105
5.5 New commands for auto-reply . . . . .	106
5.5.1 The SETAUTOR command . . . . .	107
5.5.2 Displaying auto-reply policy and WTOR information . . . . .	108
5.6 Other system updates . . . . .	109
5.6.1 SMF record type 90 subtype 33 . . . . .	110
5.6.2 SDSF SR panel . . . . .	110
5.6.3 MPFLSTxx parmlib member update . . . . .	110
5.6.4 Error detection and new system messages . . . . .	112
5.7 Security checking of CPF command routing . . . . .	113
5.7.1 Security checking with z/OS V1R12 . . . . .	113
5.8 Console buffers . . . . .	114
<b>Chapter 6. Real Storage Manager (RSM) enhancements</b> . . . . .	115
6.1 64-bit storage overview . . . . .	116
6.2 z/OS UNIX fork() processing . . . . .	116
6.2.1 z/OS UNIX fork() processing . . . . .	117
6.2.2 z/OS V1R12 enhancements . . . . .	119
6.3 RSM support of SDUMP resource . . . . .	120
6.3.1 Support in VERBX IEAVTSFS . . . . .	121
6.4 Large page enhancements . . . . .	123
6.4.1 Large page support for the nucleus . . . . .	123
6.4.2 Large page coalesce support in z/OS V1R12 . . . . .	125
<b>Chapter 7. DFSMS enhancements</b> . . . . .	127
7.1 DFSMS catalogs . . . . .	128
7.2 Contention detection for catalogs . . . . .	128
7.2.1 Detecting catalog SYSZTIOT contention . . . . .	129
7.3 DEFINE RECATALOG support enhancement . . . . .	130
7.4 OPEN/CLOSE/EOV RAS enhancements . . . . .	131
7.4.1 New DCB ABEND exit option . . . . .	132
7.4.2 SMS DASD input when missing last volume . . . . .	132
7.4.3 New reason codes for FREE=CLOSE . . . . .	132
7.5 VSAM partial release on multiple volumes . . . . .	133
7.5.1 z/OS V1R12 implementation . . . . .	133
7.5.2 z/OS V1R12 considerations . . . . .	136

7.6	VSAM enhancements . . . . .	137
7.6.1	Control area (CA) reclaim . . . . .	137
7.6.2	Disk striping support . . . . .	142
7.6.3	Dynamic trace enhancements . . . . .	142
7.7	Copy storage group volumes . . . . .	144
7.8	VOLSEMSG and TRACE parameters . . . . .	147
7.8.1	z/OS V1R12 enhancements . . . . .	148
7.9	Multitasking volume recovery from dump . . . . .	152
7.9.1	z/OS V1R12 support enhancements . . . . .	152
7.9.2	Using multitasking recovery . . . . .	152
7.9.3	Copy pool enhancements . . . . .	153
7.9.4	SETSYS command enhancements . . . . .	154
7.9.5	Fast replication recovery . . . . .	155
7.10	DFSMS SMS health check enhancement . . . . .	155
7.10.1	New health checks in z/OS V1R12 . . . . .	156
7.11	IDCAMS DELETE PDS and PDSE . . . . .	158
7.12	DFSMSrmm enhancements for z/OS V1R12 . . . . .	159
7.13	DFSMSrmm operational enhancements . . . . .	159
7.13.1	CONTROL STATUS dialog option . . . . .	160
7.13.2	Auto-reply policy and DFSMSrmm . . . . .	163
7.14	DFSMSrmm inventory management and reporting . . . . .	164
7.14.1	Retention Limit Reporting for EXPDTDROP and VRSRETAIN . . . . .	165
7.15	DFSMSrmm scalability and performance . . . . .	168
7.15.1	EAV support . . . . .	168
7.15.2	Dynamic allocation support . . . . .	169
7.15.3	IPv6 support . . . . .	169
7.15.4	DFSMSrmm reliability, availability, and serviceability (RAS) . . . . .	170
7.16	DFSMSrmm hardware support . . . . .	171
7.16.1	Creating reports about data sets and volumes that are copy exported . . . . .	171
7.16.2	Disaster recovery options . . . . .	172
7.17	DFSMSShsm space management performance . . . . .	174
7.18	DFSMS AMS DCOLLECT enhancements . . . . .	175
	<b>Chapter 8. DFSORT improvements . . . . .</b>	<b>177</b>
8.1	File size for improperly closed VSAM data sets . . . . .	178
8.2	Improved first failure data capture . . . . .	178
8.3	Improved diagnostics . . . . .	179
8.4	Dynamic allocation improvements . . . . .	179
8.5	XTIOT uncaptured UCBs and DSAB above 16 megabytes . . . . .	180
8.6	Extended address volumes . . . . .	180
8.7	Memory object intermediate work space . . . . .	180
	<b>Chapter 9. Service aids enhancements . . . . .</b>	<b>183</b>
9.1	SADMP support for EAV volumes . . . . .	184
9.1.1	IPCS SADMP dump data set utility . . . . .	184
9.1.2	Using the AMDSADDD utility . . . . .	186
9.1.3	Examples of running AMDSADDD in batch mode . . . . .	187
9.2	Superzap support for EAV3 . . . . .	188
9.2.1	Superzap control statements . . . . .	188
9.3	IPCS PDS enhancement . . . . .	189
9.3.1	Printing into a PDS . . . . .	190
9.3.2	IPCSPDS data set allocation attributes . . . . .	191
9.3.3	Directing IPCS output to different mediums . . . . .	193

9.3.4	New messages	193
9.4	IPCS SYSTRACE formatting enhancements	194
9.4.1	IPCS SYSTRACE SORTCPU	194
9.4.2	IPCS SYSTRACE PERFDATA	195
9.4.3	IPCS SYSTEM TRACE panel	197
9.5	SADMP ASID prioritization	201
9.5.1	ADDSUMM keyword of the AMDSADMP macro	202
9.6	Console dump support	204
9.7	Service aids support for BAM XTLOT	206
<b>Chapter 10. z/OS Infoprint Server</b>		<b>209</b>
10.1	IP PrintWay extended mode processing	210
10.2	ANFUXRSP user exit	211
10.3	LPD and API support for large files	212
10.3.1	Installation considerations	213
10.4	Printer Inventory for PSF enhancements	213
10.4.1	Using AFP Download Plus	213
10.5	Infoprint Central	214
10.5.1	Infoprint Central enhancements	214
10.5.2	Installation considerations	215
10.6	Migration considerations	217
10.6.1	Infoprint Port Monitor V3	218
<b>Chapter 11. TSO/E enhancements</b>		<b>221</b>
11.1	Multiple TSO/E logons	222
11.1.1	z/OS V1R12 JES2 support	222
11.2	XTLOT, nocapture UCB, and DSAB above the line support	223
11.3	Password special character support	225
<b>Chapter 12. RMF enhancements</b>		<b>227</b>
12.1	RMF Postprocessor reports in XML format	228
12.2	RMF support for SMF log stream	228
12.2.1	ISPF Postprocessor interface	229
12.3	Statistics in the CPU Activity report	231
12.4	Nominal and effective processor capacity	232
12.4.1	WEB queue distribution	233
12.4.2	Mean time to wait (MTTW)	237
12.5	Enhanced Enterprise Disk Systems report	239
12.6	Enhanced Crypto Hardware Activity report	239
12.7	RMF SMF logstream support	240
<b>Chapter 13. Language Environment</b>		<b>241</b>
13.1	C++ TR1 support	242
13.2	BSAM 64 K track support	244
13.3	Language Environment locale support	244
13.4	LE nonoverridable parmlib capability	245
13.4.1	Changing parmlib settings	247
13.5	realloc() optimization	250
13.6	VSAM EA support for KSDS AIX for C/C++	250
13.7	Resolve year 2038 problem	251
<b>Chapter 14. BCP allocation improvements</b>		<b>253</b>
14.1	Memory-based data set ENQ management	254
14.1.1	New MEMDSENQMGMT keyword	254

14.2	Suppressing SMF DD accounting . . . . .	256
14.3	Duplicate temporary data set name support . . . . .	257
14.3.1	Specifying unique temporary data set names . . . . .	257
14.3.2	TEMPDSFORMAT command options . . . . .	259
<b>Chapter 15. XML System Services . . . . .</b>		<b>261</b>
15.1	Restrict root element name . . . . .	262
15.1.1	z/OS V1R12 parsing . . . . .	262
15.2	Dynamic schema . . . . .	263
15.3	Fragment parsing . . . . .	264
15.3.1	Parsing XML document fragments with validation . . . . .	265
15.3.2	Enable offload to specialty engines . . . . .	266
<b>Chapter 16. z/OS UNIX System Services . . . . .</b>		<b>267</b>
16.1	GRS identity for latches used by z/OS UNIX . . . . .	268
16.1.1	GRS latch identity used by z/OS UNIX . . . . .	269
16.1.2	Using GRS latch identities with z/OS V1R12 . . . . .	269
16.2	Health Checker exploitation of BPX.SUPERUSER . . . . .	270
16.2.1	Using BPX.SUPERUSER resource in the FACILITY class . . . . .	270
16.3	Health Checker USS_HFS_DETECTED . . . . .	271
16.3.1	USS_HFS_DETECTED parameters . . . . .	272
16.3.2	USS_HFS_DETECTED messages . . . . .	272
16.3.3	USS_HFS_DETECTED examples . . . . .	272
16.4	z/OS UNIX mmap improvements . . . . .	274
16.4.1	USS mmap support for NFS Client files . . . . .	274
16.4.2	Serviceability improvement . . . . .	274
16.5	z/OS UNIX source address selection support . . . . .	275
16.5.1	Using source address selection . . . . .	275
16.5.2	Binding to source address selection . . . . .	275
16.5.3	Validation of source address selection . . . . .	276
16.5.4	Set and Get partner information . . . . .	277
16.6	z/OS UNIX FILEDATA=RECORD support . . . . .	278
16.6.1	Using the record file format . . . . .	278
16.6.2	z/OS UNIX shell updates . . . . .	280
16.6.3	z/OS ISHELL update . . . . .	281
16.6.4	Message update . . . . .	283
16.6.5	Exploiting FILEDATA=RECORD with correct data contents . . . . .	283
16.6.6	Migration and coexistence consideration . . . . .	285
16.7	z/OS UNIX dynamic socket limit for USS . . . . .	285
16.7.1	AF_UNIX MAXSOCKETS . . . . .	286
16.7.2	Common Inet reserved ports . . . . .	286
16.8	Remove USS support for zFS multifile system aggregates . . . . .	287
16.8.1	Changes made in z/OS V1R12 . . . . .	287
16.8.2	ISHELL panel changes . . . . .	289
16.9	USS shell and utility tsocmd . . . . .	292
16.9.1	Using tsocmd . . . . .	292
16.9.2	Installation and migration . . . . .	294
16.10	USS support for sysplex-aware on a file system basis . . . . .	295
16.10.1	Minor background explanation . . . . .	295
16.10.2	USS SPE OA29712 . . . . .	295
16.10.3	Information about F BPXOINIT display command . . . . .	296
16.10.4	Message BPXF221I . . . . .	297
16.11	zFS sysplex-aware on a file system basis . . . . .	297

16.11.1	zFS APAR OA29619.....	298
16.11.2	Background information about previous zFS behavior.....	299
16.11.3	zFS R/W mounted file system being sysplex-unaware.....	301
16.11.4	zFS being sysplex-aware in z/OS V1R11 for R/W mounts.....	302
16.12	New zFS support with APAR OA29619.....	306
16.12.1	New and older important zFS configuration options.....	307
16.12.2	Setting up the zFS parameters.....	308
16.12.3	Using sysplex=filesys.....	309
16.12.4	zFS sysplex-aware on a file system basis.....	310
16.12.5	Running zFS sysplex-aware considerations.....	312
16.12.6	Benefits of and recommendations for the new support.....	316
16.12.7	z/OS UNIX directory caching display tool.....	318
16.12.8	File system monitoring tool FSMON.....	319
16.13	DFSMSdfp indirect volume serial for zFS data sets.....	320
16.13.1	Cloning zFS file systems.....	320
16.13.2	Enabling the support.....	321
16.13.3	Cloning of zFS aggregates.....	322
16.13.4	Example of cloning a zFS aggregate.....	322
16.14	zFS usage of VSAM partial release on multiple volumes.....	325
<b>Chapter 17. z/OS UNIX-related applications.....</b>		<b>327</b>
17.1	Network File System (NFS).....	328
17.2	z/OS UNIX System Services mmap support for NFS client.....	328
17.3	NFS server new SMF records for file operations.....	328
17.4	NFS server cache monitoring and reporting.....	330
17.5	Password phrase support for the MVSlogin client utility.....	332
17.6	NFS Server displays the accounting statistics.....	334
17.7	IBM Ported Tools for z/OS: OpenSSH.....	336
17.7.1	Upgrade OpenSSH to version 5.0p1.....	336
17.7.2	SMF logging for OpenSSH.....	341
17.7.3	Providing RACF key ring support in OpenSSH.....	342
17.7.4	OpenSSH migration and coexistence considerations.....	345
17.7.5	Incorrect owner or permissions of the ssh configuration file.....	345
17.7.6	Relative path name on starting the sshd daemon.....	345
17.7.7	Using the sftp command with the -b option and further interaction.....	346
17.7.8	z/OS OpenSSH V1R2 is now an XPLINK application.....	346
17.7.9	Message numbers with OpenSSH error messages.....	346
17.7.10	Using an address containing delimiter characters.....	346
17.7.11	Using the default cipher list.....	347
17.7.12	Default MACs list.....	347
17.7.13	Port forwarding.....	347
17.7.14	New minimum RekeyLimit value.....	347
17.7.15	New minimum RSA key size.....	347
17.7.16	New DSA key size.....	348
17.7.17	New shell environment variable for ProxyCommand.....	348
17.7.18	New behavior for specifying a file name on key generation.....	348
17.7.19	Long file names.....	348
17.7.20	New behavior on key generation.....	348
17.7.21	Coexistence of old and new OpenSSH versions.....	349
17.7.22	Installation of OpenSSH V1R2.....	349
<b>Chapter 18. BCP supervisor updates.....</b>		<b>351</b>
18.1	zAAP on zIIP support.....	352

18.1.1	zAAP on zIIP implementation with z/OS V1R12 . . . . .	352
18.1.2	The DISPLAY IPLINFO command in z/OS V1R12 . . . . .	353
18.1.3	IEAOPTxx parmlib member considerations . . . . .	355
18.2	Discretionary work timeslice . . . . .	357
18.2.1	Discretionary workload . . . . .	358
18.3	Customized time slices . . . . .	358
18.3.1	The TIMESLICES parameter . . . . .	358
18.3.2	Mean-time-to-wait threshold . . . . .	359
18.3.3	Expected benefits . . . . .	359
18.4	MTTR instrumentation . . . . .	360
18.4.1	MTTR process overview . . . . .	360
18.4.2	The IEATEDS macro . . . . .	361
18.4.3	The IEAVFTED REXX exec . . . . .	361
18.4.4	Software dependencies . . . . .	363
18.5	RTM enhancements . . . . .	364
18.5.1	ESPIE percolation to RTM . . . . .	364
18.5.2	SPIEOVERRIDE for ESTAEX . . . . .	365
18.5.3	SDWALOC31 for FRRs . . . . .	365
18.5.4	RESMGR RLXPC (reusable LX PC) . . . . .	366
18.5.5	IEAARR DYNSTORAGE=NOTAVAIL . . . . .	366
18.5.6	Time of error locks for ESTAE-type recovery in the SDWA . . . . .	366
18.5.7	SDWANMFS - not my fault summary . . . . .	367
18.5.8	New TCB flag -TCBENDNG . . . . .	367
18.5.9	New TCB flag -TCBEndingAbnormally . . . . .	368
18.5.10	New STCB field - STCBCMPC . . . . .	368
18.5.11	TCB address in formatted LOGREC record . . . . .	369
18.6	Miscellaneous updates . . . . .	369
18.6.1	D SYMBOLS . . . . .	369
18.6.2	LCCA/PCCA health checks . . . . .	370
<b>Chapter 19.</b>	<b>BCP contents supervisor updates . . . . .</b>	<b>375</b>
19.1	LLA function enhancements . . . . .	376
19.1.1	Dynamic exits CSVLLIX1 and CSVLLIX2 . . . . .	376
19.1.2	Avoiding an LLA busy response . . . . .	376
19.1.3	LLA automatic restart when SUB=MSTR is omitted . . . . .	377
19.2	Dynamic exits enhancements . . . . .	377
19.2.1	Dynamic exits replace . . . . .	377
19.2.2	Dynamic exits parameter . . . . .	378
19.2.3	Dynamic exits type . . . . .	378
19.2.4	Dynamic exits option foundbuterror . . . . .	378
19.3	Dynamic LINKLIST enhancements . . . . .	379
19.4	Dynamic LPA enhancements . . . . .	379
19.4.1	Dynamic LPA AddAlias . . . . .	379
19.4.2	Dynamic LPA SVC number . . . . .	379
19.4.3	Dynamic LPA Add by fully qualified HFS pathname . . . . .	380
19.4.4	Dynamic LPA deferred LPA wait . . . . .	380
19.4.5	Dynamic LPA Query Only . . . . .	381
19.5	Defining PROGxx defaults . . . . .	381
19.6	PROGxx SYSLIB enhancement . . . . .	382
19.7	SVCR 0 trace entries for ATTACH/LINK/SYNCH/XCTL . . . . .	383
19.8	RTLS withdrawal . . . . .	383
<b>Chapter 20.</b>	<b>Extended Address Volume . . . . .</b>	<b>385</b>



20.1	Introduction	386
20.2	Extended Address Volume overview	386
20.2.1	3390 Model A	387
20.2.2	EAV basic definitions	387
20.3	EAV terminology	388
20.4	EAV key design points	389
20.4.1	Track-managed space	390
20.4.2	Cylinder-managed space	390
20.4.3	Multicylinder unit	390
20.4.4	DASD track address format	391
20.4.5	Extended address volume attributes	394
20.4.6	EAS-eligible data sets	395
20.4.7	EAS non-eligible data sets	396
20.4.8	New format DSCBs for EAVs	397
20.4.9	EATTR data set attribute	400
20.5	z/OS V1R12 enhancements for EAVs	401
20.5.1	Basic catalog structure (BCS)	402
20.5.2	VSAM volume data sets	403
20.5.3	DFSMSHsm support	404
20.5.4	DFSORT support	405
20.5.5	Catalog data sets in EAS	405
20.6	Job entry subsystems support	406
20.7	Service aids support	406
20.8	Migration considerations	406
20.8.1	EATTR attribute	407
20.8.2	EOV like concatenation	407
20.8.3	OPEN considerations	407
20.8.4	XRC considerations	408
20.8.5	Format 8 DSCBs	408
20.8.6	Tracking facility	408
20.9	Coexistence considerations	409
<b>Chapter 21. HyperSwap</b>		<b>413</b>
21.1	Basic HyperSwap overview	414
21.2	Basic HyperSwap	414
21.2.1	Relationship of GDPS HyperSwap and Basic HyperSwap	415
21.2.2	Basic HyperSwap	416
21.3	Evolution of TPC-R support of Basic HyperSwap	418
21.3.1	Metro Global Mirror with HyperSwap support	418
21.3.2	Metro Mirror with HyperSwap support	418
21.3.3	Metro Mirror suspend policy	419
21.4	Basic HyperSwap in z/OS V1R11	422
21.5	Basic HyperSwap resiliency in z/OS V1R12	422
21.6	Basic HyperSwap and data migration coexistence support	423
21.6.1	HyperSwap and data migration products	423
21.6.2	HyperSwap programming enhancement	424
21.7	Couple data set avoidance	425
21.8	Enhanced device discovery	426
21.9	HyperSwap resiliency in z/OS V1R12	426
21.9.1	Capacity planning	427
21.9.2	Usage considerations	429
21.9.3	Adding critical address spaces	429
21.9.4	New messages and reason codes	429

21.10	Software dependencies	430
21.11	Migration and coexistence	430
21.11.1	z/OS V1R12 resiliency enhancement	431
<b>Chapter 22</b>	<b>BCPii</b>	<b>433</b>
22.1	BCPii overview	435
22.1.1	Using BCPii	435
22.1.2	BCPii address space	436
22.1.3	BCPii services	437
22.1.4	BCPii services	439
22.1.5	BCPii internal interfaces	440
22.1.6	Static power saving mode	441
22.2	BCPii installation	443
22.2.1	Configure the local Support Element to support BCPii	443
22.2.2	Authorize an application to use BCPii	448
22.2.3	Configure the BCPii address space	452
22.2.4	Set up the event notification mechanism for z/OS UNIX callers	452
22.3	New BCPii diagnostics in z/OS V1R12	455
22.4	BCPii dependencies	455
22.4.1	z/OS BCPii vs. TSA BCPii	456
22.4.2	Hardware dependencies	456
22.5	Exploiters of BCPii	456
22.5.1	Current z/OS system BCPii exploiters	457
22.6	Migration and coexistence	457
22.7	BCP internal interface programming	457
22.7.1	A Metal C-example	457
22.7.2	BCPii as a set of new z/OS commands	458
22.7.3	Potential advantages of a new command set	460
<b>Chapter 23</b>	<b>About (de)ciphering</b>	<b>461</b>
23.1	Elliptic curves cryptography	462
23.1.1	From symmetric to asymmetric cryptography	462
23.1.2	Elliptic curves	464
23.1.3	Elliptic curve cryptography	466
23.1.4	System SSL elliptic curve cryptography	467
23.1.5	Length is strength	469
23.1.6	z196 specifics	471
23.2	CPACF-protected key	471
23.2.1	Principles	471
23.2.2	Protection of cryptographic keys	472
23.2.3	Architectural description	474
23.2.4	RACF field support for CPACF-protected keys	476
23.2.5	Summary of CPACF protected key	477
<b>Chapter 24</b>	<b>Predictive Failure Analysis</b>	<b>479</b>
24.1	Predictive Failure Analysis overview	480
24.1.1	PFA-detected system failures	480
24.2	How PFA chooses address spaces to track	481
24.3	Using steps to get the most out of PFA	482
24.3.1	Reduce the number of false positives	482
24.3.2	Eliminate jobs causing false positives	483
24.3.3	Automate the PFA IBM Health Checker for z/OS exceptions	483
24.3.4	PFA and IBM Health Checker for z/OS	484
24.3.5	Runtime Diagnostics with z/OS V1R12	485

24.4	PFA with z/OS V1R10 and V1R11	486
24.5	PFA with z/OS V1R12 enhancements	487
24.5.1	SMF arrival rate check	488
24.5.2	Supervised learning support	490
24.5.3	Miscellaneous improvements	491
24.6	Using PFA with commands and SDSF	492
24.6.1	SDSF support for PFA	495
24.7	PFA infrastructure	498
24.7.1	PFA parameters for health checks	499
24.7.2	Differences between PFA checks and other remote health checks	500
24.8	PFA installation	501
24.8.1	Migration considerations for PFA with z/OS V1R12	502
24.8.2	PFA in z/OS V1R12 supports one or multiple ini files	505
24.8.3	New IBM health checks with z/OS V1R12	505
24.9	Serviceability information	505
24.9.1	Serviceability improvements with z/OS V1R12	506
24.9.2	Migration actions to z/OS V1R12	507
<b>Chapter 25. C language</b>		<b>509</b>
25.1	XL C/C++	510
25.1.1	Performance via a restrict keyword	510
25.1.2	Architecture compiler suboption	513
25.1.3	C++0x standards	516
25.1.4	Usability and portability	523
25.2	Metal C enhancements	524
25.2.1	RENT option	524
25.3	Memory model	526
25.3.1	Specific-operand serialization	530
25.4	More general purpose registers	530
<b>Chapter 26. Hardware instrumentation services</b>		<b>535</b>
26.1	CPU Measurement Facility overview	536
26.1.1	Hardware-based performance analysis	536
26.2	CPU Measurement Counter Facility	537
26.2.1	Measurement counters	537
26.2.2	Cryptographic coprocessor support	538
26.2.3	Counter sets	539
26.2.4	CPU Measurement Sampling Facility	540
26.3	Hardware Instrumentation Services address space	540
26.3.1	Setting up hardware event data collection	540
26.3.2	Security specifications	541
26.3.3	Collecting data with HIS	541
26.4	z/OS V1R12 enhancements for HIS	542
26.4.1	State change detection	542
26.4.2	SMF Record Type 113 records	545
26.4.3	Load module mapping	545
26.5	Installation considerations	545
26.5.1	Requirements	545
<b>Chapter 27. FICON dynamic channel-path management</b>		<b>547</b>
27.1	LPAR clustering	549
27.1.1	LPAR clustering components	550
27.2	Value of dynamic channel-path management	551
27.2.1	Improved overall I/O performance	552

27.2.2	Simplified configuration definition	553
27.2.3	Reduced skills requirement	553
27.2.4	Maximize utilization of installed resources	553
27.2.5	Enhanced DASD subsystem availability	554
27.2.6	Reduced requirement for more than 256 channels	555
27.2.7	WLM role in dynamic channel-path management	556
27.3	FICON DCM definition	557
27.3.1	A high-level overview of the DCM process	559
27.4	I/O configuration changes	560
27.5	WLM changes	563
27.6	HMC considerations	565
27.7	Building the IOSTmmm module	568
27.8	Activating the changes	568
27.9	Invocations	569
27.9.1	SETIOS DCM=ON/OFF	570
27.9.2	Display IOS,DCM	570
27.9.3	Vary switch command	573
27.9.4	Display commands	574
27.10	Interactions and dependencies	578
27.11	Installation and planning for DCM	578
27.12	Summary considerations	580
<b>Chapter 28. Workload Manager</b>		<b>581</b>
28.1	IBM zEnterprise System	582
28.1.1	zEnterprise BladeCenter Extension	582
28.2	WLM and the z196	584
28.3	Guest Platform Management Provider	585
28.3.1	GPMP and WLM	586
28.3.2	GPMP settings and related commands	587
28.3.3	Configuring the GPMP settings	590
28.3.4	Installation considerations	591
28.4	GPMP performance monitoring	596
<b>Chapter 29. SDSF enhancements</b>		<b>597</b>
29.1	Java API	598
29.1.1	Getting documentation	599
29.1.2	Enabling your application to use the SDSF Java API	599
29.1.3	Verifying that SDSF Java is installed correctly	600
29.1.4	Writing an SDSF Java application	601
29.2	SDSF REXX enhancements	604
29.2.1	ISFLOG examples	606
29.3	Updates to SDSF panels and commands	608
29.4	Health Check History panel	608
29.4.1	Defining the logstream	609
29.4.2	New health check panel in SDSF	610
29.5	Enhanced JES2 and JES3 printer support	612
29.5.1	New action characters on the PR panel	612
29.6	New fields on the INIT panel	615
29.7	New actions on the SR panel	616
29.8	New SEARCH command	616
29.9	Coexistence with earlier releases	617
<b>Chapter 30. JES2 enhancements</b>		<b>619</b>
30.1	EAV support for spool and checkpoint data sets	620

30.1.1	Introduction to spool addressing . . . . .	620
30.1.2	Spool addressing changes in z/OS V1R12. . . . .	621
30.2	Using spool data sets in EAV cylinder-managed space . . . . .	622
30.2.1	Allocating a spool data set in EAV cylinder-managed space . . . . .	623
30.2.2	Dynamically adding a cylinder-managed spool data set to JES2 . . . . .	623
30.3	Enhanced \$S SPOOL error messages . . . . .	625
30.3.1	z/OS V1R12 enhancement . . . . .	626
30.3.2	Improved HASP414 message. . . . .	626
30.4	Better support for multiple TSO/E logons . . . . .	626
30.5	Mean time to restart improvements. . . . .	627
30.6	Stop using captured UCBs for spool volumes. . . . .	627
30.7	SSI updates. . . . .	628
30.8	Serviceability updates . . . . .	628
30.9	Migration and coexistence considerations . . . . .	629
<b>Chapter 31.</b>	<b>JES3 enhancements. . . . .</b>	<b>631</b>
31.1	EAV support for spool, checkpoint and JCT data sets . . . . .	632
31.1.1	Allocating a spool data set in EAS . . . . .	632
31.1.2	Adding the spool data set to JES3 . . . . .	632
31.1.3	Allocating a checkpoint data set in EAS . . . . .	634
31.1.4	Adding the checkpoint data set to JES3 . . . . .	635
31.1.5	The JES3 JCT utility (IATUTJCT). . . . .	635
31.1.6	BADTRACK enhancements . . . . .	635
31.1.7	JES3 data sets on EAV. . . . .	635
31.1.8	Migration considerations for EAV support. . . . .	636
31.2	SAPI enhancements . . . . .	636
<b>Chapter 32.</b>	<b>SMF enhancements . . . . .</b>	<b>639</b>
32.1	Message Flood Automation . . . . .	640
32.2	SMF records to log stream in z/OS V1R9. . . . .	641
32.3	z/OS V1R12 enhancements for flood of SMF records . . . . .	642
32.3.1	Using the log stream dump program. . . . .	642
32.4	Specifying SMF record flood options . . . . .	644
32.4.1	FLOOD parameter . . . . .	644
32.4.2	SMF flood policy definition - FLOODPOL . . . . .	645
32.5	SMF log stream buffer management. . . . .	647
32.6	SMF processor capacity data . . . . .	649
32.7	Initiator processor time . . . . .	652
32.8	SMF dump program exits . . . . .	652
<b>Chapter 33.</b>	<b>GRS enhancements . . . . .</b>	<b>655</b>
33.1	GRSQ SDUMP performance . . . . .	656
33.1.1	GRS contention notification with z/OS V1R12 . . . . .	656
33.2	GRS CTRACE improvements. . . . .	658
33.3	GRS XCF enhancements . . . . .	659
<b>Chapter 34.</b>	<b>XCF enhancements . . . . .</b>	<b>661</b>
34.1	XCF heart beating for critical system members . . . . .	662
34.1.1	New function terminology . . . . .	662
34.1.2	Enhancements to status monitoring . . . . .	663
34.1.3	Installation considerations. . . . .	663
34.1.4	XCF processing for member impairment condition. . . . .	664
34.1.5	Migration and coexistence considerations . . . . .	665
34.1.6	Messages related to an impaired system critical member . . . . .	666

34.1.7	z/OS V1R12 message text enhancements . . . . .	667
34.1.8	XCF reports impairment . . . . .	668
34.1.9	IXCJOIN macro . . . . .	669
34.1.10	Messages for members confirmed impaired . . . . .	674
34.2	SFM-based structure hang recovery . . . . .	678
34.2.1	Structure hang recovery processing . . . . .	679
34.2.2	Migration and coexistence considerations . . . . .	684
34.3	New XES/XCF health checks . . . . .	685
34.3.1	XCF_CF_PROCESSORS check . . . . .	685
34.3.2	XCF_CF_MEMORY_UTILIZATION . . . . .	686
34.3.3	XCF_CDS_MAXSYSTEM . . . . .	688
34.3.4	XCF_SFM_CFSTRHANGTIME . . . . .	689
34.3.5	XCF_CFRM_MSGBASED . . . . .	690
34.3.6	XCF_CF_STR_POLICYSIZE . . . . .	691
34.4	CF Sublist Notification enhancements . . . . .	692
34.5	Mean Time To Recovery (MTTR) for XCF . . . . .	695
34.6	Partitioning action by GDPS controlling system . . . . .	695
<b>Chapter 35.</b>	<b>z/OS HCD V1R12 . . . . .</b>	<b>697</b>
35.1	Enhancements to hardware configuration definition . . . . .	698
35.1.1	I/O autoconfiguration . . . . .	698
35.1.2	Automatic generation of D/R configuration . . . . .	699
35.1.3	Support of over-defined CIB channels on an XMP processor . . . . .	702
35.1.4	Support of third subchannel set . . . . .	704
35.1.5	New hardware support . . . . .	706
35.1.6	Enhancement of the CSS/OS device compare report . . . . .	709
35.1.7	Free format IPL parameter . . . . .	710
35.1.8	Message handling . . . . .	710
35.2	Vary CU command . . . . .	710
35.3	Displaying IOS control unit group information . . . . .	713
<b>Chapter 36.</b>	<b>Unicode enhancements . . . . .</b>	<b>715</b>
36.1	Unicode on Demand . . . . .	716
36.2	Set up a Unicode image . . . . .	716
36.3	Migration considerations . . . . .	717
36.4	Build conversation image . . . . .	718
<b>Chapter 37.</b>	<b>Capacity Provisioning . . . . .</b>	<b>719</b>
37.1	Provisioning Manager report in CPCC . . . . .	720
37.1.1	Provisioning Manager . . . . .	720
37.2	Command correlation . . . . .	727
37.3	Average performance index . . . . .	728
37.3.1	Invoke the support . . . . .	730
37.3.2	Setup and security . . . . .	730
37.4	CICS/IMS support . . . . .	733
<b>Chapter 38.</b>	<b>RRS enhancements . . . . .</b>	<b>735</b>
38.1	Performance issues . . . . .	736
38.2	Cascaded displays with z/OS V1R12 . . . . .	736
38.2.1	DISPLAY enhancements . . . . .	737
38.2.2	Extend COMMENTS field . . . . .	737
38.2.3	EXCEPTION option on the DISPLAY command . . . . .	738
38.2.4	STATUS display . . . . .	740

<b>Chapter 39. Parallel subsystems initialization</b> . . . . .	743
39.1 Mean time to recovery . . . . .	744
39.2 Subsystem initialization routines . . . . .	744
39.2.1 BEGINPARALLEL usage . . . . .	745
39.2.2 MSI dynamic exit routine . . . . .	746
39.3 Migration considerations . . . . .	746
<b>Chapter 40. z/OS Management Facility</b> . . . . .	749
40.1 IBM z/OS Management Facility . . . . .	750
40.1.1 Introduction to the new faces of z/OS . . . . .	750
40.1.2 z/OS ease-of-use enhancements . . . . .	750
40.1.3 z/OS V1R9 and the new faces of z/OS . . . . .	752
40.2 z/OSMF introduction to z/OS V1R12 . . . . .	754
40.2.1 z/OSMF system management tasks with V1R12 . . . . .	754
40.3 Browser for z/OSMF . . . . .	755
40.4 z/OSMF components . . . . .	756
40.5 z/OSMF installation . . . . .	757
40.5.1 z/OSMF administrator . . . . .	758
40.6 z/OSMF customization . . . . .	758
40.6.1 Customizing the Welcome panel for guest users . . . . .	759
40.6.2 Accessing z/OSMF . . . . .	760
40.7 Focus areas for simplification in z/OSMF V1R12 . . . . .	761
40.7.1 z/OSMF plug-ins . . . . .	761
40.7.2 z/OSMF login . . . . .	762
40.7.3 Configuration Assistant . . . . .	763
40.7.4 Links category with a set of links . . . . .	764
40.7.5 Performance category (V1R12) . . . . .	764
40.7.6 Problem Determination category . . . . .	764
40.7.7 z/OSMF Administration category . . . . .	764
40.8 Problem determination in z/OS V1R12 . . . . .	765
40.8.1 Using the z/OSMF Incident Log . . . . .	765
40.8.2 Transmitting data to IBM . . . . .	766
40.8.3 z/OS Problem Documentation Upload Utility . . . . .	766
40.8.4 Implementing the Incident Log . . . . .	767
40.9 z/OSMF Configuration Assistant . . . . .	774
40.9.1 Implementing Configuration Assistant . . . . .	775
40.9.2 z/OS V1R12 IPsec and IKE support . . . . .	777
40.10 z/OSMF Workload Management task . . . . .	778
40.10.1 WLM and z/OSMF . . . . .	779
40.10.2 z/OS V1R12 z/OSMF . . . . .	779
40.10.3 WLM customization with z/OS V1R12 . . . . .	780
40.10.4 WLM primary window . . . . .	780
40.10.5 Workload Manager policy editor . . . . .	781
40.11 Resource Monitoring application plug-in . . . . .	784
40.11.1 Monitoring Desktops task . . . . .	784
40.11.2 System Status task . . . . .	786
40.12 z/OSMF administration . . . . .	789
40.12.1 z/OSMF administrator defining users and roles . . . . .	789
40.12.2 Checking the client-side environment . . . . .	794
40.13 z/OSMF installation considerations . . . . .	796
40.14 z/OSMF dependencies . . . . .	796
40.15 Migration and coexistence . . . . .	796
40.16 Installation considerations . . . . .	797

40.16.1 SMP/E installation scenario . . . . .	797
40.16.2 Configuration process overview . . . . .	798
40.16.3 WebSphere Application Server OEM Edition configuration process. . . . .	799
40.16.4 Configure z/OS for full Incident Log functionality . . . . .	799
40.16.5 Authorizing additional users . . . . .	800
40.17 Preparing your workstation for z/OSMF . . . . .	801
40.17.1 Browser considerations. . . . .	801
40.18 Summary. . . . .	802
<b>Appendix A. SDSF Java API example . . . . .</b>	<b>803</b>
SDSF Java API complete example . . . . .	804
<b>Appendix B. WTOR messages on Auto-Reply . . . . .</b>	<b>807</b>
B.1 Auto-Reply AUTOR00 parmlib member . . . . .	808
B.2 List of Auto-Reply messages . . . . .	811
<b>Appendix C. z/OS Upload utility . . . . .</b>	<b>815</b>
C.1 z/OS Problem Documentation Upload utility . . . . .	816
C.2 z/OS Problem Documentation Upload utility JCL. . . . .	816
<b>Appendix D. BCPii Metal C-example . . . . .</b>	<b>823</b>
D.1 C code . . . . .	824
D.2 JCL to compile, assemble and link-edit . . . . .	842
D.2.1 C compiler options . . . . .	844
D.3 BCPii example as a System REXX . . . . .	844
<b>Related publications . . . . .</b>	<b>847</b>
IBM Redbooks . . . . .	847
Other publications . . . . .	847
Online resources . . . . .	849
How to get Redbooks. . . . .	850
Help from IBM . . . . .	850
<b>Index . . . . .</b>	<b>851</b>



# 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](http://www.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:

1350™	IBM®	Redbooks (logo)  ®	Tivoli®
AIX®	IMS™	Resource Link™	TotalStorage®
BladeCenter®	IMS/ESA®	Resource Measurement Facility™	VM/ESA®
BookManager®	Language Environment®	RETAIN®	VTAM®
CICS®	MQSeries®	RMF™	WebSphere®
DataPower®	MVS™	Solid®	xSeries®
DB2®	NetView®	System Storage®	z/Architecture®
DS8000®	OMEGAMON®	System x®	z/OS®
ESCON®	OS/390®	System z10®	z/VM®
FICON®	Parallel Sysplex®	System z9®	z10™
FlashCopy®	POWER7™	System z®	z9®
GDPS®	PR/SM™	System/390®	zSeries®
Hiperspace™	PrintWay™	SystemPac®	
HyperSwap®	RACF®	TDMF®	
i5/OS®	Redbooks®		

The following terms are trademarks of other companies:

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.

Java, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

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

Intel, 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.

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

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 describes changes in installation and migration when migrating from a current z/OS® V1R10 and z/OS V1R11 to z/OS V1R12. Also described are tasks to prepare for the installation of z/OS V1R12, including ensuring that driving system and target system requirements are met, and coexistence requirements are satisfied. New migration actions are introduced in z/OS V1R12. This book focuses on identifying some of the new migration actions that must be performed for selected elements when migrating to z/OS V1R12.

This book describes the following enhancements:

- ▶ z/OS V1R12 installation, HiperDispatch, System Logger, Auto-reply to WTORs, Real Storage Manager (RSM)
- ▶ DFSMS, DFSORT, Services aids, z/OS Infoprint Server, TSO/E, RMF™, Language Environment®, BCP allocation
- ▶ XML System Services, z/OS UNIX® System Services, BCP supervisor, Extended Address Volumes
- ▶ HyperSwap®, BCPii, (de)ciphering, Predictive Failure Analysis, C language, Hardware instrumentation services
- ▶ FICON® dynamic channel-path management, Workload Manager, SDSF, JES2, JES3, SMF, GRS, XCF, HCD
- ▶ Unicode, Capacity provisioning, RRS, Parallel subsystems initialization
- ▶ z/OS Management Facility (z/OSMF)

## 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 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 43 years.

**Robert Hering** is an IT Specialist at the ITS Technical Support Center, Mainz, Germany. He provides support to clients with z/OS and UNIX System Services-related questions and problems. He has participated in several ITSO residencies since 1988, writing about UNIX-related topics. Prior to supporting OS/390 and z/OS, Robert worked for many years with VM and all its different flavors (VM/370, VM/HPO, VM/XA, and VM/ESA®).

**Paulo Heuser** is a system programmer and technical support coordinator at Bannrisul, in Brazil. He has 33 years of experience in the mainframe field, including z/OS. He holds a technical background in computer networks. His areas of expertise include CICS®, Assembler programming, and computer performance evaluation. He has written extensively on SMF.

**George Kozakos** is a Senior IT Specialist with IBM Australia. He has more than 21 years of experience in MVS™ system programming. George's areas of expertise include Server Time Protocol and GDPS®. He holds degrees in Computing Science and Pure Mathematics.

**Lutz Kühner** is a Senior System z® IT Specialist with the System Technologie Group sales organization in IBM Germany. He provides technical System z support for clients in the financial market. He has 24 years of experience in the mainframe field. His area of expertise includes z/OS and z/OS UNIX, CICS, and high availability topics in general. He has written extensively on z/OS-related topics and has contributed to several IBM Redbooks publications before.

**Jean-Louis Lafitte** is Senior IT Architect at GATE Informatique SA, an IBM Premier Business Partner in Switzerland. He has 39 years of experience on IBM Large Enterprise Systems, has worked on various parallel machines (IBM RP3, CM2, KSR1, IBM SP1, SP2 and BG/L, and P), and has been associated with Parallel Sysplex® since 1984. More recently, he worked on designing hybrid structures exploiting Cell/B.E.™ Jean-Louis holds a Ph.D. in Theoretical Computer Science and several patents in 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 large zSeries® clients.

**Paulo Cesar Nascimento** is a coordinator of support in the Eletrobras Eletronuclear nuclear plant in Brazil. He has 31 years of experience in mainframes. His areas of expertise include z/OS, DB2®, SAP DB, RACF®, and z/OS manager. He has given talks at several IBM meetings, z/OS Explores, where customers share their successful projects.

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

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

Robert Haimowitz  
Alfred Schwab, editor  
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](http://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](http://ibm.com/redbooks)

- ▶ Send your comments in an email to:

[redbooks@us.ibm.com](mailto: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>





## z/OS V1R12 enhancements

The z/OS strategy has a long-term commitment to simplify the z/OS platform. The past several releases of z/OS delivered improvements in the following areas:

- ▶ Simplifying diagnosis and problem determination
- ▶ Network and security management
- ▶ Overall z/OS
- ▶ I/O configuration
- ▶ Sysplex and storage operations

These improvements can help simplify systems management, improve application programmer, system programmer, and operator productivity, and make the functions easier to understand and use.

### **Scalability and performance**

In the areas of scalability and performance, z/OS has the ability to intelligently manage workloads, reprioritize work, and dynamically reallocate system resources between applications quickly and efficiently. z/OS and System z can handle unexpected workload spikes, improve your system's efficiency and availability, all while meeting application and business priorities.

In the area of performance, there are improvements in a variety of areas. Some are focused on overall performance and others on customer pain points. One area of continuing improvement is cache and memory management; this will continue to be true for the foreseeable future. The time it takes to retrieve data from memory, while progressively shorter in the absolute sense on newer server models, has become progressively longer when measured in processor cycle time increments. In other words, though memory access is much faster than it used to be, processors spend more cycles waiting for it than they ever did. This makes the system's management of cache very important.

The z/OS platform supports many new application development technologies, such as Java™, Perl, PHP, XML, Unicode, HTML, SOAP, and other web services, in addition to C/C++ and other application development tools. And z/OS continues to update its traditional application development tools as well.

Improvement in the application development stack can reduce deployment and maintenance costs for applications deployed on the z/OS platform by bringing the leveraging technical capabilities of the z/OS software stack.

In the area of data serving, like other operating systems, z/OS provides support for current application enablement technologies, but what sets z/OS apart is the ability to operate both new and existing applications within the same system, and in close proximity to the corporate data residing on z/OS. WebSphere® applications can run on the same z/OS system as the DB2 database, which can enable tight, security-rich local connections ideal for high volume transactional throughput. Current CICS or IMS™ transactions can be extended with these new technologies to deliver value in new and innovative ways, without incurring the substantial cost required to rip and replace current core assets.

System z and z/OS are ideally suited to perform as a data serving hub for the enterprise. The platform's classic strengths of availability, security, reliability, scalability, and management have made the mainframe the de facto gold standard for data serving and OLTP. Market requirements for increased security and simplified data management, and the increasing need for real-time Business Intelligence make consolidating more data onto the mainframe an attractive option for many enterprises. New technologies, such as XML, represent net data serving workloads on the platform.

## **Standards Currency**

The Common Criteria (CC) is meant to be used as the basis for evaluation of security properties of IT products. By establishing such a common criteria base, the results of an IT security evaluation may be meaningful to a wider audience. The CC permits comparability between the results of independent security evaluations. The CC does so by providing a common set of requirements for the security functionality of IT products and for assurance measures applied to these IT products during a security evaluation. The evaluation process establishes a level of confidence that the security functionality of these IT products and the assurance measures applied to these IT products meet these requirements. The evaluation results may help consumers to determine whether these IT products fulfill their security needs.

The Common Criteria is identified in NSTISSP #11 as a requirement for the US Federal Government for all Commercial off the Shelf (COTS) acquired software.

## **Simplified monitoring and management**

IBM has embarked on a long-term commitment to simplifying the z/OS platform. The past several releases of z/OS delivered improvements in the areas of simplifying diagnosis and problem determination; network and security management; as well as overall z/OS, I/O configuration, sysplex and storage operations. These improvements can help simplify systems management, improve application programmer, system programmer, and operator productivity, and make the functions easier to understand and use.

In this chapter we provide information regarding the z/OS V1R12 improvements to the operating system.



## 1.1 Base control program (BCP) enhancements

The BCP provides essential operating system services. It 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.

### Allocation performance

In z/OS V1R12, a number of design changes are made to allocation to improve performance for address spaces that allocate a large number of data sets in a short time. These changes are expected to markedly reduce the startup time for these address spaces, such as DB2 address spaces and batch jobs that process a large number of data sets per job step.

### Parallel subsystem initialization

In z/OS V1R12, subsystem initialization is changed from serial to parallel for initialization routines that start after the SMS initialization routine (IGDSSIIN). This is intended to enable you to reduce system startup time by allowing most of these routines to run in parallel.

### Availability and RAS

Beyond server availability, applications and data must be available with good performance as well. For the System z platform this means hardware, connectivity, operating system, subsystem, database, and application availability, too. z/OS, System z servers, and System Storage® working together can provide outstanding availability: System z servers are designed to reduce planned and unplanned outages through the use of self-healing capabilities, redundant componentry, dynamic sharing, and the ability for concurrent upgrades and microcode changes.

With every release, z/OS continues to refine its error checking, fault tolerance, isolation, error recovery, and diagnostic capabilities. Beyond single system availability is z/OS Parallel Sysplex clustering and GDPS disaster recovery. Parallel Sysplex is designed to provide your data sharing applications and data with not only continuous availability for both planned and unplanned outages, but also near-linear scalability and read/write access to shared data across all systems in the Parallel Sysplex for data sharing applications.

### Enhancements to 1 MB large page support

Large (1 MB) pages were introduced in z/OS V1R10. In z/OS V1R12, the nucleus data area is planned to be backed using 1 MB pages. This is intended to reduce the overhead of memory management for nucleus pages and to free translation lookaside buffer (TLB) entries so they can be used for other storage areas. This is expected to help reduce the number of address translations that need to be performed by the system and help improve overall system performance.

### Auto reply to WTORs

In z/OS V1R12, a new timed auto reply function provides an additional way for the system to respond automatically to write to operator with reply (WTOR) messages. This new function is designed to allow you to specify message ID timeout values, and default responses in an auto-reply policy, and to be able to change, activate, and deactivate auto-reply with operator commands. Also, when enabled, it is designed to start very early in the IPL process and continue unless deactivated. An IBM-supplied auto-reply policy in a new AUTOR00 parmlib member that you can replace or modify is also available. This new function is expected to help provide a timely response to WTORs and help prevent delayed responses from causing system problems.

## CSV enhancements

In z/OS V1R12, a number of enhancements are made to the processing of PROGxx parmlib members and to Link List Lookaside (LLA) processing. These include support as follows:

- ▶ In PROGxx for passing a specified parameter to a dynamic exit
- ▶ Automatically including alias names for modules to be placed in Dynamic LPA
- ▶ Specifying volumes on SYSLIB for data sets so they need not be cataloged in the master catalog
- ▶ A REPLACE option for exits to assure there is no window during which an exit is unavailable
- ▶ A new SVCNUMDEC keyword to specify the SVC number to be added.

Additionally, a new DEFAULTS statement is available so you can specify processing defaults intended to help prevent common errors. This includes allowing you to specify that LNKLIST DEFINE always require COPYFROM, that it default to COPYFROM(CURRENT), and that it automatically process aliases for modules added to Dynamic LPA.

LLA processing is now designed to support the use of dynamic LLA exits and to process multiple MODIFY commands in parallel.

## Enhance D SYMBOLS command output

A new SUMMARY keyword of the DISPLAY SYMBOLS command is designed to provide summary information about symbols used on the system, including how many are in use. This can help you determine how many additional symbols can be defined.

## Duplicate temporary data set name support

When two or more jobs having the same job name begin to process within the same system clock second and specify the same temporary data set names, the second and subsequent jobs will fail with JCL errors while attempting to allocate data sets with duplicate names. In z/OS V1R12, you will be able to use a new parmlib option to specify that the system use the data set naming convention for unnamed temporary data sets instead, which substantially reduces the probability of this JCL error without the need to change JCL.

## RMF SMF log stream

In z/OS V1R9, support was added to write SMF data to log streams. In z/OS V1R12, it is planned to enhance RMF to read SMF records directly from a log stream. This is intended to allow you to eliminate any intermediate steps you currently use to unload SMF data from a log stream to a sequential data set for RMF postprocessing.

## SMF30 uncaptured processor times

Initiator address spaces consume processor time on behalf of starting and ending job steps that in prior releases are not associated with a particular batch job. There can be considerable variation in the processor time consumed by an initiator for different jobs. To help you better understand the resources consumed by batch jobs and improve the accuracy of chargeback programs, z/OS V1R12 is designed to record the processor time consumed for job steps in initiator address spaces using new fields in SMF Type 30 records.

The SMF30ICU and SMF30ISB fields of the processor accounting section (aka the Processor Section) of the SMF type 30 record report the time that is used by the initiator “between” job steps. These “in between” time values have been dubbed “un-captured processor times”. Some of the time that is included in these fields belongs to the step that is terminating, and some of the time belongs to job step selection and pre-allocation processes on behalf of the next step in the job or next job to be executed. Clients have reported a need to see a

breakdown of the time spent on the ending step as opposed to time spent preparing to execute the next step or next job. This line item will address that need by adding four new fields to the SMF type 30 record to report TCB and SRB processor times used during job step termination and by job step initialization.

### **SMF records for flood automation**

In z/OS V1R12, new support detects and automates the system's response to tasks that are writing SMF records at unusually high rates. SMF record flooding automation is now designed to allow you to define a policy for responding to these situations in the SMFPRMxx parmlib member by specifying whether record flooding automation is to be active, whether operators are to be warned, and the actions to take for specific SMF record types if record flooding occurs. This is intended to limit the impact of such problems by allowing less-important data to be discarded while keeping the data from critical SMF records intact. Additionally, new function is planned for the SMF dump program (IFASMF DL) to provide additional information to help you develop a record flooding policy.

### **SMF processor capacity data capture**

SMF is updated to generate event-driven type 30 and type 89 records when a change in processor capacity is detected. Fields containing processor capacity data are added to these records. This support also introduces a new SMFPRMxx parmlib option to govern the number of allowable event-driven intervals, and includes generating a new subtype of the type 90 record when a change in processor capacity is detected. The IFAURP billing report is also updated to allow for computing the SU factor from the data in the new type 89 records (as well as pull the value from the internal processor table for older records). Also, the reports are reformatted to structure the data by type and serial, rather than by type, model and serial.

### **RTM enhancements**

In z/OS V1R12, new functions for recovery and termination processing are implemented. These include a new option on ESTAEX to specify that SPIE or ESPIE exits be superseded by ESTAEX, a new option on ESPIE to request percolation to RTM, and passing information about held locks to ESTAE-type recovery routines.

### **Superzap support for EAV3**

In z/OS V1R12, superzap (AMASPZAP) supports dumping and altering data for sequential, partitioned, and direct data sets placed in the extended addressing space (EAS) on extended address volumes (EAVs).

## **1.1.1 HCD/HCM**

The I/O configurations to the operating system (software) and the channel subsystem (hardware) must be defined. The Hardware Configuration Definition (HCD) component of z/OS consolidates the hardware and software I/O configuration processes under a single interactive user interface.

Hardware Configuration Manager (HCM) is the graphical user interface to HCD. HCM interacts with HCD in a client/server relationship (that is, HCM runs on a workstation and HCD runs on the host). The host systems require an internal model of their connections to devices, but it can be more convenient and efficient for the system programmer to maintain (and supplement) that model in a visual form. HCM maintains the configuration data as a diagram in a file on the workstation in sync with the IODF on the host. While it is possible to use HCD directly for hardware configuration tasks, many customers prefer to use HCM exclusively, due to its graphical interface.

## **Coexistence of GDPS and HCD**

HCD/HCM is enhanced to allow a client to define only one OS config, and the second OS config will be generated automatically from the HCD/HCM. This enhanced function avoids unnecessary efforts, provides a consistent view for the GDPS configurations, and reduces human errors.

## **Ability to specify IPL attributes**

HCD is enhanced to allow IPL attributes to be entered in free format. This allows the central management of IPL parameters also for non-z/OS systems.

## **Predefine PSIFB connections**

HCD/HCM is enhanced to allow the CF end of the PSIFB link to be defined without being “connected.” This allows the CF to simply connect to whatever system responded when the link was physically connected without requiring an outage for the new connection.

## **1.1.2 Binder support**

The binder provided with z/OS performs all of the functions of the linkage editor. The binder link-edits (combines and edits) the individual object decks, load modules, and program objects that comprise an application, and produces a single program object or load module that you can load for execution. When a member of a program library is needed, the loader brings it into virtual storage and prepares it for execution.

### **FASTDATA enhancements**

In z/OS V1R12, the program management Binder is making program object attribute data (PMAR data) available to programs using the fast data interface, and to support programs loaded using the z/OS UNIX System Services load service (loadhfs).

### **Binder and linkage editor support**

In z/OS V1R12, the program management Binder now supports data sets having extended task I/O table (XTIOT) entries. This affects BAM XTIOT and EAV support.

### **Binder RMODE in multi-segment modules**

In z/OS V1R12, the program management binder allows you to specify that a specific residency mode (RMODE) be applied to all initial load classes of a program object, rather than the classes in the first segment containing the entry point. This new function is intended to offer application programmers more flexible options for program storage residency.

### **BCP Binder and AMBLIST improvements**

In z/OS V1R12, a number of small Binder enhancements are introduced, as follows:

- ▶ Sample programs to illustrate the use of both standard and Fastdata Binder APIs in High-Level Assembler and C.
- ▶ Character translations in AMBLIST LISTLOAD output for load modules.
- ▶ Improved AMBLIST header information for z/OS UNIX files.
- ▶ Support for long names for AMBLIST LISTOBJ for object modules in z/OS UNIX files.

### 1.1.3 Global resource serialization (GRS)

The global resource serialization component processes requests for resources from programs running on z/OS. Global resource serialization serializes access to resources to protect their integrity. An installation can connect two or more z/OS systems with channel-to-channel (CTC) adapters to form a GRS complex to serialize access to resources shared among the systems.

When a program requests access to a reusable resource, the access can be requested as exclusive or shared. When global resource serialization grants shared access to a resource, exclusive users cannot obtain access to the resource. Likewise, when global resource serialization grants exclusive access to a resource, all other requestors for the resource wait until the exclusive requestor frees the resource.

#### **GRS reporting enhancements**

RMF captures and reports information on how many units of work (WEBs) are running or waiting for processor and the overhead of SIGP processing.

#### **Unauthorized ENQ (ISGENQ) with ECB**

In z/OS V1R12, a new option is introduced for the ISGENQ service that can be used to serialize resources. This new support is designed to allow an unauthorized program to interrupt serialization processing and opt not to continue to attempt to obtain control of a resource when the resource is not available. For example, a programmer might wish to set a time limit for obtaining control of a resource. This is expected to help programmers to better manage contention delays and remove pending enqueue requests in recovery.

#### **Exploit GRS identity for latches used by USS**

In z/OS V1R12, z/OS UNIX System Services file system processing is designed to provide better information on the **DISPLAY GRS, CONTENTION** and **D GRS, ANALYZE** commands by identifying itself as the holder of held latches to GRS. This is intended to help you diagnose and take corrective actions for latch contention problems that involve file system processing.

### 1.1.4 Sysplex enhancements

A sysplex refers to a tightly-coupled cluster of independent instances of the z/OS operating system. A sysplex can be either basic or parallel. A basic sysplex can communicate using channel to channel (CTC) connections between LPARs. Parallel Sysplex uses something called a Coupling Facility (CF).

#### **SFM CF structure related hangs**

In z/OS V1R12, XES builds on the existing hang-detect monitoring capability. A new CFSTRHANGTIME SFM policy now allows you to specify how long CF structures can have pending requests. When the time is exceeded, SFM now drives corrective actions to try to resolve the hang condition. This is intended to help you avoid sysplex-wide problems that can result from an affected CF structure that is waiting for timely responses from all the systems using it.

#### **MTTR enhancements in couple data set initialization**

In z/OS V1R12, a number of improvements are implemented for XCF to help reduce IPL time. This change is expected to help improve availability by reducing the time required to IPL the systems in a Parallel Sysplex environment.

## 1.1.5 Workload Manager (WLM)

For z/OS, the management of system resources is the responsibility of the workload management (WLM) component. WLM manages the processing of workloads in the system according to the company's business goals, such as response time. WLM also manages the use of system resources, such as processors and storage, to accomplish these goals.

### Resource groups for batch management

In z/OS V1R12, WLM now considers resource group maximums and the projected increase in system or sysplex demand before starting initiators during resource adjustment and policy adjustment processing when the service class has been assigned to a resource group and a resource group maximum has been defined. The Type 99 SMF record is also extended to show when the number of initiators to be started was limited for this reason. These changes are intended to improve WLM batch management.

### Discretionary work time slicing

In z/OS V1R12, changes to the dispatching of discretionary work have been made. The system is now designed to run discretionary work for a longer period of time before dispatching other discretionary work, while still interrupting it after short periods for nondiscretionary work. This change is intended to help improve the throughput for systems with a high percentage of discretionary workloads.

### WLM policy editor

WLM policy editor functionality is to be integrated into z/OSMF R12, which will facilitate the creation and editing of WLM service definitions, installation of WLM service definitions and activation of WLM service policies, and monitoring of the WLM status of a sysplex and the systems in a sysplex.

### Return OOCOD information

In z/OS V1R12, the WLM service for requesting LPAR-related data (REQLPDAT) is enhanced to include character-based data about the machine model, a Model-Permanent-Capacity Identifier, a Model-Temporary-Capacity Identifier, the Model-Capacity Rating, the Model-Permanent-Capacity Rating, and the Model-Temporary-Capacity Rating. This new data is intended to be used for reporting and use by vendors.

### Manage tasks in queue server correctly

WLM honors the behavior of environments such as WebSphere and now manages work in enclave server address spaces and queue server address spaces that are running outside of enclaves. This solves the problem of not managing swapped-out server address spaces that run only work outside of enclaves and solves the problem of running the work outside an enclave, as for example the WebSphere garbage collector with a priority that is too low.

## 1.1.6 z/OS Management Facility (zOSMF)

The z/OS Management Facility is intended to enable system programmers to more easily manage and administer a mainframe system by simplifying day-to-day operations and the administration of a z/OS system.

## **z/OSMF incident log enhancements**

In z/OSMF V1R12, a number of improvements are made and intended to help you manage problem data more easily, as follows:

- ▶ For the incident log function
- ▶ For encryption of dumps to be sent to IBM
- ▶ Breaking dumps into multiple data sets that can be sent via FTP in parallel to reduce transmission time
- ▶ Specifying additional data sets to an incident
- ▶ Adding free-form comments to new fields for problem descriptions
- ▶ FTP destinations and a function to display FTP status

## **zOSMF currency browser support**

Support for Microsoft® Windows® Vista and Windows 7 has been implemented. This release of z/OSMF also supports Microsoft Internet Explorer 7, Internet Explorer 8, Mozilla Firefox 3.0, and Firefox 3.5.

## **z/OSMF navigation services**

In z/OSMF V1R12, an interface to allow the addition of non-z/OSMF launch points and links to the navigation tree is implemented.

## **Multiple EAR support**

Changes to z/OSMF packaging to support delivery of applications via multiple EAR files have been implemented.

## **1.1.7 Cross-system coupling (XCF)**

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.

### **Extend XCF heartbeat for critical members**

In z/OS V1R12, XCF now monitors components that identify themselves as critical to the function of a Parallel Sysplex, and initiate partitioning actions if a monitored component fails to respond when polled for status or indicates impairment. This function is intended to help reduce the incidence of sysplex sympathy sickness due to unresponsive critical components. GRS is planned to exploit these XCF critical member and XES enhanced exit hang detection functions in both ring and star modes. Additionally, GRS will be designed to monitor key global enqueue and queue scan processing tasks and to inform XCF if it detects that GRS is impaired.

### **SETXCF command**

The **SETXCF** command includes a **REALLOCATE** option that can be used to cause XCF to analyze and reallocate structures in a Coupling Facility. In z/OS V1R12 a new **TEST** option allows you to get information about the changes the command would make and any errors that would be encountered if it were issued without the new option. The report created by the command (similar to that already provided by message IXC545I) is intended to provide information you can use to decide whether to take any needed actions to resolve errors or simply to reissue the command without **TEST** to make the changes effective. Also, consolidated information about any exceptions that were encountered is planned to be provided by **REALLOCATE** processing to make it easier to find.

## 1.1.8 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.

### **System Logger RAS for enforcement of share options**

In z/OS V1R12, System Logger is enhanced to correct the VSAM SHAREOPTIONS for new log stream data sets when it detects that they are not correctly set. Messages are issued to indicate that Logger has detected and corrected a data set's SHAREOPTIONS settings. This new function is intended to prevent data set access problems from arising when SHAREOPTIONS(3,3) has not been set in the DATACLAS or ACS routine used when allocating log stream data sets.

### **System Logger VSCR**

System Logger is providing z/OS common storage constraint relief to aid in an overall z/OS reduction goal. The objective is to aid in the z/OS common storage constraint relief goal by reducing the 31-bit common storage usage below the bar. Many System Logger modules currently loaded and running in LPA are moved into load modules within the System Logger (IXGLOGR) private address space.

## 1.1.9 Capacity Provisioning

Dynamic and sometimes unexpected system workload demands coupled with error-prone manual processes may cause you to miss service-level commitments or overpay for system capacity. IBM Implementation Services for System z Capacity Provisioning provides IBM skills and an understanding of autonomic management to help you improve system utilization and control the activation and deactivation of additional resources to meet business needs.

The Capacity Provisioning Control Center is now supporting displaying provisioning reports supported by the Capacity Provisioning Manager. This is intended to simplify the investigation of Capacity Provisioning reports and operation of the Capacity Provisioning server from z/OSMF. The Capacity Provisioning Manager client is updated to provide support for Windows Vista.

### **BCP support of CICS/IMS transaction classes**

In z/OS V1R12, Capacity Provisioning is using the delay data for transaction service classes provided by RMF starting with z/OS V1R11, or an equivalent product, to help determine whether a provisioning action is required for servers on which CICS and IMS are running. Monitoring delay data for CICS and IMS transaction classes is intended to help improve capacity provisioning decisions for servers with LPARs running CICS and IMS.

## 1.1.10 Predictive failure analysis (PFA)

In z/OS V1R12, a new predictive failure analysis check is introduced to monitor the rate at which SMF records are generated. When the rate is abnormally high for a particular system, the system will be designed to issue an alert to warn you of a potential problem.

In z/OS V1R12, function is introduced for PFA to allow you to specify that PFA ignore data related to certain jobs or address spaces when you expect their behavior to be atypical. This can help you improve the overall accuracy of PFA checks for LOGREC, message, and SMF record arrival rates.



Four changes are also introduced to improve the quality of PFA modeling. These changes are designed to do the following:

- ▶ Capture data when exceptions are issued to help you identify problems
- ▶ Use dynamic modeling intervals based on system stability
- ▶ Discard the last hour's LOGREC data from before a shutdown
- ▶ Monitor smaller increments of common storage assigned to a system

### 1.1.11 XML System Services

z/OS XML System Services (z/OS XML) is an XML processing component of the z/OS operating system. It contains an XML parser intended for use by system components, middleware, and applications that need a simple, efficient, XML parsing solution. z/OS XML can parse documents either with or without validation. XML allows you to tag data in a way that is similar to how you tag data when creating an HTML file. XML incorporates many of the successful features of HTML, but was also developed to address some of the limitations of HTML.

#### Dynamic schema

In z/OS V1R12, XML System Services is enhanced to provide XML schema validation support by allowing applications to extract schema-related information from an XML instance document without the application first performing a separate parse. This is designed to improve the usability of the validating parsing interface and intended to reduce the processing cost of obtaining this information.

#### Fragment parsing

In z/OS V1R12, z/OS XML System Services now allows you to validate part of an XML document, rather than the entire document. Called fragment parsing, this capability is intended to reduce the processing cost of performing validation by allowing you to validate only a portion of a document rather than requiring the validation of the entire document. For example, this can be useful when only a subset of a large document containing multiple fragments has changed.

#### Restrict root element name

In z/OS V1R12, XML System Services now provides a new validating parse capability that allows applications to restrict the set of element names to be accepted as valid root elements to a subset of those allowable in an XML schema. This is intended to provide an additional level of validation capability beyond that provided by the W3C schema language.

### 1.1.12 C/C++ enhancements

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.

z/OS XL C/C++ includes:

- ▶ A C compiler (referred to as the z/OS XL C compiler)
- ▶ A C++ compiler (referred to as the z/OS XL C++ compiler)
- ▶ Performance Analyzer host component, which supports the IBM C/C++ Productivity Tools for the IBM OS/390 product
- ▶ A set of utilities for C/C++ application development

## **C/C++ enhancement**

In z/OS V1R12, program management Binder now complements the existing Binder C/C++ API DLL functions (iewbndd.so, iewbndd.x) with an XPLINK version (iewbnddx.so, iewbnddx.x). This is designed to offer XPLINK applications improved performance by eliminating expensive XPLINK to non-XPLINK transitions when the Binder functions are called. Also, a C/C++ header provides a way to map the IEWBMMP structure (\_\_iew\_modmap.h). For C and C++ users, this will simplify the task of processing the module map, which the Binder creates in programs when the MODMAP option is used.

### **1.1.13 Common Information Model (CIM)**

The Common Information Model (CIM) is a standard data model for describing and accessing systems management data in heterogeneous environments. It allows system administrators or vendors to write applications (CIM monitoring clients) that measure system resources in a network with different operating systems and hardware. With z/OS CIM, it is possible to use the DMTF CIM open standard for systems management, which is also implemented on further major server platforms (for example, Linux® on zSeries, Linux on xSeries®, i5/OS®, or AIX®).

#### **CIM standards currency**

In z/OS V1R12, the CIM Server is upgraded to the 2.10 version of the OpenPegasus CIM Server. Also, the CIM Servers Schema repository is updated to CIM Schema version 2.22. This is intended to keep the z/OS CIM Server and schema current with the CIM standard from OpenGroup and DMTF, and to allow z/OS management applications manage z/OS in an enterprise environment.

#### **z/OS storage management CIM Provider-SMI-S Host Profile provider**

The CIM Server includes providers for z/OS to be interoperable with the SMI-S storage management standard v1.3. It includes providers for the host discovered resources (HDR) and host bus adapter (HBA) profiles.

## **1.2 z/OS and hardware**

The following items are related to z/OS and the hardware on which z/OS executes.

#### **Exploit System z10 API command correlation support**

Currently, successful or failing command completion events cannot be associated to a specific request. With the new command correlation support provided by the hardware, it is now possible to handle such command completions correctly, which increases reliability of hardware management considerably.

#### **Hardware Instrumentation Services (HIS)**

HIS support for dynamic processor speed can be changed dynamically without a re-IPL. This can occur for clients using replacement capacity, On/Off Capacity on Demand, or during cooling problems with the machine. HIS currently relies on the processor speed to remain constant throughout an instrumentation run. If the processor speed were to change from underneath HIS without its knowledge, data collected for that run would either be thrown out or misinterpreted.

## BCPii support for System z hardware

The BCPii callable services continues to expand its support of more System z API commands and attributes related to the CPC and images. This allows a BCPii application to have more options in its control of the System z hardware.

## 1.3 z/OS UNIX System Services (z/OS UNIX)

The UNIX System Services element of z/OS 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.

### TSOCMD command support

Previously, the `tsocmd` shell command was available only from the Tools and Toys section of the z/OS UNIX System Services website. In z/OS V1R12, z/OS support for this function is now integrated. Unlike the existing TSO command, the `tsocmd` command can be used to issue authorized TSO/E commands.

### USS shell and utilities support for FILEDATA=RECORD

In z/OS V1R12, there is support in z/OS UNIX System Services for the record file format in the `cp`, `mv`, `ls`, `pax`, and `extattr` shell commands as well as the `ISHELL` command. In addition to binary and text format, files can be handled in record file format. z/OS applications accessing these files by using QSAM, BSAM, VSAM, or BPAM and coding `FILEDATA=RECORD` will be able to take advantage of the record file format to read and write data as records.

### USS support for FILEDATA=RECORD

z/OS UNIX supports a number of file formats, including NA (not defined), Binary, and delimited text formats including NL (text delimited using newline characters), CR (carriage return characters), LF (using line feed characters), CRLF (both carriage return and line feed characters), LFCR (both line feed and carriage return characters), and CRNL (both carriage return and newline characters). In z/OS V1R12, there is new support for a record file format, comprising prefixed records in which the prefix defines the record length.

### z/OS UNIX kernel RAS

The following RAS items are implemented in z/OS V1R12:

- ▶ Schedule synchronous SRBs for creating pthreads to better serialize use of below-the-line storage.
- ▶ Retry Fork processing when VSM list processing causes Fork to fail with EAGAIN.
- ▶ Currently, fork COW processing is turned off when MaxSharePages reaches 62.5%. The limits detection code could be changed to put out the limits message for this limit when it reaches 62.5% rather than when it reaches 85%.
- ▶ Clients have reported problems with latches being held for an excessive period of time, but UNIX System Services did not notify them of the condition because no contention existed. A new message is planned to notify customers about the condition so actions can be taken to resolve the issue before the latch impedes other work in the system. The system will query all held latches and issue a message to indicate a latch has been held a long time, and identify the job that owns it. A new message (BPXM123E) is created for this purpose.

### **Dynamic socket limits**

In z/OS V1R12, z/OS UNIX System Services supports changing the number of reserved ports for Common Inet, the maximum number of sockets that can be obtained for a given file system type, by using the **SET OMVS** and **SETOMVS** operator commands. This can help you improve availability by making it unnecessary to restart OMVS to change these values

### **UNIX System Services (USS) support for RFC 5014**

USS and Communications Server support for Source Address Selection/Get Partner Info has USS updates for the new socket options for Default Address Selection as described in RFC 5014. There are macro updates for the new socket options and common INET support for these options when there are multiple TCPIP stacks configured.

### **Support for mmap for NFS client**

z/OS UNIX System Services supports the memory mapping (mmap) function for files in zFS and HFS file systems. In z/OS V1R12, new support allows applications to use memory mapping for NFS client files. This will enable NFS-mounted file systems to be used by applications that use memory mapping.

## **1.4 zFS support**

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).

### **Removal of multisystem aggregate support**

The zFS strategic direction is to remove multifile system aggregate support. In z/OS V1R12 USS is removing the existing commands and displays that allow users to manipulate file systems in a multifile system aggregate.

## **1.5 JES components**

JES2 and JES3 are the primary subsystems that are required when using the z/OS operating system.

### **JESXCF design change**

In z/OS V1R12, the JESXCF component allows you to log on to multiple systems within a sysplex using the same TSO/E user ID.

### **JES2 SAPI notifications for all data sets**

In z/OS V1R12, JES2 provides function you can use to specify, using the SYSOUT application programming interface (SAPI), that a program receive ENF 58 notifications when SYSOUT data sets have been deleted. This new function is designed to help applications to monitor the progress of print data sets through the system.

### **JES3 SAPI notifications for all data sets**

In z/OS V1R12, JES3 provides function you can use to specify, using the SYSOUT application programming interface (SAPI), that a program receive ENF 58 notifications when SYSOUT data sets have been deleted. This new function is designed to help applications to monitor the progress of print data sets through the system.

### **JES2 EAV support for spool and checkpoint data sets**

In z/OS V1R12, JES2 now allows both spool and checkpoint data sets to reside in extended addressing space (EAS) on an extended address volume (EAV), making it possible to place both spool and checkpoint data sets anywhere on an EAV, and to define spool data sets up to the maximum size of 1,000,000 tracks (approximately 56 GB).

### **JES3 EAV support for spool data sets**

In z/OS V1R12, JES3 now allows the spool, checkpoint, and Job Control Table (JCT) data sets to be placed anywhere on an extended address volume (EAV). When allocated as extended-format data sets, spool data sets up to 16,777,215 tracks (approximately 922 GB) will be supported.

## **1.6 TSO/E**

TSO/E is a base element of z/OS. TSO/E allows users to interactively share computer time and resources. In general, TSO/E makes it easier for people with all levels of experience to interact with the MVS system.

### **TSO/E XTIO T support**

In z/OS V1R12, TSO/E now supports extended task I/O tables (XTIO Ts), uncaptured UCBs, and DSABs above 16 MB for data sets allocated by programs.

### **TSO/E password special character support**

In z/OS V1R12, TSO/E accepts passwords that include one or more special characters. This is intended to leave the checking for acceptable password characters to an external security manager such as RACF.

## **1.7 Library Server**

z/OS information is accessible using panel readers with the BookServer/Library Server versions of z/OS books in the Internet library at:

<http://www.ibm.com/systems/z/os/zos/bkserv/>

### **Library Server performance**

Indexing and administration improvements are introduced with z/OS V1R12 Library Server to improve performance when building new catalogs and supporting multiple users on heavily loaded systems. Also, indexing is implemented to capture the author's intended definition of primary nodes for an InfoCenter's Table of Contents. Administrative improvements include long filename support, programmatically checking for the required level of Java, and generation of a new Test and Diagnostics page for use by Library Server administrators and IBM support personnel.

### **User support for personalized document sets**

A new Personal BookCase function in z/OS V1R12 Library Server allows you to create, use, and share your own subset of the documents from a Library Server catalog. This function is designed to allow you to configure a Personal BookCase that includes the shelves and documents, as well as the infocenters and topics, that you are interested in so you can have the reference documents you routinely use available quickly.

### **User and administrative user usability enhancements**

In z/OS V1R12, Library Server usability enhancements are improved for user interfaces, including improved navigation between certain dialogs, modernized icons, and descriptive hover popups for documents on a shelf.

## **1.8 z/OS V1R12 new components**

Following is a new component with z/OS V1R12.

### **z/OS Runtime Diagnostics (RTD)**

In z/OS V1R12, a new component called z/OS Runtime Diagnostics is implemented. This function is designed to help you reduce the time spent deciding what actions you should take to resolve a problem and identify potentially related symptoms and causes when it appears that there is a significant system problem that affects its ability to process your workloads, but is still running. Often, you must analyze problems like this in a short time to preserve application availability.

Runtime Diagnostics is designed to be activated using the START operator command and to process the operator messages recorded in the OPERLOG log stream to identify critical messages. In addition, it is designed to search for serialization contention, address spaces consuming a high amount of processor time, and to analyze the system trace table for patterns common to looping address spaces. Runtime Diagnostics are expected to run in less than one minute to return results quickly enough to help you make decisions about alternative corrective actions and help you maintain high levels of system and application availability.

## **1.9 Data Facility Storage Management Subsystem (DFSMS)**

DFSMS comprises a suite of related data and storage management products for the z/OS system. DFSMS is an operating environment that helps automate and centralize the management of storage based on the policies that your installation defines for availability, performance, space, and security. The heart of DFSMS is the Storage Management Subsystem (SMS). Using SMS, the storage administrator defines policies that automate the management of storage and hardware devices. These policies describe data allocation characteristics, performance and availability goals, backup and retention requirements, and storage requirements for the system.

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.

## DFSMS catalog

In z/OS V1R12, the catalog address space (CAS) is now designed to check for contention periodically. Based on an interval you specify and the reason for contention, CAS now writes a logrec record and terminates certain tasks that have waited for longer than the specified maximum when contention checking occurs. A new MODIFY CATALOG,CONTENTION command allows you to specify a different interval than the 30 minute default, or to disable CAS contention detection. This new function is intended to prevent tasks that take excessive time to complete, or that never complete, from affecting catalog performance.

## DFSMS OCE RAS enhancements

Open/Close/EOV (OCE) RAS items are providing support for the following items:

- ▶ A new DCB abend exit ignore option to additionally bypass the associated determinant abend message.
- ▶ EOV processing now detects a missing last volume condition when reading, and if the JFCB and TIOT built by allocation have room, EOV will call catalog services to determine the additional volumes and then update the JFCB with the volume serial numbers and the TIOT with the device address. If this recovery is not possible, EOV will abnormally terminate the job with abend 637-B0.
- ▶ Externalize in a new message the reason codes documenting the specific reason for a FREE=CLOSE failure.
- ▶ Modernize the component by changing this processing to issue existing abend ABEND50D with new reason code X'24'. IEC999I will continue to be issued with the explanation text.
- ▶ Add new reason codes for abend 413 and abend 637 when detected for messages IEC709I, IEC710I, IEC711I, and IEC712I. Also, a recovery option will be made available to the DCB abend exit.
- ▶ Improve DSAB chain search performance by exploiting the improved GETDSAB hash algorithm in common services module IFG019RA.

## DFSMS disk striping

In z/OS 1R12, VSAM record level sharing (RLS) supports striped data sets. This is designed to bring the benefits of VSAM striping, such as allowing single application requests for records in multiple tracks or control intervals (CIs) to be satisfied by concurrent I/O requests to multiple volumes. Using striped data sets can result in improved performance by transferring data at rates greater than can be achieved using single I/O paths.

## BSAM 64K tracks for Language Environment record I/O

In z/OS V1R7, support was introduced in DFSMSdfp for large format sequential data sets (DSNTYPE=LARGE). In z/OS V1R8, Language Environment added support for these data sets using noseek (QSAM). Support for seek (BSAM) was limited to data sets no larger than 64 K tracks on any volume when opened for read. In z/OS V1R12, seek (BSAM) support is extended to data sets up to the maximum size when using record I/O. Binary and text I/O with seek continue to be supported for data sets up to 64 K tracks in size on any volume when opened for read.

## Extended addressable catalogs

In z/OS V1R12, DFSMS there is new support for catalogs with extended addressability (EA). This is designed to make it possible to define and use integrated catalog facility (ICF) basic catalog structures (BCS) with extended addressability (EA), allowing catalogs larger than 4 GB.

## **DFSMS EAV supports additional data set types**

In z/OS V1R12, DFSMS supports the following additional data set types in the extended addressing space (EAS):

- ▶ Sequential (both basic and large) data sets
- ▶ Partitioned (PDS/PDSE) data sets
- ▶ Catalogs
- ▶ BDAM data sets
- ▶ Generation data groups (GDGs)
- ▶ VSAM volume data sets (VVDS)

Overall, EAV helps to relieve storage constraints as well as simplify storage management by providing the ability to manage fewer large volumes as opposed to many small volumes.

## **DFSMS support for the ATTREXX interface**

The System Data Mover (SDM) component provides a REXX interface for many of the functions of the SDM programming interface (ANTRQST). This new function is designed to provide a REXX programming interface to FlashCopy®, Global Mirror, z/OS Global Mirror (XRC), and Peer-to-Peer Remote Copy (Metro Mirror and Global Copy) copy services.

## **ISMF function to copy underlying volumes**

The Interactive Storage Management Facility (ISMF), which you use to manage your SMS configuration, allows you to copy storage group definitions from one control data set (CDS) to another. In z/OS V1R12, ISMF allows you to specify that the volume list for pool-type storage groups be copied at the same time. This allows you to copy entire storage groups from one configuration to another without having to add their volumes to the destination CDS afterward.

## **DFSMS VSAM serviceability for dynamic trace**

In z/OS 1R12, VSAM serviceability is enhanced by improving the usability of VSAM Record Management Trace. A new interface is introduced for the user to enable VSAM Record Management Trace dynamically without using the DD card of the JCL. This new interface extends the usability of VSAM Record Management Trace to support data sets that are allocated dynamically and allow the user to enable VSAM Record Management Trace without taking the application or data set offline.

## **DFSMS catalog**

In z/OS V1R12, IDCAMS is enhanced to avoid DFSMSHsm recalls for any generation data sets that are migrated when deleting entire generation data groups (GDGs). Instead, IDCAMS will call DFSMSHsm to delete such data sets without recalling them. This is expected to reduce processing time, particularly when one or more generation data sets have been migrated to tape.

In z/OS V1R12, DFSMSDfp allows a zFS data set to be recataloged with an indirect volume serial or system symbol. This is designed to allow the zFS file systems used for z/OS system software files (such as version root file systems) to be cataloged using an indirect volume serial or a system symbol the same way as non-VSAM data sets to make cloning and migration easier.

On prior releases, partial release operations for VSAM data sets supported releasing space only on the last volume containing data for each data set. In z/OS V1R12, partial release is extended to support releasing unused volumes in addition to releasing space on the last volume of a multivolume VSAM data set that contains data.



In z/OS V1R12, the IDCAMS DEFINE RECATALOG command is enhanced for multivolume and striped data sets. This new function is designed to automatically create catalog entries with correctly ordered volume lists while eliminating any duplicate volumes that might have been specified. This will make it easier to recatalog multivolume and striped VSAM data sets.

### **DFSMS separates the VOLSELMSG and TRACE parameters**

Currently SMS TRACE and VOLSELMSG facilities share three parameters: TYPE, ASID and JOBNAME. The specification of these shared parameters applies to both TRACE and VOLSELMSG facilities simultaneously. In order to provide more flexibility and usefulness for these two facilities, in z/OS 1R12 SMS enhances these shared parameters by allowing different settings to be specified for either facility separately. In addition, when both TRACE and VOLSELMSG are active and there are a large number of candidate volumes, numerous module and message trace entries may be created for the detailed volume selection analysis messages, IGD17389I. With this support, SMS limits the number of trace entries that are created for these IGD17389I messages for each allocation.

### **DFSMS AMS PDSE empty command**

In z/OS V1R12, IDCAMS is enhanced to allow you to delete all members of a partitioned data set in a single operation by specifying a wildcard character (\*) as the member name for a data set when using the DELETE command. This new support is designed to allow you to remove all members of a PDS or PDSE data set in a single command.

### **DFSMS RLS**

Over time, VSAM key-sequenced data sets (KSDS) for which records are added and deleted have often become fragmented, causing many control area (CA) splits, the use of additional index levels, and consumption of additional extents on DASD volumes. Performance and DASD space utilization can usually be improved copying these data sets, deleting and reallocating them, and reloading them. This requires scheduled outages for applications using these data sets.

In z/OS V1R12, DFSMSdfp allows you to specify that VSAM dynamically reclaim unused control areas for KSDSs, including those used for record-level sharing (RLS), and adjust the number of index levels as needed. This new function is intended to help you preserve performance and minimize space utilization for KSDSs, improve application availability, and allow you to discontinue the use of jobs whose sole purpose is to reorganize one or more KSDSs.

### **DFSMS PDSE verification tool**

In z/OS V1R12, new PDSE functions are introduced. A new utility verifies that the structure of a PDSE is valid, and a programming service performs similar checking to help programs verify the state of a PDSE before and after critical operations. These new functions are intended to help you detect errors in PDSE structures that might otherwise go undetected.

### **DFSMS PDSE**

When a corrupt PDSE is detected in the link list during IPL, the system enters a wait state. In z/OS V1R12, the system issues a message identifying the corrupt PDSE prior to entering the wait state.

### **DFSMS storage group management and volume selection**

In z/OS V1R12, DFSMS enhancements are implemented for storage group management and volume selection performance. As volume sizes increase, one percent of a volume represents an increasingly large amount of storage. For example, on a 223 GB volume, 1% is over 2 GB of storage. In z/OS V1R12, the limit on the high threshold you can specify for space utilization

for pool storage groups is increased from 99% to 100%. In most cases, IBM recommends a high threshold value less than 100% for storage groups. This allows data sets to expand without an increased risk of encountering out-of-space abends. The 100% specification is intended to be used to make more storage capacity available for storage groups that hold static data. Also, SMS processing of volume lists is changed in a way intended to improve allocation performance for large volume lists.

### **DFSMS ISMF DCOLLECT currency**

The Integrated Storage Management Facility (ISMF) includes a data collection application, DCOLLECT, that provides storage-related measurement data that can be used as input to the DFSMSrmm Report Generator to create customized reports or to feed other applications such as billing applications. In z/OS V1R12, DCOLLECT data class (DC) records are updated to include information about all data class attributes. Also, data set (D) records include job names, and storage group (SG) records include information about OAM protect retention and protect deletion settings.

### **DFSMS BAM XTIO support**

The LE/C runtime library data set name retrieval, used for the fldata() function when opening a member of a partitioned data set concatenation, is updated to support the XTIO. Allowing XTIO (task I/O table) to be above the line, all the open data sets on your tasks can now live above 16 MG. This is called the extended TIO, and this requires a 4-byte pointer versus the current 3-byte pointer. LE needs a new interface that will work for the previous systems and for release 12.

## **1.9.1 DFSORT**

DFSORT is the IBM high-performance sort, merge, copy, analysis, and reporting product for z/OS. With DFSORT, you can sort, merge, and copy data sets. You can use DFSORT to do simple tasks such as alphabetizing a list of names, or you can use it to aid complex tasks such as taking inventory or running a billing system. DFSORT gives you versatile data handling capabilities at the record, field and bit level.

### **DFSORT ICE083A error message**

In z/OS V1R12, DFSORT now provides additional information when DFSORT cannot dynamically allocate sufficient work space on a specified volume. New messages help to resolve these problems quickly.

### **DFSORT support for uncaptured UCBs (XTIO)**

In z/OS V1R12, DFSORT now supports extended task I/O tables (XTIOs), uncaptured unit control blocks (UCBs), and data set access blocks (DSABs) above the line. This is intended to help you take advantage of those functions to allow more concurrently-open data sets and provide for virtual storage constraint relief.

### **DFSORT calculation of work space for VSAM data sets**

When VSAM data sets are not closed properly, the statistics, including information used to determine file sizes, might be incorrect. In z/OS 1R12, DFSORT attempts to estimate the required work space for VSAM data sets that have not been closed correctly, and allocate work data sets of the needed size for certain kinds of input data sets.

### **DFSORT diagnostic messages in error situations**

In z/OS V1R12, DFSORT issues diagnostic messages automatically, without the need to add a SORTDIAG DD statement or specify the DIAGSIM=YES installation option. This is intended

to make it easier to provide the information you need to resolve many DFSORT problems without having to rerun a sort to collect diagnostic information, and to improve first failure data capture.

### **DFSORT EAV support for SORTWKnn data sets**

In z/OS V1R12, there is new support for non-extended format large sequential data sets to reside in the extended addressing space (EAS) on extended address volumes (EAVs). DFSORT supports these data sets up to the maximum size. This is designed to allow sort work data sets up to approximately 16,000,000 tracks in size.

### **DFSORT dynamic allocation improvements**

DFSORT's dynamic allocation of work data sets has been enhanced to improve reliability in cases where the disk work space required to complete a sort is higher than originally expected. This can happen when incorrect file size information is provided to DFSORT or when contention for central storage limits the amount of Hiperspace™ storage DFSORT is able to use for intermediate storage. Users will now be able to request DFSORT to allocate additional work data sets that are allocated with zero space and only used if needed. DFSORT has also improved its algorithms to dynamically adjust the size of additional extents that are obtained on the work data sets when the work space requirements unexpectedly increase.

## **1.9.2 Constraint relief for XTIO**

Support for XTIO (eXtended Task Input Output Table), uncaptured UCBs, and DSAB above the line will allow BSAM, BPAM, QSAM and EXCP to support the XTIO, nocapture and DSAB-above-the-line options of dynamic allocation for DASD, tape and dummy data sets and when PATH= is coded. The main purpose is for VSCR and the main VSCR would come from DASD and tape support. DB2 requested this for VSCR reasons and JES2 requested the UCB nocapture option. DFSORT will support it to help with VSCR for DB2. With this new support in BAM, DFSORT will allow its caller (such as DB2) to use these VSCR options and DFSORT will automatically use them when it allocates work files.

### **Support for XTIO, uncaptured UCBs, and DSAB**

Some workloads require an increasing number of open data sets. In z/OS V1R12, the BSAM, QSAM, and BPAM (basic and queued sequential, and basic partitioned access methods) and execute channel program (EXCP) processing is designed to support the use of an extended task I/O table (XTIO) with uncaptured UCBs, and support data set association blocks (DSABs) above the 16 MB line. This is expected to allow more data sets to be allocated by an address space and to provide virtual storage constraint relief for DASD and tape data sets.

## **1.9.3 NFS Server**

The NFS Server is a component of z/OS that allows remote access to z/OS host processor data from workstations, personal computers, or any other system on a TCP/IP network that is using client software for the Network File System protocol.

### **Exploitation of C APIs for alternate group database functions**

In z/OS V1R12, NFS made changes to utilize new C interfaces rather than existing USS callable services, which potentially will have performance benefits.

## **NFS Server SMF records**

In z/OS V1R12, NFS allows you to specify that SMF records be created when a file system object is created, renamed, or removed. New SMF42 subtype 26 records can be specified with a new site attribute or enabled by operator commands for both z/OS UNIX file objects including directories and files, and data sets and members. In conjunction with z/OS UNIX Type 80 and RACF Type 92 records, this support is intended to improve auditability for NFS file operations.

## **NFS Server cache monitoring and reporting**

In z/OS V1R12, the NFS Server performs additional cache monitoring and reporting. The NFS Server issues warning messages when the value specified for the `bufhigh` site setting is being approached, and issues additional messages when a buffer shortage or impending buffer shortage has been relieved. This is intended to allow you to use automation to respond to impending buffer shortage events.

## **NFS Server displays accounting statistics**

NFS Client displays the statistics about the NFS Client activities to the z/OS NFS Server. The displayed statistics include the number of NFS V2, NFS V3 and NFS V4 operations being used by the NFS Clients in their communication with the z/OS NFS.

## **NFS password phrase support for the MVSlogin client utility**

In z/OS V1R12, the NFS Server supports password phrases up to 100 characters in length for `mvslogin`, in addition to existing support for passwords up to 8 characters long. This support will require password phrase support from an external security manager such as RACF. This is intended to allow you to migrate to password phrases, which offer a much larger name space than passwords.

# **1.10 DFSMSHsm**

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 DFSMSdftp family to provide efficient and effective storage management.

## **DFSMSHsm space management performance**

In z/OS 1R12, DFSMSHsm space management performance improvements support a new option to allow you to specify that primary space management, interval migration, and command volume migration be done in parallel.

## **DFSMSHsm multi-task recover from dump**

In z/OS V1R12, DFSMSHsm supports parallel processing for recovery from dump tape volumes when the dumps reside on multiple tape volumes and multiple tape drives are available. This new function is intended to allow you to specify that up to 64 concurrent tasks be used to help speed recovery processing. Also, this is designed to allow you to restore fast recovery copy pools from tape using DFSMSHsm.

## **Fast reverse restore**

In z/OS V1R12, DFSMSHsm is exploiting the fast reverse restore feature of the IBM System Storage DS8000® Series. This function is designed to allow recovery to be performed from

an active, original FlashCopy target volume to its original source volume without having to wait for the background copy to finish when the volume pair is in a full-volume FlashCopy relationship. DFSMSShsm FlashCopy backup and recovery operations will be designed to create full-volume FlashCopy relationships when the devices support it, and Fast Reverse Restore function to support recovery of volumes associated with copy pool backups including Space Efficient and Incremental FlashCopy targets. A new SETSYS parameter is planned to allow you to specify whether extent or full-volume FlashCopy relationships are to be established between volume pairs when DFSMSShsm invokes DFSMSdss to perform fast replication backup and recovery.

### **DFSMSShsm dump stacking**

In z/OS V1R12, the DFSMSShsm DUMP function used to copy source disk volumes to target tape volume has been enhanced. The dump stacking function is designed to allow up to 255 source volumes to be dumped to a single tape volume, up from the prior limit of 99. This is intended to help you take better advantage of large capacity tape cartridges.

## **1.11 DFSMSrmm**

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, a room with tapes, shelves, and drives, you might have several automated, virtual, and manual tape libraries. You probably also have both on-site libraries and off-site 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, 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 EAV and IPv6 support**

In z/OS V1R12, new support makes all data sets used by DFSMSrmm eligible for allocation in the extended addressing space of an EAV. This includes the DFSMSrmm journal and dynamically allocated temporary files. DFSMSrmm support for IPv6 is also implemented.

### **DFSMSrmm simplified monitoring and management**

DFSMSrmm provides for z/OS V1R12 that when a retention limit is reached, it will be added to the ACTIVITY file. New reports created from the ACTIVITY and extract files can help you see why retention limits were triggered. Also, OPENRULE ignore processing is available for duplicate tape volumes, and DFSMS recovery from multivolume tape label anomalies will be based on policy information you defined to DFSMSrmm.

This is planned so you can exploit OPENRULE ignore processing for duplicate tape volumes. DFSMS is going to automatically handle multi-volume label anomalies as described by messages IEC709I, IEC710I, IEC711I and IEC712I. This enables automated recovery based on information defined to DFSMSrmm.

## **DFSMSrmm RAS enhancements**

In z/OS 1R12, DFSMSrmm creates additional trace records for processing outside the subsystem address space to enable improved diagnostics.

## **DFSMSrmm hardware support**

In z/OS 1R12, DFSMSrmm supports copies of logical tape volumes that are exported from virtual tape subsystems (VTS) such as the TS7700 virtualization engine, including copies that might be sent to a secure offsite storage facility for disaster recovery. This new function is planned to manage exported copies and their locations in addition to managing original virtual tape volumes residing in a VTS.

## **DFSMSrmm competitive enhancements**

In z/OS 1R12, DFSMSrmm allows you to define policy rules that govern whether residual data on tape volumes should be erased during close processing. This new function to erase unused data is designed to help you ensure that tapes sent outside your installation contain only the data you wish to send.

## **1.12 DFSMSdss**

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

### **DFSMSdss BSAM I/O support**

In z/OS V1R12, DFSMSdss now supports using larger blocks when possible for DUMP, COPYDUMP, and RESTORE operations, and supports extended-format sequential dump data sets on DASD for DUMP, RESTORE, and COPYDUMP. The use of larger block sizes, up to 256 K when dumping to tape, is intended to improve performance for these operations. With the use of extended format dump data sets on DASD it is intended to support striping and compression.

### **Support for removal of IMBED and REPLICATE**

The creation of new VSAM data sets with IMBED, REPLICATE, or KEYRANGE attributes has been unsupported since z/OS V1R3. These attributes, originally introduced to improve performance on older DASD, typically act only to occupy additional space and slow performance on modern cached DASD. In z/OS V1R12, DFSMSdss removes the IMBED and REPLICATE attributes during logical restore and DFSMSHsm recall processing. An informational message is introduced to confirm that newly-restored data sets no longer retain these attributes or encountered data sets with these attributes and could not remove them. During DFSMSdss data set dump processing an information message is also introduced to indicate that a data set with the IMBED, REPLICATE, and KEYRANGE attribute was encountered and requires some action to be taken by the user. The messages may also be encountered when DFSMSHsm uses DFSMSdss to process a data set with these attributes during migrate and backup.

## **1.13 DFSMSOam**

The Object Access Method (OAM) is a component of DFSMSdfp, the base of the storage management system (SMS) of DFSMS. OAM uses the concepts of system-managed storage, introduced by SMS, to manage, maintain, and verify tape volumes and tape libraries within a tape storage environment.

## **DFSMSoam RAS enhancements**

In z/OS 1R12, OAM provides API support for the object storage and retrieval function (OSR) to run in a CICS threadsafe environment. This is intended to allow exploiters to take advantage of the improved multitasking and throughput capabilities provided by threadsafe programming. Additionally, the volume recovery utility improves performance when recovering object data stored on optical and tape media.

## **1.14 SDSF**

SDSF provides a powerful and secure way to monitor and manage your z/OS sysplex. Beginning with z/OS V1R10, there is support for JES3 systems.

### **SDSF JES managed device support**

In z/OS V1R12, SDSF is designed to support printer displays for JES3. Also, initiator displays for JES2 no longer require MQSeries® when all systems in the JES2 MAS are at z/OS V1R12 JES2, or for JES2 device displays when all systems in the MAS are at or above z/OS V1R11 JES2.

### **Health check history display in SDSF**

In z/OS V1.12, SDSF is enhanced to augment the CK panel by displaying recorded checks on a new health check history panel. The default is to display up to ten prior iterations of each check from the log stream, and this support allows you to browse and print check output from the history display as you can on the primary CK panel.

### **SDSF data to Java**

In z/OS V1R12, SDSF makes Java classes available to provide a new means of accessing SDSF. Classes are provided for each of the SDSF panels that can be used by applications to request SDSF functions. This new support is designed to allow Java-based applications to easily access SDSF.

### **SDSF REXX**

In z/OS V1R12, SDSF introduces a new ISFLOG command for SDSF REXX. It is designed to read the system log and return its records in stem variables, and to support options to limit the number of records returned and specify start and end times. This new function will simplify access to the system log for SDSF REXX.

### **SDSF JES3-managed device support**

In z/OS V1R12, SDSF supports printer displays for JES3. Also, initiator displays for JES2 no longer require MQSeries when all systems in the JES2 MAS are at z/OS V1R12 JES2, or for JES2 device displays when all systems in the MAS are at or above z/OS V1R11 JES2.

## **1.15 z/OS Health Checker**

The Health Checker provides a mechanism to check the health of an active system and can help you check for and proactively resolve potential problems before application availability is impacted or system or sysplex-wide outages occur. The Health Check framework allows for unique autonomic functions for the z/OS platform by alerting you about potential problems and allowing their correction before they impact your business.

## **Issue messages without message table**

In z/OS V1R12, the Health Checker framework is enhanced to allow health checks to be registered without a message table and for them to issue messages directly without using one. This makes it easier to write health checks quickly.

## **METAL C headers and sample check**

The Health Checker framework provides headers to enable an installation to write health checks using METAL C, in addition to existing support for High-Level Assembler and REXX. Providing high-level language support can make it easier to write complex health checks. Additionally, sample health checks written in METAL C are available.

## **XES/XCF Health Checks**

z/OS V1R12 has new health checks for the Parallel Sysplex components XCF and XES. They are designed to warn you about the following events:

- ▶ When a structure's specified size is more than double its initial size
- ▶ When a configuration data set's (CDS's) maximum system limit is lower than the primary sysplex CDS's limit when shared CPs are used for Coupling Facility partitions
- ▶ When the message-based communications protocol can be used for recovery processing but the policy-based protocol is specified
- ▶ When your Sysplex Failure Management (SFM) policy does not specify the action to be taken to relieve hangs caused by a connector failure

These checks can help you correct and prevent common sysplex management problems.

## **SMB migration Health Check**

In z/OS V1R12, the SMB server adds two health checks. The first is designed to detect SMB running in a zFS sysplex-aware environment and alert you that SMB cannot export zFS read/write file systems in this environment, and the second determines whether SMB is configured to support the RPC protocol (DCE/DFS) and display a message to remind you that IBM plans to withdraw support for this protocol in a future release.

## **HFS to zFS Migration Health Check**

IBM recommends that you use zFS file systems for z/OS UNIX System Services. In z/OS V1R12, a migration health check identifies HFS file systems you should consider migrating to zFS file systems. This is intended to help you easily obtain and track the list of remaining file systems to be converted.

## **DFSMS SMS health checker enhancements**

In z/OS V1R12, DFSMS adds new health check functions for the communications and active configuration data sets (COMMDS and ACDS). One new check is designed to alert you that OMMDS and ACDS are on the same volume. The other is intended to identify COMMDS and ACDS data sets that were defined without the REUSE attribute, which is recommended. These new checks can help you manage your SMS environment.

## **Facility class profile for BPX.SUPERUSER**

In z/OS V1R12 the Health Checker started task supports running with an assigned user ID that has access to the BPX.SUPERUSER profile in the FACILITY CLASS. This will make it unnecessary to run the Health Checker address space with a user ID having UID(0).



## IOS health checks

In z/OS V1R12, new health checks are designed for the I/O Supervisor (IOS). IBM recommends using the relatively new MIDAWs and Captured UCB Protection functions introduced in recent releases, and locating eligible I/O-related control blocks above the 16 MB line. These health checks are designed to notify you when these functions are not being used, to help you manage system performance and the use of virtual storage.

## z/OS OMPROUTE checks

z/OS V1R12 Communications Server enhances the z/OS Health Checker for z/OS by adding two new checks, as follows:

- ▶ A check for IPv4 routing and a check for IPv6 routing. The checks determine whether the total number of indirect routes in the TCP/IP stack routing table exceeds a maximum threshold (default 2000). When this threshold is exceeded, OMPROUTE and the TCP/IP stack can potentially experience high processor consumption from routing changes.
- ▶ Two new maximum threshold parameters are available to override the default values for the total number of IPv4 and IPv6 indirect routes in a TCP/IP stack routing table before warning messages are issued.

# 1.16 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 VTAM® and include Subarea, Advanced Peer-to-Peer Networking, and High Performance Routing protocols.

## Resolver reaction to unresponsive DNS name servers

z/OS Communications Server V1R12 provides notification to the operator console when a domain name system (DNS) name server does not respond to a certain percentage of resolver queries sent to the name server during a sliding 5-minute interval. In addition to the notification, statistics regarding the number of queries attempted and the number of queries that received no response are displayed for each currently unresponsive name server at 5-minute intervals. This can alert you to a possible problem with your DNS name server configuration that may be adversely affecting applications on your z/OS system.

The default value for the TCPIP.DATA RESOLVERTIMEOUT configuration statement, which controls the timeout value for UDP requests sent to a name server, is modified to be 5 seconds instead of 30 seconds.

## Resolver support for IPv6 connections to DNS name servers

z/OS Communications Server V1R12 allows the system resolver to send requests to Domain Name System (DNS) name servers using IPv6 communication. This function allows you to use the existing NSINTERADDR and NAMESERVER resolver configuration statements in the TCPIP.DATA data set to define the IPv6 address of the name server.

## Policy table and Socket API support for IPv6

z/OS V1R12 Communications Server supports RFC3484 by providing a configurable policy table for default address selection for IPv6. The source address selection algorithm and

destination address selection algorithm are enhanced to support additional address selection rules in conjunction with the configured or default policy table. For example, you might choose to prefer IPv4 communication over IPv6 by providing a custom policy table for default address selection.

z/OS V1R12 Communications Server also supports RFC5014 by providing IPv6 socket API for source address selection. Applications can indicate they prefer temporary IPv6 addresses over public IPv6 addresses or public IPv6 addresses over temporary IPv6 addresses.

Additionally, z/OS V1R12 Communications Server has enhanced the SRCIP configuration to allow an administrator to indicate that the TCP/IP stack should prefer public IPv6 addresses over temporary IPv6 addresses. This will allow you to override the preferences specified by an application using the IPv6 socket API for source address selection.

## **Enhancements to IPv6 router advertisement**

In z/OS V1R12 Communications Server, the improvements include:

- ▶ The ability to learn indirect prefix routes from IPv6 router advertisement messages
- ▶ The ability to associate preference values with all routes that are learned from IPv6 router advertisement messages

Use of this function is expected to reduce the number of IPv6 static routes that must be defined and the ability to route around network failures when not using OMPROUTE to install routes learned via a dynamic routing protocol, such as OSPF.

## **Trusted TCP connections**

z/OS V1R12 Communications Server introduces trusted TCP connections, to enable sockets programs to retrieve sysplex-specific connection routing information and partner security credentials for connected sockets. Partner security credentials can be retrieved if both endpoints of a TCP connection reside in the same z/OS image, z/OS sysplex, or z/OS subplex, and the endpoints are within the same security domain. In such a topology, partner programs can use trusted connections to authenticate each other as an alternative to using an SSL/TLS connection with digital certificates for client and server authentication.

## **Digital certificate access server**

z/OS V1R12 Communications Server enhances the digital certificate access server (DCAS) to allow modification of the debug level without restarting the application.

## **Internet Key Exchange version 2 support**

Internet Key Exchange version 2 (IKEv2) is the second version of the Internet Key Exchange (IKE) protocol, which is used by peer nodes to perform mutual authentication and to establish and maintain security associations (SAs). In z/OS V1R12 the Communications Server IKE daemon (IKED) is enhanced to support IKEv2, in addition to its existing IKEv1 support. The z/OS Communications Server support for IKEv2 includes:

- ▶ IPv4 and IPv6 support
- ▶ A new identity type called KeyId
- ▶ Authentication using pre-shared keys or digital certificates; certificates may use RSA or elliptic curve keys
- ▶ Re-keying and re-authentication of IKE SAs and child SAs
- ▶ Hash and URL certificates and certificate bundles

## IPSec support

z/OS V1R12 Communications Server introduces the following enhancements to network security services (NSS) processing, IPSec certificate trust chains, and certificate revocation lists:

- ▶ All the certificate authorities in the trust chain are considered when creating or verifying a signature for certificate authorities that are in the key ring.
- ▶ Certificate revocation information is used when available when verifying a certificate.

The z/OS Internet Key Exchange daemon (IKED) is planned to be enhanced to use these new NSS daemon (NSSD) functions when a stack is configured as a network security client.

## IPSec support for cryptographic currency

z/OS V1R12 Communications Server is introducing the following enhancements to IPSec and IKE support for cryptographic currency:

- ▶ Support for the Advanced Encryption Standard (AES) algorithm in Cipher Block Chaining (CBC) mode for IP Security. In addition to the previously-existing support of AES with a 128-bit key length, z/OS V1.12 Communications Server is designed to support AES with a 256-bit key length in CBC mode. You can use longer key lengths for more sensitive data.
- ▶ Support for the Advanced Encryption Standard (AES) algorithm in Galois Counter Mode (GCM) and in Galois Message Authentication Code (GMAC) mode for IP Security. AES in GCM mode is intended to provide both confidentiality and data origin authentication. AES-GCM is a very efficient algorithm for high-speed packet networks. AES in GMAC mode is intended to provide data origin authentication but does not provide confidentiality. AES-GMAC, like AES-GCM, is also a very efficient algorithm for high-speed packet networks. z/OS V1R12 Communications Server supports both 128-bit and 256-bit key lengths for these algorithms.
- ▶ Support for the use of hashed message authentication mode (HMAC) in conjunction with the SHA2-256, SHA2-384, and SHA2-512 algorithms. These algorithms are intended to be used as the basis for data origin authentication and integrity verification. The new algorithms, HMAC-SHA2-256-128, HMAC-SHA2-384-192, and HMAC-SHA2-512-256, are designed to help ensure that data is authentic and has not been modified in transit. Versions of these algorithms that are not truncated are available as Pseudo-Random Functions (PRFs). These algorithms are named PRF-HMAC-SHA2-256, PRF-HMAC-SHA2-384, and PRF-HMAC-SHA2-512.
- ▶ Support for an authentication algorithm, AES128-XCBC-96, that can help ensure data is authentic and not modified in transit.
- ▶ Support for elliptic curve digital signature algorithm (ECDSA) authentication.

z/OS V1R12 Communications Server is introducing support for the standard FIPS 140 security requirements for cryptographic modules for IP security. This standard is expected to be useful to organizations that use cryptographic-based security systems for protection of sensitive or valuable data. Protection of a cryptographic module within a security system is thought necessary to maintain the confidentiality and integrity of the information protected by the module. FIPS 140 dictates security requirements that should be satisfied by a cryptographic module to obtain higher degrees of assurance about the integrity of the module. FIPS 140 provides four increasing, qualitative levels of security intended to cover a wide range of potential applications and environments. z/OS V1R12 Communications Server support is designed to meet the requirements for security level 1.

## Enforce RFC 4301 compliance for IPSec filter rules

In z/OS V1R12, RFC 4301 compliance for IPSec filter rules is mandatory. RFC 4301 “Security Architecture for the Internet Protocol” specifies the base architecture for IPSec-compliant

systems, including restrictions on the routing of fragmented packets. Compliance enforcement may require minor changes to IP filters for IP traffic that is routed through z/OS. The Configuration Assistant is changed to identify any noncompliant IP filters, and policy agent will not install an IPSec policy that contains any non-compliant IP filters.

### **Common storage reduction for the TN3270E server**

In z/OS V1R12 Communications Server, the TN3270E Telnet server provides access method control block (ACB) sharing for logical units (LUs) as a way to reduce ECSA usage. Prior to z/OS V1R12 Communications Server, every Telnet LU name opened its own ACB to VTAM. You can code a new SHAREACB statement to allow multiple Telnet LUs to share a single ACB, which reduces the overall amount of ECSA (and Telnet private) storage allocated to support Telnet sessions. Telnet LU ACB sharing can benefit your installation if you currently run a large number of connections (8 K or greater) to a given Telnet server.

### **Performance improvements for fast local sockets**

z/OS V1R12 Communications Server enhances the performance of fast local sockets for TCP connections. This function is automatically enabled.

### **Performance improvements for streaming bulk data**

In z/OS V1R12 Communications Server, processing for OSA-Express in QDIO mode supports inbound workload queueing. Inbound workload queueing uses multiple input queues for each QDIO data device (subchannel device) to improve TCP/IP stack scalability and general network optimization. When inbound workload queueing is enabled for a QDIO interface, inbound streaming bulk data is processed on an ancillary input queue (AIQ). This function is expected to improve throughput while reducing processor consumption for inbound streaming bulk data.

### **Multipath control for Enterprise Extender**

z/OS Communications Server allows the coding of MULTIPATH in the TCP/IP profile that enables multipath support for IP packets. You might want this behavior for TCP connections but not for Enterprise Extender (EE) connections. In z/OS Communications Server V1R12, the multipath function is disabled by default for EE connections regardless of the value specified in the TCP/IP profile. You can use the VTAM start option MULTIPATH to control the multipath function for EE.

### **Performance improvements for Sysplex Distributor connection routing**

In z/OS V1R12 Communications Server, processing for OSA-Express in QDIO mode now supports inbound workload queueing. Inbound workload queueing uses multiple input queues for each QDIO data device (subchannel device) to improve TCP/IP stack scalability and general network optimization. When inbound workload queueing is enabled for a QDIO interface, inbound Sysplex Distributor traffic is designed to be processed on an ancillary input queue (AIQ). This function is expected to improve performance for inbound Sysplex Distributor traffic that is routed to a target stack.

### **Improvements for AT-TLS performance**

In z/OS V1R12 Communications Server, AT-TLS processing is now designed to provide reduced processor usage for encryption and decryption of application data while improving throughput for some types of workloads. This function is automatically enabled.

### **Drop all connections for a server**

z/OS V1R12 Communications Server has extended the **VARY TCPIP, ,DROP** command to allow the dropping of all established TCP connections for servers that match the specified filter

parameters. When issued, each server that is found to match the specified filter parameters will have all its established TCP connections dropped. You can filter by port, jobname, or server ASID. This function is expected to make it easier to move workloads from one application instance to another application instance.

### **Packet trace filtering for encapsulated packets**

In z/OS V1R12 Communications Server, packet trace filtering is enhanced to support:

- ▶ Including the next hop IP address on the trace output. This can be obtained from the fully formatted packet trace using IPCS. The next hop IP address is also available to applications that consume the real-time packet trace through the real-time TCP/IP networking monitoring API.
- ▶ Making packet trace filtering available to encapsulated packets that are used in VIPAROUTE traffic

### **Control joining the sysplex XCF group**

In z/OS V1R12 Communications Server, support now allows you to specify that a TCP/IP stack be kept isolated from the sysplex; a new configuration parameter prevents a stack from automatically joining the sysplex group at startup. You can have the stack join the sysplex group at a later time by issuing the **VARY TCP,IP, ,SYSPLEX, JOININGROUP** command.

### **Enhancements to sysplex autonomics**

In z/OS V1R12 Communications Server, sysplex problem detection and recovery are enhanced to detect when the TCP/IP stack has abended five times in less than a minute. Existing sysplex recovery logic is applied when this problem is detected.

### **Management data for CSSMTP**

z/OS V1R12 Communications Server provides enhancements to improve the management of the CSSMTP application by adding the following new SMF 119 record subtypes:

- ▶ 048 - CSSMTP Configuration data records
- ▶ 049 - CSSMTP Target server connection records
- ▶ 050 - CSSMTP Mail records
- ▶ 051 - CSSMTP Spool records
- ▶ 052 - CSSMTP Statistics records

Applications that want to process the new SMF 119 subtypes can obtain them from a traditional MVS SMF exit routine or in real-time from the z/OS Communications Server Network Management Interface (NMI) for SMF, SYSTCPSM. CSSMTP issues the SIOCSAPPLDATA ioctl to add application data (apldata) to the TCP connections used to connect to target mail servers. You can see the application data (apldata) displayed in the Netstat All/-A, AllConn/-a, and Conn/-c reports.

### **Enhancements to EZBNMIFR - network interface and device statistics**

z/OS V1R12 Communications Server uses new TCP/IP callable NMI requests to provide TCP/IP stack network interface information and network interface and global statistics. Network management applications can use the request output to monitor interface status and TCP/IP stack activity. z/OS V1R12 Communications Server provides the following new requests:

<b>GetGlobalStats</b>	Provides TCP/IP stack global counters for IP, ICMP, TCP, and UDP processing.
<b>GetIfs</b>	Provides TCP/IP network interface attributes and IP addresses.
<b>GetIfStats</b>	Provides TCP/IP network interface counters.

**GetIfStatsExtended** Provides data link control (DLC) network interface counters.

### **SMF event records for sysplex events**

z/OS V1R12 Communications Server introduces sysplex event notification through new SMF 119 event records (subtypes 32 - 37) that describe the following events:

- ▶ DVIPA removed (subtype 33)
- ▶ DVIPA status change (subtype 32)
- ▶ DVIPA target added (subtype 34)
- ▶ DVIPA target removed (subtype 35)
- ▶ DVIPA target server ended (subtype 37)
- ▶ DVIPA target server started (subtype 36)

The new SMF 119 event records are written to the MVS SMF data sets, and can also be obtained from the real-time TCP/IP network monitoring NMI (SYSTCPM).

### **Data trace records for socket data flow start and stop**

z/OS V1R12 Communications Server enhances TCP/IP data tracing (DATTRACE) to provide two new trace records:

- ▶ A Start record with State field API Data Flow Starts that indicates the first data sent or received by the application for the associated TCP or UDP socket.
- ▶ An End record with State field API Data Flow Ends that indicates the socket has been closed.

### **Performance improvement to EZBNMIFR GetConnectionDetail request**

z/OS V1R12 Communications Server reduces processor utilization for TCP/IP Callable Network Management Interface (NMI), EZBNMIFR, GetConnectionDetail. All the filters that are specified for the request must contain the complete identification (4-tuple) of established TCP connections. The 4-tuple of a TCP connection consists of the local IP address, local port, remote IP address, and remote port for the connection.

### **Verify Netstat message catalog synchronization**

In z/OS V1R12 Communications Server, the Netstat function provides support for verifying that message catalogs being used are at the correct level when they are opened. This function is intended to prevent Netstat from abending or not functioning correctly when the message catalog is out of synch with the Netstat program.

### **Enterprise Extender connection health verification**

z/OS V1R12 Communications Server provides the option to check the health of an Enterprise Extender (EE) connection during the activation of the connection. The health of active connections can also be verified. z/OS Communications Server sends an LDLC probe to the remote partner using all five ports. If the LDLC probe cannot reach a port for any reason, you will not be able to activate the EE connection and you will receive an error message. You can check the health of all active EE connections periodically. VTAM issues an error message if it cannot reach the remote partner.

### **Communications Server IP address selection**

Address Selection/Get Partner Info request from Communications Server involving header file updates is summarized in Section 14 of RFC 5014. This allows a network administrator to control the source and destination IP address selection algorithms.

## 1.17 Serviceability items

Dump management is keeping pace with the growth in diagnostic data that results from larger systems and larger programs using ever-increasing amounts of memory. These improvements are meant to help you keep dumping time and dump transmission time under control.

### **Standalone dump support for EAV2**

In z/OS V1R12, standalone dump supports extended format dump data sets in the extended addressing space (EAS) on extended address volumes (EAVs).

### **RSM support of SDUMP resource management**

In z/OS V1R12, SVC dump and real storage management (RSM) processing are designed to reduce dump capture time. The time required to capture data on auxiliary storage is expected to be reduced significantly. Additionally, dump exit exploiters can take advantage of improved RSM services to reduce the system impact of collecting large amounts of data. In z/OS V1R12, the GRS dump exit takes advantage of the improved RSM services.

### **Improved standalone dump ASID prioritization**

In z/OS V1R12, standalone dump has better capability to better prioritize data capture for address spaces, and to dump a number of system address spaces first irrespective of their ASID numbers. This is intended to attempt to capture the data most often needed to diagnose system problems more quickly in case there is not enough time to take a complete standalone dump. Also, standalone dump will now allow you to specify additional address spaces to be added to the predefined list using a new ADDSUMM option.

### **Integrate the SUMTRACE REXX exec into the IPCS command**

The SUMTRACE CLIST, which is used to analyze data output by the SYSTRACE, is now integrated into the SYSTRACE command. Summary information is to be displayed right after the SYSTRACE listing. This greatly improves the operation's efficiency, and removes the dependency on the SYSLIST command's output format.

### **Dump health scan**

In z/OS V1R12, IPCS has a new feature to aid with problem determination. IPCS is providing ways to extract and store important information from a dump into a PDS data set. IPCS subcommands or any REXX exec can be used to extract the information. The goal is to expedite problem determination. This would also help when analysis is passed from a client system programmer to IBM support personnel.

### **Console DUMP**

In z/OS V1R12, dump processing is changed to act on a new option for the CHNGDUMP and DUMP commands, and in parmlib member IEADMCxx. The new DEFERTND option allows you to specify that task's non-dispatchability for address spaces being dumped as a result of a DUMP operator command be delayed until after global data capture is complete. This is intended to reduce the amount of time tasks and address spaces being dumped are set nondispatchable to capture volatile data to reduce the impact of command-initiated SVC dumps.

### **System trace formatting enhancement**

A new subparameter, SORTCPU, is added to existing IPCS SYSTRACE command, which will process each processor's trace entries separately in ascending order by processor address. If

options “time” and “number of entries N” are specified for SORTCPU, “N” entries before and after the “time” will be displayed for each processor.

## 1.18 Language Environment

Language Environment gives you a common environment for all Language Environment-conforming high-level language (HLL) products. An HLL is a programming language above the level of assembler language and below that of program generators and query languages. Language Environment establishes a common runtime environment for all participating HLLs. It combines essential run-time services, such as routines for runtime message handling, condition handling, and storage management. All of these services are available through a set of interfaces that are consistent across programming languages. You may either call these interfaces yourself, or use language-specific services that call the interfaces. With Language Environment, you can use one runtime environment for your applications, regardless of the application's programming language or system resource needs.

### **VSAM EAV support for KSDS AIX for C/C++ in Language Environment**

In z/OS VR12, Language Environment provides support for C/C++ to access alternate indexes (AIXs) for extended format VSAM key-sequenced data sets (KSDS) that reside in the extended addressing space (EAS) on an extended address volume (EAV).

### **Non-overrideable parmlib capability**

In z/OS V1R7, Language Environment allowed overrideable runtime options to be defined in a new CEEPRMxx member of parmlib. In z/OS V1R12, support is added for non-overrideable (NONOVR) options. This support can allow you to specify the options for Language Environment without user modifications, eliminating a repetitive migration task.

### **Language Environment realloc() optimization support**

In z/OS V1R12, Language Environment improves performance for string manipulation intensive applications, such as certain applications written in the Perl language.

### **Language Environment locale support**

In z/OS V1R12, Language Environment provides Euro currency support for Slovakia in the C/C++ Run-time Library. Both Euro and pre-Euro support are provided and the default locale for Slovakia is changed to use the Euro symbol.

In z/OSV1R9 the C/C++ Run-time Library iconv() family of functions began to use Unicode Services to perform most character conversions. In z/OS V1R12, the ucmmap source or genxlt source for character conversions is removed from the C/C++ Run-time Library. You can create customized conversion tables using Unicode Services to replace these functions.

### **Support for constructed calendar times beyond 2038**

Calendar times, represented by time\_t, will overflow in January 2038. In z/OS V1R12, the Language Environment C/C++ Run-time Library has support to include new services including time64\_t, which will support constructed calendar times up to and including 23:59:59 UTC on December 31, 9999.



## 1.19 Security Server

The need for strong platform security is a core attribute and one of the basic values of the z/OS platform. The mainframe is an ideal security hub for the enterprise. A security-rich holistic design helps protect the system from malware, viruses, and insider threats. Also, z/OS system and virtualization integrity features mean you can confidently place data, applications, and mixed workloads on System z. Flexible, centralized role-based access controls resource access to the System z.

Encryption solutions integrated into z/OS help secure data from theft or compromise, both on the network or on storage media. Built on the time-tested System z encryption infrastructure, availability, disaster recovery, and access controls, z/OS encryption solutions are more robust, resilient, and scalable than encryption solutions that are pieced together from disparate technologies. Also, a tamper-resistant encryption System z hardware module can protect encryption keys from detection and tampering with the highest certification at FIPS 140-2 Level 4.

In addition to encryption, z/OS can provide end-to-end security solutions for your enterprise. z/OS can provide centralized, highly secure and resilient key management for IBM tape encryption across the enterprise. Create and manage an enterprise-wide user registry with the help of IBM Tivoli® Directory Server for z/OS. Be your own enterprise-wide digital certificate authority with full life cycle management with z/OS PKI Services.

z/OS can also help you address your compliance needs with more confidence. Extensive audit capabilities of z/OS can facilitate regulatory compliance. Add Tivoli Compliance Insight Manager for integrated audit, monitoring and compliance for an enterprise scope. Independent Common Criteria certifications attest that z/OS and System z solutions have been methodically designed, tested, and reviewed for secure operations.

### **RACF XTIOt support**

In z/OS V1R12, RACF now supports extended task I/O tables (XTIOts), uncaptured UCBs, and DSABs above 16 MB for data sets allocated by programs.

### **RACF RAS enhancements**

A discrete general resource profile with generic characters (\*,%,&) in its name, defined in a class enabled for generics (GENCMD or GENERIC) is often called a “ghost” profile. Such profiles are not referenced by RACF for authorization checking. However, when defined, they can confuse and annoy RACF administrators and system programmers. In z/OS V1R12, RACF provides a new NOGENERIC keyword for the RDELETE command to enable you to delete these profiles. Also a GENERIC=N option is implemented for R\_admin DELETE.

### **Generic profile load performance**

Support is added to z/OS V1R12 that improves performance when loading large numbers of generic profiles, and allows you to modify the number of generic anchor tables that are kept per address space.

### **Digital certificate long distinguished name support**

In z/OS V1R12, RACF and PKI Services now support longer distinguished names in digital certificates. For certificates with distinguished names longer than 246 characters that use MD2, MD5, SHA224, SHA256, SHA384, and SHA512 hash algorithms, a new format for the profile string supports distinguished names of up to 1,024 characters in length. This is intended to support your use of certificates with very long distinguished names.

## **New schema syntaxes and matching rules**

In z/OS V1R12, IBM Tivoli Directory Server for z/OS is providing additional schema syntaxes and matching rules to support standard schema definitions available on other LDAP servers, such as open LDAP, SunOne, and non-z/OS IBM Tivoli Directory Server. This enables IBM Tivoli Directory Server for z/OS to support applications that require attributes using these additional syntaxes and matching rules.

### **1.19.1 PKI extensions**

In z/OS V1R12, a number of improvements for PKI Services are implemented. Certain events, such as restoring a prior level of the security database, or removing and reinstalling the Certificate Authority (CA) certificate, can cause the security manager to return serial numbers to be used for new certificates that have been used before. PKI Services is designed to detect this and find the first unused serial number before issuing a new certificate, to avoid issuing two or more certificates with duplicate serial numbers.

Also, a new utility allows you to post existing certificates in LDAP, avoiding the need to post them manually. Additionally, another new utility will be designed to allow you to post updates to Certificate Revocation Lists (CLRs) immediately when you need to, rather than waiting for the interval you have specified.

Last, PKI Services performs certain tasks, such as removing old or expired certificates and requests, and processing certificate expiration notification warning messages, once a day. These housekeeping tasks have historically consumed considerable processing time when you have a large number of certificates. A new PKI Services design is intended to markedly improve the performance and reduce the processing time of these tasks.

#### **PKI extensions**

In z/OS V1R12, PKI Services supports passing the reason that a certificate request was rejected from the administrator to the requester in the rejection e-mail. Also, PKI Services supports custom extensions to X.509 Version 3 certificates; for example, creating a Domain Controller certificate with an extension called Certificate Template Name, with an OID, with BMP data DomainController for use with Microsoft Exchange or Smart Card Login. Last, PKI Services allows you to create certificates with multiple instances of name forms in Subject Alternate Name as described by RFC5280.

#### **Certificate management protocol (CMP) support**

Certificate management protocol (CMP) is an Internet protocol used to manage X.509 digital certificates described by RFC4210, which uses the certificate request message format (CRMF) described by RFC4211. In z/OS V1R12, PKI Services provides support for parts of the CMP standard, allowing CMP clients to communicate with PKI Services to request, revoke, suspend, and resume certificates. This is intended to allow you to use CMP in a centralized certificate generation model.

#### **Date enhancement for PKI Services and RACDCERT**

In z/OS V1R12, the RACF RACDCERT command and PKI Services are enhanced to support the creation of certificates with expiration dates in the far future (up to the year 9997). This is expected to be valuable for certificates that require a validity period of more than 30 years, such as Certificate Authority certificates, which usually have a long life.

#### **Security check sysplex-scope CPF commands**

In z/OS V1R12, the command prefix facility (CPF), which you can use to route commands from one system to another within a sysplex, now supports security checking similar to that

provided for the ROUTE operator command. Defining a new MVS.CPF.ROUTE.CHECK profile in the RACF OPERCMDS class will specify that the system use the MVS.ROUTE.CMD profile in the RACF OPERCMDS class to determine whether the operator is allowed to send a command to the specified system. This is intended to add the same level of checking to CPF that exists for the MVS ROUTE command

### **Kerberos sysplex support**

In z/OS V1R12, the network authentication service for z/OS utilizes a RACF function to help improve the availability of applications that use Kerberos or GSSAPI services when deployed in a DVIPA environment. This new support allows you to remove the dependency on which image of the Sysplex a Kerberos or GSSAPI application request is routed to. This can help improve application availability by enabling transparent failover for improved application availability and improved workload balancing between images in a sysplex.

### **Password policy**

In z/OS V1R12, IBM Tivoli Directory Server for z/OS provides support for configurable password policy rules that can be applied to user passwords in the directory. New support is available for automatic password revocation, expiration, formatting checks, history, and a password change mechanism that can be implemented on an individual, group, or directory basis. This new function helps you ensure that users change their passwords periodically, that new passwords meet your password requirements, that recently-used passwords not be reused, and that users can be locked out after a defined number of failed attempts.

### **Activity log management**

In z/OS V1R12, IBM Tivoli Directory Server for z/OS supports continuous activity logging. This new function allows you to specify the time of day or size of an activity log file, and to allow you to use the console command to initiate an activity log file switch. The server is designed to close the current log file or generation data set and open a new one. Also, the new function allows you to specify that log entries be filtered by IP address.

## **1.19.2 Cryptography**

Cryptography includes a set of techniques for scrambling or disguising data. The scrambled data is available only to someone who can restore the data to its original form. The purpose is to make data unintelligible to unauthorized persons, but readily decipherable to authorized persons. Cryptographic systems combine two elements:

- ▶ A process or algorithm that is a set of rules that specify the mathematical steps needed to encipher or decipher data.
- ▶ A cryptographic key (a string of numbers or characters), or keys. The algorithm uses the key to select one relationship between plaintext and ciphertext out of the many possible relationships the algorithm provides. The selected relationship determines the composition of the algorithm's result.

ICSF supports two main types of cryptographic processes:

- ▶ Symmetric, or secret key, algorithms, in which the same key value is used in both the encryption and decryption calculations.
- ▶ Asymmetric, or public key, algorithms, in which a different key is used in the decryption calculation than was used in the encryption calculation.

## **ICSF high performance secure key**

An enhancement to Central Processor Assist to Cryptographic Function (CPACF) on IBM System z10® servers with the CEX3C feature is designed to help facilitate the continued privacy of cryptographic key material when used by the CPACF for high performance data encryption. Leveraging the unique z/Architecture®, protected key CPACF is designed to help ensure that key material is not visible to applications or operating systems when used for encryption operations. Protected key CPACF is designed to provide significant throughput improvements for large volumes of data and low latency for small blocks of data. In z/OS V1R12, ICSF exploits the enhancements made to the CPACF in support of separate key wrapping keys for DES/TDES and AES. This is designed to provide the same functions available using the PCI card, but with the advantage of CPACF performance.

## **ICSF RSA private key support for smart card personalization**

In z/OS V1R12, ICSF provides support for translation of external RSA tokens wrapped with key encrypting keys into one of three smart card formats. A new callable service, PKA key translate (CSNDPKT), is designed to translate an existing RSA private key in CCA external format and into a specified smart card (SC) format in support of VISA, or the common ME or CRT format. To use this new function, you need an IBM System z9® or System z10 server with the Crypto Express2 feature with a minimum driver and microcode level. This function is also available on z/OS V1R8 and higher with the z/OS V1R8, z/OS V1r9 or z/OS V1r10 with the cryptographic support for z/OS V1R8-V1R10 and the z/OS.e V1R8 web deliverable and PTF UA46713.

## **System SSL elliptic curve cryptography phase 1**

The U.S. National Security Agency (NSA) announced Suite B Cryptography in 2005, and implementation guidelines in 2009. One newly listed technology was elliptic curve cryptography (ECC), which is regarded as a faster algorithm that requires a smaller key than RSA cryptography. This type of cryptography is expected to be attractive for use with small devices such as smart cards, which have limited computing power. In z/OS V1R12, PKI Services allows you to create and sign certificates with ECC keys in addition to RSA keys. In V1R12, System SSL provides support for ECC-related data structures, signing data, and verifying signed data using elliptic curve digital signature algorithm (ECDSA). This is intended to allow exploiters of z/OS System SSL to import ECC style certificates and private keys into key database files or PKCS#11 tokens and use ECDSA certificates to sign and verify operations.

## **ICSF growth (PKCS#11)**

With the focus on data security, cryptographic services must be continuously available. New ICSF designs that use integration between hardware and software components are intended to help improve the availability of z/OS host applications. In z/OS V1R12, a software cryptographic engine function embedded in ICSF is designed to be used so that no optional cryptographic coprocessors will be required for PKCS11 processing. Additional algorithms to be supported are DSA, DH, EC, AES GCM, BLOWFISH, and RC4. Furthermore, this support provides cryptography services and packaging that is designed to meet NIST FIPS 140-2 level 1 criteria, and support to comply with RFC4869 Suite B Cryptographic Suites for Ipsec and the following new clear key algorithms, as follows:

- ▶ Galois/Counter Mode encryption for AES (GCM)
- ▶ Elliptic curve Diffie-Hellman key derivation (ECDH)
- ▶ Elliptic curve digital signature algorithm (ECDSA)
- ▶ HMAC

In addition, support has been added for several new cryptographic algorithms:

HMACSHA-256-128; HMAC-SHA-384-192; HMAC-SHA-512-256; PRF-HMAC-SHA-256;  
PRF-HMAC-SHA-384; PRF-HMAC-SHA-512; AES-128-GCM; AES-256-GCM

**Note:** This function is also available with the z/OS Cryptographic Support for z/OS V1R9-V1R11 web deliverable.

### **PKCS#11 certificate object upgrade**

In prior releases, System SSL supported X.509 certificates with RSA key sizes up to 2048 bits for use in PKCS#11 tokens. In z/OS V1R12, System SSL gskkyman is enhanced to support the creation and management of X.509 certificates and keys within a PKCS#11 token that have RSA key sizes up to 4096-bits, DSA keys and Diffie-Hellman keys. These X.509 certificates and keys are usable through the System SSL APIs.

### **RACDCERT ECC certificate support**

Elliptic curve cryptography (ECC) is regarded as a faster algorithm that requires a smaller key than RSA cryptography. In z/OS V1R12, the RACF RACDCERT command supports certificates with the ECC keys, in addition to RSA and DSA keys.

### **PKI Services ECC-based certificates**

In z/OS V1R12, PKI Services allows you to create and sign certificates with ECC keys in addition to RSA and DSA keys.

### **RACF field support for ICSF HPSK**

This support provides a new attribute in the ICSF (SYMCPASSISTWRAP(YES/NO) segment to allow/disallow CPACF wrapping of secure keys stored in the CKDS. This is an update to the RACF ICSF segment in support of the ICSF high performance secure key CPACF support in z/OS V1R12.

The ICSF HPCK, “ICSF high performance secure key” on page 38 aims to do the following:

- ▶ Provide a “secure key” high performance bulk encryption solution, where “clear key” material is not present in z/OS host addressable memory or the ICSF CKDS.
- ▶ Provide a solution that combines the high performance characteristic of clear key and the high security characteristic of secure key.

The RACF support for this includes adding a new field to the ICSF segment to indicate whether or not a secure key (that is, a key wrapped under Symmetric, DES, or AES master key) stored in the CKDS can be used as input to the CSNBSYE, CSNBSYD, CSNBSMG, or CSNBSMV APIs and be rewrapped with the CPACF Wrapping Key.

The ICSF segment applies to profiles in the CSFKEYS/GCSFKEYS and XCSFKEY/GXCSFKEY classes that protect symmetric and asymmetric keys stored in the ICSF CKDS and PKDS, respectively.

### **ACL by IP address**

In z/OS V1R12, IBM Tivoli Directory Server for z/OS provides an extension to access control lists (ACLs) to provide the ability to dynamically transform base ACLs using filter ACLs you specify, to add or remove permissions based on:

- ▶ Bind distinguished name (DN)
- ▶ Alternate DNs
- ▶ Pseudo DNs
- ▶ Groups a bind or alternate DN belongs to
- ▶ IP address of the client connection

- ▶ Time of day that directory entry was accessed
- ▶ Day of week that directory entry was accessed
- ▶ The bind mechanism used
- ▶ Whether bind encryption was used

This function is designed to provide additional flexibility in access controls for LDAP connections.

### **Salted SHA-1 encryption**

In z/OS V1R12, IBM Tivoli Directory Server for z/OS provides Salted SHA-1 encryption support. Intended to make dictionary attacks against SHA-1 encrypted data much more difficult, stored Salted SHA-1 password values in LDAP include a random 20-byte string so that encrypting the same password more than once will usually result in differing encrypted values. This is intended to make it much more difficult to determine the encrypted password value. This support is designed to be functionally equivalent to that currently provided by the IBM Tivoli Directory Server, and can allow easier migration of LDAP server workloads to z/OS.

## **1.20 Service aids**

Service aids are programs designed to help you diagnose and repair failures in system or application programs. The AMBLIST and AMASPZAP service aids can be used to perform some program management tasks. Both AMBLIST and AMASPZAP support program objects, long names up to 1024 bytes, and multiple text classes.

### **Support for uncaptured UCBs (XTIOT)**

In z/OS V1R12, the SNAP/SNAPX services and dump processing (including that for SVC, SYSABEND, SYSMDUMP, and SYSUDUMP dumps), and the AMASPZAP program, are now supporting extended TIOT (XTIOT).

## **1.21 Infoprint Server for z/OS**

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.

### **Relief for data set capacity limits**

In z/OS V1R12, Infoprint Server for z/OS has enhanced extended mode processing to support more SYSOUT data with similar attributes, the maximum number of active jobs allowed by the job entry subsystem (JES2 or JES3), and line printer daemon (LPD) support for file sizes up to 4 GB. Support for large file sizes is available on z/OS V1R9 and higher with the PTF for APAR OA28795. Also, Infoprint Server will now prioritize spooling and printing for existing jobs higher than receiving new work. These changes are intended to help relieve constraints and reduce spool occupancy for Infoprint Server jobs.

## **LPD server now supports files larger than 2 GB**

The Infoprint Server's API has been enhanced to provide a function that returns the current API version. The DocumentInfo structure of the API now contains eight bytes of size information.







## z/OS V1R12 installation

This chapter describes changes in installation and migration when migrating from z/OS V1R11 to z/OS V1R12, and tasks to prepare for the installation of z/OS V1R12, including ensuring system and target system requirements are met, and coexistence requirements are satisfied. New migration actions are introduced in z/OS V1R12. This chapter focuses on identifying some of these actions that must be performed for selected elements when migrating to z/OS V1R12. Some of the new migration tasks for selected elements, such as BCP, are highlighted.

## 2.1 z/OS V1R12 installation considerations

The program number for z/OS V1R12 is 5694-A01. Use this program number when ordering z/OS V1R12. Ordering for z/OS V1R12 began on September 10, 2010, which was also the first date for ordering z/OS V1R12 ServerPac, SystemPac®, and CBPDO using CFSW configuration support, or ShopzSeries, the Internet ordering tool.

z/OS V1R12 was generally available on September 24, 2010 via ServerPac, CBPDO and SystemPac. When ordering z/OS V1R12, ensure that you order all the optional features that you were licensed for in previous releases of z/OS.

**Note:** There are no new elements added in z/OS V1R12. One element, Managed System Infrastructure for Setup (msys for Setup) has been withdrawn in z/OS V1R12. Therefore, the FMIDs HMSI707 and HMSI737 are not included in z/OS V1R12. z/OS V1R11 is the last release to include these FMIDs.

### 2.1.1 Installation consideration for z/OS V1R12 CBPDO

If installing CBPDO, delete the msys for Setup element from the target system after z/OS V1R12 is installed. You can use sample job CLNOS390 to delete msys for Setup FMIDs. Delete the obsolete data sets and paths for msys for Setup, and remove the associated DDDEFs.

*z/OS Migration, GA22-7499*, identifies the deleted data sets and paths.

## 2.2 Driving system requirements for z/OS V1R12

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. Once 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 shares 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.

**Important:** DVD is the newest medium and, like tape and the Internet, can be used for all products, that is, z/OS and products that run on z/OS, as well as for PTF maintenance. Choosing DVD delivery reduces the size of shipments to you, and eliminates the need to introduce foreign tapes into your site. Your order is placed and processed through ShopzSeries as a DVD (4.7 GB single-sided single-layer layered) media order.

### Driving system requirements for installing z/OS V1R12

These are the driving system requirements:

- ▶ The driving system must be z/OS V1R10 or z/OS V1R11.
- ▶ You must use the latest release of Program Management Binder, HLASM, SMP/E to install CBPDO wave 1 and 2 FMIDs, and install service for CBPDO and ServerPac. The Program Management Binder changed in z/OS V1R12.

**Note:** The Customized Offering Driver (COD) may also be used as the driving system. See *z/OS Planning for Installation, GA22-7504*, which provides a detailed description of the driving system requirements.

### DASD storage requirements

If you are migrating to z/OS V1R12 from z/OS V1R11 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.

The DASD storage requirements to install z/OS V1R12 are as follows (approximate):

- ▶ Target libraries: total space required 5891 3390 cylinders
- ▶ Distribution libraries: total space required 8599 3390 cylinders
- ▶ Root file system: total space required 3100 3390 cylinders

**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% free space to accommodate the installation of maintenance.

## 2.3 Methods of installing z/OS

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), while others are available for an additional fee. Choose the software delivery method. The choices are:

- ▶ DVD
- ▶ Tape
- ▶ Internet

If you are migrating to z/OS V1R12 from z/OS V1R10 or V1R11, use any of the installation packages to install z/OS V1R12 such as the entitled packages (ServerPac and CBPDO).

This section describes each package, as follows:

- ▶ Customized Offerings Driver V3 (5751-COD) - for new users to z/OS.
- ▶ ServerPac
- ▶ CBPDO

### Choosing DVD as delivery

DVD is a new delivery medium available with z/OS V1R12 and, like tape and the Internet, can be used for all products, that is, z/OS and products that run on z/OS, as well as for PTF maintenance. Choosing DVD delivery reduces the size of shipments to you, and eliminates the need to introduce foreign tapes into your site. Your order is placed and processed through ShopzSeries as a DVD (4.7 GB single-sided single-layer layered) media order.

With a DVD delivery, consider the following:

- ▶ If you are installing from a DVD, you will need to make the order contents available to the ServerPac Install Dialogs.

- ▶ To install a DVD order, you will need to make the order contents available to the CBPDO Jobs.

**Note:** z/OS is no longer delivered on 3480, 3480C, and 3490E tapes.

### 2.3.1 Customized Offerings Driver V3 (5751-COD)

The Customized Offerings Driver is intended to run in single-system image and monoplex modes only. Its use in multi-system 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. For example, IBM does not support the use of the Customized Offerings Driver to run any production workload.

**Note:** IBM has installed the service on the Customized Offerings Driver to allow it to be IPLed and used (if necessary) after the new system has been IPLed and used.

#### Using the Customized Offerings Driver

The Customized Offerings Driver V3 (5751-COD) is an entitled driving system you can use as follows:

- ▶ If you do not have an existing system to use as a driving system.
- ▶ If 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 V1R10 system.

The Customized Offerings Driver is in DFSMSdss dump/restore format and supports 3390 triple-density or higher DASD devices. It has the following requirements:

- ▶ A locally attached non-SNA terminal and a system console from the IBM (or equivalent) family of supported terminal types: 317x, 327x, 319x, or 348x.
- ▶ An IBM (or equivalent) supported tape drive is also required to restore the driver.

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, but these items are not customized. However, existing environments can be connected to, and used from, the Customized Offerings Driver system.

#### Installing the Customized Offerings Driver

Depending on the level of your existing system, the Customized Offerings Driver might be at higher product and service levels. Therefore, as is true of the level of software you plan to install, fallback service might be necessary to let you IPL and use your existing level of software after the Customized Offerings Driver has been IPLed and used in any environment. You must one of the following:

- ▶ Use the Customized Offerings Driver in a completely isolated environment.
- ▶ Install the needed fallback service on your existing system before the Customized Offerings Driver is IPLed. A completely isolated environment shares no DASD with any other system and will not be used to IPL any lower level of software.

Installing the service on your existing system will also satisfy the requirements for falling back from the Customized Offerings Driver. This will allow you to IPL and use your current level of software after using either the Customized Offerings Driver or the new system.

### 2.3.2 Driving system requirements using ServerPac (entitled)

ServerPac is an entitled software delivery package consisting of products and services 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.

**Note:** Of the two entitled installation packages available for installing z/OS, most clients choose ServerPac rather than CBPDO.

Two types of ServerPac installation are available to you. You choose the type when you install, not when you order.

- ▶ A full system replacement installs a complete z/OS system. It installs all the data sets you need to IPL, to log on to the target system, and to run a z/OS image in order to complete other installation and customization tasks. The installed data sets fall into two major categories:
  - System software and related data sets (such as distribution and target libraries, SMP/E CSI data sets, and sample libraries).
  - Operational data sets (such as page data sets, system control files, and a master catalog).
- ▶ 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. When new operational data sets are required, you must allocate and initialize them before you IPL. For example, you might need to add parameters required by the new software level or change data sets so they will work with both the old and new levels.

### 2.3.3 Custom-Built Product Delivery Option (CBPDO)

CBPDO is an entitled software delivery package consisting of uninstalled products and unintegrated service. There is no dialog program to help you install, as there is with ServerPac. You must use SMP/E to install the individual z/OS elements and features, and their service, before you can IPL. Installation instructions are in the publication *z/OS Program Directory*.

z/OS and all products that run on z/OS are available by way of CBPDO. When enhancements (such as the z/OS V1R4 z990 Exploitation Support feature) are provided as features of a release subsequent to the general availability of the release, the enhancements are available by themselves in CBPDO. There is no need to reorder and reinstall all of z/OS. In addition, the enhancements are available for Internet delivery in CBPDO when ordered using ShopzSeries. This support provides a quick and easy way for you to order and receive these post-release features.

## 2.4 Coexistence, migration, and fallback considerations

z/OS systems can coexist with specific prior releases. This is important because it gives you flexibility to migrate systems in a multi-system configuration using rolling IPLs rather than requiring a systems-wide IPL.

**Note:** A rolling IPL is the IPL of one system at a time in a multi-system configuration. You might 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. By using LPAR technology, you can use rolling IPLs to upgrade your systems without losing either availability or capacity.

### Understanding coexistence

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 multi-system configuration, or they could be shared over a period of time by the same system in a single-system configuration. 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).

### Coexistence PTFs

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. You should 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 intended to allow new functions provided in later releases to work on earlier-level systems.

Coexistence and fallback play an important part in planning for migration to the latest release. IBM's policy regarding the releases that are supported for coexistence and fallback, as well as migration, and the states in which specific releases are supported is as follows:

- ▶ Coexistence of a z/OS V1R12 system with a z/OS V1R10 or z/OS V1R11 system is supported.
- ▶ Migration from a z/OS V1R10 system or z/OS V1R11 system to z/OS V1R12 is supported.
- ▶ Fallback from a z/OS V1R12 system to a z/OS V1R10 or z/OS V1R11 system is supported.

### Understanding fallback

Fallback or backout is a return to the prior level of a system. Fallback can be appropriate if you migrate to z/OS V1R12 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.

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

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.

## 2.4.1 Installation considerations for coexistence PTFs

The required coexistence service must be installed on z/OS V1R10 and z/OS V1R11 systems that will coexist with z/OS V1R12 to enable the lower z/OS releases to tolerate changes in z/OS V1R12, as follows:

- ▶ Receive the latest HOLDDATA on the z/OS V1R10 and z/OS V1R11 systems.  
Use SMP/E V3R5, which is the level of SMP/E in z/OS V1R10, V1R11, and V1R12, to identify required coexistence PTFs that must be installed on z/OS V1R10 and z/OS V1R11 systems in preparation for migration to z/OS V1R12.
- ▶ Acquire and RECEIVE the latest HOLDDATA onto your z/OS V1R10 and z/OS V1R11 systems. Use your normal service acquisition portals, or download the HOLDDATA directly from:

<http://service.software.ibm.com/holddata/390holddata.html>

- ▶ Run the SMP/E REPORT MISSINGFIX command on your z/OS V1R10 and z/OS V1R11 systems, and specify a Fix Category (FIXCAT) value of IBM.Coexistence.z/OS.V1R12. The report will identify any missing coexistence and fallback PTFs for that system. For complete information about the REPORT MISSINGFIX command, see *SMP/E V3R5.0 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.

**Important:** z/OS V1R12 will not identify coexistence and fallback PTFs. IBM plans to remove the Enhanced PSP Tool (EPSPT) and the extract files from the web on December 31, 2010. Use the SMP/E MISSINGFIX report and FIXCAT instead of EPSPT to identify missing coexistence PTFs.

With z/OS V1R12, *z/OS Migration*, GA22-7499 will no longer document the required coexistence and fallback PTFs since the SMP/E MISSINGFIX command, in conjunction with the latest HOLDDATA, will identify the current coexistence PTFs that are required.

### Coexistence levels with z/OS V1R12

z/OS systems can coexist with specific prior releases; see Figure 2-1 on page 50. This is important because it provides you with the flexibility to migrate systems in a multi-system configuration using rolling IPLs rather than requiring a systems-wide IPL.

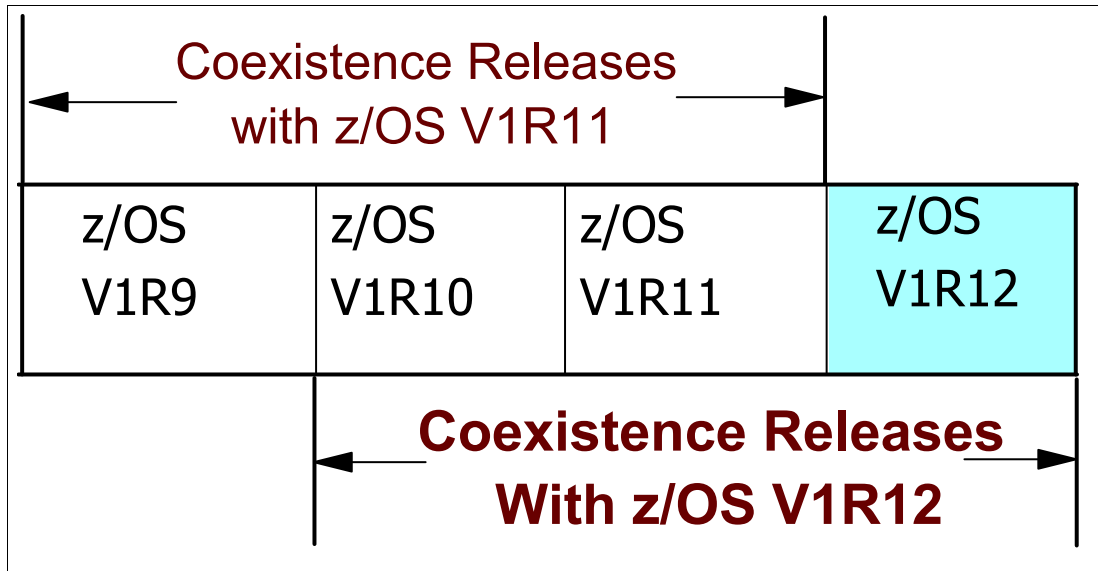


Figure 2-1 Coexistence levels

## 2.5 Migration actions for z/OS V1R10, z/OS V1R11 systems

This section describes new migration actions for the BCP element that can be performed on current coexistence-supported z/OS release systems, z/OS V1R10 and z/OS V1R11.

### 2.5.1 Trace options with the CTIGRSxx member

Before z/OS V1R12, the default buffer value (BUFSIZE) for the trace option with the GRS component in the IBM-supplied CTIGRS00 parmlib member was 128 K. Starting with z/OS V1R12, the default size in CTIGRS00 is increased to 16 M.

#### Migration action

If you specify your own CTIGRSxx parmlib member, change the BUFSIZE in that parmlib member to 16 M.

### 2.5.2 Programs that use system-generated temporary data sets

For locally written programs with z/OS V1R12, MVS device allocation, which is responsible for generating data set names of temporary data sets (that is, data sets created and deleted within the same job), has been modified to use a more unique format for the data set name by default.

The pre-z/OS V1R12 format for temporary data set names is as follows, where the label is defined in JCL using the DSN=&&label syntax:

```

SYSyyddd.Thhmmss.RA000.jobname.tempname.Hnn (nn: sysid)
SYSyyddd.Thhmmss.RA000.jobname.Rggnnnnn (gg: sysid)

```



## **z/OS V1R12 implementation**

In z/OS V1R12, device allocation does not use the label in the data set name that it selects. Instead, device allocation uses another supported format, as follows:

```
SYSyyddd.Thhmmss.jobname.RA000.Rggxxxxx
```

This format is more unique and allows installations to start using facilities with JES that allow multiple jobs with the same job name to execute at the same time. Most programs already allow either format of data set name to be included, and these programs are not affected by this change. Locally written programs can contain dependencies about what the JCL used to invoke. Thus it may assume that the DSN=label syntax will always be used.

### **Notes:**

1. The ability to use the DSN=&&label syntax as a reference label throughout the JCL job continues to be supported. Only the physical name of the data set is changed.
2. The SYSTEM TEMPDSFORMAT option affects only the data sets that specify DSN=&&mysdn, but not the data sets that do not specify DSNAME at all.

## **Migration action**

Examine the program source for locally written production programs for the commonly known temporary data set name string RA000. If the name string appears, analyze the code to determine whether it assumes that the next qualifier in the data set name is a label. Update the program to eliminate this dependency.

Specify the ALLOCxx parmlib option SYSTEM TEMPDSFORMAT(INCLUDELABEL) to allow installations to migrate to z/OS V1R12 and test the JCL, instead of attempting to analyze any locally written programs before installing the product.

## **2.5.3 Track CSVRTLS services**

The change is required if you use CSVRTLS services. z/OS V1R5 was the last release of z/OS to support Run-Time Library Services (RTLS) for Language Environment. In z/OS V1R12, the underlying CSVRTLS services are removed from z/OS. A way to track CSVRTLS usage, and to let you find any programs that might be using these services, is available for z/OS V1R11, and rolled back to z/OS V1R10 with APAR OA29019.

## **Migration action**

Exploit the z/OS tracking facility to help you determine whether you are using any of the CSVRTLS services with the following commands:

- ▶ SET RTLS
- ▶ DISPLAY RTLS

The CSVRTLS macro is removed from z/OS V1R12 and for older releases, as follows:

- ▶ For z/OS V1R11, see APAR OA29995.

Via this APAR, changes are made to exploit the z/OS tracking facility so that you can tell whether you are using any of the following in preparation for the announced withdrawal of these functions in the release after z/OS V1R11:

- Use of the SET RTLS command
- Use of the DISPLAY RTLS command

- Use of the CSVRTLS macro
- Use of the RTLS system parameter
- In addition to those cases, use of the RTLS system parameter in IEASYSxx parmlib member will be tracked, but will be displayable only after having issued the **DISPLAY RTLS** command, since the RTLS system parameter is processed before the tracking facility can be activated. And, of course, if you issue the **DISPLAY RTLS** command itself, then that usage will be shown as well.

**Note:** You should report uses of the CSVRTLS macro to the owner of the product that invoked it. The display might help you determine this, as it shows the job name. Uses of the operator command and system parameter will not show a useful jobname. In general, there is no need to report these entries to IBM.

- ▶ For z/OS V1R10, install PTF UA50068 for APAR OA29019.

Figure 2-2 is an example of the Tracker report output (note, some columns were compressed to fit in this description).

```

D OPDATA, TRACKING
CNZ1001I 15.47.47 TRACKING DISPLAY
STATUS=ON      NUM=2    MAX=1000 MEM=n/a EXCL=0    REJECT=0
--TRACKING INFORMATION-- -VAL- JOBNAME  PROGRAM+OFF-- ASID NUM
WTO: IEC350I CATALOG ADD    00 CATALOG  IGGOCLX0 80BAC  1A  1
WTO: IEF677I WARNING MES    00 JES2     IEFNB903  C9AA  13  1

```

Figure 2-2 Tracker report output for RTLS

## Migration assistance tracker commands

The tracking facility can be started and display what it finds with the following commands:

- ▶ The SETCON command is used to activate and deactivate the Console ID Tracking facility.
- ▶ The DISPLAY OPDATA,TRACKING command is used to display the current status of the console ID tracking facility, along with any recorded instances of violations.

## CNIDTRxx parmlib member

The CNIDTRxx parmlib member is used to list violations that have already been identified to prevent them from being recorded again.

An optional CNIDTRxx parmlib member can be defined to exclude instances from being recorded. The exclusion list is picked up when the tracker is started or via the SET command. The filters for the exclusion list are the three items that make an instance unique. You can exclude all SMS instances or just LSPACE instances, or exclude all instances created by job names that begin with BRS\*. There are many more possibilities. The filters support wildcarding on these three fields.

To learn more about the tracking facility, see Appendix A in *z/OS MVS Planning: Operations*, SA22-7601. The tracking facility supports an Exclusion list that informs the facility which instances have already been reported and should no longer be tracked. This list is updated from the data sent to IBM. Current lists are available for download from:

<http://www-1.ibm.com/servers/eserver/zseries/zos/downloads/>

## 2.5.4 Remove classification rules with the EWLM transaction class

This is a recommended task, because otherwise you will have to delete the classification rules the next time you use the WLM ISPF application to modify the EWLM subsystem type.

In z/OS V1R12, the workload management (WLM) service definition no longer supports the work qualifier EWLM transaction class name (ETC) for classification rules of the subsystem type EWLM. Although z/OS V1R12 disregards classification rules with the ETC work qualifier, you should consider removing them. If you do not remove the rules, you will have to delete them the next time you use the WLM ISPF application to modify the EWLM subsystem type.

### Migration action

If your WLM service definitions contain classification rules for subsystem type EWLM with the ETC work qualifier, start the WLM ISPF application and choose the Classification Rules option from the Definition Menu. Use the Modify option (3) for the IBM-supplied subsystem type EWLM. Delete all rows with the ETC qualifier type by using the Delete row Option (D).

## 2.5.5 Update the SFM policy

To control automatic termination of impaired critical members, it is recommended to designate how long XCF will wait before initiating termination of the impaired critical member.

Starting with z/OS V1R12, a member of an XCF group can identify itself as being critical to the operation of the group or the system. If a critical member appears to be inoperative (impaired) and the condition persists long enough, XCF automatically terminates the member in an attempt to resolve the problem. For a member that is critical to the operation of the system, this termination causes the system to be removed from the sysplex.

### SYSGRG group members

Members of the SYSGRS group, for instance, are critical to the operation of the system. If any GRS member is impaired, ENQ processing is likely impacted throughout the sysplex. Failure to perform ENQ processing in a timely fashion has significant negative impact. Thus if a GRS member appears to be impaired, XCF will automatically remove from the sysplex the system on which that member resides.

You can set the MEMSTALLTIME parameter in your sysplex failure management (SFM) policy to control how long XCF allows a critical member to persist in an impaired state before it initiates termination of the member (or the member's system). If the MEMSTALLTIME specification resolves to NO (either implicitly or explicitly), XCF will terminate an impaired critical member if the condition persists as long as the failure detection interval (INTERVAL) of the system on which the member resides, or if the condition persists as long as two minutes, whichever is greater. To determine which groups are using the critical support, issue the appropriate XCF display command.

### MEMSTALLTIME parameter

The MEMSTALLTIME parameter also determines how long XCF allows a signalling sympathy sickness condition to persist before terminating a stalled group member that is contributing to the problem. The parameter indicates the number of seconds that XCF should wait before it terminates a member that is impacting the sysplex. A MEMSTALLTIME value of 120 (two minutes) seems to suit many installations because it provides some additional time for the system to resume normal operation, yet allows automatic action to resolve the problem before the sympathy sickness condition critically impacts the sysplex. Installations that resolve such conditions through manual intervention sometimes use a higher value to allow time for such

intervention to be accomplished. Installations that are less able to tolerate sympathy sickness conditions sometimes set lower values.

### Migration actions

If you do not have an SFM policy, or if the SFM policy specifies (either implicitly or explicitly) MEMSTALLTIME(NO), determine whether the default time that XCF will wait before terminating an impaired critical member is acceptable. The default time is the maximum of the effective failure detection interval, or two minutes, whichever is greater.

If you have an SFM policy that specifies MEMSTALLTIME other than NO, confirm that the current specification is also acceptable for the termination of critical members.

If changes are necessary, take the following steps:

1. As needed, use the IXCMIAPU utility to create a function couple data set for SFM policies.
2. Use the IXCMIAPU utility to create or modify an SFM policy with an acceptable MEMSTALLTIME specification.
3. Issue the SETXCF command to activate the desired SFM policy.

## 2.5.6 Accommodate new REUSASID default

This is a *required* action. If this migration action is not taken, 0D3 abends might occur with downlevel products that provide no toleration support for reusable ASIDs. Because reusable ASIDs have been available since z/OS V1R9, it is reasonable to expect that the current levels of products are tolerant of reusable ASIDs.

In z/OS V1R9, the REUSASID(YES | NO) parameter in parmlib member DIAGxx was introduced with a default of NO. Starting with z/OS V1R12, the default is changed to YES.

When a reusable ASID is requested by the START command or the ASCRE macro, this reusable ASID is assigned if REUSASID(YES) is specified in DIAGxx. If REUSASID(NO) is specified in DIAGxx, an ordinary ASID is assigned. The default is REUSASID(YES). The use of reusable ASIDs might result in system 0D3 abends, if products or programs have not been upgraded to tolerate reusable ASIDs.

### Migration actions

Take the following migration actions:

1. On z/OS V1R11 or z/OS V1R10 systems, specify REUSASID(YES) in the parmlib member DIAGxx. On z/OS V1R12 systems, keep REUSASID(YES) or allow it to default to YES.
2. Verify that no 0D3 abends occur as a result.
3. If 0D3 abends do occur, apply appropriate maintenance to the affected code.
4. If this is not possible, specify REUSASID(NO) in DIAGxx on z/OS V1R12 to override the new default of REUSASID(YES).

## 2.5.7 Review the list of WTORS in parmlib member AUTOR00

This is a *required* action. In z/OS V1R12, the DDDEF parmlib provides an AUTOR00 member. This member should be found in your parmlib concatenation during IPL and will result in auto-reply processing being activated. If the WTORS listed in AUTOR00 are automated by your existing automation product, ensure that the replies in AUTOR00 are appropriate.

## Migration action

Examine the WTOR replies in the AUTOR00 parmlib member. If the replies or delay duration are not desirable, you can create a new AUTORxx parmlib member and make corresponding changes. Also compare the replies to what your automation product would reply to these WTORs. Make sure that the AUTOR00 replies are in accordance with the replies from your automation product. IBM does not recommend making updates to AUTOR00, because updates to AUTOR00 might be made by the service stream or in new z/OS releases.

If you have created an AUTORxx parmlib member, update the IEASYSyy parmlib member that you use for IPL. Add the following statement to the IEASYSyy member: AUTOR=(xx,00). Here xx corresponds to the AUTORxx parmlib member that you created. The IEASYSyy members specifying AUTOR cannot be shared with prior z/OS releases. If you only need the default AUTOR00 settings, you can omit specifying AUTOR= in IEASYSyy, and other z/OS levels can continue to use IEASYSyy. Even if AUTOR= is not specified in IEASYSyy, AUTOR00 is used if it exists.

If you do not want to activate auto-reply processing, specify AUTOR=OFF in the parmlib member IEASYSxx or in response to message IEA101A SPECIFY SYSTEM PARAMETERS. It is not recommended that you remove AUTOR00 from parmlib, because service or new releases might reinstall AUTOR00. If there is no AUTOR00 member in parmlib, auto-reply is not activated and the following messages are produced:

```
CNZ2600I AUTO-REPLY POLICY ATTEMPTING TO USE AUTOR=00.  
IEA301I AUTOR00 NOT FOUND IN PARMLIB  
CNZ2601I AUTO-REPLY POLICY NOT ACTIVATED.  
NO ENTRIES SPECIFIED
```

The IEASYSyy members specifying AUTOR=OFF cannot be shared with prior z/OS releases.

## 2.5.8 Removing some references to Unicode services

Starting in z/OS V1R12, the pre-built image CUNIDHC2 has been eliminated. This pre-built image contained all the conversion tables supported by DB2 and would be loaded into storage when you had an empty Unicode environment (no UNI=xx in the IEASYSxx member) and the first requestor of a Unicode conversion service would be DB2.

Given that most clients would use only a handful of these tables and given that Unicode Services has the capability to dynamically load tables into storage, the need for pre-built image has become obsolete. Unicode Services will no longer ship the pre-built image SYS1.SCUNIMG(CUNIDHC2) and will no longer automatically load the pre-built image. We recommend that you use the Unicode On Demand capability to load all tables.

## Migration action

Take the following actions:

- ▶ Remove SYS1.SCUNIMG from the LNKLST specification and APF authorization list.
- ▶ Remove the catalog entry for SYS1.SCUNIMG.
- ▶ Remove the following DDDEF entries:
  - DDDEF ACUNIMG for SYS1.ACUNIMG
  - DDDEF SCUNIMG for SYS1.SCUNIMG

## 2.6 Migration considerations before first IPL of z/OS V1R12

Migration actions need to be done for the following components and functions of the operating system before doing the first IPL of z/OS V1R12.

### 2.6.1 Infoprint Server

For z/OS V1R12, the format of the Infoprint Server Printer Inventory files has changed from Version 1 to Version 2 format. When you start Infoprint Server on z/OS V1R12 for the first time, Infoprint Server reformats the Version 1 Printer Inventory files and creates Version 2 Printer Inventory files. The Version 1 Printer Inventory files are not removed so that if you need to fall back to the previous z/OS release, Infoprint Server can use the Version 1 Printer Inventory files. Therefore, the Printer Inventory file system requires more space in z/OS V1R12 than in previous releases. You might need to increase space in the Infoprint Server Printer Inventory.

#### Migration actions

The Infoprint Server element has new migration actions to perform before installing z/OS V1R12.

Increase the space in the Printer Inventory file system. The Infoprint Server Printer Inventory conversion upon initialization of z/OS V1R12 is automatic. The size of the inventory is increased.

To increase the size of the file system, you can use the z/OS UNIX **zfsadm grow** (zFS) or **confighfs** (HFS) commands.

**Tip:** Set the **aggrfull** (zFS) or **FSFULL** (HFS) file system option so that warning messages are issued if the Infoprint Server base directory (`/var/Printsrv`) is getting full.

An IBM Health Checker for z/OS check has been implemented to assist with this.

Run the **df** command to display the current utilization of the Printer Inventory file system:

```
df -P /var/Printsrv
```

Printer Inventory files are located in the Infoprint Server base directory. The default base directory name is `/var/Printsrv`. You might have changed the base directory name in the `base-directory` attribute in the `aopd.conf` configuration file. The `aopd.conf` default location is `/etc/Printsrv/aopd.conf`. However, you might have specified a different location in environment variable `AOPCONF`.

The free space required is 200% of the sum of the Version 1 Printer Inventory and historical Printer Inventory files (`master.db`, `jestoken.db`, `pwjestoken.db`, `hinu/hinv.db`, and `logdb/log.db`). If the “Capacity” is greater than 33%, increase the size of the file system.

#### Fallback actions

If you need to fall back, remove the Version 2 Printer Inventory files. During the fallback, an older inventory is used and updates the inventory. The older inventory from pre-z/OS V1R12 systems will not be reflected in the newer inventory (from z/OS V1R12).

If Version 2 Printer Inventory files exist after falling back to z/OS V1R11 or z/OS V1R10, remove them from the Infoprint Server base directory. Be careful not to remove any Version 2

files while running z/OS V1R12 because Infoprint Server on z/OS V1R12 requires Version 2 Printer Inventory files. Version 2 files have the extension v2db. The default base directory is /var/Printsrv.

You might have changed the base directory name in the base-directory attribute in the aopd.conf configuration file. The aopd.conf default location is /etc/Printsrv/aopd.conf. However, you might have specified a different location in environment variable AOPCONF.

**Example:** These z/OS UNIX commands switch to an effective UID of 0, remove all files with the v2db extension from directory /var/Printsrv, and switch back to the original UID:

```
su
rm -f $(find /var/Printsrv/ -name "*.v2db")
exit
```

To remove Printer Inventory files, you must have an effective UID of 0 or be a member of the RACF AOPADMIN group.

**Note:** In z/OS V1R12, the format of the Infoprint Server Printer Inventory files has changed from Version 1 to Version 2 format. When you start Infoprint Server on z/OS V1R12 for the first time, Infoprint Server reformats the Version 1 Printer Inventory files and creates Version 2 Printer Inventory files. Both Version 1 and Version 2 Printer Inventory files exist in the Infoprint Server base directory. Infoprint Server on z/OS V1R12 uses the Version 2 Printer Inventory files. If you fall back to z/OS V1R11 or V1R10, Infoprint Server uses the Version 1 Printer Inventory files.

If you start Infoprint Server on z/OS V1R12 a second time after falling back to a previous z/OS release, Infoprint Server uses the existing Version 2 Printer Inventory files that it created the first time you started Infoprint Server on z/OS V1R12. It does not reformat the Version 1 Printer Inventory files again.

If you want Infoprint Server to reformat the Version 1 Printer Inventory files again, remove the Version 2 Printer Inventory files before you start Infoprint Server on z/OS V1R12. Because the Version 2 Printer Inventory files no longer exist, Infoprint Server reformats the Version 1 Printer Inventory files and creates a new set of Version 2 Printer Inventory files. In most cases, you should remove the Version 2 Printer Inventory files if they exist. If you do not remove the Version 2 Printer Inventory files, any changes that the administrator made to the Version 1 Printer Inventory on z/OS V1R11 or z/OS V1R10 are not in the Version 2 Printer Inventory. In addition, Infoprint Central on z/OS V1R12 cannot display historical information for jobs that Infoprint Server processed on z/OS V1R11 or V1R10.

## Upgrade Java for the Internet Printing Protocol (IPP) Server

The Infoprint Server element requires an upgrade to the IBM XML Toolkit V1.10 if using Infoprint Central. This is required as of R12 if you use IPP Server and specify the JAVA\_HOME environment variable.

You are using IPP Server if the start-daemons={ippd} attribute is specified in the Infoprint Server configuration file. The configuration file's default location is /etc/Printsrv/aopd.conf. However, you might have specified a different location in environment variable AOPCONF.

In z/OS V1R12, the Internet Printing Protocol (IPP) Server component of Infoprint Server requires Java V6.0. If the JAVA\_HOME environment variable specifies the location of an earlier version of Java, you must update the variable.

**Migration action:**

You can do this before installing z/OS V1R12 if APAR OA28720 is applied. Otherwise, do it after installing z/OS V1R12.

Also install IBM 31-bit SDK for z/OS, Java 2 Technology Edition, V6 (5655-R31). If you use IPP Server, edit the aopstart EXEC to update the directory path specified in the JAVA\_HOME environment variable. IPP Server requires the 31-bit version of Java V6.0.

If you installed Java V6.0 in the default Java directories, you do not need to specify the JAVA\_HOME environment variable. If JAVA\_HOME is not specified, IPP Server looks for Java files in the /usr/lpp/java/J6.0 directory.

**Upgrade XML for Infoprint Central**

This is required if you use Infoprint Central. In z/OS V1R12, the Infoprint Central component of Infoprint Server, which you can use to work with IP PrintWay™ extended mode print jobs and printers, requires the IBM XML Toolkit V1.10 product.

**Migration actions:**

1. Install IBM XML Toolkit V1.10 (5655-J51).
2. Specify the XML V1.10 libraries in the LIBPATH environment variable in your z/OS IBM HTTP Server environment variables file (default location is /etc/httpd.envvars). After z/OS V1R12 is installed, Infoprint Central requires the XML V1.10 libraries:
  - LIBPATH: Change /usr/lpp/ixm/IBM/xml4c-5\_6/lib to /usr/lpp/ixm/IBM/xml4c-5\_7/lib
  - LIBPATH: Change /usr/lpp/ixm/IBM/xslt4c-1\_10/lib to /usr/lpp/ixm/IBM/xslt4c-1\_11/lib
  - ICU\_DATA: You can remove this variable because XML no longer uses it.
3. Restart the z/OS IBM HTTP Server to pick up the changes to the environment variables file.

## 2.6.2 Language Environment (LE)

The LE element has some new migration actions to perform before the first IPL of z/OS V1R12, as follows:

Set run-time options as overrideable or nonoverrideable in the CEEPRMxx parmlib member. This is a recommended action for z/OS V1R12, so that you can eliminate use of the assembler language usermods to specify installation-wide run-time options, and use the CEEPRMxx parmlib member instead.

In z/OS V1R12, you can set run-time options as overrideable or nonoverrideable in the CEEPRMxx parmlib member using the OVR or NONOVR attribute or with a SETCEE command. The ability to specify an option as overrideable or nonoverridable removes a barrier to using CEEPRMxx.

**Migration action**

Set run-time options in the CEEPRMxx parmlib member using the OVR or NONOVR attribute or by issuing the SETCEE command. You can eliminate future Language Environment migration actions by no longer using the USERMODs at the installation default level to mark run-time options as nonoverrideable.



**Note:** In a future release, IBM plans to remove the capability to change the default Language Environment run-time options settings via SMP/E installable USERMODs. IBM recommends using the CEEPRMxx parmlib member to change the default.

Use Unicode Services to create conversion tables since the Language Environment library CEE.SCEEUMAP has been removed.

### 2.6.3 z/OS UNIX

The following migration actions need to be performed before installing z/OS V1R12.

#### **Remove the MAXSOCKETS value**

Before z/OS V1R12, a value had to be specified for the MAXSOCKETS keyword in the NETWORK statement for AF\_UNIX in the BPXPRMxx parmlib member of SYS1.PARMLIB if the maximum number of AF\_UNIX sockets for the system needed to be greater than the default of 100. As of z/OS V1R12, the value does not need to be specified because a maximum value of 10,000 has been set for MAXSOCKETS for AF\_UNIX. The MAXSOCKETS keyword is still allowed on the NETWORK statement for AF\_UNIX, but will be ignored on a z/OS V1R12 system.

#### **Migration action**

Remove any MAXSOCKETS statement from NETWORK statements for DOMAINNAME(AF\_UNIX). In a configuration with a shared BPXPRMxx parmlib, MAXSOCKETS should only be removed when all systems are at z/OS V1R12.

#### **Connection scaling**

Discontinue use of z/OS UNIX System Services Connection Scaling because this is recommended as of R12. In a future release, IBM plans to discontinue support of z/OS UNIX System Services Connection Scaling, specifically the Connection Manager and Process Manager components.

z/OS UNIX System Services Connection Scaling consists of the following FMIDs:

- ▶ HCMG110 (Connection Manager)
- ▶ JCMG1J0 (Connection Manager Japanese)
- ▶ HPMG110 (Process Manager)
- ▶ JPMG1J0 (Process Manager Japanese)

### 2.6.4 Shell and Utilities version of the tsocmd command

Before z/OS V1R11, the **tsocmd** command was obtained from the Tools and Toys section of the z/OS UNIX website. Starting with z/OS V1R12, Shell and Utilities support of the **tsocmd** command has been added. The supported version differs from the Tools and Toys version in a number of ways, as follows:

- ▶ Exit values are consistently sent if the issued TSO/E command fails.
- ▶ The tsoin and tsoout environment variables are not supported. Instead, stdin and stdout are supported, as is done for most other Shell and Utilities commands.
- ▶ The TSO PROFILE environment variable is supported.

## Migration action

Look for current use of the Tools and Toys version of the **tsocmd** command. If there is no current use, no actions or changes are required. If there is current use of this command, determine if the command is located in `/bin` or in another directory. Also, determine if you want to preserve the Tools and Toys version in addition to the officially shipped version.

Consider the following options:

1. If you want to preserve the Tools and Toys version, ensure the Tools and Toys version of **tsocmd** is not located in `/bin` prior to the installation of z/OS V1R12.
2. If you do not want to preserve the Tools and Toys version, and it is located in `/bin`, then the installation of z/OS V1R12 will automatically replace the Tools and Toys version with the new officially supported version. If the Tools and Toys version is not located in `/bin`, remove it from its current location. In either case, you will also need to remove the Tools and Toys **tsocmd** command load module from the authorized load library (either prefix `TSOCMD.LOADLIB` or `SYS1.LINKLIB`) as described in the **tsocmd** command Tools and Toys README documentation available at the following site:

<ftp://ftp.software.ibm.com/s390/zos/tools/tsocmd/tsocmd.readme.txt>

**Note:** A new version of the **tsocmd** command that matches the version shipped with z/OS V1R12 will be made available on the Tools and Toys website. If you have multiple systems at different releases and you want to have the same version of the tool on all releases, or if you want to try out the new command prior to z/OS V1R12, you can download the new version to the earlier systems and replace the previous Tools and Toys version of **tsocmd** if appropriate. If this action is taken, you will need to clean up the authorized load library as described in Step 2 prior to downloading the new tool.

## 2.7 BCP migration actions after the first IPL

These are migration actions that you can perform only after you have IPLed z/OS V1R12. You need a running z/OS V1R12 system to perform these actions. An example is issuing RACF commands related to new functions. Note that the term “first IPL” does not mean that you have to perform these actions after the very first IPL, but rather that you need z/OS V1R12 to be active to perform the task. You might perform the task quite a while after the first IPL.

### 2.7.1 Hardware instrumentation services (HIS)

Hardware instrumentation services (HIS) is a function that collects hardware event data for processors in SMF records type 113, subtype 2, as well as UNIX System Services output files. You can only use HIS for IBM System z10™ or later machines. In z/OS V1R12 (and in z/OS V1R11 and z/OS V1R10 with the PTF for APAR OA30486 installed), functionality is added to the HIS component that causes changes in the output filename formats produced by HIS (.CNT, .SMP, and .MAP), as well as introducing additional lines to the .CNT file, possibly causing incompatibilities. In addition, an increase in SMF Type 113 records might be noticed.

Any tools that programmatically open the HIS output files (.CNT, .MAP, or .SMP), and any tools that programmatically analyze the HIS output .CNT file, should be analyzed and updated to accommodate the new formats.

## Migration action

The following items are what you should look for after the first IPL of z/OS V1R12:

- ▶ The new filename output format is an indication that the support is installed.
- ▶ The first line of the .CNT file indicates output version 2.
- ▶ New output message HIS032I STATE CHANGE DETECTED ACTION=action.

An additional line in the output of the DISPLAY HIS command, which describes the action that should be taken should a state change event occur (specified by the operator at the start of a collection run). Following are actions that should be taken:

- ▶ If programmatically opening files, ensure the new output file format is handled.
- ▶ If programmatically parsing the .CNT file, ensure to check the VERSION identifier in the header. If the identifier is VERSION 2, be prepared for the new STATECHANGE line.

## 2.7.2 BCP program management Binder

The following migration actions can be taken after the first IPL of z/OS V1R12.

### Detecting program modules with multiple INITIAL LOAD segments

This required action has the following conditions:

- ▶ Only users or ISVs that produce program object programs, which might have multiple INITIAL LOAD segments, need to take action if all of the items listed below are true:
  - Program is required to reside in a program object.
  - Program has multiple segments.
  - Program has multiple initial load classes.
  - Program has mixed RMODEs.
  - Program is link-edited with the RMODE option to override the Binder default.

**Note:** In most cases, even if a program has multiple segments containing INITIAL LOAD classes, no action is required.

### Binder RMODE option

Prior to z/OS V1R12, the binder RMODE option only applied to the first module segment, which contains some but possibly not all the initial load classes. Subsequent segments containing other initial load classes were not affected by the RMODE option, and thus the binder determined the RMODE based on the attributes of the classes contained therein.

Therefore, beginning with z/OS V1R12, the binder RMODE option applies to all initial load classes by default. Thus all segments containing initial load classes are affected. This new behavior takes effect only when the RMODE binder option is specified. Also the RMODE option has been expanded so that either the new or previous behavior can be explicitly requested.

### Migration action

To determine whether you might be affected by this change, you can run the AMBLIST service aid against your program objects and examine the output. However, if you do not explicitly specify the RMODE option when you bind the program, it is not affected even if the following two conditions are true.

Run AMBLIST using the LISTLOAD control statement. Using the OUTPUT=MAP option results in a smaller amount of output that still contains the pertinent information. The following two conditions must both be true for the program to be affected by the new RMODE option behavior:

1. In the Module Summary, only programs that are program objects and are PO FORMAT: 3 or higher are affected.
2. In the SEGMENT MAP, only programs for which there is the class entry SEGMENT 2, and TYPE INITIAL, are affected.

**Note:** The RMODE differs for the SEGMENT 1, INITIAL load classes, and the SEGMENT 2, INITIAL load classes.

Given that the two conditions above are true, there are then generally three possible situations:

- ▶ If the RMODE option is not being specified, there is no need to do anything, because the behavior is identical in z/OS V1R12.
- ▶ If you specified RMODE=ANY, expected all segments to be RMODE=31, but find that one segment is RMODE=24, part of the program is using below-the-line storage. In most cases, this is not desirable and the new Binder behavior will remedy the situation. It is possible that you rely on having part of the program use below-the-line storage. You can use RMODE(ANY,COMPAT) to revert to the pre-z/OS V1R12 behavior.
- ▶ If you specified RMODE=24, expected all segments to be RMODE=24, but find that one segment is RMODE=31, part of the program is using above-the-line storage. In most cases, this is desirable and the new Binder behavior could cause a problem by causing the program to use more below-the-line storage. It is also possible that you rely on having part of the program use above-the-line storage. You can use RMODE(24,COMPAT) to revert to the pre-z/OS V1R12 behavior.

### 2.7.3 Cryptographic Services

This action is required if you do not want the PKI Services daily maintenance task to run at midnight. The Cryptographic Services element has a migration action to perform after first IPL of z/OS V1R12, as follows:

For PKI Services, change the time at which the daily maintenance task runs. With z/OS V1R12, the PKI Services daily maintenance task runs when you start PKI Services and daily at midnight by default. If you do not want to use the default, update as required.

**Note:** Before z/OS V1R12, PKI Services ran a daily maintenance task when you started the PKI Services daemon, and every twenty-four hours after that. Starting with z/OS V1R12, by default the task runs when you start the PKI Services daemon, and daily at midnight. If running this task at midnight causes problems, for example performance problems because you have other tasks scheduled to run at midnight, you can change the time at which the PKI Services daily maintenance task runs.

#### Migration action

Change the value of the MaintRunTime variable in the PKI Services configuration file to specify the time at which you want the daily maintenance task to run.

### 2.7.4 Unicode Services

Use Unicode Services to create conversion tables. This is required if you use the iconv() family of functions to test to a “known conversion result” and experience test case failures. Also, if you use custom conversion tables replacing those listed in either ucmapt.lst or genx1t.lst.

In z/OS V1R12, the C/C++ run-time library will no longer include any ucmmap source code or genxlt source code for character conversions now being performed by Unicode Services.

### Migration action

If you use customized conversion tables, you should now generate custom Unicode Services conversion tables.

If you use the iconv() family of functions testing to a “known conversion result” and experience test case failures, you need to update your expected results to the new conversion results.

If you want to create custom conversion tables involving any of the CCSIDs related to the conversion table source no longer being shipped, you should now generate custom Unicode Services conversion tables instead of custom Language Environment conversion tables.

**Note:** The installation prefix.SCEEUMAP data set will no longer be shipped. The /usr/lib/nls/locale/ucmap HFS directory will no longer be shipped.

## 2.8 JES2, JES3, and SDSF installation considerations

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.

You could migrate to the JES2, JES3, and SDSF that comes with z/OS V1R12 at the same time you migrate to the rest of z/OS V1R12, or as soon as possible thereafter. In this way, you benefit directly from the new functions in the z/OS V1R12 level of JES2, JES3, and SDSF and enable other elements and features to benefit from this level.

However, because such a migration is not always practical, certain prior levels of JES2 and JES3 are supported in z/OS V1R12 so that you can stage your migration to z/OS V1R12 (that is, migrate your JES2 or JES3 as shown in Figure 2-3, and SDSF later).

BCP Release	JES2 Release allowed JES3 Release allowed	SDSF Release allowed
R12	R10	R10
R12	R11	R11
R12	R12	R12

Figure 2-3 Allowable BCP, JES2, JES3, and SDSF combinations

**Note:** With z/OS V1R10 and later releases, SDSF is supported with JES3.

### JES2 migration actions

There is a JES2 migration action to perform before installing z/OS V1R12, as follows:

- ▶ Update the code to remove references to PDBLENG from installation exits. Determine whether your installation has exits that reference PDBLENG and update to use the field PDBSIZE.
- ▶ Ensure that calls to JES property information services SSI can support information returned for multiple members.

### **SDSF migration actions**

The following actions need to be performed after the first IPL of z/OS V1R12:

- ▶ Set a default for the Initiators panel. With z/OS V1R12, SDSF displays WLM initiators along with JES-managed initiators on the Initiator (INIT) panel. To change the default, update the ISFPRMxx parmlib member.
- ▶ Set the format of device names on the Printers panel. With z/OS V1R12, SDSF displays printer names in longer format on the Printer panel. To change the default to shorter names, update the ISFPRMxx parmlib member.
- ▶ Set the view of OPERLOG. With z/OS V1R12, SDSF displays only active log data for OPERLOG. To change the default to also display inactive log data, update the ISFPRMxx parmlib member.

## **2.9 Elements withdrawn from z/OS V1R12**

The Managed System Infrastructure for Setup (msys for Setup) FMIDs HMSI707 and HMSI737 are not included in z/OS V1R12. z/OS V1R11 is the last release to include these FMIDs.

- ▶ If you plan to install the CBPDO deliverable for z/OS V1R12, you must delete the msys for Setup element from the target system after z/OS V1R12 is installed. A sample job, CLNOS390, is provided to delete the msys for Setup FMIDs from the target system.
- ▶ The z/OS V1R12 Program Directory provides instructions for running the sample job CLNOS390.
- ▶ The obsolete libraries, paths and associated DDDEFs for the msys for Setup element must be removed from the target system after z/OS V1R12 has been installed and the msys for Setup element has been deleted.
- ▶ *z/OS Migration*, GA22-7499 identifies the obsolete libraries, paths and DDDEFs that must be deleted.

## **2.10 IBM zEnterprise 196 (z196)**

IBM announced the next generation of our leading workload optimization and consolidation system, the IBM zEnterprise System, on July 22nd. The IBM zEnterprise 196 (z196) platform is designed with performance and capacity for growth and large-scale consolidation, improved security, resiliency and availability while helping you to lower both risk and cost. As environmental concerns raise the focus on energy consumption, z196 offers new efficiencies enabling dramatic reduction of energy usage and floor space when consolidating workloads from distributed servers. For organizations looking to build green data centres, optional water cooling and high-voltage DC power allow a bold step into the future of cooler computing without changing the footprint. The z196 will deliver unique specialty engines to help deliver greater efficiencies and expand the use of the mainframe for a broader set of applications, while helping to lower the total cost of ownership (TCO).

## 2.10.1 IBM System z Discovery and Auto-Configuration (zDAC)

A new IBM System z Discovery and Auto-Configuration (zDAC), available with z/OS V1R12 on IBM zEnterprise 196 (z196) servers, can help simplify the configuration and reduce the complexity and setup time for new and changed disk and tape I/O configuration changes. zDAC is designed to automatically perform a number of I/O configuration definition tasks for new and changed disk and tape control units connected to a switch or director when attached to FICON channel. zDAC can save time by discovering new and changed devices for you, and suggesting configurations aligned with best practices for availability and with the I/O policies that you set. For example, with zDAC, adding controllers to an existing I/O configuration can take as little as a few minutes.

zDAC is designed to perform discovery for a single system or for all the systems in a sysplex that support the function. It proposes new configurations that incorporate the current contents of your I/O definition file (IODF) with additions for new and changed controllers and devices based on a policy you define in the Hardware Configuration Dialog (HCD), which can include preferences for availability and bandwidth, including parallel access volume (PAV), HyperPAV, and DCM specifications, and control unit and device number ranges. These capabilities are integrated with HCD and z/OS Hardware Configuration Manager (HCM) functions.

### Discovery and Auto-Configuration (zDAC) function

zDAC is designed to perform discovery for a single system or for all the systems in a sysplex. These capabilities are integrated with HCD and z/OS Hardware Configuration Manager (HCM). When new controllers are added to an I/O configuration or changes are made to existing controllers, the system is designed to discover them, and propose configuration changes based on a policy you define in the Hardware Configuration Dialog (HCD). Your policy can include preferences for availability and bandwidth including parallel access volume (PAV), HyperPAV, DCM specifications, and preferred control unit and device number ranges.

## 2.10.2 Three subchannel sets for zEnterprise 196 servers

z/OS V1R12 supports three subchannel sets on IBM zEnterprise 196 (z196) servers. This helps relieve subchannel constraints, and can allow you to define larger I/O configurations that include large numbers of Metro Mirror (PPRC) secondaries and Parallel Access Volume (PAV) aliases. As with the prior support for two subchannel sets, you can define base devices, aliases, and secondaries in the first subchannel set (set zero), and define only aliases and secondaries in subchannel sets one and two. All three subchannel sets support ESCON®, FICON, and zHPF protocols. This support is also available on z/OS V1R10 and z/OS V1R11 with the PTF for APAR OA30677.

z/OS V1R12 on zEnterprise 196 servers with Coupling Facility Control Code (CFCC) Level 17 supports up to 2047 structures per Coupling Facility (CF) image, up from the prior limit of 1023. This allows you to define a larger number of data sharing groups, which can help when a large number of structures must be defined, such as to support SAP configurations or to enable large Parallel Sysplex configurations to be merged. This function requires the PTF for APAR OA32807; PTFs are also available for z/OS V1R10 and z/OS V1R11.

z/OS V1R12 on zEnterprise 196 servers with CFCC Level 17 also supports more connectors to list and lock structures. XES and CFCC already support 255 connectors to cache structures. With this new support XES also supports up to 247 connectors to a lock structure, 127 connectors to a serialized list structure, and 255 connectors to an unserialized list structure. This support requires the PTF for APAR OA32807; PTFs are also available for z/OS V1R10 and z/OS V1R11.

z/OS V1R12, with z196 servers and Coupling Facility control code (CFCC) Level 17, is designed to capture Coupling Facility (CF) data nondisruptively in some circumstances, allowing the CF to continue operating. This new function is intended to help improve Parallel Sysplex availability when it is necessary to capture CF data.

### 2.10.3 IBM zEnterprise 196 server performance

The IBM zEnterprise 196 server adds additional scalability and performance capabilities for your z/OS environment, as follows:

- ▶ The new 96-way core design (with 80 cores that are client configurable) delivers massive scalability for secure data serving and transaction processing for large-scale businesses.
- ▶ The performance of a z196 (2817) processor is expected to be 1.3 to 1.5 times the performance of a z10 EC (2097) based on workload and model.
- ▶ The largest z196 (2817-780) is expected to exceed 1.6 times the capacity of the largest z10 (2097-764).
- ▶ It has up to twice the available real memory, 3 terabytes (TB) per server (with up to 1 TB real memory per LPAR) compared to the z10 EC Model E64.
- ▶ New quad-core 5.2 GHz processor chips, with more than 100 new instructions to enable improved code efficiency, are also designed to help improve the execution of Java and processor intensive workloads. For example, it is anticipated that z/OS and zEnterprise 196 servers can provide a significant performance improvement for Java workloads.

#### Storage capacity

In addition, you now have the ability to extend the amount of addressable storage capacity to help facilitate storage growth with the introduction of:

- ▶ A third subchannel set
- ▶ An additional 64 K subchannels

This should help complement other functions such as “large” or extended address volumes (EAV) and HyperPAV. This may also help facilitate consistent device address definitions, simplifying addressing schemes for congruous devices. The first subchannel set (SS0) allows definitions of any type of device (such as bases, aliases, secondaries, and devices that do not implement the concept of associated aliases or secondaries).

The second and third subchannel sets (SS1 and SS2) can now both be used for disk alias devices (both primary and secondary devices) and/or Metro Mirror secondary devices only. The third subchannel set supports ESCON, FICON, and zHPF protocols, and is supported by z/OS V1AR12. This support is also available for z/OS V1R10 and z/OS V1R11 with PTFs.

### 2.10.4 Parallel Sysplex environments

zEnterprise 196 provides many enhancements to a z/OS Parallel Sysplex environment, as follows:

- ▶ Connectivity improvements with up to 80 coupling links

zEnterprise 196 increases the number of external coupling links allowed from 64 to 80. This allows the full configuration of 32 PSIFB links and 48 ISC-3 links to be used. In addition, you can also configure up to 32 (internal) IC links for coupling between images defined on the same server. Having more coupling links is important to provide sufficient coupling connectivity for larger single Parallel Sysplexes, as well as for configurations where the same server hosts multiple Parallel Sysplexes and Coupling Facility images.



- ▶ **Parallel Sysplexes and Coupling Facility images**  
z/OS V1R12 on zEnterprise 196 servers with CFCC Level 17 also supports more connectors to list and lock structures. XES and CFCC already support 255 connectors to cache structures. With this new support XES also supports up to 247 connectors to a lock structure, 127 connectors to a serialized list structure, and 255 connectors to an unserialized list structure. This support requires the PTF for APAR OA32807; PTFs are also available for z/OS V1R10 and z/OS V1R11.
- ▶ **Connectivity improvements with 128 coupling CHPIDs per server**  
To support larger Parallel Sysplexes with ever-increasing amounts of data sharing traffic to the Coupling Facility, the throughput and capacity of more coupling CHPIDs is also required. With z196, the number of coupling CHPIDs per server has been increased from 64 to 128. Since IFB links allow for multiple (logical) CHPIDs over the same (physical) link, this can also allow for larger Parallel Sysplexes without requiring more coupling link hardware.
- ▶ **Connectivity improvements with up to 2047 structures**  
CFCC Level 17 increases the number of structures that can be allocated in a CFCC image from 1023 to 2047. Allowing more CF structures to be defined and used in a sysplex permits more discrete data sharing groups to operate concurrently, and can help environments requiring many structures to be defined, such as to support SAP or service providers. z196 and CFCC Level 17 also provide improved serviceability of Coupling Facilities with enhanced data collection and triggering of nondisruptive CF dumps.

## 2.10.5 Networking performance

z/OS VR12 on z196 servers is designed to provide improved networking performance with OSA-Express-3 in QDIO mode with inbound workload queuing (IWQ). IWQ creates multiple input queues and allows OSA to differentiate workloads “off the wire” and then assign work to specific input queues to z/OS. With each input queue representing a unique type of workload, each having unique service and processing requirements, the IWQ function allows z/OS to use appropriate processing resources for each input queue.

This approach allows multiple concurrent z/OS processing threads to process each unique input queue, avoiding traditional resource contention. In a heavily mixed workload environment, this “off the wire” network traffic separation provided by OSAExpress3 IWQ reduces the conventional z/OS processing required to identify and separate unique workloads, which is expected to result in improved overall system performance and scalability.

- ▶ It is anticipated that networking performance for interactive workloads can be improved significantly, depending on amount of data being transferred, presence of bulk-data traffic in the mix, and whether communication is z/OS to z/OS, or z/OS to distributed system. For example, interactive networking response time improvements of 30-50% on a System z10 model 2097-E64 were recorded.
- ▶ It is also anticipated that bulk-data (streaming) workloads can also benefit with OSA Express-3 IWQ and its ability to reduce the amount of costly network retransmissions (by reducing the incidence of out-of-order packets). A streamlined CommServer execution path for Sysplex Distributor over IWQ is expected to improve performance for sysplex-distributed traffic as well.
- ▶ When inbound workload queueing is enabled for a QDIO interface, inbound streaming bulk data is processed on an ancillary input queue (AIQ). This function is expected to improve throughput while reducing processor consumption for inbound streaming bulk data.

- ▶ When inbound workload queuing is enabled for a QDIO interface, inbound Sysplex Distributor traffic is processed on an AIQ. This function is expected to improve performance for inbound Sysplex Distributor traffic that is routed to a target stack.
- ▶ IWQ is supported on z/OS V1R12 and is exclusive to OSA-Express3 on z196 and System z10, where CHPID types OSM and OSX are exclusive to z196.

### OSA-Express3

In addition to IWQ, OSA-Express3 also introduces the capability for the operating system to directly query and display the current OSA configuration information (similar to OSA/SF). z/OS exploits this new OSA capability by introducing the new TCP/IP operator command DISPLAY OSAINFO. This command allows the operator to monitor and verify the current OSA configuration, which can help you improve the overall management, serviceability, and usability of OSA-Express3. The D OSAINFO command requires OSA-Express3 CHPID types OSD, OSM, and OSX, and z/OS V1R12.

## 2.10.6 IBM zEnterprise Unified Resource Manager

The new zEnterprise System also provides end-to-end workload monitoring and other systems management capabilities for System z ensembles, through the new IBM zEnterprise Unified Resource Manager. System z ensembles are collections of one or more zEnterprise System nodes in which each node comprises a z196 server and its optionally attached IBM zEnterprise BladeCenter® Extension (zBX) Model 002, as shown in Figure 2-4 on page 68.

An ensemble can consist of a single z196 server running z/ OS images and z/VM® hosting Linux for System z images but without a zBX attached, or it can consist of from 1 to 8 z196 servers, at least one of which has a zBX attached. The resources of a zEnterprise System ensemble are managed and virtualized as a single pool, integrating system and workload management across the multisystem, multitier, multi-architecture environment. The zEnterprise Unified Resource Manager uses the intranode management network (INMN) for communication.



Figure 2-4 zEnterprise 196 and zEnterprise BladeCenter Extension

## **Guest Platform Management Provider (GPMP)**

z/OS integrates with this new management environment. A new agent, Guest Platform Management Provider (GPMP), in z/OS V1R12 communicates with z/OS WLM and provides basic data (such as system resource utilization, system delays, and paging delays) back to the zEnterprise Unified Resource Manager over the INMN network. The zEnterprise Unified Resource Manager can add additional workload relationships from the ensemble components to your z/OS workload; for example, linking a transaction that started on the zBX back to DB2 on z/OS data.

## **XL C/C++ support**

z/OS V1R12 XL C/C++ also exploits new instructions in the IBM zEnterprise System 196 server. The z/OS V1R12 XL C/C++ compiler provides new ARCHITECTURE(9) and TUNE(9) options to help you exploit new instructions that are available on z196 servers. These options are designed to provide better performing applications tuned for the new server. Additional optimization and tuning have been made to improve the floating-point performance. These changes can improve the performance of generated code without the need for changes to the source code. A performance improvement of over 11% was observed using compute-intensive integer workload code generated by the z/OS V1R12 XL C/C++ compiler with high optimization when compared to code generated using the z/OS V1R11 XL C/C++ compiler. Performance improvements are based on internal IBM lab measurements using the ILP32, XPLINK, ARCH(9), TUNE(9), HGPR, O3, HOT, and IPA(LEVEL(2)) with PDF compiler options. Performance results for specific applications will vary; some factors affecting performance are the source code and the compiler options specified.





# HiperDispatch

z/OS workload management and dispatching have been enhanced to take advantage of the System z10 hardware design. The IBM z10 processor supports a new mode of dispatching called HiperDispatch (HD), which increases the system capacity by up to 10%. The amount of improvement varies according to the system configuration and workload.

HiperDispatch was introduced with z/OS V1R10 and rolled down to JBB772S, HBB7730, and HBB7740. This chapter discusses the following:

- ▶ HiperDispatch overview
- ▶ Activating HiperDispatch
- ▶ Monitoring HiperDispatch
- ▶ HiperDispatch enhancements in z/OS V1R10
- ▶ HiperDispatch enhancements in z/OS V1R11
- ▶ HiperDispatch enhancements in z/OS V1R12

## 3.1 HiperDispatch overview

HiperDispatch is a combination of hardware, Hypervisor, and z/OS that increases system capacity. It does this by increasing the probability of cache hits when executing z/OS instructions. Each processor has its own level 1 (L1) cache. This is the best place to find data because it requires the fewest machine cycles to access the data. processors are grouped at the hardware level in books.

All processors in the same book share a common level 2 (L2) cache. This is the second best place to access data. A processor can also access the L2 cache of other books, but this requires more machine cycles. The difference in machine cycles required to access a piece of data found in the L1 cache versus the same book L2 cache is relatively small. However, there is a significant difference in the number of machine cycles to access a piece of data in the same book L2 cache versus a different book L2 cache. To optimize for same book L2 cache hits, a unit of work must run on a subset of the available processors in the same book.

### 3.1.1 Without HiperDispatch

To describe the interaction between z/OS and Hypervisor we use a hypothetical example of a z10 processor with eight physical processors. The z10 is running two LPARs with the same weight and eight logical processors. Each LPAR will receive four physical processors at execution time which will be distributed across the eight logical processors defined to each LPAR, as shown in Figure 3-1.

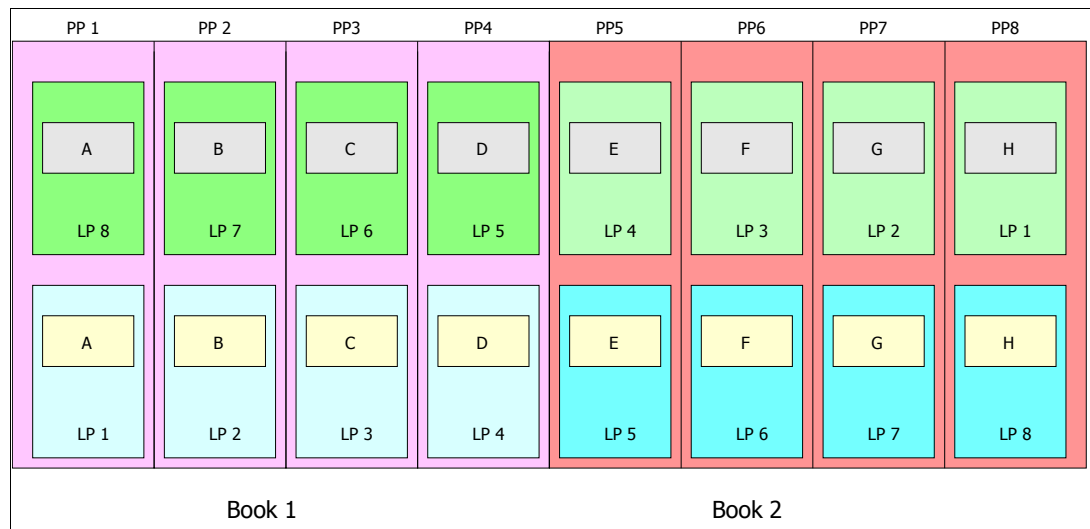


Figure 3-1 z10 without HiperDispatch

Without HiperDispatch, the Hypervisor can dispatch any logical processor on any physical processor and z/OS can dispatch any thread on any logical processor. This results in threads being dispatched randomly across physical processors. This has the effect of randomizing which physical processor is chosen, reducing L1 and L2 cache hits, which reduces the throughput of the system.

### 3.1.2 With HiperDispatch

HiperDispatch optimizes for same book L2 cache hits by dispatching threads intelligently on physical processors. First, Hipervisor needs to dispatch a given logical processor on the

same physical processor. Then z/OS needs to dispatch a thread among a relatively static collection of processors in the same book to increase the probability of a same book L2 cache hit. Since a thread is going to be dispatched among a small group of processors, the probability of an L1 cache hit also increases.

Using our hypothetical example in Figure 3-2, if both LPARs consume their full LPAR weight, each LPAR will still receive four physical processors worth of execution time. With HiperDispatch=YES, the four physical processors worth of execution time is distributed amongst a subset of the logical processors defined to each LPAR. Normally, each LPAR uses four of its logical processors to process its work, as shown in Figure 3-2.

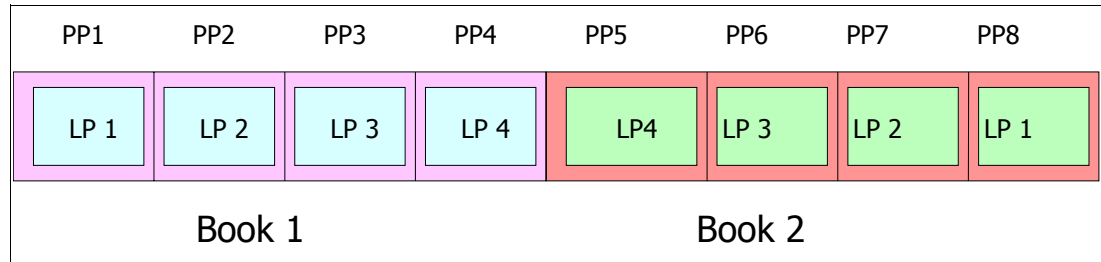


Figure 3-2 z10 with HiperDispatch and two LPARs of equal weight

Since HiperDispatch optimizes the same book L2 cache, Hipervisor assigns the four logical processors 1, 2, 3, and 4 from the LPAR to physical processors 1, 2, 3, and 4 in book 1 and the four logical processors 1, 2, 3, and 4 from the LPAR to physical processors 8, 7, 6, and 5 in book 2. These logical processors are referred to as vertical highs because one logical processor is mapped to exactly one physical processor. When more than one logical processor is mapped to the same physical processor, that logical processor is referred to as a vertical medium. Multiple vertical mediums share the processing power provided by one physical processor.

There are four logical processors in each LPAR that are unaccounted for. Those remaining four logical processors are not mapped to any physical processors. These logical processors are referred to as vertical lows or discretionary processors and are normally in a parked state. When the z10 is busy and starts to approach 100% utilization, the vertical lows and then vertical mediums are parked. A parked processor is online but will not dispatch work or process I/O interrupts. An LPAR's vertical lows float on top of a different LPAR's vertical highs or mediums so that an LPAR that needs more capacity can receive additional capacity from a different LPAR that is not consuming its full weight. An LPAR will only expand into its vertical lows when both of the following conditions are met:

- ▶ An LPAR is consuming its full LPAR weight.
- ▶ A different LPAR is not consuming its full weight.

When both conditions are met, the LPAR consuming its full weight can unpark one or more of its vertical lows to use another LPAR's unused capacity. Once a vertical low is unparked it will dispatch work on behalf of that LPAR. The vertical low will be parked once the overworked LPAR no longer needs the extra capacity or the other LPAR starts consuming its full weight.

### Grouping processors into affinity nodes

z/OS groups logical processors into entities called *affinity nodes*. Processors assigned to an affinity node tend to be in the same book to increase the probability of an L2 cache hit. All processors in an affinity node have the same processor type (CP, zAAP, or zIIP). Each affinity node has its own WUQ and all processors assigned to that affinity node dispatch work from that WUQ. HiperDispatch optimizes cache hits by ensuring that threads are redispached on the same affinity node.

To ensure that the responsiveness of high priority work is not impacted with HiperDispatch, high priority work is assigned to a new WUQ called the high performance WUQ (HPWUQ). All standard CPs dispatch work from the HPWUQ before dispatching work on their affinity WUQ. The HPWUQ contains work element blocks (WEBs) for high priority work that include:

- ▶ WEBs with dispatching priority 255. Address spaces with service class SYSTEM are assigned dispatching priority 255.
- ▶ SRB WEBs with dispatching priority 254. Address spaces with service class SYSSTC are assigned dispatching priority 254.
- ▶ Lock promotion WEBs. These have dispatching priority 255.

HiperDispatch responds to utilization spikes within an affinity node by assigning processors from another affinity node to perform work from the busy affinity node's WUQ. In addition, WLM gathers statistics every two seconds to distribute the workload evenly across the affinity nodes.

There is a significant improvement in the capacity of the system (0%-10% depending on system configuration and the workload) by optimizing for level 2 cache hits with HiperDispatch. The more physical processors a z10 has or the larger the overcommit ratio between logical processors and physical processors, the larger the capacity gain from HiperDispatch.

## 3.2 Activating HiperDispatch

HiperDispatch is only supported by the IBM z10 processor, device type 2097. It is part of the z/OS V1R10 base code and available for JBB772S, HBB7730, and HBB7740 via APARs:

- ▶ OA20633 and OA23333 for supervisor
- ▶ OA20418 for WLM
- ▶ OA12774 for RMF

### **IEAOPTxx parmlib member**

To activate HiperDispatch, a new keyword in IEAOPTxx is used:

- ▶ HiperDispatch = YES
- ▶ The default is HiperDispatch = NO

HiperDispatch can be activated at IPL by specifying HiperDispatch = YES in the IEAOPTxx parmlib member used during IPL. It can also be activated and deactivated dynamically by updating IEAOPTxx or creating a new IEAOPTxx specifying HiperDispatch = YES or NO and issuing the following MVS command:

```
SET OPT=xx
```

### **HiperDispatch messages**

When HiperDispatch is activated, the following message is issued:

```
IRA860I HIPERDISPATCH MODE IS NOW ACTIVE
```

When HiperDispatch is deactivated, the following message is issued:

```
IRA861I HIPERDISPATCH MODE IS NOW INACTIVE
```



## 3.3 Monitoring HiperDispatch

HiperDispatch has implications for workload management (WLM) and help processing. There are also changes in RMF reports to display HiperDispatch information.

### 3.3.1 WLM considerations

With HiperDispatch the prioritization of workloads via WLM policy definitions becomes more important because access to processors changes. To optimize cache hits a work unit has access to a smaller number of processors, which increases the potential for queuing delays.

As HiperDispatch changes how work is dispatched among the processors, additional attention and review of the WLM policy may be needed to ensure proper workflow through the system. With HiperDispatch it is important that critical work, highly interactive work, and processor intensive work is prioritized appropriately.

Prior to z/OS V1R10, only the Master address space and WLM are automatically assigned with service class SYSTEM and cannot be assigned a different service class. In z/OS V1R10 there are extra system addresses that are classified into SYSTEM and cannot be changed. They are XCFAS, GRS, CONSOLE, IEFSCHAS, IXGLOGR, SMF, CATALOG, SMSPDSE, and SMSPDSE1.

With HiperDispatch, work for address spaces with service class SYSTEM runs on the High Performance WUQ (HPWUQ), ensuring high access to processors.

### 3.3.2 RMF reports

With HiperDispatch=Yes different processor utilizations are seen in the RMF CPU Activity report depending on whether a processor is a vertical high, vertical medium, or vertical low.

With the RMF HiperDispatch APAR OA12774, the RMF CPU Activity report has a new column for PARKED time% and is enhanced to indicate the high, medium, or low share via the LOGICAL PROCESSOR SHARE% column. In addition, changes are introduced with APAR OA24074, which adds an indication whether HiperDispatch is active on the processor type and model information line. OA24074 also changes the calculation of the MVS view of processor utilization to take into account that logical processors can be parked.

With APAR OA24074, the calculation is:

$$\text{MVS UTIL(\%)} = \frac{\text{Online Time} - (\text{Wait} + \text{Parked Time})}{\text{Online Time} - \text{Parked Time}} * 100$$

This also affects the AVG MVS UTIL(%) for all logical processors because the MVS UTIL(%) for each logical processor is weighted by the time being online and unparked. This effect becomes obvious with processors being parked partially during the interval. For example, an MVS UTIL of 100% for a processor parked 80% means the processor was unparked for 20% of the interval and busy the whole time it was unparked. The interval for this processor adds less to the overall average than an MVS UTIL of 100% for a processor that was not parked.

Figure 3-3 on page 76 shows an example of a processor activity for a system with HiperDispatch=Yes and OA24074 installed running on a z10. The logical processor share for the partition of 640.0% was allocated across five logical processors with a high share of 100%, two logical processors with a medium share of 70%, and one discretionary logical processor, CP 7, with a low share of 0%, which was parked 85.78% and thus unparked

14.22% of the online interval time. During this same interval, CP 7 was busy processing 13.84% of the time.

With HIPERDISPATCH=NO, the logical processor share would be 80% for each of the 8 logical processors. There are 12 vertical highs, 7 vertical mediums, and 14 vertical lows.

C P U A C T I V I T Y									
z/OS V1R10				SYSTEM ID S59				DATE 11/28/200	
RPT VERSION V1R10 RMF				TIME 16.45.00				CYCLE 1.000 SECONDS	
CPU 2097 MODEL 732 H/W MODEL E40 SEQUENCE CODE 0000000000DC6CE HIPERDISPATCH=YES									
---CPU--- ----- TIME % ----- LOG PROC --I/O INTERRUPTS--									
NUM	TYPE	ONLINE	LPAR BUSY	MVS BUSY	PARKED	SHARE %	RATE	% VIA TPI	
0	CP	100.00	96.33	97.34	0.00	100.0	5.80	48.75	
1	CP	100.00	95.96	97.07	0.00	100.0	4.59	55.30	
2	CP	100.00	95.79	96.84	0.00	100.0	5.10	55.18	
3	CP	100.00	95.46	96.68	0.00	100.0	2.40	53.75	
4	CP	100.00	95.08	96.41	0.00	100.0	8435	10.05	
5	CP	100.00	73.92	96.86	0.00	70.0	20.74	4.95	
6	CP	100.00	74.33	97.13	0.00	70.0	14.15	19.39	
7	CP	100.00	13.84	98.89	85.78	0.0	0.00	0.00	
TOTAL/AVERAGE			80.09	96.94		640.0		8488	10.14

Figure 3-3 RMF CPU Activity report

### 3.4 Help processing

Help processing occurs when an affinity node is overcommitted and the dispatcher determines that it needs help. This is done by assigning the WUQ for the overcommitted affinity node to another less busy processor. In HiperDispatch=NO all processors are candidates to give help, while in HiperDispatch=YES preference is given to processors in the same affinity node.

When a CP affinity node needs help:

- ▶ In HiperDispatch=NO, all CPs are candidates to give help.
- ▶ In HiperDispatch=YES, waiting CPs in the same affinity node are the first processors chosen for help. If there are no waiting CPs in the same affinity node, a good candidate CP with the same book is chosen. If there are no good candidates in the same book and the affinity node needs help badly enough, a CP in a different book can be chosen for help.

When a zAAP affinity node needs help:

- ▶ In HiperDispatch=NO, help from another zAAP is preferred. If all other zAAP processors are busy, a CP is chosen if IFAHONORPRIORITY=YES.
- ▶ In HiperDispatch=YES, waiting zAAPs in the same affinity node are the first processors chosen for help. If all the other zAAP processors in the same affinity node are busy, help can be provided by a zAAP in a different affinity node or a CP if IFAHONORPRIORITY=YES.

When a zIIP affinity node needs help:

- ▶ In HiperDispatch=NO, help from another zIIP is preferred. If all other zIIP processors are busy, a CP is chosen if IFAHONORPRIORITY=YES.

- ▶ In HiperDispatch=YES, waiting zIIPs in the same affinity node are the first processors chosen for help. If all the other zIIP processors in the same affinity node are busy, help can be provided by a zIIP in a different affinity node or a CP if IFAHONORPRIORITY=YES.

### 3.4.1 Alternate wait management

The alternate wait management (AWMT) and honor priority values specified via the IEAOPTxx parmlib member affect help processing for both HiperDispatch=YES and HiperDispatch =NO.

The keywords in the IEAOPTxx parmlib member are:

- ▶ CCCAWMT
  - Alternate Wait Management (AWM) value for normal CPs
  - For HiperDispatch = NO the valid range is 1-1,000,000 microseconds. Specifying CCCAWMT >= 500000 disables AWM.
  - Default for HiperDispatch = NO is 12000 microseconds.
  - For HiperDispatch =YES the valid range is 1600-3200 microseconds. Specifying a value outside that range will result in 3200 microseconds being assigned.
  - Default for HiperDispatch = YES is 3200 microseconds.

**Note:** For a dedicated LPAR, AWM is always inactive. For a shared LPAR, AWM is always active with HiperDispatch=Yes. For HiperDispatch=No, AWM can be disabled by specifying CCCAWMT >= 500000.

- ▶ ZAAPAWMT
  - Alternate Wait Management (AWMT) value for zAAP processors.
  - For HiperDispatch = NO the valid range is 1-499999 microseconds.
  - Default for HiperDispatch = NO is 12000 microseconds.
  - For HiperDispatch = YES the valid range is 1600-3200 microseconds. Specifying a value outside that range will result in 3200 microseconds being assigned.
  - Default for HiperDispatch =YES is 3200 microseconds.

**Note:** The valid range for ZAAPAWMT has changed from 1600 - 3200 to 1600 - 499999 microseconds with APAR OA26789. See *OA26789 - Improvements to PARK processing and AWMT* for more details.

- ▶ ZIIPAWMT
  - Alternate Wait Management (AWMT) value for zIIP processors.
  - For HiperDispatch = NO the valid range is 1-499999 microseconds.
  - Default for HiperDispatch = NO is 12000 microseconds.
  - For HiperDispatch = YES the valid range is 1600-3200 microseconds. Specifying a value outside that range will result in 3200 microseconds being assigned.
  - Default for HiperDispatch =YES is 3200 microseconds.

**Note:** The valid range for ZIPPAWMT has changed from 1600 - 3200 to 1600 - 499999 microseconds with APAR OA26789. See *OA26789 - Improvements to PARK processing and AWMT* for more details.

- ▶ IFAHONORPRIORITY=YES/NO
  - Specifying IFAHONORPRIORITY=YES means that normal CPs will help when zAAPs need help.
  - Specifying IFAHONORPRIORITY=NO means that normal CPs will not be eligible to help zAAPs.
- ▶ IIPHONORPRIORITY=YES/NO
  - Specifying IFAHONORPRIORITY=YES means that normal CPs will help when zIIPs need help.
  - Specifying IFAHONORPRIORITY=NO means that normal CPs will not be eligible to help zIIPs.

### Help for affinity nodes

An affinity node is deemed to need help if either:

- ▶ The interval used to check whether a processor needs help is also based on the AWMT values. This means the AWMT values affect how responsive help processing is to spikes in workload and also the criteria to determine whether a processor needs help.
- ▶ In HiperDispatch = YES, help is given for a certain number of dispatches by the chosen CPU while in HiperDispatch = NO the processor chosen to give help will continue to dispatch work from the WUQ needing help until the WUQ is empty.

When a system is IPLed specifying HiperDispatch = NO, the default value for CCCAWMT, ZAAPAWMT and ZIIPAWMT of 12000 micro-seconds is used. This value is in 64-bit TOD format where bit 51 corresponds to 1 micro-second. This means that these values can be converted to microseconds by dropping the last three digits (nibbles). X'02EE0000' = X'02EE0' microseconds = 12000 microseconds.

## 3.5 HiperDispatch enhancements to z/OS V1R10

There have been a number of important improvements to HiperDispatch for z/OS V1R10 and lower via new APARs that have been included with base z/OS V1R11. This section discusses the improvements made via these APARs. It is highly recommended that they be installed on z/OS V1R10 and lower systems.

### OA26789 - Improvements to PARK processing and AWMT

With HiperDispatch enabled, when an LPAR is very close to 100% busy and the CPC has white space available, a vertical low (discretionary) processor should be unparked to allow the LPAR to expand into the available processor capacity in the CPC. With the initial implementation of HiperDispatch vertical low (discretionary) processors were not necessarily unparked when an LPAR was very busy and there was white space in the CPC. This effect was mainly observed on low weighted LPARs with only a few or no vertical high CPs. This issue has been addressed by APAR OA26789, which has PTFs for JBB772S, HBB7730, HBB7740 and HBB7750.

In addition, with HiperDispatch enabled, the range accepted for ZAAPAWMT and ZIIPAWMT has been increased. ZAAPAWMT and ZIIPAWMT control how aggressive zAAPs and zIIPs

are in asking for help. When ZAAPAWMT or ZIIPAWMT is set too low, for LPARs with discretionary zAAPs or zIIPs, it can result in zAAPs or zIIPs asking CPs for too much help. When zAAPs or zIIPs get too much help from CPs, the zAAP or zIIP utilization can significantly decrease.

With OA26789 the valid range for zAAPAWMT and ZIIPAWMT with HiperDispatch enabled increases from 1600 - 3200 microseconds to 1600 - 499999 microseconds. This allows a higher setting for zAAPAWMT or ZIIPAWMT to be used as appropriate for LPARs with discretionary zAAPs or zIIPs available.

The APAR description of the problem and conclusion is shown in Figure 3-4.

**PROBLEM DESCRIPTION:**

1. The algorithm in IRABAADJ tries to adjust the CPU capacity of a Lpar as needed by unparking / parking of vertical low CPs. Prior to the changes with this APAR the algorithm aggressively tried to park vertical low CPs as soon as they appeared to be no longer useful. This led to situations that unparking did not take place even though there was demand in the Lpar and also available CPU capacity on the CEC. This effect was mainly observed with low weighted Lpars with only a few or no vertical high CPs.

2. The range of the IEAOPT parameters ZAAPAWMT and ZIIPAWMT must be extended to allow values greater than 3200 in case of HIPERDISPATCH=YES.

**PROBLEM CONCLUSION:**

This APAR changes the unpark / park algorithm in module IRABAADJ. A vertical low CP will be unparked as soon as MVS\_busy of the Lpar is getting above 95% and there is sufficient CEC capacity available.

Parking of vertical low CPs is done when the average efficiency of the low CPs is getting too low. The average low CPs efficiency is stored in variable VCM\_LparCapVlAvgEffect (IRABAVCM) and calculated as  $VCM\_LparCapVlAvgEffect = \frac{VCM\_LparCapUsedDiscr + 256}{VCM\_LparCapNonGuaran}$ .

- VCM\_LparCapUsedDiscr represents the used capacity on vertical low CPs.
- VCM\_LparCapNonGuaran represents the CP capacity provided by unparked vertical low CPs.

The algorithm to unpark / park vertical low CPs is performed for every CPU type (general CP, zAAPs, zIIPs) individually. Major changes were done in macro IRABAVCM substructure VCM\_LparCaps(PtIxDim) to make room for new variables replacing old no longer needed variables. All variables of VCM\_LparCaps(PtIxDim) will also be written to the SMF99 subtype 12 record. For this IRASMF99 was changed accordingly.

The range of the IEAOPT keywords ZAAPAWMT and ZIIPAWMT has been extended to 1600-499999 which corresponds to a timeframe from 1.6 to 500 milliseconds.

Figure 3-4 APAR OA26789

### **OA24920 - Performance degradation with AWMT off**

Alternate wait management (AWMT) is always enabled for shared LPARs with HiperDispatch enabled. AWMT is disabled on dedicated LPARs and can be disabled for shared LPARs with HiperDispatch disabled.

Systems running on dedicated LPARs or shared LPARs with AWM off can suffer a performance degradation when processors get parked and unparked due to CONFIG CPU ONLINE/OFFLINE activity. The park and unpark processing results in the processor's wait time slice in LCCAWTSC being set too high. Both dedicated LPARs and shared LPARs with AWM off require a short wait time slice so that processors wake up frequently enough to dispatch work.

The APAR description of the problem and conclusion is shown in Figure 3-5.

<p>PROBLEM DESCRIPTION: Dedicated LPAR(s) or shared LPAR(s) with alternate wait management (AWM) off have the CPU wait time slice (LCCAWTSC) too high for those environments. Depending on the workload running on the dedicated LPAR(s) or shared LPAR(s) with alternate wait management (AWM) off, the LPAR(s) may see a performance degradation.</p> <p>PROBLEM CONCLUSION: CPU wait time slice (LCCAWTSC) is set properly for dedicated LPARs and shared LPARs with alternate wait management (AWM) off.</p>
--

Figure 3-5 APAR OA24920

### APAR OA25825 - help for affinity nodes

The system maintains processor masks that are used to select a good candidate processor for help when an affinity node needs help. All processors in an affinity node have the same characteristics as all the other processors in that affinity node. When a processor is assigned to an affinity node, the processor masks would not necessarily be updated in a timely fashion before APAR OA25825 (Figure 3-6 on page 81). Prior to APAR OA25825, when a processor joined an affinity node, the processor masks would only be updated when the affinity node's state changed. Since taking a processor offline removes a processor from the processor masks, processing like IRD processor management, which brings processors offline and online, can result in corrupting the processor masks to a point where help processing is ineffective.

In HiperDispatch=NO, over many hours, days, or weeks, zAAPs/zIIPs could be unable to ask CPs for help. HiperDispatch=NO was the most likely environment to experience this problem because the interval used to determine the node's state was too long. If the node's state does not change after IPL, then processors configured online after IPL may be unable to be chosen for help.

The same problem also existed with HiperDispatch=YES, but it was less likely to occur because the interval used to determine the node's state was significantly shorter. This shorter interval makes it much more likely that the *need help* state of a given node will change and avoid the problem.

**PROBLEM DESCRIPTION:**

In a HiperDispatch=NO environment, when zAAP/zIIP processors are not able to handle all the zAAP/zIIP workload, HELP processing may not signal any standard processors for HELP for the zAAP/zIIP workload. Part of the HELP processing decision process uses the AWUQ High\_SigWait\_Mask to determine which processors are available to provide HELP. When the mask is binary zeroes, no standard CPUs are available to provide HELP for the zAAP/zIIP workload.

The problem is that there are no CPs in this mask even though there are standard CPs have a high significant wait. As a result, the zAAPs/zIIPs do not choose CPs for help due to a high significant wait.

The need help CPU masks are only refreshed when the CPUs assigned to a given node change state. All CPUs in a node have the same state as all the other CPUs in the same node. So the cumulative statistics of all CPUs in each node determine the state of each node. These statistics are maintained in internal CPU masks which are used in the need help logic. These internal CPU masks are only updated when the node's state changes.

For example, each node has a characteristic known as significant wait. There are 3 categories of significant wait: high significant wait, low significant wait, and no significant wait. When a CPU comes online, it is assigned to the appropriate node. That CPU does not get added into the appropriate significant wait mask until after the node's significant wait state changes.

This problem is most likely to appear while running with HiperDispatch=NO. With HD=NO, the interval used to determine the need help state of each node is too long. The longer interval makes it less likely the need help state of a given node will change. Should a node's state not change at all after IPL, then any CPUs configured online after IPL may not be added into the internal need help CPU masks.

This problem also exists in HiperDispatch=YES, but it is less likely to occur. In HD=YES, the interval examined to determine the need help state of each node is much smaller. This shorter interval makes it more likely the need help state of a given node will change.

**PROBLEM CONCLUSION:**

Refresh the internal need help CPU masks when CPUs come online and are assigned to a node.

Supporting APAR OA25841 will shorten the interval being used to determine the need help state of each node in HD=NO.

*Figure 3-6 APAR OA25825*

## 3.6 HiperDispatch enhancements with z/OS V1R11

Significant enhancements have been made to HiperDispatch in z/OS V1R11. This includes improvements in:

- ▶ How dispatching priority promotion is handled for a task that holds a LOCAL or CML lock
- ▶ Help processing

## Promotion of LOCAL or CML lock holder

There are significant RAS improvements in handling the promotion of a task holding a LOCAL or CML lock with HiperDispatch enabled. New fields have been added to the ASSB and ENCB to allow WLM to factor lock promote time into their calculations. The supervisor code to add this new function was updated via APAR OA27855 and the WLM code was updated via APAR OA27810.

The APAR description of the problem and conclusion is shown in Figure 3-7.

The HiperDispatch lock promotion algorithm has been enhanced to record the amount of time an enclave / address space has been promoted. WLM apar OA27810 makes use of these new times. See OA27810 for more information.

Figure 3-7 APAR OA27855

A problem with the original implementation was that a task running at a promoted dispatching priority could monopolize a processor. This was addressed via APAR OA28744 by limiting the amount of time a task will be promoted for and then forcing it to run at its normal dispatching priority.

The APAR description of the problem and conclusion is shown in Figure 3-8.

**PROBLEM DESCRIPTION:**  
To improve RAS in HD=YES, a unit of work which was promoted for holding a local / CML lock that was not released in a timely fashion must give up its lock promotion priority until all units of work between the lock promotion priority and the unit of work's base dispatch priority has completed.

**PROBLEM CONCLUSION:**  
In HD=YES, when a unit of work was promoted for holding a local / CML lock which was not released in a timely fashion, force the unit of work to run at its base dispatch priority until all units of work between the lock promotion priority and the unit of work's base dispatch priority has completed.

Figure 3-8 APAR OA28744

**Note:** APARs OA27810, OA27855, and OA28744 have been rolled down to HBB7730, HBB7740, and HBB7750.

## Improvements to help processing

The help algorithms have been enhanced so that CPs will not automatically give help to zAAPs and zIIPs before going into a wait. This prevents too much zAAP and zIIP processing from being offloaded to CPs when a zAAP or zIIP requests help from a CP.

## 3.7 HiperDispatch enhancements with z/OS V1R12

HiperDispatch enhancements have been made to support the z196 Servers. There are enhancements to support greater than 64 processors and also to take advantage of the z196 Server architecture.



This section discusses the new hardware feature for the z196 Servers and how HiperDispatch supports and exploits these features.

### 3.7.1 z196 Server features

The z196 Server increases throughput and performance by:

- ▶ Supporting up to 80 processors. The z10 only supports up to 64 processors and the z9 only supports up to 32 processors.
- ▶ Increased processor speed, 30% faster than z10
- ▶ A new cache level called chip level cache.

Table 3-1 compares the z10 cache architecture with the IBM zEnterprise 196 cache architecture.

*Table 3-1 z10 cache and memory latency*

Z10 Caches and Memory Latency	z196 Caches and Memory Latency
L1 (CPU, 64 K I/128 K D)	L1 (CPU 64 K I/ 64 K-128 K D)
L1.5 (CPU, 3 M, 16-97c)	L2 (CPU, 1.5 M, 14-48c)
-----N / A-----	L3 (Chip, 24 M, 50-110c)
Local L2 (Book, 48 M, 90-240c)	Local L4 (Book, 192 M, 150-310c)
Remote L2 (Book, 240-350c)	Remote L4 (Book, 330-490c)
Memory (970-1110c)	Memory (970-1180c)

Abbreviations in Table 3-1 are:

Key: I=I-Cache, D=D-Cache, c=Cycles

The IBM zEnterprise 196 has multiple processor cores within a chip.

- ▶ Maximum of 4 processor cores per chip.
- ▶ Some chips have 1 or 2 bad processor cores and so chips with 3 or 2 processors are possible.
- ▶ It is wise to optimize for chip level cache because it is a big cache with low latency.

### 3.7.2 HiperDispatch enhancements in z/OS V1R12 for the IBM zEnterprise 196

In z/OS V1R12 HiperDispatch has been enhanced to:

- ▶ Support greater than 64 processors.
- ▶ Optimize performance for chip level cache.
- ▶ Improve performance when running on an IBM zEnterprise 196.

#### Software dependencies

There are APARs for both the supervisor component and WLM to support HiperDispatch on the IBM zEnterprise 196:

- ▶ Supervisor APAR OA30476 is for z/OS V1R10, z/OS V1R11, and z/OS V1R12.
- ▶ WLM APAR OA30308 for z/OSV1R10 and z/OS V1R11 are included in the base for z/OS V1R12.

## Support for greater than 64 processors

On large LPARs with greater than 64 processors on the IBM zEnterprise 196 with HiperDispatch=YES can provide a 25% increase in capacity over HiperDispatch=NO for some workloads. In this environment it is not easy to switch into HiperDispatch=NO to avoid an HiperDispatch=YES issue because a lot more capacity is needed to contain the workload.

In view of this, systems running on LPARs with greater than 64 processors must run with HiperDispatch=YES, because of the following considerations:

- ▶ LPARs with 64 processors or less can run with HiperDispatch=YES or NO.
- ▶ LPARs with greater than 64 processors must run with HiperDispatch=YES.

LPARs with greater than 64 processors, specifying in the IEAOPTxx parmlib member with a HiperDispatch=NO, during or after IPL, will be forced to use or remain in HiperDispatch=YES with the following message:

```
IRA865I HIPERDISPATCH=YES FORCED DUE TO GREATER THAN 64 LPS DEFINED
```

For LPARs IPLed with 64 processors or less in HiperDispatch=NO:

- ▶ An attempt to dynamically add processors that will result in greater than 64 processors will result in the message:

```
ISN012E HIPERDISPATCH MUST BE ENABLED TO CONFIGURE CPU IDS GREATER THAN 3F  
ONLINE
```

- ▶ An attempt to configure more than 64 processors online in HiperDispatch=NO will result in the message:

```
IEE241I CPU(x) NOT RECONFIGURED ONLINE -- REQUIRES HIPERDISPATCH ENABLED
```

For LPARs IPLed with greater than 64 processors in HiperDispatch=YES:

- ▶ An attempt to switch to HiperDispatch=NO will result in the message:

```
IRA865I HIPERDISPATCH=YES FORCED DUE TO GREATER THAN 64 LPS DEFINED
```

**Note:** An LPAR with greater than 64 processors online and so running in HiperDispatch=YES cannot switch back to HiperDispatch=NO even if processors are configured offline.

## New HiperDispatch health check

In z/OS V1R12, a new health check has been added to support greater than 64 processor support:

```
SUP_HIPERDISPATCHCPUCONFIG
```

The new health check is added for the z196 only because it is the only server that supports greater than 64 processors. It is a customizable health check that raises an exception when a system running on an IBM zEnterprise 196 in HiperDispatch=NO is getting close to being forced to run HiperDispatch=YES due to the number of processors approaching 64.

The SUP\_HIPERDISPATCHCPUCONFIG health check parameter, CPUSLEFTB4NEEDHD, accepts values between 0 to 63 with default of 8.

CPUSLEFTB4NEEDHD represents the minimum number of installable and activated processors while being allowed to run in HiperDispatch=NO.

**Note:** Specifying CPUSLEFTB4NEEDHD forces the health check to always be successful. The parameter is meaningless in HiperDispatch=YES.

The SUP\_HIPERDISPATCHCPUCONFIG is supplied as shown in Figure 3-9.

```
CHECK(IBMSUP, SUP_HiperDispatchCPUConfig)
ACTIVE
SEVERITY(LOW) INTERVAL(ONETIME) DATE(20081015)
PARM('CpusLeftB4NeedHd(8)')
REASON('Your reason for making the update.')
```

Figure 3-9 SUP\_HiperDispatchCPUConfig health check

- ▶ For a system in HiperDispatch=YES or HiperDispatch=NO, when the difference between 64 and the number of processors online is greater than the CpusLeftB4NeedHd keyword, the following informational message is issued:

IEAVEH080I CPU configuration supported with HiperDispatch *state*

- ▶ For a system with HiperDispatch=NO, when the difference between 64 and the number of processors online is less than or equal to the CpusLeftB4NeedHd keyword, the following warning message is issued:

IEAVEH081E CPU configuration supported with HiperDispatch disabled. numcpus more CPU(s) can be added with HiperDispatch disabled





# System Logger

This chapter describes the changes to System Logger in z/OS V1R12. 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 different systems across a sysplex.

A log stream is an application-specific collection of data that is used as a log. The data is written to and read from the log stream by one or more instances of the application associated with the log stream. A log stream can be used for such purposes as a transaction log, a log for recreating databases, a recovery log, or other logs needed by applications.

A System Logger application can write log data into a log stream, which is simply a collection of data. Data in a log stream spans two kinds of storage:

- ▶ Interim storage, where data can be accessed quickly without incurring the overhead of DASD I/O.
- ▶ DASD log data set storage, where data is hardened for longer term access. When the interim storage medium for a log stream reaches a user-defined threshold, the log data is offloaded to DASD log data sets.

This chapter describes the following changes for z/OS V1R12:

- ▶ System Logger SHAREOPTIONS
- ▶ Log stream dat set support

## 4.1 System Logger enhancements with z/OS V1R11

Logger staging data sets provide a recoverable copy of log data when it is initially written into a log stream. A staging data set is allocated on the first connection to a log stream from that particular system in the sysplex, as shown in Figure 4-1.

Prior to z/OSV1R11, all staging data sets on one system were allocated under one logger subtask. As work began processing and exploiters began connecting to their log streams, this resulted in a sequential handling of allocations and data set preparation. Therefore, in z/OS V1R11, System Logger began to allocate each staging data set under its respective log stream connection task. This provided a more distributive hashing scheme when assigning DASD-only log streams to a connection task. As a result, this reduced general path length for any log stream connection. Thus, removing the single task bottleneck allowed for more concurrent staging data set allocations and much faster turnaround for large amounts of simultaneous log stream connections, especially when staging data sets are used.

### IXGCONN service

The changes due to these enhancements are seen when IXGCONN REQUEST=CONNECT is issued for connection to a log stream for both:

- ▶ DASD-only log streams
- ▶ CF structure-based log streams with staging duplexing

The log stream staging data set is allocated under the connection task and the logger assigns them to the log stream. For DASD-only log streams, the assignment to a logger connection task is enhanced by a more distributive name hashing scheme. All log streams are mapped to a CF structure and are assigned to the same logger connection task (no change in this area).

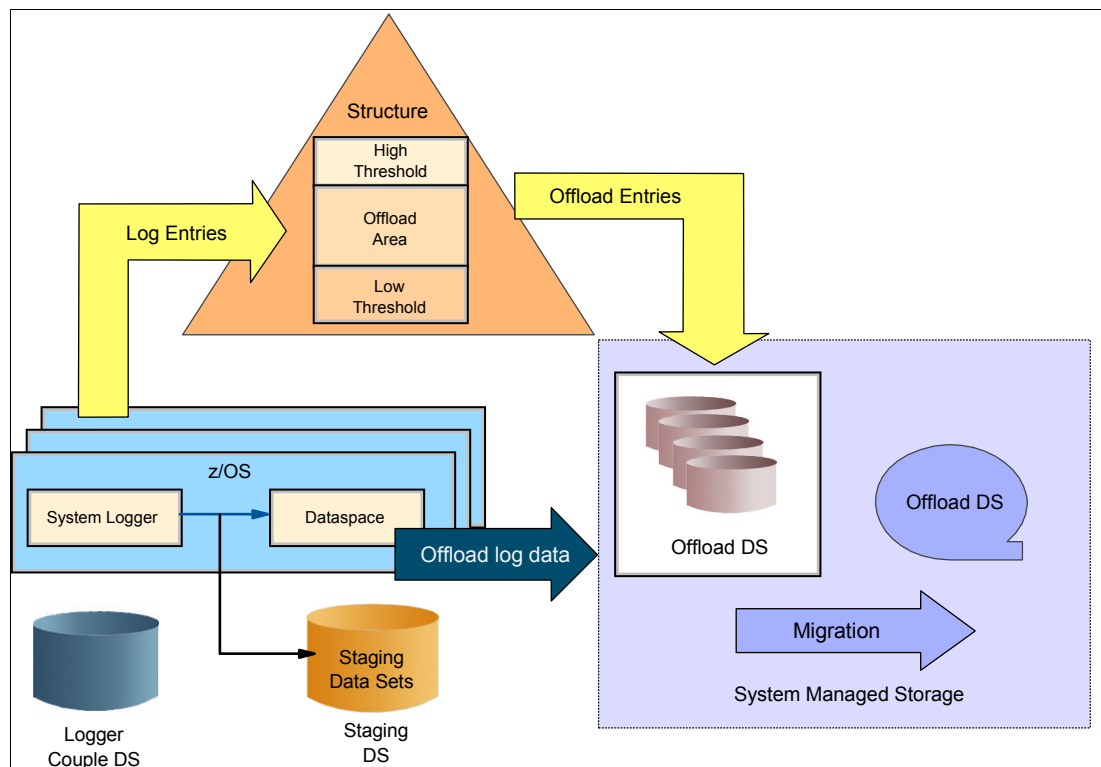


Figure 4-1 System Logger

## System logger task monitoring messages

When the logger event monitor determines that a System Logger service task is not responding while attempting an allocation, deletion, HSM recall, or HSM deletion for a log stream data set, an IXG271I message is issued, as shown in Figure 4-2. Delays in System Logger service tasks can impact not only the specific log stream, but also other log streams on the system or in the sysplex.

```
IXG271I  LOGGER DATA SET REQUEST IN group taskname SERVICE TASK DELAYED DURING  
THE PAST seconds SECONDS FOR LOGSTREAM logstream staging DSN=dsname, DIAG=diag
```

Figure 4-2 IXG271I message

**Operator response:** Check for any conditions in the installation that might be preventing the task from proceeding.

**System action:** System Logger will prompt the operator for action with message IXG272E as shown in Figure 4-3. If the operator does not respond, the data set request might still finish normally. If this occurs, System Logger issues a DOM for both IXG271I and IXG272E messages and continues processing. Until the data set request completes, System Logger on this system might not be able to process many functions such as log stream connect, disconnect, deletion requests, offloads, or browses.

Message IXG271I accompanies the IXG272E message. System Logger waits for the operator to specify an action, or for the request to complete resulting in message IXG272E being DOMed.

```
IXG272E  LOGGER group taskname TASK DELAYED, REPLY "MONITOR", "IGNORE", "FAIL",  
"EXIT".
```

**Explanation:** The system logger Event Monitor is requesting that an action be specified for a task that is not responding in the system logger address space.

The actions are as follows:

1. MONITOR – Continue monitoring this delayed request.
2. IGNORE – Stop monitoring this delayed request.
3. FAIL – Fail the current request this task is processing.
4. EXIT – Terminate system logger Service Task Monitoring.

Figure 4-3 IXG272E message

If any of the waiting recalls complete, System Logger issues a DOM for both IXG271I and IXG272E messages and continues processing. Message IXG281I may also be issued, indicating System Logger has data set recall requests pending.

**Operator response:** Contact the system programmer. Then respond to the message.

When the taskname is MIGRATED DATASET, the resource information for the log stream and data set identifies the oldest recall request outstanding for that group.

**Note:** The PRODUCTION group can have up to 24 recall requests waiting for DFSMSHsm to respond, and the TEST group can have up to 8 recalls.

## 4.2 Logger SHAREOPTIONS

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. A System Logger application can be supplied by:

- ▶ IBM. For example, CICS log manager and the operations log stream (OPERLOG) are IBM-supplied System Logger applications. Other IBM examples are IMS, RRS, and SMF.
- ▶ Independent software vendors.
- ▶ Your installation application.

A System Logger application can write log data into a log stream, which is simply a collection of data. Data in a log stream spans two kinds of storage:

- ▶ Interim storage, where data can be accessed quickly without incurring DASD I/O.
- ▶ DASD log data set storage, where data is hardened for longer term access.

When the interim storage medium for a log stream reaches a user-defined threshold, the log data is off-loaded to DASD log data sets.

There are two types of log streams:

- ▶ Coupling Facility log streams
- ▶ DASD-only log streams

### System Logger log streams

The main difference between the two types of log streams is the storage medium System Logger uses to hold interim log data:

- ▶ In a Coupling Facility log stream, interim storage for log data is in Coupling Facility list structures.
- ▶ In a DASD-only log stream, interim storage for log data is contained in local storage buffers on the system. Local storage buffers are data space areas associated with the System Logger address space, IXGLOGR.

Your installation can use just Coupling Facility log streams, just DASD-only log streams, or a combination of both types.

### Using SHAREOPTIONS(3,3) with z/OS V1R12

When running a Parallel Sysplex, System Logger requires the use of data set sharing across multiple systems. When DFSMS manages System Logger data sets, a data class definition (or ACS routine) allows an installation to specify SHAREOPTIONS(3,3).

When System Logger data set is non-DFSMS managed, and no sharing attributes are defined, improper data set sharing attributes can lead to log stream access errors. Access errors to the log stream can cause an outage of an exploiter such as RRS, CICS, and IMS.

**Note:** If you have multiple systems in the sysplex, it is typical for System Logger to require access to log stream data sets and staging data sets from multiple systems. For this reason, you must specify VSAM SHAREOPTIONS(3,3) for log stream data sets and staging data sets.

If you are using a single system sysplex, there is no requirement for any particular SHAREOPTIONS.



## 4.2.1 System Logger enhancement

In z/OS V1R12, System Logger is enhanced to correct the VSAM SHAREOPTIONS for new log stream data sets when it detects they are not as required. Messages are issued to indicate that System Logger has detected and corrected a data set's SHAREOPTIONS settings.

The SHAREOPTIONS detection and change are done only when log stream data sets are newly allocated. The checking is done only for certain log stream types and sysplex configurations. It means the CF structure-based log streams in a multi-system sysplex have their new data sets checked.

**Note:** Since the automatic updating of existing data set attributes might not be appropriate for all users, logger will continue to perform its detection and provide warning messages when it recognizes inappropriate attributes for existing log stream data sets

## 4.2.2 Using SHAREOPTIONS(3,3)

This new function is intended to prevent data set access problems arising when SHAREOPTIONS(3,3) has not been set in the DATACLAS or ACS routine used when allocating log stream data sets in a shared environment.

System Logger detected that the SHAREOPTIONS settings for the data set did not meet the (3,3) required setting. When this happens, the data set attributes will be updated (via an IDCAMS ALTER). This ensures correct operation of the systems in the sysplex using this data set. An incorrect setting can cause logger being unable to read log data for requests during log stream recovery or browse requests.

The new IXG282I message is shown in Figure 4-4. The message indicates that System Logger detected that a data set which has just been newly allocated had incorrect attributes, so the data set was modified as noted.

```
IXG282I DataSetType DATASET DataSetName WAS ALTERED FOR LOGSTREAM logstream
UPDATED ATTRIBUTE attribute
```

Figure 4-4 IXG282I message

**Note:** System Logger corrects the incorrect SHAREOPTIONS setting, so you should check the log stream LS\_DATACLAS and STG\_DATACLAS specifications to ensure they reference an SMS data class that has the correct options. To determine the current data class values, run the IXCMIAPU Utility and specify LIST LOGSTREAM NAME(logstream) DETAIL(YES) LISTCAT. If the log stream data sets are not defined as intended, you need to update your ACS routines, or other allocation defaults on your system. Alternatively, the SMS data class can be altered, but this type of change will impact any data set that is associated with the data class.

### IXG283I message

If System Logger corrects the SHAREOPTIONS, then the IXG283I message is issued, as shown in Figure 4-5 on page 92.

```
IXG283I OFFLOAD DATASET IXGLOGR.EXAMPLE.A0000000 ALLOCATED NEW FOR LOGSTREAM
IXGLOGR.EXAMPLE CISIZE=4K, SIZE=1474560
```

Figure 4-5 System Logger message correcting SHAREOPTIONS value

System Logger allocated a new log data set as noted in the message shown in Figure 4-5.

The message text indicates:

<b>DataSetType</b>	The type of logger data set
<b>OFFLOAD</b>	A log stream offload data set was altered.
<b>Staging</b>	A staging (recovery) data set was altered.
<b>DataSetName</b>	The name of the data set that was allocated new.
<b>LOGSTREAM</b>	The name of the log stream.
<b>CISIZE</b>	The Control Interval (CI) size for the data set.
<b>SIZE</b>	The size of the data set in bytes.

The log stream data set is used by System Logger to either duplex log data or to move log data from primary/interim storage to secondary storage (STAGING or OFFLOAD respectively) for the log stream. The data set was allocated using the CISIZE and SIZE attributes displayed in the message.

### Incorrect SHAREOPTIONS specification

When the VSAM SHAREOPTIONS attribute for a data set is initially incorrect, then logger attempts to alter it through IDCAMS. You should check the log stream LS\_DATACLAS and/or STG\_DATACLAS specifications to ensure that they reference an SMS data class that has the correct options.

**Important:** To determine the current data class values, run the IXCMIAPU utility with this JCL statement:

```
//SYSIN DD *
DATA TYPE(LOGR)
LIST LOGSTREAM NAME(ATR.PLEX75.MAIN.UR) DETAIL(YES) LISTCAT
/*
```

If the log stream data sets are not defined as intended, then you may need to update your ACS routines, or other allocation defaults on your system. Alternatively, the SMS data class can be altered, but this type of change will impact any data set associated with the data class.

When the LISTCAT parameter is used, System Logger will request from the IDCAMS system services VSAM utility the equivalent of the following command:

```
LISTCAT ENTRIES (log-stream-data-set-name) ALL
```

The LISTCAT output messages will be provided for each log stream offload data set. The combined logger and catalog information in the SYSPRINT output allows for easier determination of data sets needing attention and/or correction by the systems programmer.

**Tip:** The use of wildcards on the NAME keyword should be used only when necessary, since the LISTCAT option can produce a large amount of output for each offload data set.

## Publications

Refer to the following publications:

- ▶ *z/OS MVS Diagnosis: Reference*, GA22-7588 in the topic "LISTCAT (IDCAMS) messages for offload data sets" for samples and description.
- ▶ *z/OS MVS Setting Up a Sysplex*, SA22-7625 in the topic "LIST LOGSTREAM Keywords and Parameters."

## 4.3 Log stream data set support

The 2 GB log stream data set size limitation was a known constraint. Exploiters that generate a large amount of log data could cause frequent offload data set switches. Data set switches can be costly for performance since during an offload the allocation and preparation of the data set is in line with the actual offload. This can cause the following effects:

- ▶ Can cause data set extents to fill quickly.
- ▶ Limit primary storage available when staging data sets are in use.
- ▶ Lead to more frequent offloading than perhaps desired.
- ▶ For CF log streams, may lead to under-utilization of the CF structure space available.

### 4.3.1 Maximum log stream data set size to 4 GB (OA30548)

System Logger is enhanced to support log stream data set sizes up to 4 GB (increased from the previous logger 2 GB limit). This change applies to both log stream offload and staging data set types. For both DASD-only log streams and Coupling Facility-based log streams using staging data sets, this support allows for primary storage sizes up to the new 4-GB size. SMF exploitation of log streams has been the driving force behind many of the recent requests for larger log data set size support.

**Important:** Before installing this PTF on any system in the sysplex, the PTFs (UA52327, UA52328, and UA52329) for APAR OA31461 *must* be installed and active on every system in the sysplex.

The PTF for APAR OA30548 may be applied to one system at a time in the sysplex. Systems with and without the PTF for APAR OA30548 activated can coexist in the same sysplex. If a system with the PTF for OA30548 activated performs a log stream offload and allocates the next log stream offload data set, then a data set size up to 4 GB can be obtained and used by all systems in the sysplex. However, if a system without PTF OA30548 activated performs an offload and allocates the next log stream offload data set, then a data set size up to 2 GB is allocated.

To exploit this function, installations will use the same APIs or utilities as currently used to define or update data set size parameters on the log stream definition.

### IXGQBUF macro

Applications can ascertain the length of the control information generated by System Logger using the IXGQUERY service, which returns the information in a buffer mapped by the IXGQBUF macro (QBUF\_CONTROL\_INFO\_SIZE field).

IXGQBUF version 4 is added in z/OS V1R12 to support new fields QBUF\_FULL\_LS\_DS\_SIZE and QBUF\_FULL\_STG\_DS\_SIZE, which contain the size of the

current log stream offload data set and staging data set, respectively. The fields are added using existing available storage in the QBUF area.

**Note:** A compatible change is made to the following IXGQBUF mapping macro field by APAR OA30548. Use the IXGQUERY service to retrieve information about a log stream in the sysplex. The information is returned in a buffer mapped by IXGQBUF. The information returned by IXGQUERY includes:

- ▶ Information related to the size of the current offload data set and staging data set in the IXGQBUF. Note that the maximum value returned in fields QBUF\_LS\_DS\_SIZE and QBUF\_STG\_DS\_SIZE is 2 GB-1 ('7FFFFFFF'X).
- ▶ For data set sizes greater than 2 GB, '7FFFFFFF'X will be returned, but the full value will be available in fields QBUF\_FULL\_LS\_DS\_SIZE and QBUF\_FULL\_STG\_DS\_SIZE.

### The HIGHOFFLOAD parameter

The HIGHOFFLOAD parameter acts against the structure space use or staging data set. The IXCMIAPI utility with a specification of TYPE(LOGR) or the service IXGINVNT were updated for this support, as follows:

- ▶ Use LS\_Size and STG\_Size parameters to request 4K blocks (regardless of CI\_Size).

**Note:** This value is only one factor in the resulting size of the log data set.

- ▶ The offload data set parameters are:
  - Ls\_Size()
  - Ls\_Dataclas()
- ▶ The staging data set parameters are:
  - Stg\_Size()
  - Stg\_Dataclas()

For example, an LS\_Size(872000) with 24 K CI Size would yield a data set size of approximately 4,286,545,920 bytes.

New informational hardcopy messages, detailing key log data set allocation activity and attributes such as data set size are in messages IXG283I and IXG284I.

**Note:** The HIGHOFFLOAD parameter specifies the point, in percent value of space consumed, where System Logger will begin offloading Coupling Facility log data to the DASD log data sets for this log stream. It corresponds to the HIGHOFFLOAD parameter in the LOGR policy.

## 4.3.2 Coexistence and migration considerations

This support is rolled back to z/OS V1R9 and above. With APAR OA30548, PTFs UA52443, UA52444, and UA52445 provide 4 GB log data set support on z/OS V1R9 and above releases.

Prior to APAR OA30548 support, log data sets sized (using logger parameters) greater than 2 GB were being reallocated under 2 GB. With OA30548 PTFs or z/OS V1R12 applied and

activated in the sysplex (and no other changes to log stream parameters), these data sets will no longer be reallocated less than 2 GB.

**Important:** Toleration APAR OA31461 (PTFs UA52327, UA52328, and UA52329) must be installed and active on every pre-z/OS V1R12 system in the sysplex before bringing the V1R12 system into the sysplex.

### Migration considerations for using the new fields

Take the following steps if your application uses either of these new fields:

1. Review the use of IXGQBUF fields QBUF\_LS\_DS\_SIZE and QBUF\_STG\_DS\_SIZE.
2. Ensure that your application can tolerate a maximum value of 2 GB-1 for those fields when the actual data set size is 2 GB or greater.
3. If your program requires the full data set size, update it to use new fields QBUF\_FULL\_LS\_DS\_SIZE and QBUF\_FULL\_STG\_DS\_SIZE when the QBUF\_VERSION\_NUMBER is 4 or greater.
4. With this support applied and activated, users specifying QBUF length = 200 bytes will be returned QBUF\_VERSION\_NUMBER = 4. Ensure that your program can tolerate a QBUF\_VERSION\_NUMBER greater than 3 in this case.

### 4.3.3 Testing log data set parameter modifications

When defining or updating a log stream, you need to understand what the resulting size of the offload data set, or the staging data set, or both will be. The maximum size for both types of log data sets is 4 GB. If the System Logger parameters and other factors cause allocation to attempt to create a log data set greater than the maximum 4 GB size, the request fails and any attempt to define or connect to the log stream also fails. If the size change is the result of an update request, System Logger might not be able to offload data when it is necessary to allocate a new offload data set.

Many factors affect the resulting size of both offload and staging log data sets, so it is helpful to test changes to the log stream configuration for DASD before you implement the changes on the production log stream. IBM suggests that you implement the following procedure before you make any parameter or configuration changes that can affect the size of the log data sets. Consider the following actions:

- ▶ For offload data set configuration changes: Define a GROUP(TEST) log stream. This log stream should use the same configuration definitions as the production log stream apart from any parameter changes that you are making for testing purposes. For example, unless it is one of the items that you are planning to change, ensure that the LS\_SIZE, CI\_SIZE, SMS classes, DASD device, and so forth are the same values as those used for the production log stream. If the log stream define request is successful, it is likely that the update to the production log stream will not result in offload allocation failures if the resulting offload data set is allocated with more than the maximum 4 GB size. If the offload data set exceeds that size, System Logger issues message IXG251I to the hardcopy log with return code 140 and reason code 110.
- ▶ For staging data set configuration changes: Follow a similar procedure as that for the offload data set. For example, define the same values for STG\_SIZE and other parameters for the staging data set as those for production. To test the allocation of a staging data set, you must connect to the log stream. IXGCONLS provides a mechanism for doing this. If the log stream connect request is successful, it is likely that the update to the production log stream will also not result in allocation failures of the staging data set. This procedure causes a log stream resource to be in use until you are able to delete the

definition. See IXGCONLS in *z/OS MVS Diagnosis: Reference*, GA22-7588. If the data set size is larger than 4 GB, the allocation request fails, and System Logger issues message IXG251I to the hardcopy log that contains message IGD306I with return code 140 and reason code 110.

**Note:** See *z/OS MVS Setting Up a Sysplex*, SA22-7625, for more details.



## Auto-reply to WTORs

With the auto-reply policy for WTORs, you can get an automatic response from the system to WTOR messages; when there is no automation, the operator is unaware of the outstanding request, or spends a long time determining what response should be given.

This new support in z/OS V1R12 with the use of an auto-reply policy determines if an operator or client-supplied automation has not provided any reply to a WTOR in a specified amount of time, and the auto-reply policy contains this WTOR, the system will use the reply from the policy to reply to the message.

This chapter describes the following features for this new support:

- ▶ Auto-reply policy for WTORs
- ▶ IBM-supplied AUTOR00 parmlib member
- ▶ Using the auto-reply policy
- ▶ New commands for auto-reply
- ▶ Operating system components updates for auto-reply
- ▶ Security checking of CPF commands
- ▶ Console buffer enhancements

## 5.1 Auto-reply policy for WTORs

Reply delays can affect all systems in a sysplex. Customers notice that WTOR (write-to-operator with reply) messages may take at least 30-45 minutes to be answered. In some cases it is unacceptable to wait so long. It happens for the following reasons:

- ▶ Most customers do not have operators closely monitoring the system, just waiting to issue an immediate reply to a WTOR.
- ▶ Some WTORs are so infrequent that operators may never have seen them before.
- ▶ Operators typically do not have authority, experience, or system understanding to make their own decision on what to reply for uncommon WTORs.

While traditional automation products already provide this support, auto-reply can be used for most WTORs issued during NIP when inboard automation is not available. It means that simple replies can easily be handled without involving complex automation.

### 5.1.1 Auto-reply policy

With the auto-reply policy for WTORs, you can get an automatic response from the system for WTOR messages. When there is no automation, the operator is unaware of the outstanding request, or spends a long time determining what response should be given. The auto-reply policy provides the following enhancements for z/OS operating systems:

- ▶ If an operator or client-supplied automation has not provided any reply to a WTOR in a specified amount of time, and the auto-reply policy contains this WTOR, the system will use the reply from the policy to reply to the message.
- ▶ A default auto-reply policy is activated during an IPL, unless you explicitly request that the policy not be activated. If you do not activate the default policy, WTORs issued during NIP cannot be automated.
- ▶ You can add to or alter the default auto-reply policy, or provide your own auto-reply policy.
- ▶ You can use an operator command to:
  - Activate or deactivate the auto-reply policy on a system.
  - Display the auto-reply policy and the current outstanding WTORs that are being monitored by auto-reply processing.
  - Deactivate auto-reply processing or to stop monitoring a current outstanding WTOR.
- ▶ You can specify the system parameter AUTOR=xx in the IEASYSxx parmlib member or in response to the following message:

```
IEA101A SPECIFY SYSTEM PARAMETERS
```

This allows your installation to provide a set of parmlib members that contain the auto-reply policy, or to request that auto-reply processing not be activated.

If you do not want to activate auto-reply processing, specify AUTOR=OFF in the IEASYSxx parmlib member or in response to message IEA101A SPECIFY SYSTEM PARAMETERS. It is not recommended that you remove AUTOR00 from parmlib, because service or new releases might reinstall AUTOR00. If there is no AUTOR00 member in parmlib, auto-reply is not activated and the following messages are produced:

```
CNZ2600I AUTO-REPLY POLICY ATTEMPTING TO USE AUTOR=00.  
IEA301I AUTOR00 NOT FOUND IN PARMLIB  
CNZ2601I AUTO-REPLY POLICY NOT ACTIVATED.  
NO ENTRIES SPECIFIED
```



## 5.2 Activating an auto-reply policy for WTORs

For z/OS V1R12 systems, IBM supplies a default auto-reply AUTOR00 parmlib member. During an IPL, if the AUTOR00 parmlib member exists, auto-reply processing is activated.

If the replies or delay duration in the AUTOR00 member are not desirable, you can create a new AUTORxx parmlib member and make corresponding changes. Also compare the replies to what your automation product would reply to these WTORs.

**Note:** Make sure that the AUTOR00 replies are in accordance with the replies from your automation product. It is not recommended to make updates to AUTOR00, because updates to AUTOR00 might be made by the service stream or in new z/OS releases.

With z/OS V1R12, with the new AUTOR00 parmlib member, there is also a new system parameter, AUTOR=, specified in the IEASYSxx parmlib member. During an IPL, a default policy is used unless it is requested not to be activated or another AUTORxx policy is specified. You can add your own policy or override the AUTOR00 policy.

If WTORs listed in the AUTOR00 parmlib member are automated by an existing automation product, examine the WTOR replies in the AUTOR00 parmlib member before IPLing, to avoid possible conflicts.

### IEASYSxx parmlib member

This parameter enables you to specify your own auto-reply policy during IPL, or to request that auto-reply processing not be activated. If auto processing is not to be activated, specify AUTOR=OFF or remove AUTOR00 from the member; see Figure 5-1.

The parameter options are:

- ▶ AUTOR={xx }
- ▶ AUTOR={{xx[,xx]...}}
- ▶ AUTOR={OFF }
- ▶ AUTOR={{(OFF) }

If AUTOR= is specified in response to IEA101A SPECIFY SYSTEM PARAMETERS, the AUTOR= value overrides any AUTOR= specification in the IEASYSxx parmlib member.

**Value Range:** Any two characters (A-Z, 0-9, @, #, and \$).

**Default Value:** AUTOR=00

Figure 5-1 AUTOR= options in the IEASYSxx parmlib member

### 5.2.1 AUTORxx parmlib member

Use the AUTORxx parmlib member to activate auto-reply processing on a system. The AUTOR00 parmlib member contains the auto-reply policy suggested by IBM. You can modify the member (which is not recommended), or define another AUTORxx member to customize the auto-reply policy. The AUTOR00 member also contains comments that provide the message text for each WTOR and the rule used to select the WTOR.

## AUTORxx parameter syntax

The following syntax rules apply to the AUTORxx parmlib member:

- ▶ Data is specified in columns 1-71.
- ▶ Comments can start in any column and can span lines, and must start with /\* and end with \*/. Comments may appear in any place where a blank is accepted, except within quoted strings.
- ▶ Generally, syntax errors cause the auto-reply changes to be rejected, while in the following cases, syntax errors are ignored and allow the changes to become active:
  - The maximum number of message IDs is reached.
  - Duplicate message IDs are specified.
  - NOTIFYMSGSGS is specified in different members.

## AUTORxx parmlib member parameters

The parameters shown in Figure 5-2 can be specified with the AUTORxx parmlib member.

```
[NOTIFYMSGSGS({HC|CONSOLE})]  
[MSGID(['msgid']) {NOAUTORREPLY  
                {DELAY(nnn{M|S}) REPLY(['replytext'] [,['replytext']...])}]
```

Figure 5-2 AUTORxx syntax format

**Note:** If you create an AUTORxx parmlib member, update the IEASYSxx parmlib member that you use for IPL. Add the following statement to the IEASYSxx member where xx corresponds to the AUTORxx parmlib member that you create:

```
AUTOR=(xx,00)
```

Where:

▶ NOTIFYMSGSGS

May only appear once in a parmlib member. If multiple members are specified in the SET AUTOR= command, the first NOTIFYMSGSGS value is accepted. The other specifications are ignored.

- HC - Indicates that the auto-reply notification messages (CNZ2605I, CNZ2606I, and CNZ2608I) only appear in the hardcopy log. If NOTIFYMSGSGS is not specified, HC is the default value.
- CONSOLE - Indicates that the auto-reply notification messages (CNZ2605I, CNZ2606I, and CNZ2608I) are displayed on consoles receiving routing codes 2 (operator information) or 10 (system programmer information) and also appear in the hardcopy log.

▶ MSGID()

The first keyword of the message definition. This keyword must be in the range of 1 to 10 characters. The message ID must be enclosed in quotes if it contains non-alphanumeric characters, such as equal signs or parentheses. The quotes do not count as part of the message ID. If enclosed in quotes, the msgid is not converted to uppercase. Wildcards are partially supported in the msgid, as follows:

- The question mark "?" is supported, because some messages (for example, JES2 messages) start with an installation-specified character (not always \$). For instance, ?HASP1234 is allowed.

- The asterisk "\*" is not treated as a wildcard in a msgid. If a msgid contains both a question mark and an asterisk, the message definition is rejected.
- Multiple question marks in the msgid is supported. For instance, eight question marks ???????? would match on any eight character message ID.

**Note:** The maximum number of message IDs that do not contain wildcards is limited to 10,413. The maximum number of message IDs that contain wildcards is limited to 1,500.

▶ NOAUTORREPLY

Can be specified to cause subsequent MSGID() specifications of this message ID (in this member or subsequent members) to be ignored. With the NOAUTOREPLY option, you can have your own AUTORxx member and use it to remove messages specified in the AUTOR00 member without actually removing the AUTOR00 statements. No auto-reply processing is performed for this message ID.

▶ DELAY(nnn{M | S})

Can appear in any order and on different lines from REPLY() and MSGID().

- nnn - Is the minimum amount of time, in minutes (M) or seconds (S) to delay after the WTAAOR is issued but before the system issues a reply. Only three digits are supported, which gives a limit of 999 minutes (16.65 hours) or 999 seconds (16.65 minutes). The delay value of 0 is supported, but specifying the value of 0 can prevent automation or an operator from providing a reply.

▶ REPLY - REPLY(['replytext['] [,['replytext[']...])

Can appear in any order and on different lines from DELAY() and MSGID(). A null reply can be provided by specifying REPLY(' ').

- replytext - This is limited to 64 characters. If the reply contains blanks, non-alphanumeric characters, or is not to be folded to uppercase, enclose the reply in single quotes. The single quotes count towards the total of 64 characters. If the reply is too long for the WTOR requestor, the notification message CNZ2608I is issued when the WTOR is issued. Where message CNZ2608I appears depends on the setting of NOTIFYMSGs. It is displayed on consoles receiving routing codes 2 (operator information) or 10 (system programmer information), and also appears in the hardcopy log.

- To specify quotes in the reply, specify two contiguous single quotes, like REPLY('That's all folks.'). The reply may contain system symbolics that are resolved when the parmlib member is processed. If you want to have the symbolic resolved when the REPLY command is issued, specify two contiguous ampersands, like REPLY('Here is the '&&SYSNAME','). The symbolic must be in uppercase and enclosed in quotes along with the two contiguous ampersands &&.

- If the auto-reply policy entry contains a symbolic (such as &&name) that is to be resolved when the reply is issued, auto-reply is unable to validate that the reply length would be acceptable by the WTOR issuer. If the reply with the symbolic resolved is longer than what the WTOR issuer expects, when auto-reply processing issues the reply, the REPLY command rejects the reply with:

```
IEE700I REPLY xxxx IGNORED; REPLY TOO LONG FOR REQUESTOR
```

Auto-reply no longer monitors this WTOR, and the response to the D AUTOR,WTORS command (CNZ2604I) indicates that the WTOR has been replied to.

- Do not have the reply text span parmlib source lines. Otherwise, the use of symbolics might cause problems. For example, if you specify:

```
REPLY('system=&SYSNAME.,option1 ,option2')
```

And &SYSNAME resolves to SY1, the reply text is:

```
'system=SY1,option1 ,option2'
```

To prevent the substitution from causing problems, split the reply text into separate strings. A correct specification is:

```
REPLY('system=&SYSNAME.',',',option1,option2')
```

And the reply text is:

```
system=SY1,option1,option2'
```

- Auto-reply processing can concatenate the strings specified in REPLY. For example, if you specify REPLY('String1'), the quotes are part of the reply 'String1'. If you specify REPLY('String1','String2'), auto-reply processing concatenates the strings and the reply is 'String1String2'. The leading and trailing quotes are part of the reply. You can also have a mixture of strings. By specifying:

```
REPLY(Word1,'String1',Word2,'String2')
```

You will get the reply:

```
WORD1'String1'WORD2'String2'
```

- The **REPLY** command accepts only one quoted string if the first character of the reply is a quote. For example:

```
'This is the reply' but not this
```

The **REPLY** command will only pass

```
This is the reply
```

To the WTOR issuer. The WTOR issuer does not see

```
but not this
```

Although it is possible that auto-reply can build this type of string, you should be aware that it may not yield the results you wanted.

#### Notes:

1. Both DELAY() and REPLY() are required if NOAUTOREPLY is not specified.
2. Duplicate DELAY() or REPLY() keywords for an MSGID() are rejected and cause the MSGID() specification to be rejected.
3. Specification of the same message ID in different message definitions results in the first definition being used.

### Syntax example of AUTORxx parmlib member

Figure 5-3 shows an example of an AUTORxx parmlib member.

```
notifmsgs(hc)  
/* $HASP811 REPLY Y TO CONTINUE OR N TO TERMINATE START PROCESSING */  
Msgid(?HASP811) Delay(30S) Reply(Y)  
/* ANTU2220D "READY FOR FLASHCOPY. REPLY 'I' TO INITIATE, 'C' TO */  
/* CANCEL" */  
Msgid(ANTU2220D) Delay(60S) Reply(C)
```

Figure 5-3 IBM-supplied default for the AUTORxx parmlib member

**Note:** If NOTIFYMSGSGS is not specified, NOTIFYMSGSGS(HC) is the default value.

## 5.3 IBM-supplied default AUTOR00 parmlib member

The AUTOR00 parmlib member describes each predefined rule on comments and identifies also which of the six auto-reply rules is applied for each automated message. The AUTOR00 member is available in SYS1.IBM.PARMLIB(AUTOR00). A copy of this member is shown in Appendix B, “WTOR messages on Auto-Reply” on page 807.

**Note:** If AUTOR= is not specified in the IEASYSxx parmlib member, the default AUTOR00 parmlib member is used if it exists.

### Auto-reply rules

To develop the list of WTORS to include in the default policy, a number of selection rules were used.

Following are the details of each of the six selection rules and examples:

#### 1. System detected problem (Figure 5-4)

If the system detected an error during execution of an operator-initiated action, for example, if an operator reply could cause corruption, an auto-reply is needed.

**Reply:** Cancel the action

**Rationale:** Resources could be held up; respond quickly. Worst case is that the user has to reinitiate the action.

**Delay time:** 60 seconds, which allows the operator to reject quickly and allows the user to reinitiate.

```
ARC0380A RECALL WAITING FOR VOLUME volser IN USE BY HOST procid, FUNCTION
function. REPLY WAIT, CANCEL, OR MOUNT          */
/* Rule: 1                                     */
  Msgid(ARC0380A)  Delay(60S) Reply(CANCEL)
/*****
$HASP294 WAITING FOR RESERVE (VOL volser). REPLY 'CANCEL' TO END WAIT
/* Rule: 1                                     */
  Msgid(?HASP294)  Delay(30S) Reply(CANCEL)
```

Figure 5-4 Examples of messages applied on rule 1 on AUTORxx parmlib member

#### 2. System detected recovery issue (Figure 5-5)

Tough love on sick but not dead situations.

**Reply:** Terminate.

**Rationale:** Want to quickly address before situation deteriorates further.

**Delay time:** 60 seconds.

```

$HASPO70 SPECIFY RECOVERY OPTIONS ('RECOVER' OR 'TERMINATE' OR 'SNAP' AND,
OPTIONALLY, ',NODUMP')                               */
/* Rule: 2                                           */
Msgid(?HASP070) Delay(30S) Reply(TERMINATE)

```

Figure 5-5 Example of message applied on rule 2 on AUTORxx parmlib member

### 3. System detected dynamic change (Figure 5-6)

For dynamic changes that were made, choose the latest active system configuration when there is a recognized discrepancy.

**Reply:** Option that reflects latest configuration

**Rationale:** Dynamic changes were intended, but not hardened. Return to intended state (dynamic).

**Delay Time:** 30 seconds

```

IXC289D REPLY U TO USE THE DATA SETS LAST USED FOR typename OR C TO USE THE
COUPLE DATA SETS SPECIFIED IN COUPLExx              */
/* Rule: 3                                           */
Msgid(IXC289D) Delay(60S) Reply(U)

```

Figure 5-6 Example of message applied on rule 3 on AUTORxx parmlib member

### 4. Confirmation WTORs (Figure 5-7)

If a generic confirmation message, reply negative.

**Reply:** Negative confirmation (for example, NO, CANCEL ...)

**Rationale:** If the operator has not responded immediately to the message, assume there is some confusion and do not automatically assume the original command was correctly entered. Allow the operator to re-enter the command.

**Delay time:** 60 seconds

```

CNZ9009D CONTINUE WITH MIGRATION? REPLY N TO ABORT OR Y TO CONTINUE
/* Rule: 4                                           */
Msgid(CNZ9009D) Delay(60S) Reply(N)

```

Figure 5-7 Example of message applied on rule 4 on AUTORxx parmlib member

### 5. Continue with IPL (Figure 5-8)

If there is a condition that is preventing the system from IPLing, reply to allow the system to continue to IPL.

**Reply:** Option to allow IPL to continue (for example, GO, CONTINUE ...).

**Rationale:** If the condition is preventing the system from IPLing and the system needs to be up to correct the condition, reply to allow the system to continue the IPL.

**Delay time:** 60 seconds

```

$HASP811 REPLY Y TO CONTINUE OR N TO TERMINATE START PROCESSING
/* Rule: 5
Msgid(?HASP811) Delay(30S) Reply(Y) */

```

Figure 5-8 Example of message applied on rule 5 on AUTORxx parmlib member

#### 6. Component or product recommended values (Figure 5-9 on page 105)

Component level expert specified values.

**Reply:** Specified by component level expert.

**Rationale:** Component level expert specifications may override auto-reply rules.

**Delay time:** Specified by component level expert.

```

ARC0825D RECYCLE TAPE LIST CREATED, DSN=dsname. DO YOU WISH TO CONTINUE? REPLY
'N' TO STOP RECYCLE OR 'Y' WHEN READY TO MOUNT TAPES.
/* Rule: 6
Msgid(ARC0825D) Delay(60S) Reply(Y) */

```

Figure 5-9 Example of message applied on rule 6 on AUTORxx parmlib member

### Examples of messages with the auto-reply policy

In Appendix B, “WTOR messages on Auto-Reply” on page 807, you can see all messages already included in the AUTOR00 parmlib member. IBM can make future changes on AUTOR through service maintenance or in future z/OS releases.

Figure 5-10 shows how the system reacts to message IEE800D, included in the AUTOR00 parmlib member.

```

V 1800,OFFLINE,FORCE
*016 IEE800D CONFIRM VARY FORCE FOR 1800 - REPLY NO OR YES
CNZ2605I At 09.35.17 the system will automatically 133
reply: NO
to the following WTOR:
0016 IEE800D CONFIRM VARY FORCE FOR 1800 - REPLY NO OR YES
R 0016,NO
CNZ2606I System has automatically replied: 135
NO
to the following WTOR:
0016 09.34.18 2010130 IEE800D CONFIRM VARY FORCE FOR 1800 - REPLY N
OR YES
IEE600I REPLY TO 016 IS;NO
IEE756I VARY COMMAND CANCELLED BY OPERATOR

```

Figure 5-10 Example of automated reply to message IEE800D

## 5.4 Changing the auto-reply policy

If the replies or delay duration are not desirable, you can create a new AUTORxx parmlib member and make corresponding changes. Also compare the replies to what your automation product would reply to these WTORs. Make sure that the AUTOR00 replies are in accordance with the replies from your automation product. It is not recommended to make

updates to AUTOR00, because updates to AUTOR00 might be made by the service stream or in new z/OS releases.

**Note:** If you have specific needs for WTOR automation or need to avoid system automation product conflicts with the AUTORxx parmlib member, then:

- ▶ Create a new AUTORxx policy.
- ▶ Keep default AUTOR00 and change only NOAUTORREPLY on the AUTOR00 parmlib member.

If you have created an AUTORxx parmlib member, update the IEASYSxx parmlib member that you use for IPL. Add the following statement to the IEASYSxx parmlib member:

```
AUTOR=(xx,00)
```

Here the xx corresponds to the AUTORxx parmlib member that you created.

**Important:** The IEASYSxx member specifying AUTOR cannot be shared with prior z/OS releases.

If you only need the default AUTOR00 settings, you can omit specifying AUTOR= in the IEASYSxx parmlib member, and previous z/OS levels can continue to use the IEASYSxx parmlib member. Even if AUTOR= is not specified in the IEASYSxx parmlib member, AUTOR00 is used if it exists.

If you do not want to activate auto-reply processing, specify AUTOR=OFF in the IEASYSxx parmlib member or in response to the following message:

```
IEA101A SPECIFY SYSTEM PARAMETERS.
```

It is not recommended that you remove AUTOR00 from parmlib, because service or new releases might reinstall AUTOR00. If there is no AUTOR00 member in parmlib, auto-reply is not activated and messages are issued as follows:

```
CNZ2600I AUTO-REPLY POLICY ATTEMPTING TO USE AUTOR=00.  
IEE252I MEMBER AUTOR00 FOUND IN SYS1.IBM.PARMLIB  
CNZ2600I AUTO-REPLY POLICY ACTIVATED.
```

If a policy is active, the following messages are issued:

```
CNZ2600I AUTO-REPLY POLICY ATTEMPTING TO USE AUTOR=00.  
IEE252I MEMBER AUTOR00 FOUND IN SYS1.IBM.PARMLIB  
CNZ2600I AUTO-REPLY POLICY ACTIVATED.
```

### Auto-reply notification messages

The purpose of auto-reply notification messages is to record in the hardcopy log the fact that auto-reply plans to take action and whether auto-reply does take action. One of the messages is for notification if the reply would be too long for the WTOR issuer. Where the notification messages are displayed depends on the setting of the NOTIFYMSGs statement in the AUTORxx parmlib member.

## 5.5 New commands for auto-reply

The following operator commands are available to control the auto-reply policy for WTORs:



<b>SET AUTOR=(xx[,xx]..)</b>	Activate auto-reply processing on a system by specifying the AUTORxx parmlib member or members that the system is to use.
<b>D AUTOR,POLICY</b>	Display the auto-reply policy active on the system.
<b>D AUTOR,WTORS</b>	Display the current outstanding WTORS that are being monitored by auto-reply processing.
<b>SETAUTOR OFF</b>	Deactivate auto-reply processing and stop monitoring all WTORS issued on the system.
<b>SETAUTOR IGNORE</b>	Request that auto-reply processing stop monitoring an outstanding WTOR.

### 5.5.1 The SETAUTOR command

Use the SETAUTOR command to deactivate auto-reply processing or to stop monitoring a current outstanding WTOR.

The complete syntax for the SETAUTOR command is shown in Figure 5-11.

```

SETAUTOR {OFF
          {IGNORE,RID={rpid }
          {0,SYS=sysname}

```

Figure 5-11 SETAUTOR command syntax

The SETAUTOR parameters are:

- OFF** An optional parameter indicating that the auto-reply processing should deactivate the auto-reply processing and stop monitoring all WTORS issued on the system.
- IGNORE** An optional parameter requesting that the auto-reply processing should stop monitoring a WTOR.
- RID=rpid** The reply ID that identifies the WTOR that will no longer be monitored by auto-reply processing. The values supported are 0-9999.

**Note:** Different WTORS might each have a reply ID of zero. When the limit of outstanding WTORS is reached, each system will have one more ID (ID zero) that is to be assigned to one important WTOR at a time. Therefore, to distinguish which reply ID zero is desired, a system name must also be specified.

**sysname** The name of the system where the reply ID zero message is issued.

For the output of the SETAUTOR command, see the descriptions of messages CNZ2600I and CNZ2607I using LookAt or in *z/OS MVS System Messages Volume 4 (CBD - DMO)*, SA22-7634.

#### Command examples

Assume the following VARY command is issued to generate message IEE800D:

```

V 3D0,OFFLINE,FORCE
0009 IEE800D CONFIRM VARY FORCE FOR 3D0 - REPLY NO OR YES

```

If message IEE800D (reply ID 0009) is to be ignored by auto-reply processing, issue the command shown in Figure 5-12.

```
SETAUTOR IGNORE,RID=9
CNZ2607I AUTO REPLY WILL NO LONGER OCCUR FOR THE FOLLOWING WTOR: 0009
06.46.55 2008130 IEE800D CONFIRM VARY FORCE FOR 3D0 - REPLY NO OR YES
```

Figure 5-12 Command to remove a reply ID 9

If auto-reply processing is to be deactivated, issue the command shown in Figure 5-13.

```
SETAUTOR OFF
CNZ2600I AUTO-REPLY PROCESSING DEACTIVATED.
```

Figure 5-13 Turn off auto-reply processing

## 5.5.2 Displaying auto-reply policy and WTOR information

Use the DISPLAY AUTOR command to display the current auto-reply policy and the current WTORS being monitored by auto-reply processing. The complete syntax for the DISPLAY AUTOR command is shown in Figure 5-14.

```
D AUTOR[,POLICY|,P]
      [,WTORS|,W]
      [,L={a|name|name-a}]
```

Figure 5-14 DISPLAY AUTOR command options

The **D AUTOR** parameters are:

- |                          |  |
|--------------------------|--|
| <b>POLICY or P</b>       | Requests that the auto-reply policy active on the system be displayed in message CNZ2603I. This command displays all the current WTORS that are being monitored.   |
| <b>WTORS or W</b>        | Requests that the current outstanding WTORS being monitored by auto-reply processing be displayed in message CNZ2604I.   |
| <b>L=a, name, name-a</b> | 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 a complete description of messages CNZ2603I and CNZ2604I, use LookAt or see *z/OS MVS System Messages Volume 4 (CBD - DMO)*, SA22-7634.

### Command examples

Assume that the AUTOR00 parmlib member contains the settings shown in Figure 5-15. There are two auto-replies in the policy, one specifying one-minute delay and reply continue and the second specifying one=minute delay and a reply of no.

```

/*-----\
| IXC371D CONFIRM REQUEST TO VARY SYSTEM xxx OFFLINE. |
| REPLY SYSNAME=xxx TO REMOVE xxx OR C TO CANCEL. |
\-----*/
Msgid(IXC371D) Delay(1m) Reply(C)
/*-----\
| IEE800D CONFIRM VARY FORCE FOR xxx - REPLY NO OR YES |
\-----*/
Msgid(IEE800D) Delay(1m) Reply(NO)

```

Figure 5-15 Auto-reply policy with two reply IDs being monitored

If the auto-reply policy is displayed, the output of the **D AUTOR,P** command is as shown in Figure 5-16.

```

D AUTOR,P
CNZ2603I 06.46.34 AUTOR POLICY
POLICY ACTIVATED AT 06.45.32 ON 12/25/2008 NOTIFYMSG(S(H)C)
FROM PARMLIB MEMBERS 00
--MSG ID-- DELAY ----REPLY TEXT---
IEE800D 1M NO
IXC371D 1M C

```

Figure 5-16 A display of the existing auto-reply policy

If WTORs IEE800D and IXC371D are issued and are waiting to be replied to by auto-reply processing, the output of the **D AUTOR,W** command is as shown in Figure 5-17.

```

D AUTOR,W
CNZ2604I 06.47.02 AUTOR WTORS
0009 STATUS=06.47.58 SYS=SY1
MSG=IEE800D CONFIRM VARY FORCE FOR 3D0 - REPLY NO OR YES
REPLY=NO
0008 STATUS=06.47.46 SYS=SY1
MSG=IXC371D CONFIRM REQUEST TO VARY SYSTEM SY1 OFFLINE. REPLY
SYSNAME=SY1 TO REMOVE SY1 OR C TO CANCEL.
REPLY=C

```

Figure 5-17 Display the outstanding WTORs

## 5.6 Other system updates

The following changes have been made in the z/OS V1R12 operating system for the auto-reply function:

- ▶ SMF record type 90 updates
- ▶ SDSF in the SR panel
- ▶ MPFLSTxx parmlib member updates

## 5.6.1 SMF record type 90 subtype 33

There is a new SMF record type 90 subtype 33 for the auto-reply implementation. These records list the AUTORxx suffixes specified in a SETAUTOR command and are shown in Figure 5-18 on page 110.

Offsets	Name	Length	Format	Description
0	0 SMF90T33_Timestamp	8	binary	Time of auto-reply policy change.
8	8 SMF90T33_#_Suffixes	4	binary	Count of AUTORxx suffixes used to set the policy.
16	16 SMF90T33_Suffixes	76	EBCDIC	Array of 2-byte suffixes of specified AUTORxx parmlib members (38 maximum).

Figure 5-18 SMF record type 90 subtype 33 for auto-reply

## 5.6.2 SDSF SR panel

SDSF has support for the console auto-reply function, which includes a new action character and columns on the SR panel. Figure 5-19 shows an example from the SR panel of a current outstanding action message. IEE800D is a message ID in the AUTOR00 parmlib member for auto-reply. Auto-reply will respond to this action message.

Display Filter View Print Options Search Help						
-----						
SDSF SYSTEM REQUESTS RM 1 IM 11 CEM 136 EM 43 LINE 1-23 (191)						
COMMAND INPUT ==>				SCROLL ==> HALF		
NP	REPLYID	SysName	JobName	Message-Text		
	51	SC74		&051 IEE800D CONFIRM VARY FORCE FOR D800 - REPLY NO OR YES		

Figure 5-19 Display from the SR panel

The following tasks are associated with exploiting this enhancement:

- ▶ If you have customized field lists in your ISFPRMxx parmlib member, review the columns added with this enhancement for possible updates to the field lists.
- ▶ Authorize users to the AI action character on the SR panel using the CMDLEV parameter in ISFPARMS (command level 3) or SAF.

## 5.6.3 MPFLSTxx parmlib member update

The MPFLSTxx parmlib member (message processing facility list) is used to control message presentation and management. It was updated to add the capability to differentiate color attributes for messages that are controlled by the auto-reply support.

Message presentation refers to color, highlighting, and intensity attributes that the system uses when displaying messages on an operator console. You can specify these attributes on the .MSGCOLR statement in the MPFLSTxx parmlib member; of course, the console must support the attributes you specify. On the .MSGCOLR statement, you can indicate the display attributes that the system is to use in one of the following ways:

- ▶ IBM-supplied defaults (DEFAULT)
- ▶ Attributes specified in the previously used MPFLSTxx parmlib member or concatenation of MPFLSTxx parmlib members (NOCHANGE)
- ▶ Color (c), highlighting (h), and intensity (i) attributes specified in the MPFLSTxx parmlib member for a message area (msgarea).

### The .MSGCOLR statement

The syntax for the .MSGCOLR statement is shown in Figure 5-20.

```
{DEFAULT } [/*comments*/]
.MSGCOLR {NOCHANGE }
{msgarea(c,h[,i]) [,msgarea(c,h[,i])] ...}
```

Figure 5-20 .MSGCOLR statement

Where:

**msgarea(c,h[,i])** Specifies the message area (message type or field) and contains the accepted values and defaults for msgarea. This operand allows you to specify the color (c), highlighting (h), and intensity (i) attributes for a message area.

Color attributes (c) are specified as follows:

- ▶ R-Red, W-White, G-Green, B-Blue, P-Pink, Y-Yellow, T-Turquoise

Highlighting attributes (h) are specified as follows:

- ▶ N - Normal (colored characters on a black background)
- ▶ B - Blinking
- ▶ R - Reverse video (black characters on a colored background)

Intensity attributes (i) are specified as follows:

- ▶ N-Normal intensity
- ▶ H-High intensity

If you do not specify a .MSGCOLR statement, the system uses IBM-supplied defaults. If you specify a .MSGCOLR statement with no operands, the system defaults to NOCHANGE. NOCHANGE indicates that the color, highlighting, and intensity attributes are to be the same as those specified in the previously used MPFLSTxx parmlib member (or concatenation of MPFLSTxx parmlib members).

### New values for msgarea

z/OS V1R12 has a new value for msgarea, called AUTOR and is used as shown in Table 5-1.

Table 5-1 Values for msgarea

Values for msgarea	Meaning and Use	Defaults for msgarea (color, highlight, intensity)
AUTOR	Specifies attributes for write-to-operator with reply (WTOR) messages that are being monitored by auto-reply processing.	(W, N, H)

It is possible to check your setting with the DISPLAY MPF command, shown in Figure 5-21.

```
D MPF
IEE677I 17.26.27 MPF DISPLAY 891
MESSAGE SUPPRESSION AND USER EXITS INACTIVE - NOT INITIALIZED
GENERAL WTO USER EXIT (IEAVMXIT) INACTIVE
SUBSYSTEMS RECEIVING FOREIGN MESSAGES AND DOMS:
*ALL
FIELD      -MPF COLOR  HLIGHT  INTEN      FIELD      -MPF COLOR  HLIGHT  INTEN
URGATTN    -DFL RED    NONE    HIGH      IMEDACTN   -DFL WHITE   NONE    HIGH
EVETACTN   -DFL GREEN   NONE    NORM      GENMSG     -DFL GREEN   NONE    NORM
PPMSG      -DFL GREEN   NONE    NORM      SELPEN     -DFL BLUE    NONE    NORM
INSTRERR   -DFL WHITE   NONE    HIGH      ENTRYARA   -DFL GREEN   NONE    NORM
WARNLGEN   -DFL BLUE    NONE    NORM      WARNRGEN   -DFL BLUE    NONE    NORM
WARNRURG   -DFL RED    BLINK   HIGH      OOLCNTL    -DFL TURQU   NONE    NORM
OOLLABEL   -DFL TURQU   NONE    NORM      OOLDATA    -DFL GREEN   NONE    NORM
AUTOR     -DFL WHITE  NONE    HIGH
COMMAND-EXIT -MPF
  CDRPL     -00
```

Figure 5-21 Display MPF command

#### 5.6.4 Error detection and new system messages

Auto-reply will attempt to detect whether the reply is too long for the requestor. If so, the reply is not issued, a warning message, as shown in Figure 5-22, is issued (by default to hardcopy) and the WTOR is no longer monitored if the reply contains symbolics, as there is no length detection.

```
CNZ2608I REPLY FOR WTOR msgid IS TOO LONG FOR REQUESTOR.
AUTO-REPLY WILL NOT PROCESS THIS WTOR: rpid msgtxt
```

Figure 5-22 Error message for incorrect reply lengths

Auto-reply attempts to detect a “run away.” If auto-reply attempts to reply to the same WTOR more than 20 times in a second, monitoring is stopped. This could happen if one of the following situations occurs:

- ▶ The policy contains a typo so the reply is always invalid.
- ▶ The policy requests a delay of zero.
- ▶ A WTOR loop (“run away”) occurs.

Figure 5-23 shows the error message for the situations that might occur.

```
CNZ2609I RECURSIVE AUTO-REPLY DETECTED FOR MESSAGE xxxxxxxxxx.
AUTO-REPLY WILL NOT PROCESS THIS WTOR:
rpid msgtext
```

Figure 5-23 Auto-reply error conditions message

## New system messages related to auto-reply

Following are messages related to the auto-reply function being active:

- ▶ When auto-reply replies during NIP processing:

```
CNZ2602I REPLY TO 00 IS:replytxt <- Auto replied
```

- ▶ The post NIP auto-reply message is as follows:

```
CNZ2605I At hhmss the system will automatically reply:replytxt to the following WTOR:rpid msgtxt
```

```
CNZ2606I System has automatically replied:replytxt to the following WTOR: rpid hhmss yyyyddd msgtxt [msgtxtcont]
```

- ▶ Activating, modifying, or deactivating auto-reply has the following message:

```
CNZ2600I AUTO-REPLY POLICY action
```

The action field will contain one of the following:

ACTIVATED

DEACTIVATED

DEACTIVATED – OUTSTANDING WTORS WILL NOT BE AUTO-REPLIED

ALREADY DEACTIVATED

MODIFIED

MODIFIED - OUTSTANDING WTORS USING PREVIOUS POLICY ATTEMPTING TO USE AUTOR=00

## 5.7 Security checking of CPF command routing

The command prefix facility (CPF) macro allows you to manage command prefixes. A command prefix enables an operator to enter a command from a system in a sysplex, and route that command to the appropriate subsystem for execution. The CPF macro allows an application to use command prefixes to associate an operator command with a "target" system. The command prefixes are available to any system in the sysplex.

CPF allows you to define a unique command prefix for each system or subsystem in the sysplex so that the operator can enter the command from any console in the sysplex and expect the command to run on the appropriate subsystem.

### 5.7.1 Security checking with z/OS V1R12

z/OS V1R12 is implementing security checks to verify that an operator can use the prefix to transport commands to other systems in the sysplex. Commands that are sent to other systems via prefix routing are now checked by RACF resource profiles that authorize the use of prefix routing. Following is the RACF profile to authorize prefix routing:

```
MVS.ROUTE.CMD.sysname
```

For compatibility, the security check is only performed if the following resource profile is defined in the OPERCMDS class. This allows installations to now control whether operators may issue a CPF prefixed command that will be transported to another system.

```
MVS.CPF.ROUTE.CHECK
```

## 5.8 Console buffers

The MVS system keeps some WTO and WTOR messages in buffers in virtual storage. The WTO buffers hold the messages that the system has not yet displayed at the eligible consoles; the WTOR buffers each hold one WTOR message that the system has already displayed but that an operator has not responded to. The maximum number of WTO and WTOR buffers is determined by the MLIM and RLIM parameters on the INIT statement in the CONSOLxx parmlib member. If these parameters are not coded, the system defaults, as described in *z/OS MVS Initialization and Tuning Reference*, SA22-7592, are in effect.

The CONTROL Q command is used to remove messages that are backed up to a console. On earlier z/OS releases, the CONTROL Q command could only remove backed-up messages that were in 24-bit storage. Once removed, the system would then move messages that were in 31-bit storage down to 24-bit storage. This would give the appearance that the CONTROL Q command did not work since the console was still backed up. Typically, the operator would have to issue several CONTROL Q commands to clear the backlog.

With z/OS V1R12, since there is no movement of the messages, CONTROL Q can remove all the messages destined for the console.

- ▶ Auto-reply required enlarging some console buffers. Buffers have been moved to 31-bit storage and the WQEs (IHAWQE) were already in 24-bit or 31-bit storage.
- ▶ RDCM (IEERDCM) and TDCM (IEETDCM) are moved to 31-bit storage.

There are changes to use Format 1 CCWs for console I/O, and the **CONTROL Q** command is now more effective in message backlog conditions.





## Real Storage Manager (RSM) enhancements

The Real Storage Manager (RSM) controls the allocation of central storage during initialization and pages in user or system functions for execution.

Some RSM functions:

- ▶ Allocate central storage to satisfy GETMAIN requests for SQA and LSQA.
- ▶ Allocate central storage for page fixing.
- ▶ Allocate central storage for an address space that is to be swapped in.

This chapter describes RSM enhancements introduced in z/OS V1R12. These include:

- ▶ LOCALSYSAREA for IAVR64
- ▶ RSM support of SDUMP resource
- ▶ Large page enhancements

## 6.1 64-bit storage overview

The 2-GB address in the address space is marked by a virtual line called the bar (Figure 6-1). The bar separates storage below the 2-GB address, called below the bar, from storage above the 2-GB 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 called 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.

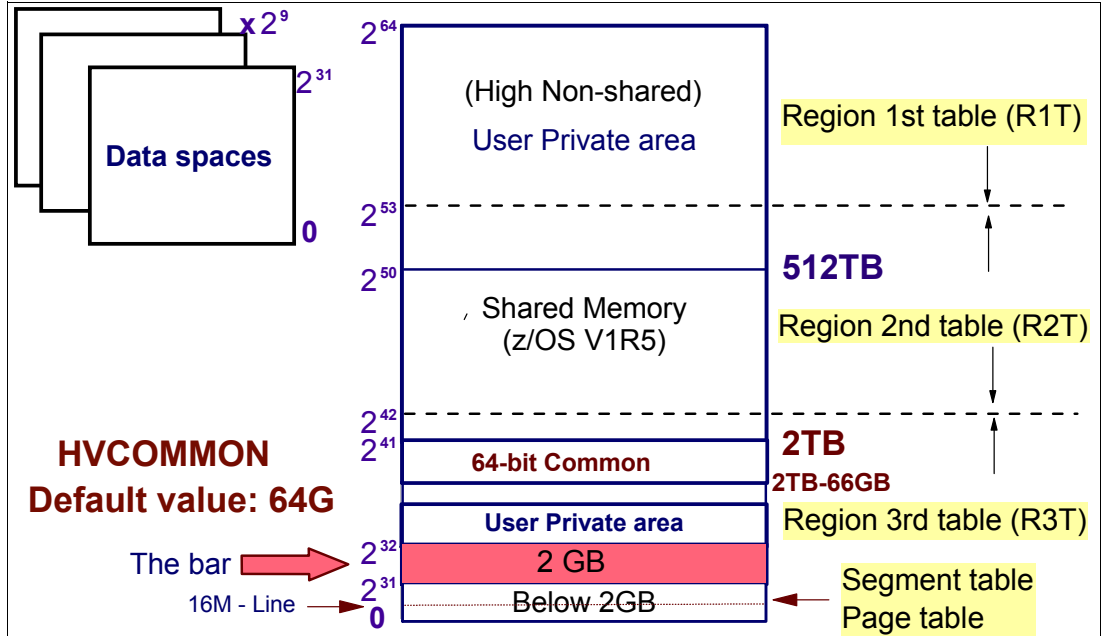


Figure 6-1 z/OS V1R11 map of virtual storage areas

## 6.2 z/OS UNIX fork() processing

Fork() is a POSIX/XPG4 function that creates a duplicate process referred to as a child process. The process that issues the fork() is referred to as the parent process. With z/OS UNIX, a program that issues a fork() function creates a new address space that is a copy of the address space where the program is running. The fork() function does a program call to the kernel, as shown in Figure 6-2 on page 117, which uses WLM/MVS facilities to create the child process. The contents of the parent address space are then copied to the child address space.

### WLM and the kernel

The kernel interfaces to WLM to create the new address space for a fork() function. WLM uses an IBM-supplied procedure, BPXAS, to start a new address space. This new address space then initializes the UNIX child process to run in the address space.

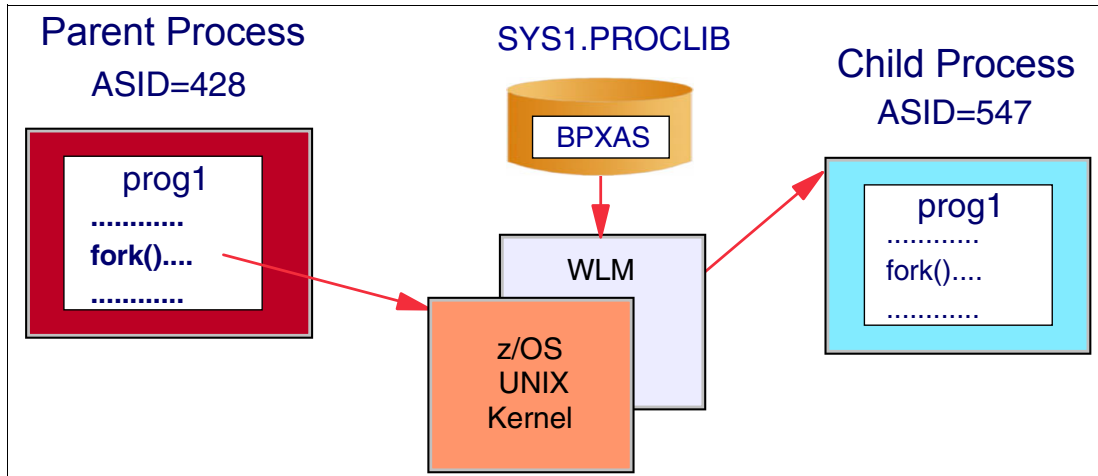


Figure 6-2 *fork()* processing with z/OS UNIX

## 6.2.1 z/OS UNIX fork() processing

Prior to z/OS V1R12, during `fork()` processing, the 64-bit copy processing fails when high virtual storage is allocated in the child space at the time the copy of the parent 64-bit virtual is copied into the child space.

During the `fork()` creation of the new address space, the program in the child address space starts at the same instruction as the program in the parent address space. Control is returned to both programs at the same point. The only difference that the program sees is the return code from the `fork()` function. A return code of zero is returned to the child after a successful `fork()`. All other return codes are valid only for the parent.

### Child address space

Once the child address space has been created, the child gets the required storage from a `STORAGE` request. The kernel then copies the contents of the parent address space to the child address space using the `MVCL` instruction, as shown in Figure 6-3 on page 118. Once the copy has been completed, a short conversation between the kernel and the child process takes place. At this point, both the parent and child process are activated. The program in the child address space gets control at the same point as the program in the parent address space. The only difference is the return code from the `fork()` function.

**Note:** The child address space is almost an identical copy of the parent address space. User data, for example private subpools, and system data, such as Recovery Termination Management (RTM) control blocks, are identical.

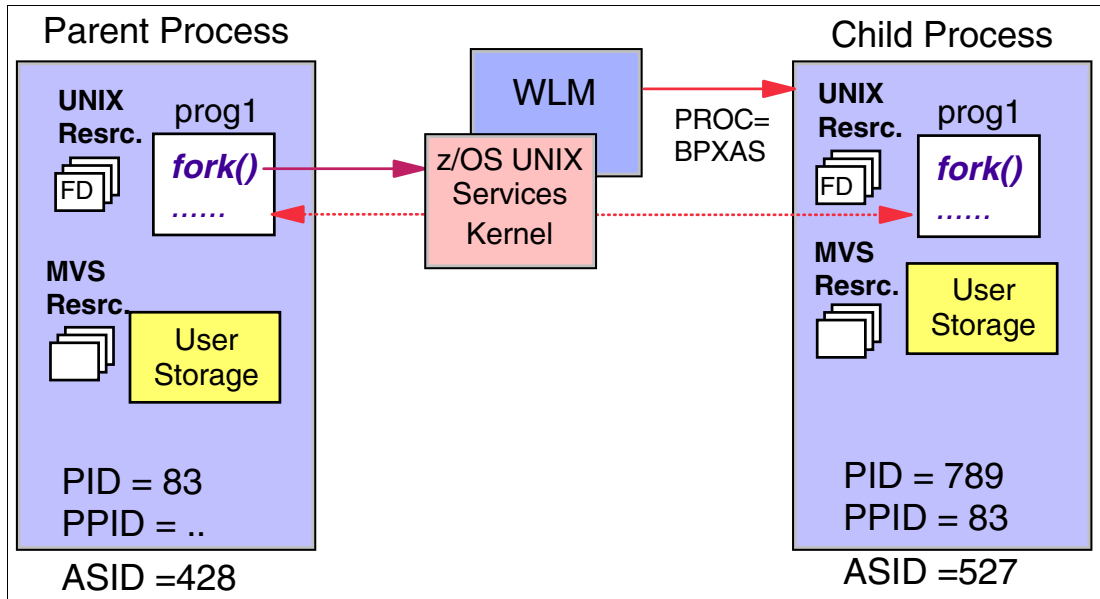


Figure 6-3 `fork()` processing with z/OS UNIX

### Problems prior to z/OS V1R12

With z/OS versions prior to z/OS V1R12, `fork()` 64-bit copy processing fails when the high virtual storage is allocated in the child address space at the time when the copy of the 64-bit parent address space is copied into the child space.

This causes the following to occur:

- ▶ RSM fails the copy with return code 8 reason code '6E000300'x and the child address space contains memory objects that were previously allocated.
- ▶ The `fork()` function fails with a return code 70, reason code '0B1505C1', as the invocation of the IARV64FC service fails. The documented action for this failure was to retry the operation at a later time.

Figure 6-4 on page 119 shows the TCB structure for the fork initiator child address space.

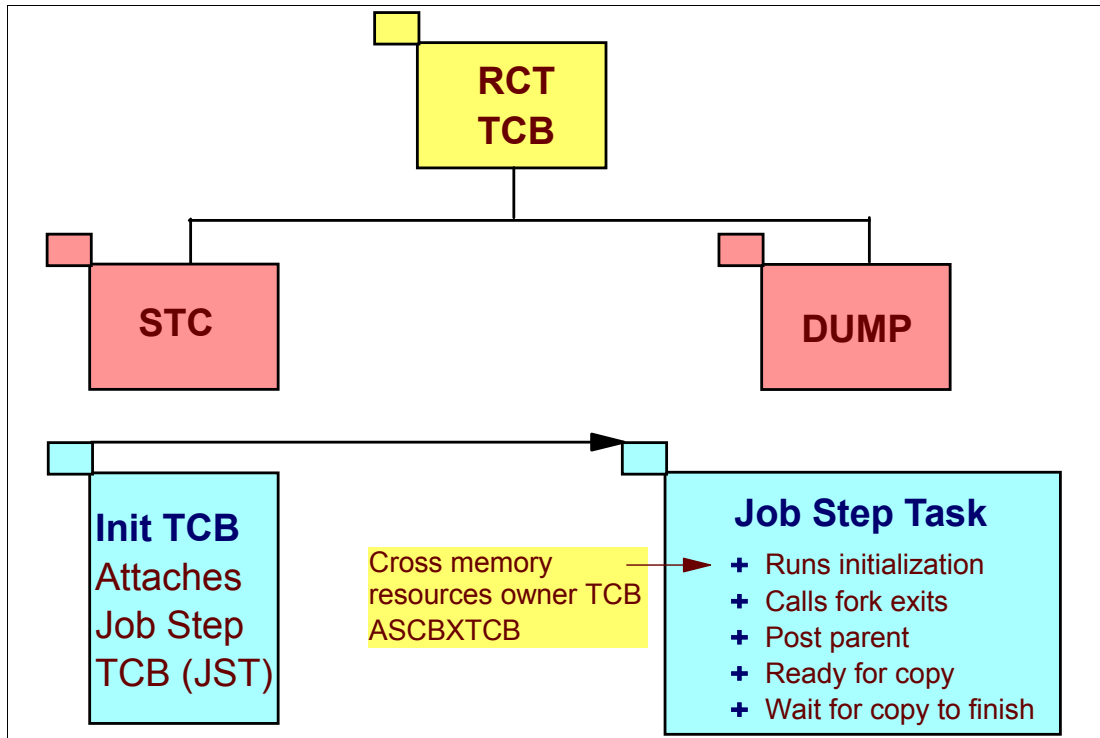


Figure 6-4 TCB structure for the fork initiator address space

As part of the fork() process address space initialization processing is done. During this processing RACF uses storage in the newly created address space. This causes the RSM copy of the parent storage to the fork() address space to fail because some of the storage is already used.

## 6.2.2 z/OS V1R12 enhancements

A new function in z/OS V1R12 supports a new system area in the storage above the 2-GB bar, as shown in Figure 6-5 on page 120. This storage is designed to be used for system storage as an equivalent to LSQA below the 2-GB bar. Storage in this area will not be copied during the fork() process when RSM copies the parent storage to the child address space.

**Note:** A new keyword is added to IARV64 REQUEST=GETSTOR, LOCALSYSAREA=NOIYES, to indicate that the memory object should 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 new keyword.

The new system area is shown in Figure 6-5.

### fork() processing with z/OS V1R12

When a REQUEST=GETSTOR is processed, a new required parameter, 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. At completion, the memory object is created in the address space indicated.

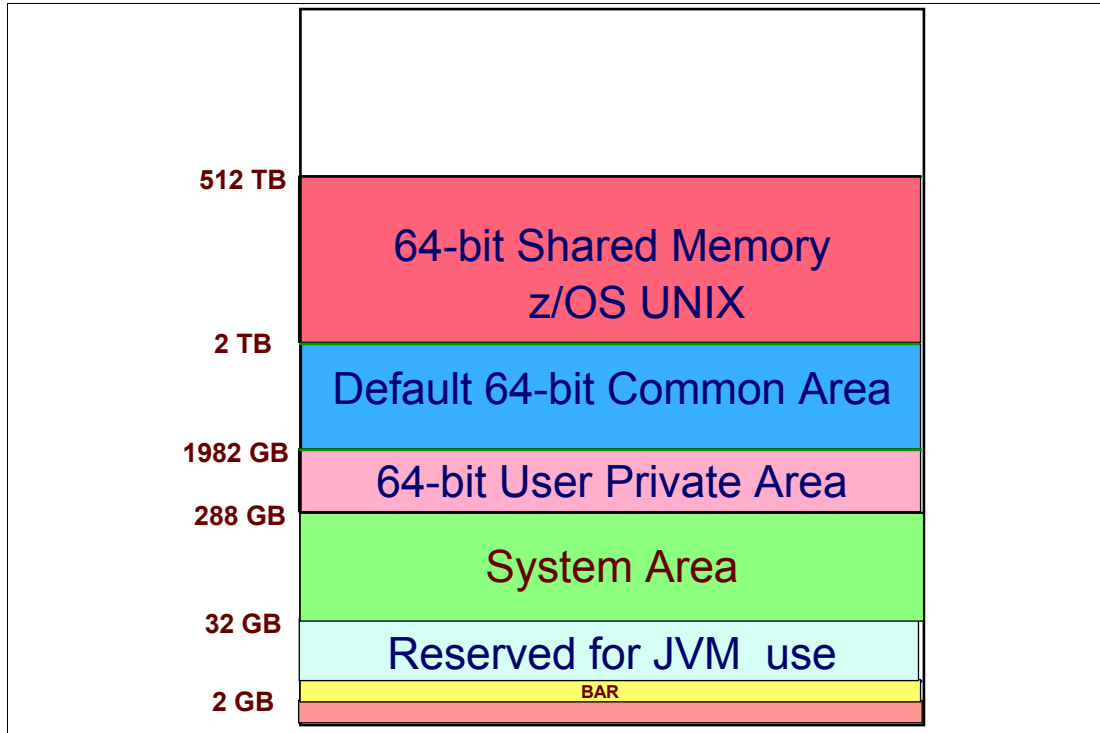


Figure 6-5 Private area storage map showing the new system area

### 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 6-5.

This new function allows fork() processing to continue to use 64-bit virtual under any circumstances.

## 6.3 RSM support of SDUMP resource

SVC dump capture time to capture dump storage can be excessive in certain environments. Taking SVC dumps can create real storage constraints in the following situations:

- ▶ When a large amount of data needs to be paged in from auxiliary storage for inclusion in the dump
- ▶ When capturing large amounts of component SDUMP exit data

- ▶ Data paged in from auxiliary storage puts pressure on real memory availability. The paged-in data looks like recently referenced data to the system, which may force the page-out of more important production or application data.

### **z/OS V1R12 SDUMP improvements**

This is addressed in z/OS V1R12 by improvements to the RSM data copy service used by SDUMP when capturing dump data, as follows:

- ▶ Batch page-in I/O operations during the SDUMP capture phase rather than performing one I/O at a time eliminates much of the delay and waiting associated with performing the I/O serially.
- ▶ Data captured for the dump will no longer look recently referenced; thus this data will be paged out before other potentially more important data is paged out.
- ▶ Certain components, for example GRS for SDATA GRSQ data, IOS for CTRACE data, and configuration dataspace now exploit a more efficient capture method in their SDUMP exits.

These improvements reduce the dump capture time and system non-dispatchable time. This should cause less impact to other application and production work.

Prior to z/OS V1R12, the target area passed by SDUMP was always within a dataspace. Now the target area can be a dataspace or an area within a 64-bit memory object.

### **z/OS V1R12 RSM improvements**

The following changes in RSM processing have been made to enhance data capture and benefit SDUMP processing:

- ▶ Reduced dump capture time in certain environments, such as large amounts of data paged in from aux storage and large amounts of component exit data.
- ▶ There will be less likelihood of non-DUMP-related pages being paged out due to real storage memory pressure.
- ▶ More efficient capture of GRS and IOS exit data allows the caller to use 64-bit virtual memory objects as the target of the copy.
- ▶ Use the multipage option of the WAITINIO service (NUMPAGES) when paging in pages to be captured (rather than one page at a time).
- ▶ Artificially age all captured pages to make them look old and more likely steal candidates than more important production or application data.
- ▶ Maintain a set of counts and statistics for various operations performed during the copy, such as the number of pages requiring page-in and the number of waits for frame availability.

## **6.3.1 Support in VERBX IEAVTSFS**

The VERBX IEAVTSFS command has been enhanced to display the SDUMP data capture statistics. Figure 6-6 on page 122 shows sample output from the VERBX IEAVTSFS command showing the SDUMP statistics for the global data capture phase of the dump and for the capture phase of ASID X'13'.

The IPCS VERBX IEAVTSFS command displays the statistics for various phases of the dump capture process:

- ▶ A set of statistics for the global storage capture phase
- ▶ A set of statistics for each address space included in the dump

- ▶ A set of statistics for the dump exit phase of the dump

Global storage start	04/30/2010 12:06:31.726449
Global storage end	04/30/2010 12:06:31.808150
Global storage capture time	00:00:00.081700
Defers for frame availability	0
Pages requiring input I/O	59
Source page copied to target	9966
Source frames re-assigned	59
Source AUX slot IDs re-assigned	0
Asid 0013:	
Local storage start	04/30/2010 12:06:31.851937
Local storage end	04/30/2010 12:06:31.903105
Local storage capture time	00:00:00.051167
Tasks reset dispatchable	04/30/2010 12:06:31.903248
Tasks were nondispatchable	00:00:00.051310
Defers for frame availability	0
Pages requiring input I/O	50
Source page copied to target	1708
Source frames re-assigned	66
Source AUX slot IDs re-assigned	0

Figure 6-6 VERBX IEAVTSFS data capture statistics

## RSM statistics in z/OS V1R12

The new function in z/OS V1R12 provides data for the RSM statistics at the end of the report:

- ▶ Defers for frame availability - The number of WAITFRAM requests. A non-zero value indicates that RSM had to wait one or more times for frame availability. These defers adversely affect the dump capture time.
- ▶ Pages requiring Input I/O - The number of pages read via the WAITINIO service. High values here indicate the need to page in data for inclusion in the dump.
- ▶ Source page copied to target - The number of source-real to target-real copies performed via the BLKMVC service. This should be the most common operation performed during the dump capture.
- ▶ Source frames re-assigned – This is the count of source frames that were “robbed” (ROBREAL). Source frames are mainly robbed when the source page was paged in and the copy in real is unchanged from the aux copy.
- ▶ Source AUX slot IDs re-assigned – The aux slot ID for the source page was transferred to the target page.

Figure 6-7 on page 123 shows sample output from the VERBX IEAVTSFS command showing the SDUMP statistics for the SDUMP exits phase of the dump.



Dump Exits	
Exit address	0755E6A0
Home ASID	0005
Exit start	04/30/2010 12:06:31.740223
Exit end	04/30/2010 12:06:31.740226
Exit time	00:00:00.000002
Exit attributes: Sdump, Early Global	
Defers for frame availability	0
Pages requiring input I/O	3
Source page copied to target	5239
Source frames re-assigned	3
Source AUX slot IDs re-assigned	0

Figure 6-7 VERBX IEAVTSFS dump exits statistics

## 6.4 Large page enhancements

The large page support increasing 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 could incur. The performance benefits turned out to be even better than expected.

### 6.4.1 Large page support for the nucleus

A new function in z/OS V1R12 introduces 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. It should be noted 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 % of memory size, as follows:

- ▶ 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, as well as 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. Some applications can be severely degraded by using 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 new 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%     | 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.<br><br>If the amount specified exceeds this value, the specification is rejected and the system issues a prompt for a new value to be specified on the LFAREA parameter.<br><br>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 below. |
| xxxxxxG | 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 new 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 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 new 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.

## 6.4.2 Large page coalesce support in z/OS V1R12

With the large page support, installations defined how much central storage was to be used for large pages via the LFAREA parameter in the IEASYSxx parmlib member. The disadvantage of central storage backing unused large pages is that they cannot be used during page shortage situations.

The new function in z/OS V1R12 introduces 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. Once defined, this area can only be used for large page requests. The only exception to this rule is when the system is storage constrained. When storage is constrained, RSM breaks up a large page into 256 4 K pages in order to satisfy 4 K page requests. When storage is no longer constrained, RSM performs a large page coalesce (this 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 could result in a permanent loss of those large pages and a single large page.





## DFSMS enhancements

DFSMS comprises a suite of related data and storage management products for the z/OS system. DFSMS is an operating environment that helps automate and centralize the management of storage based on the policies that your installation defines for availability, performance, space, and security. The heart of DFSMS is the Storage Management Subsystem (SMS). Using SMS, the storage administrator defines policies that automate the management of storage and hardware devices. These policies describe data allocation characteristics, performance and availability goals, backup and retention requirements, and storage requirements for the system.

DFSMS is an exclusive element of the z/OS operating system. It is a software suite that automatically manages data from creation to expiration. This chapter describes the enhancements to Data Facility Storage Management Subsystem (DFSMSrmm) for z/OS V1R12.

This chapter describes the functional enhancements to DFSMS in z/OS V1R12:

- ▶ DFSMS catalogs contention detection
- ▶ DEFINE RECATALOG support
- ▶ OPEN/CLOSE/EOV RAS support
- ▶ VSAM partial release on multiple volumes
- ▶ New VSAM support
- ▶ Copy storage group volumes
- ▶ VOLSELMSG and TRACE parameters
- ▶ Multitasking volume recovery from dump
- ▶ DFSMS health checks
- ▶ IDCAMS DELETE PDS and PDSE
- ▶ DFSMSrmm support improvements

## 7.1 DFSMS catalogs

Today, catalogs are limited to 4 GB in size, affecting scalability. As volume sizes have increased, the need for larger catalogs has become urgent. The solution is to use the current extended addressability function in VSAM. This function requires the data sets to be extended format (EF), thus the catalog must be EF.

**Note:** The new support in z/OS V1R12 is for basic catalog structures (BCS) only; it does not apply to VVDSs.

### EA catalogs

For extended addressable (EA) catalogs, the z/OS V1R12 enhancement allows catalogs to grow beyond the previous 4-GB size limit, using extended addressability. Extended addressability (EA) support already exists for VSAM data sets; this new support lets you define basic catalog structure (BCS) catalogs as extended format data sets to give them extended addressability.

**Restriction:** Unlike other extended format VSAM data sets, which can also be striped or compressed, extended format ICF BCSes cannot be striped or compressed.

**Note:** This BCS catalog enhancement does not apply to VVDS data sets, which continue to have a 4-GB size limit.

Several requirements must be met in order to DEFINE EA catalogs. They must be SMS-managed and be on extended format (EF) volumes. In order for SMS to select an extended format volume, the ISMF Data Class panel must have a data set name type of EXT, meaning extended, and the IF EXT field below it must be P for preferred or R for required.

### Changed messages

There are two changed messages, IGD17080I and IGD17163I. The present versions of these messages state that they can be caused by trying to make user catalogs or master catalogs EF. For z/OS V1R12, this EF restriction is removed and these messages no longer have references to catalogs.

## 7.2 Contention detection for catalogs

This enhancement provides a way to identify certain catalog service tasks waiting on the input/output table resources (SYSZTIOT) to see if any are waiting beyond the default or specified wait time. The first time the system detects a CAS task waiting on the SYSZTIOT resource, it creates a SYMREC record to the logrec data set and the IEC393I message. If the task is still waiting the next time the system checks the CAS for tasks waiting on the SYSZTIOT resource, the system issues only system message IEC393I displaying information about the waiting task or tasks. You can specify a different wait time using the following command:

```
MODIFY CATALOG,CONTENTION(SYSZTIOT,wait_time)
```

The MODIFY CATALOG,REPORT command is also enhanced to display the wait time that is in effect, in a new CONTENTION line in the report output.

## Specifying a wait time for catalog service task contention

You can use the new `MODIFY CATALOG,CONTENTION` command to specify a wait time in minutes for catalog service tasks to wait in contention for an event, after which a notification message is issued. The command has two parameters:

- ▶ Reason class, which specifies the group of events which the wait time field will apply to. Currently the only supported reason class is `SYSZTIOT`.
- ▶ Wait time, which specifies the length of wait in minutes before the message is issued.

The wait time value must be an integer in the range of 5 to 9999. The default value is 10, for 10 minutes. You can specify a value of 0 to disable the wait time notification function. The command is as follows:

```
MODIFY CATALOG,CONTENTION(SYSZTIOT,20)
```

### 7.2.1 Detecting catalog `SYSZTIOT` contention

In order to monitor contention for the task input/output table resources (`SYSZTIOT`), the system checks the catalog address space (CAS) for tasks waiting for the `SYSZTIOT` beyond the specified wait time (by default, 10 minutes). Once a task is identified as waiting beyond the specified wait time, the system writes a `SYMREC` record to the logrec data set and issues `IEC393I` message displaying information about the waiting task or tasks. If the task is still waiting after 5 more minutes, this message is repeated and a new `SYMREC` record is written. This message will be repeated and a new `SYMREC` will be written every 15 minutes thereafter if nothing changes. If a new hung task is identified or an old hang is resolved, the notification is reset to the new state at the time of the next system check (within 30 seconds).

#### Commands to analyze `SYSZTIOT` contention

If you receive system message `IEC393I` notifying you that tasks are waiting on the `SYSZTIOT` resource, you can use the information in the message to determine if any action is needed to resolve the wait. Note that receiving this message does not mean that an error or problem exists—you may not need to take any action at all. If you cancel any of the jobs listed, consider first taking a dump of the CAS by issuing:

```
F CATALOG,TAKEDUMP
```

To get more information about the waiting jobs, use the information in message `IEC393I` as follows:

- ▶ To get job name information, issue one of the following commands:
  - To gather additional information about the waiting task, issue this command:  

```
F CATALOG,LISTJ(jobname)
```
  - To gather information about all currently executing service tasks, issue this command:  

```
F CATALOG,LIST
```
- ▶ After getting task information, you can redrive the holder of the `SYSZTIOT` so that the waiting task may complete:

```
F CATALOG,END(taskid),REDRIVE
```

If the above command does not resolve the contention, you can use the following command to terminate the waiting task:

```
F CATALOG,ABEND(taskid)
```

- ▶ To list all the tasks currently holding `SYSZTIOT`, issue this command:  

```
D GRS,RES=(SYSZTIOT,*)
```

This is useful if the task with an exclusive hold on SYSZTIOT is not a CAS task. By default, the system lets a task wait for the SYSZTIOT task for 10 minutes before writing a SYMREC record to the logrec data set and issuing message IEC393I. You can specify a different wait time with this command:

```
MODIFY CATALOG,CONTENTION(SYSZTIOT,wait_time)
```

If you specify a non-default SYSZTIOT wait time on the **MODIFY** command, you must respecify that value after each system IPL. A non-default value does, however, persist through restarts of the CAS task.

## 7.3 DEFINE RECATALOG support enhancement

A catalog is a data set that contains information about other data sets. It provides users with the ability to locate a data set by name, without knowing the volume where the data set resides. Thus, data sets can be moved from one device to another, without requiring a change in JCL DD statements that refer to an existing data set.

The basic catalog structure (BCS) 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. When we talk about a catalog, we usually mean the BCS. The VVDS can be considered an extension of the volume table of contents (VTOC). The VVDS is volume-specific, whereas the complexity of the BCS depends on your definitions. The relationship between the BCS and the VVDS is many-to-many. That is, a BCS can point to multiple VVDSs and a VVDS can point to multiple BCSs.

### z/OS V1R12 enhancement

In pre-R12, when using the DEFINE RECATALOG command, the volumes were entered into the catalog record mostly in the specified order. The volume containing the VVR with RBA 0 was first, but after that the prime volumes were in the same order as in the DEFINE VOLUME parameter. When extent constraint relief for VSAM came about, an index was built over internal control blocks that map the extents. The DEFINE RECATALOG caused the index build to fail because VSAM Open expected the volumes in ascending RBA sequence. Not having an extent control block index may present performance problems for VSAM should a data set be recataloged with the volumes out of RBA sequence. Also, some DSS-like ISV products had difficulty with the recataloged data sets.

The solution is to place the volumes in the catalog record in ascending RBA sequence. This new function is in response to FIN APAR OA24010. The IDCAMS DEFINE RECATALOG command is enhanced for multivolume and striped data sets, as follows:

- ▶ Automatically creates catalog entries with correctly ordered volume lists, while eliminating any duplicate volumes that might have been specified.
- ▶ Makes it easier to recatalog multivolume and striped VSAM data sets.

The new support is invoked by the IDCAMS DEFINE RECATALOG command. Catalog entries for volumes are now ordered by RBA starting from 0 for data and index components regardless of the order they are in the DEFINE VOLUMES parameter. Candidate volumes follow the prime volumes. Prime volumes are those that have space assigned to them. There is no need for you to do anything new or different. An example of this support follows:

```
DEFINE CLUSTER -  
    ( NAME(TEST.DS1) -
```



```

INDEXED -
RECATALOG ) -
DATA -
( VOLUME(SMS001 SMS002 SMS003) ) -
INDEX -
( VOLUME(SMS001 SMS003 SMS002) )

```

This enhancement puts the index portion volumes in the following order:

```
SMS001 SMS002 SMS003
```

**Note:** There are some new return and reason codes that indicate when volumes are missing in the user-supplied volume list. Since the catalog now orders the volumes by RBA, it can check to see if there are any gaps. Previously, only the first prime volume could be detected as missing.

### New reason codes for message IDC3009I

Following are the message IDC3009I reason codes for return code 86 (recatalog error).

- |                       |   |
|-----------------------|---|
| <b>Reason Code 24</b> | DEFINE RECATALOG detected that a volume for a data component is missing from the volume list.   |
| <b>Reason Code 26</b> | DEFINE RECATALOG detected that the last volume for a data component is missing from the volume list.  |
| <b>Reason Code 28</b> | DEFINE RECATALOG detected that the last volume of a stripe for a data component of a striped VSAM data set is missing from the volume list.   |
| <b>Reason Code 30</b> | DEFINE RECATALOG detected that a volume for an index component is missing from the volume list.   |
| <b>Reason Code 32</b> | DEFINE RECATALOG detected that the last volume for an index component is missing from the volume list.  |
| <b>Reason Code 34</b> | DEFINE RECATALOG detected that the last volume of a stripe for an index component of a striped VSAM data set is missing from the volume list. |

## 7.4 OPEN/CLOSE/EOV RAS enhancements

Open/Close/EOV (OCE) provides the following enhancements in z/OS V1R12:

- ▶ A new indicator for the DCB ABEND exits ignore option, to suppress the write-to-programmer message that is normally issued when a given Open/Close/EOV abend occurs.
- ▶ Open/Close/EOV also provides the following additional diagnostic improvements in z/OS V1R12:
  - New reason codes for ABENDs 413 and 637 in messages IEC026I and IEC145I, which indicate errors in reading multivolume tape data sets returned by the label anomaly exit.
  - A new IEC988I message indicating the reason for a FREE=CLOSE failure.
  - A new reason code 24 for ABEND code 50D, pointing to IEC999I messages for details on indeterminate errors.

## 7.4.1 New DCB ABEND exit option

When an OCE determinant ABEND is detected and the DCB ABEND exit selects the ignore option, the associated ABEND message is issued even though there is no abnormal termination of the task.

In z/OS V1R12, a new DCB ABEND exit ignore option is provided to additionally bypass the associated determinant ABEND message. This eliminates the contradiction of externalizing an ABEND when the task does not actually abnormally terminate. For example some applications may expect an out of space error to recover via dynamically obtaining more space. This provides more application level control when ignoring OCE determinant ABENDs.

The write-to-programmer messages are problem determination messages for the specific abend as they describe the abend, and normally appear before OCE passes control to the abend exit routine, if any, named in the DCB. The new message suppression option is valid only when you have specified the existing option to ignore the ABEND condition (okay to ignore).

### IFG0199I exit option considerations

Use this exit to bypass the associated determinant ABEND message. By using this new DCB ABEND exit option, you can eliminate messages for OCE determinant ABEND conditions that are subsequently ignored via the DCB ABEND exit. This should improve application level control when ignoring OCE determinant abnormal termination conditions.

## 7.4.2 SMS DASD input when missing last volume

When reading an SMS DASD data set and message IEC710I another volume expected is issued, the job completes successfully even though not all the data is read. New support in z/OS V1R12 attempts to recover this situation by locating the next volume via the catalog to read the volume. If recovery is not possible, then the task is ABENDED with an ABEND 637-BC.

This guarantees that data integrity is maintained by ensuring that when the task completes with condition code zero, all the data for the associated multivolume SMS DASD data set has been read.

Potentially, this problem might be caused by having two DD statements for the same data set and one of them is used to extend the data set to a new volume. The other DD will not reflect the new volume.

With this automatic recovery, you can:

- ▶ Be certain that all SMS DASD data is read when there are multiple DD statements for the same data set within a job step or across steps.
- ▶ Be certain that data integrity when reading multivolume SMS DASD data sets is guaranteed.

## 7.4.3 New reason codes for FREE=CLOSE

A FREE=CLOSE is requested (unallocate when the data set is closed) in the allocation request, but is then bypassed for the following expected reasons:

- ▶ An ABEND is processing.
- ▶ The disposition of the data set is LEAVE or REWIND.
- ▶ The data set open count is not 0.

- ▶ There is a concatenation in process for the data set.

These reasons are not externalized, making analysis of the failure time-consuming and difficult. z/OS V1R12 is now providing a new IEC988I message to externalize the reason for not honoring the FREE=CLOSE request. This simplifies diagnosis of FREE=CLOSE failures.

### **z/OS V1R12 implementation for FREE=CLOSE**

Therefore, when FREE=CLOSE is specified in the JCL or in the CLOSE macro but was not honored, message IEC988I is issued.

```
IEC988I jobname,stepname,ddname{concatenated number},device #,volser,dsn DATA  
SET NOT UNALLOCATED DURING CLOSE RCxx
```

If CLOSE is not deallocating the data set, this causes no problem; this is an informational message and no application change is required. If it does cause a problem, or you want to eliminate the message, correct the application based on the reason for the failure and rerun the job. This provides improved system communication when using the FREE=CLOSE function.

### **New reason codes for FREE=CLOSE**

For the new message IEC988I DATA SET NOT UNALLOCATED DURING CLOSE RCxx, the reason codes are as follows:

- 0 – ABEND in process
- 1 – Disposition of leave
- 2 – Disposition reread
- 3 – DSAB data set open count not zero (DD name has more than one open DCB)
- 4 – Concatenation involved

## **7.5 VSAM partial release on multiple volumes**

Partial release is when unused space is released at the end of an extended format data set. In non-EAV data sets, the starting point for freeing space must be a CA boundary. So, all space after the last used CA boundary up to the high allocated RBA is freed in partial release. In the case of EAV, partial release must start at a multicylinder unit (MCU) boundary. Again, in the case of EAV, partial release will free all space after the last used MCU boundary up to the high allocated RBA.

Partial release is activated on a VSAM CLOSE request, when either the SMS management class or the JCL DD statement (for example, SPACE=(,RLSE)) is set for partial release. Space also can be released when DFSMSHsm is performing space management or when an authorized program issues the PARTREL macro.

### **7.5.1 z/OS V1R12 implementation**

In previous z/OS releases, VSAM partial release did not support releasing unused space that spanned multiple volumes.

z/OS V1R12 provides a means to partial release space in VSAM data sets which are over-allocated and spanning volumes. Currently, the system releases space only on volumes where the high-used RBA and the high-allocated RBA are on the same volume. This means that VSAM partial release today does not support releasing space which spans multiple

volumes. This release extends the current VSAM partial release support to include data sets that span multiple volumes.

Extended format data sets that are over-allocated and span volumes (best fit data sets) are partially releasing space on the last volume containing data and all unused empty volumes will be scratched and marked as candidates.

### **Partial release**

Partial release is used to release unused space from the end of an extended format data set and is specified through SMS management class or by the JCL RLSE subparameter. For data sets whose ending high-used RBA is in track-managed space, all space after the high-used RBA is released on a CA boundary up to the high-allocated RBA. If the high-used RBA is not on a CA boundary, the high-used amount is rounded to the next CA boundary. For data sets whose ending high-used RBA is in cylinder-managed space, all space after the high-used RBA is released on an MCU boundary up to the high-allocated RBA. If the high-used RBA is not on an MCU boundary, the high-used amount is rounded to the next MCU boundary. Partial release restrictions include:

- ▶ Partial release processing is supported only for extended format data sets.
- ▶ Only the data component of the VSAM cluster is eligible for partial release.
- ▶ Alternate indexes opened for path or upgrade processing are not eligible for partial release. The data component of an alternate index when opened as a cluster could be eligible for partial release.
- ▶ Partial release processing is not supported for temporary close.
- ▶ Partial release processing is not supported for data sets defined with guaranteed space.

For extended format data sets, partial release can release unused space across multiple volumes, from the high-used RBA (or next CA/MCU boundary) to the high-allocated RBA. Partial release requires that the primary volumes in the data set be in ascending order by RBA. For example, the first volume could have RBAs 1 to 1000, the next 1001 to 2000, and so on. If the primary volumes appear out of order, partial release issues an error message and releases no space.

### **VSAM partial release**

The partial release of unused data takes effect at VSAM CLOSE, or when invoked by DFSMSHsm space management, and requires that the primary volumes be in ascending order. Partial release applies only to extended format data sets, and to the data component of the VSAM cluster. It does not apply to:

- ▶ Alternate indexes opened for path or upgrade processing
- ▶ Temporary close
- ▶ Data sets defined with guaranteed space

With partial release, CLOSE releases the unused space only on the current volume except for VSAM extended format data sets or for a striped data set that has a stripe count of greater than 1. For VSAM extended format data sets, partial release at CLOSE releases unused space from multiple volumes as follows:

- ▶ In non-EAV data sets, the starting point for freeing space is at a CA boundary; all space after the last used CA boundary is freed in a partial release, up to the high-allocated RBA.
- ▶ In EAV data sets, the starting point for freeing space is an MCU boundary; all space after the last used MCU boundary is freed up to the high-allocated RBA.

## Example of the new function

Consider the example shown in Figure 7-1 as the general mechanism that is now used to solve partial release when the starting and ending locations for freeing space do not reside on the same volume. A 100-cylinder data set is defined with best fit in an SMS management class where partial release is active. SMS has selected a storage group with three volumes, as shown in Figure 7-1 (volumes A, B, and C). Each volume has 40 cylinders that are currently empty. 60 cylinders worth of data is written to the data set prior to closing.

Upon closing, the system retrieves the high-used RBA and the high-allocated RBA. The starting RBA for freeing data for non-EAV data sets is rounded up to the next CA boundary, and in the case of an EAV volume, the starting RBA for freeing data is rounded up to the next multicylinder unit (MCU).

Now, each primary volume is inspected in order and processed. It is expected that the primary volumes will be in order. This means that the RBAs from volume to volume will be in ascending order. For example, the first volume could be 1 to 1000, the next 1001 to 2000, and so on. If the primary volumes appear out of order, an error is generated and no space is released.

In the example, all data in volume A is below the calculated starting RBA and is not processed. Volume B has both used and unused space. In this case, all of the unused space in volume B above the calculated starting RBA is freed in a call to DADSM. Additionally, the local RBAs are updated. Lastly, in volume C the data set is scratched and the volume is returned to the candidate list. Returning to the candidate list evolves cleaning up the BCS volume information associated with this data set and making the volume a potential candidate for future use when extending.

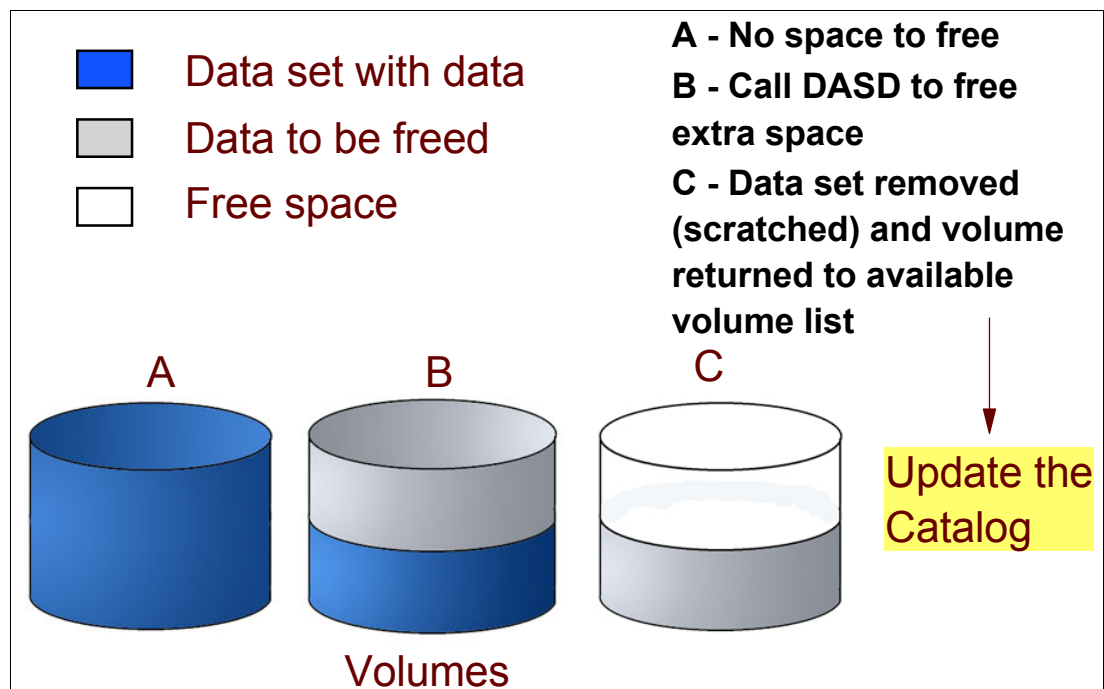


Figure 7-1 Example of the new function

**Note:** The best way to see this enhancement to partial release is to allocate a best fit data set, which must then span volumes. Do not fill the data set to the last volume. Then close the data set having either specified partial release in SMS or in the job. To request partial release for this DD statement, code SPACE=(,RLSE) even if the data set exists.

## 7.5.2 z/OS V1R12 considerations

Before completing, the catalog for the data set is updated with the new data set characteristics. Before z/OS V1R12, with the current implementation, the data set format1 DSCB and VVDS VVR would be orphaned on volume C, also leaving the BCS data to be inaccurate concerning this volume. This problem occurs only in the scenario of best fit where over-allocation occurs and the close is expected to shrink to the used size.

With z/OS V1R12, the new support is invoked normally when any of the following occurs:

- ▶ In a VSAM CLOSE request, when either the SMS management class specifies partial release or the JCL DD statement specifies it with SPACE=(,RLSE)).
- ▶ When DFSMSHsm performs space management.
- ▶ An authorized program issues the PARTREL macro.

DADSM supports the release of unused space that is allocated to sequential or partitioned data sets, PDSEs, sequential extended-format data sets, and VSAM extended-format data sets. The partial release function is called when:

- The data set is closed (if the RLSE subparameter of SPACE was specified on its DD statement or the storage administrator specified an appropriate value for the partial release option in the management class definition).
- A restart is processing from a checkpoint taken before the data set was extended.
- DFSMSHsm performs a space management cycle and an appropriate value is specified in the management class definition.
- A PARTREL macro is issued.

**Note:** The PARTREL macro supports sequential and partitioned data sets on volumes with or without indexed VTOCs. It supports PDSEs and extended format data sets on SMS-managed volumes for which indexed VTOCs are required.

### Requirements for compression

The following requirements apply for compression of extended format data sets:

- ▶ The data set must already have met the requirements for extended format.
- ▶ Compression management services requires the data set to have a primary allocation of at least 5 MB in order to be allocated as a compressed data set.
- ▶ Compression management services requires the data set to have a primary allocation of 8 MB if no secondary allocation is specified, in order to be allocated as a compressed data set.
- ▶ Compression management services requires that the data set have a minimum record length of 40 bytes, not including the key offset or key length.

For VSAM KSDS, the compression management services (CMS) requirement to have a primary allocation of five megabytes is due to the amount of sampling needed to develop a dictionary token. The 5 MB allocation must be for the data component only. If the amount is specified at the cluster level, then the index component will use a portion of the 5-MB

allocation. This will result in the data set not meeting the CMS space requirement and the data set will not be eligible for compression processing. This is also true for the 8 MB primary allocation, if no secondary is specified.

The CISIZE for a nonspanned compressed KSDS must be at least 10 bytes larger than the maximum record length.

The VSAM KSDS data set is not eligible for compression because the CISIZE is not large enough to contain the data record's key field. When a data record spans CIs, the record's key field must be contained within the first CI. The CISIZE for a spanned compressed KSDS must be at least 15 bytes larger than the key field.

## 7.6 VSAM enhancements

VSAM enhancements in z/OS V1R12 provide the ability to automatically reclaim empty CA space for key-sequenced data sets (CA Reclaim), which allows access to striped data sets from VSAM/RLS, and improves the usability of VSAM record management trace.

To enable the trace dynamically when diagnostic data is needed, z/OS V1R12 introduces the following enhancements for VSAM and VSAM RLS:

- ▶ Control area (CA) reclaim
- ▶ Disk striping support
- ▶ Dynamic trace enhancements

### 7.6.1 Control area (CA) reclaim

You can now request that empty CA space for a key-sequenced data set (KSDS) be reclaimed automatically. In previous releases, VSAM KSDSs (key-sequenced data sets) must be “reorganized” on a regular basis in order to do the following:

- ▶ Reclaim space previously used by deleted records
- ▶ Improve sequential read performance
- ▶ A “reorganization” consists of the following functions:
  - Closing the KSDS
  - Unloading the KSDS to a backup file
  - Deleting and redefining the KSDS
  - Reloading the file from the backup file
  - Reopening the KSDS

As a result, this affects many aspects of a z/OS environment, such as:

- ▶ Customer application data
- ▶ IMS and CICS files
- ▶ Catalogs
- ▶ Control data sets for many IBM products such as DFSMSHsm and DFSMSrmm

#### **z/OS V1R12 VSAM improvements**

This release enhances VSAM and VSAM RLS to reclaim empty space (control areas) in KSDSs. This applies to SMS and non-SMS data sets and resolves the main reasons for

KSDS reorganizations to be able to reclaim space and improve both sequential and direct performance.

To implement this, empty sequence sets and high-level index records will be placed on a free list after the last record in the CA is erased. The reclaimed CA may then be reused as new records are inserted anywhere in the data set.

Pre-existing empty CAs will not be reclaimed by CA reclaim. Once all sharing systems are upgraded to z/OS V1R12 and enabled for the CA reclaim function, the data sets may need to be reorganized one last time to reclaim any pre-existing empty CAs.

**Benefit:** CA reclaim will address the main reasons for reorganizations that will help make data sets available 24 hours a day.

### IGDSMSxx parmlib member

Because CA reclaim is disabled by default on a system level, but is enabled by default for all KSDSs without having to redefine the data set, which allows for:

- ▶ CA reclaim to be enabled on a system level or on a data set level.
- ▶ A new system level parameter in the IGDSMSxx parmlib member to enable this function:

```
CA_RECLAIM({NONE | DATACLAS | DATACLASS})
```

These parameters specify whether or not to use CA reclaim for KSDSs according to the CA reclaim attribute in their data classes, as follows:

#### **NONE**

Indicates that none of the KSDSs will be using CA reclaim, regardless of the data class specification. The default is NONE. If CA\_RECLAIM is not specified in SYS1.PARMLIB, the CA\_RECLAIM(NONE) default will be in effect, which means the default is CA reclaim disabled.

#### **DATACLAS | DATACLASS**

Indicates that the SMS-managed KSDSs and non-SMS-managed KSDSs will go by the data class specification of CA reclaim=YIN at define time or subsequent ALTER setting. Only the resultant catalog setting of CA Reclaim=Y will do CA reclaim; CA Reclaim=N will not do CA reclaim. It is the way to specify some KSDSs not to use CA reclaim. **Default:** NONE

### DFSMS data class considerations

The CA reclaim attribute in data classes defined prior to z/OS V1R12, when CA reclaim was introduced, defaults to Yes. If CA reclaim is disabled for all KSDSs on this system when CA\_RECLAIM(NONE) is specified in the IGDSMSxx parmlib member, or a SETSMS CA\_RECLAIM(NONE) command is issued, and one of the following occurs:

- ▶ The system IPLs.
- ▶ RLS VSAM address space is recycled.
- ▶ A SET SMS=xx command is issued.

IDCAMS ALTER can be used to disable or enable individual KSDSs after DEFINE time as follows:

```
ALTER 'ksdsname' RECLAIMCA/NORECLAIMCA
```



**Note:** ALTER does not immediately disable or enable CA reclaim, unlike the SETSMS command. ALTER immediately updates the catalog but the attribute in the catalog is not looked at by the system until the next time the control block structure is built, which means it will not take effect until all the current OPENs against the control block structure have been closed and a subsequent first OPEN is issued.

### CA reclaim considerations

An example of a data set that should benefit greatly from CA reclaim is one where many records are deleted. CA reclaim provides for improved DASD space usage, but it requires additional I/O to keep track of the reclaimed CAs so that they can be reused. Some special cases that might have a small performance degradation are:

- ▶ Many erases but very few or no PUTs that will cause CA splits. The overhead of maintaining the free CAs may exceed the faster index search.
- ▶ The erased records are reinserted to bring back the CAs that CA reclaim has labored to put on the free chains for reuse.
- ▶ Many applications have been designed around the reorg problem and have reorgs built into the logic. IBM suggests stopping the deleting and reloading of these files.

Consider the following possibilities:

- ▶ KSDSs can be classified into three groups with respect to CA reclaim:
  - CAs are never completely erased. Using CA reclaim will have no impact.
  - CAs are emptied; however, the similar record key ranges are reinserted in a timely manner, reusing the CA. Using CA reclaim may cause a slight performance impact.
  - CAs are emptied and erased and key ranges are not reinserted in a timely manner (if ever). Using CA reclaim will provide performance and space improvements.
- ▶ Most KSDSs will likely benefit from CA reclaim, but to give more control to applications, all KSDSs can be enabled or disabled on an individual basis.

**Restrictions:** CA reclaim cannot reclaim space for the following:

- ▶ Partially empty CAs
- ▶ Empty CAs that already existed when CA reclaim was enabled
- ▶ CAs with RBA 0
- ▶ CAs with the lowest and highest key of the KSDS
- ▶ Index CIs with the highest keys in each index level
- ▶ Data sets processed with global shared resources (GSR)
- ▶ Existing data sets with the IMBED or REPLICATE attributes

**Note:** Reclaimed space remains in the allocated space for the data set and the tracks are not released.

### LISTCAT output for CA reclaim

LISTCAT output will indicate whether CA reclaim is enabled for this data set (unless disabled by the system option); see Figure 7-2 on page 140.

```

SMSDATA
STORAGECLASS ---SXPXS02   MANAGEMENTCLASS---(NULL)
DATACLASS  -----KSCR000Y   LBACKUP ---0000.000.0000
CA-RECLAIM----- (YES)   BWO STATUS-----00000000   BWO TIMESTAMP---00000
00:00:00.0
BWO----- (NULL)

```

Figure 7-2 LISTCAT output showing the SMSDATA with CA-RECLAIM

## DATATEST output

DATATEST reads the sequence set from the index component and the entire data component of the KSDS cluster. So, it should take considerably more time and more system resources than INDEXTEST.

If you are using EXAMINE to document an error in the data component, run both tests. If you are using EXAMINE to document an error in the index component, it is usually not necessary to run DATATEST.

If you are using EXAMINE to confirm a data set's integrity, your decision to run one or both tests depends on the time and resources available.

EXAMINE DATATEST output will indicate how fragmented the data set is by listing the current count of empty (un-reclaimed CAs).

```
IDC01728I FOUND n EMPTY CONTROL AREAS THAT HAVE NOT BEEN RECLAIMED
```

The existing EXAMINE DATATEST or DATATEST INDEXTEST message will continue to display CAs that have been orphaned, but it will become a "minor" error (RC=4), not "major" (RC=8) because the situation by itself only loses DASD space but does not cause data integrity problems, as follows:

```
IDC11724I DATA COMPONENT CA NOT KNOWN TO SEQUENCE SET
```

For z/OS V1R12, the message is changed as follows:

**Explanation:** Some abnormal situations, such as ABENDs or logical errors, can cause a control area (CA) split or CA reclaim to be interrupted. A logical error results in VSAM backing out the CA split that is in progress and returning a nonzero return code to the PUT that caused the CA split. When a CA split is aborted in an abnormal condition, an unreachable new CA (commonly referred to as an orphan CA) can be left behind within the data set. This in itself does not cause a data integrity problem and the data set is fully functional and accessible.

IDCAMS EXAMINE INDEXTEST DATATEST generates the IDC11724I message and sets the return code to 4 if only the orphan CA condition is present and no data integrity problems are detected. However, if the EXAMINE also detects data integrity problems, it will also display other messages and will return with return code = 8 (and not 4). Because EXAMINE INDEXTEST DATATEST checks both the index and data components, its return code is more accurate in reflecting the true severity of the situations it detects than that of EXAMINE DATATEST NOINDEXTEST; it is also more thorough in detecting errors.

Figure 7-3 on page 141 shows an output example.

```

EXAMINE NAME(XXXXXXXX) INDEXTEST DATATEST
IDC01700I INDEXTEST BEGINS
IDC11724I DATA COMPONENT CA NOT KNOWN TO SEQUENCE SET
IDC21700I MINOR ERRORS FOUND BY INDEXTEST
IDC01701I DATATEST BEGINS
IDC11768I [CI SPLIT | CA RECLAIM] IN PROGRESS
IDC01713I DATA CONTROL INTERVAL DISPLAY AT RBA
yyy FOLLOWS
...
IDC01714I ERROR LOCATED AT OFFSET zzzzzzzz
IDC11724I DATA COMPONENT CA NOT KNOWN TO SEQUENCE SET
IDC21703I MINOR ERRORS FOUND BY DATATEST
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 4

```

Figure 7-3 Example of message IDC01724I

## SMF recording

SMF type 64 record counts how many CAs have been reused by CA splits (not reclaimed) since the data set was defined. The following field contains the information:

SMF64DAU - 32 bit integer

## IDCAMS VERIFY RECOVER option

You can use the IDCAMS VERIFY command to verify a VSAM data set. When you issue this command, IDCAMS opens the VSAM data set for output, issues a VSAM VERIFY macro call, and closes the data set. The IDCAMS VERIFY command and the verification by VSAM OPEN are the same. Neither changes the data in the verified data set.

It completes any interrupted CA reclaim. The use of this command is optional in that the following cases will do CA reclaim recovery automatically:

- ▶ Non-RLS or RLS VSAM PUT
- ▶ An ERASE that starts a new CA reclaim
- ▶ An index search that runs into an index CI that was involved in an interrupted CA reclaim
- ▶ RLS recovery that automatically gets control as a result of the interruption completes the interrupted CA reclaim

Even if CA reclaim recovery is not done, with the partially completed CA reclaim the data set is still 100% accessible.

## Migration considerations

CA reclaim will be recognized and not supported in earlier releases. PTFs for these APARs will be available for coexistence for CA reclaim, as follows:

- ▶ OA25108 (RLS)
- ▶ OA26256 (IDCAMS)
- ▶ OA26466 (VSAM)
- ▶ OA27557 (SMS)

Apply these PTFs before or at the same time that you go to z/OS V1R12.

## 7.6.2 Disk striping support

A striped data set has tracks that spread across multiple devices, as is the case for the sequential access method or the CIs for VSAM. This format allows a single application request for records in multiple tracks or CIs to be satisfied by concurrent I/O requests to multiple volumes. The result is improved performance for sequential data access by achieving data transfer into the application at a rate greater than any single I/O path. The scheduling of I/O to multiple devices to satisfy a single application request is referred to as an I/O packet.

VSAM data striping applies only to data sets that are defined with more than one stripe. Any data set listed with one stripe is in the extended format and is not considered to be a striped data set.

**Restriction removed:** In z/OS V1R12, the previous restriction on accessing striped data sets from VSAM RLS is removed. All current VSAM striping rules now apply to VSAM RLS.

## 7.6.3 Dynamic trace enhancements

In previous releases, you could enable VSAM record management trace only by specifying the trace parameter in the JCL DD card:

```
AMP=( 'TRACE=(subparameters)')
```

That method limited VSAM record management trace from supporting data sets that are allocated dynamically, such as catalog data sets. The previous method also limited users from enabling the trace dynamically and required users to take down the application to insert trace parameters and open the data set before the trace could be enabled.

VSAM serviceability is enhanced by improving the usability of VSAM record management trace. A new IDAVDTxx parmlib member is introduced to enable VSAM record management trace dynamically, without using the DD card of the JCL. This new interface extends the usability of VSAM record management trace to support data sets that are allocated dynamically and allow the user to enable VSAM record management trace without taking the application or data set offline. New trace functions are also added.

### IDAVDTxx parmlib member

You can predefine a maximum of eight VSAM record management trace entries in an IDAVDTxx parmlib member. Each trace entry consists of the target data set name, the target job name, and the usual trace parameters such as Hook, PARM1, PARM2, and so on. After the trace entries are defined, you can invoke the new started task, IDAVDT, with the START console command to enable VSAM record management trace. Once you start IDAVDT, you can interact with it with the MODIFY command to invoke different functions.

The parameters for the IDAVDTxx parmlib member are:

<b>VTRACE</b>	Specifies that the current entry is to define a VSAM record management trace entry.
<b>DSNAME</b>	<b>(data_set_name)</b> - Specifies which data set the user would like to enable VSAM record management trace.
<b>JOBNAME</b>	<b>(job_name)</b> - Specifies the job where the targeted data set was opened or will be opened.
<b>HOOK</b>	<b>(hook_id [,hook_id])</b> - Specifies the list of VSAM predefined hook points where the trace should execute during VSAM record management

processing. This statement consists of 1 to 3 other statements (SLIP, DUMP, or ABEND).

- PARM1** (trace\_parm\_1) - Specifies which control blocks VSAM record management trace should capture.
- PARM2** (trace\_parm\_2) - Specifies which control blocks VSAM record management trace should capture when tracing the AIX.
- ECODE** (ANY | rsnocode) - Specifies that tracing should occur only if an error code is being returned to the caller.
- ANY** - Tracing is performed for any nonzero return code.
- rsnocode** - Tracing is performed only if the RPLFDBK code matches the rsnocode.

**Note:** The default member is IDAVDT00.

### Operator command for trace

Each trace entry consists of the target data set name, the target job name, and the usual trace parameters. After you start IDAVDT, you can interact with IDAVDT with the MODIFY command to perform the following functions for VSAM dynamic trace parameters:

- READIN** Parse the trace entries in the specified IDAVDTxx parmlib member and insert it into the dynamic trace save area located in common storage.
- ENABLE** Enable a specific trace entry that is stored in the dynamic trace area.
- DISABLE** Disable a specific trace entry that is stored in the dynamic trace area.
- DISPLAY** Display a specific trace entry that is stored in the dynamic trace area.
- VALIDATE** Validate a specific trace entry, or all trace entries, that are stored in the dynamic trace area.
- INVALIDATE** Invalidate a specific trace entry, or all trace entries, that are stored in the dynamic trace area.

**Note:** You can predefine up to the maximum of 8 trace entries with different trace parameters. Each trace entry consists of the target data set name, the target job name, and the usual trace parameters. When the trace entries are defined, you can use the START console command to call IDAVDT to enable VSAM record management trace.

### Installation considerations

Take the following steps to use this new interface:

- ▶ Before using the new VSAM dynamic trace started task, ensure that the shipped IDAVDT proclib member and the IDAVDT00 parmlib member are updated in your parmlib and proclib concatenations.
- ▶ Read about the new parameters and values for the new IDAVDT proclib member and the IDAVDTxx parmlib member in *z/OS DFSMSdfp Diagnosis*, GY27-7618 and *z/OS MVS Initialization and Tuning Reference*, SA22-7592.
- ▶ Initialize the new proclib and parmlib members with appropriate entries for the desired trace or traces.
- ▶ Include these new proclib and parmlib members in your working library.
- ▶ Copy the IDAVDTxx parmlib member from the data set pointed to by your parmlib DD definition to the system's working parmlib data set.

- ▶ Copy the IDAVDT PROCLIB member from the data set pointed to by your PROCLIB DD definition to the system's working PROCLIB data set.
- ▶ Invoke the new started task IDAVDT using the START console command.

## 7.7 Copy storage group volumes

To simplify the creation of new SMS classes, storage groups, aggregate groups, and SCDSs, you can copy existing ones and modify them.

However, the volumes defined in the storage groups cannot be copied. The storage administrator will have to manually add volumes to the storage groups. This process can be difficult at times.

With z/OS V1R12, using ISMF, a new option under the Copy function for pool type storage group function is provided to allow you to indicate whether the volume list of the from storage group should also be copied/added to the target storage group. The default is NO. To access this new option, do the following:

1. From the ISMF Primary panel (panelid DGTSMMD2), select Option 6.
2. (Panelid DGTSCSG1) - Enter your CDS Name and select Option 1. Generate a list of storage groups (names shown are in our system).
3. (Panelid DGTLGP23) - Select a STORGRP NAME (**EAV**) and type copy, as shown in Figure 7-4.

LINE OPERATOR	STORGRP NAME	SG TYPE	VIO MAXSIZE	VIO UNIT	AUTO MIGRATE
--- (1) ---	-- (2) ---	----- (3) -----	-- (4) --	(5) -	-- (6) ---
	DUMPS	POOL	-----	----	NO
<b>copy</b>	<b>EAV</b>	POOL	-----	----	NO
	LOGR	POOL	-----	----	NO
	NULLGRP	POOL	-----	----	NO
	OPENMVS	POOL	-----	----	NO
	SEQ	POOL	-----	----	NO
	SEQNEW	POOL	-----	----	NO
-----	-----	-----	BOTTOM	OF	DATA

Figure 7-4 Storage group list

4. Panel DGTDCCA2 is displayed. Enter a "/" to do the copy and specify a new Construct Name, EAV1, as shown in Figure 7-5 on page 145.

```

Panel  Utilities  Help
-----
DGTDCCA2                COPY ENTRY PANEL
Command ===>

Construct will be Copied from:

  Data Set Name . . : SYS1.STPPLEX.SCDS
  Construct Name  . : EAV
  Construct Type  . : STORAGE GROUP

Specify "Copy To" Construct:

  Data Set Name . . . 'SYS1.STPPLEX.SCDS'
                          (1 to 46 Characters)

  Construct Name . . EAV1_____
                          (If Copy Pool - 1 to 23 characters, fully specified)
                          (Others - 1 to 8 characters, fully specified)
Enter "/" to select option  - Replace like-named Construct
                          - Perform Alter
                          / Copy Storage Group Volumes (Pool SG only)

```

Figure 7-5 Copy storage group panel with new option

If duplicate volume names are found, SMS returns reason code CSRDUVPV(6707) and lists the volume name in the ISMF log data, without adding it to the target storage group; see Figure 7-6 on page 146 and Figure 7-7 on page 146.

```

Panel  Utilities  Help
-----
                                COPY ENTRY PANEL          SMS RETCODE:      4
Command ===> _____

Construct will be Copied from:

Data Set Name . . . : SYS1.STPPLEX.SCDS
Construct Name . . . : SEQ
Construct Type . . . : STORAGE GROUP

Specify "Copy To" Construct:

Data Set Name . . . 'SYS1.STPPLEX.SCDS'
                                (1 to 46 Characters)

Construct Name . . . SEQNEW
                                (If Copy Pool - 1 to 23 characters, fully specified)
                                (Others - 1 to 8 characters, fully specified)
Enter "/" to select option      Replace like-named Construct
                                Perform Alter
                                / Copy Storage Group Volumes (Pool SG only)

Use ENTER to Perform COPY;
Use HELP Command for Help; Use END Command to Exit.

-----
- RSNCODE:          6707 CSECT:IGDCSC50 SUBRETCODE:          0 SUBRSNCODE: -
- 0                                                         e
-----

```

Figure 7-6 Message indicating error 6707 in Copy storage group

```

SEQ
RETURN CODE(0004); REASON CODE(6707)
MODULE(IGDCSC50); PROCEDURE()
MESSAGE ID( - ); LAST PANEL(DGTDCCA2)
SERVICE(SMS/CS); FEEDBACK(SUBRET:          0 SUBRSN:          0)
APPLICATION(DGT6 - SG); FUNCTION(SG COPY)
SEQ
RETURN CODE(0004); REASON CODE(1327)
MODULE(DGTFCFCV); PROCEDURE(DGTFCFCV)
MESSAGE ID(DGTUC051 - DGTUC051); LAST PANEL(DGTDCCA2)
SERVICE(CSI); FEEDBACK(SG COPY VOLUME , VOLUME SBOX3W)
APPLICATION(DGT6 - SG); FUNCTION(SG COPY)

```

Figure 7-7 List log ISMF with duplicate volume Dump Stacking Volume Limit Expansion

The DFSMSshm dump function copies source disk volumes, in their entirety, to a target tape volume. The dump function allows stacking multiple source volumes on a single target tape volume. Previously, the dump function allowed up to 99 source disk volumes to be dumped to a single target tape volume. With increasing tape capacity, this limit has been increased, and in z/OS V1R12, the maximum number of disk volumes allowed to be stacked on a single tape



volume is increased from 99 to 255 volumes. This provides better utilization of the capacity of the current tape technology.

To use this enhancement, specify the STACK parameters with the value of up to 255 on the DFSMSHsm BACKVOL DUMP OR DEFINE DUMPCLASS commands, as shown in Figure 7-8.

```
DEFINE DUMPCLASS(MONTHLY DAY(1) RETPD(180) AUTOREUSE NORESET STACK(255)-  
DATASETRESTORE VTOCCOPIES(4) )  
  
F DFSM, BACKVOL VOLUMES(DATA01, DATA02, DATA03) DUMP(DUMPCLASS(MONTHLY))  
STACK(255)
```

Figure 7-8 Show the new value on the commands

**Note:** A coexistence PTF is required for z/OS V1R10 and later DFSMSHsm hosts within an HSMplex to tolerate the increased dump stacking volume limit.

Systems prior to z/OS V1R12 with the coexistence PTF can process a dump volume with more than 99 volumes (for example, when using RECOVER, AUDIT, or list).

Defining a dump class with a dump stacking value greater than 99 and specifying a dump stacking value greater than 99 using the BACKVOL command can be done on z/OS V1R12 only. Releases prior to z/OS V1R12, with the applicable PTF, will support a value greater than 99, but will fail and issue a message if an attempt is made to define a value greater than 99 on the pre-V1R12.

## 7.8 VOLSELMSG and TRACE parameters

To understand how to use the VOLSELMSG and related parameters for Volume Selection Messages and Traces, see the IGDSMSxx parmlib member in *z/OS MVS Initialization and Tuning Reference*, SA22-7592.

### IGDSMSxx parmlib member

In the IGDSMSxx parmlib member, using the VOLSELMSG keyword allows you to control volume selection analysis messages issued when you create or extend an SMS-managed data set to a new volume. These analysis messages are written to the hardcopy log and the job log.

Currently, SMS TRACE and VOLSELMSG facilities share three parameters:

TYPE, ASID and JOBNAME

Specification of these shared parameters applies to both facilities simultaneously. There are situations where the operator may need to use TRACE and VOLSELMSG on different events, ASID and JOBNAME without affecting each other.

The new support in z/OS V1R12 allows different values to be specified on the shared parameters for TRACE and VOLSELMSG facilities. This allows the operator to have better control over SMS TRACE and VOLSELMSG diagnostic tools.

The keyword, VOLSELMSG, is defined as shown in Figure 7-9 on page 148 before z/OS V1R12.

**VOLSELMSG({ON | OFF | 0 | nnnnn | ALL})**

**ON | OFF** Controls whether SMS volume selection analysis messages are to be issued.

**Default:** OFF

**0|nnnnn|ALL** Controls whether SMS volume selection analysis messages are to be issued.

**Default:** 0

**0** - Indicates only summarized analysis messages are to be issued.

**nnnnn** - The number of volumes to be included in the message with a range of 0 to 65535.

**ALL** - Indicates that all volumes used for volume selection will be included in detailed analysis messages.

If you specify VOLSELMSG(nnnnn), where nnnnn has a value greater than 0, with TYPE(ALL), you must also specify one of the following parameters to limit the number of detailed analysis messages:

- ▶ JOBNAME
- ▶ ASID
- ▶ STEPNAME
- ▶ DSNAME

When all volumes are to be included, volumes are listed by storage group. If only a subset of volumes is to be included, volumes are listed in volume selection preference order without an association to storage group. The system can issue an excessive number of analysis messages to the spool for the following conditions:

- ▶ The job or address space creates or extends many SMS-managed data sets
- ▶ Many volumes are to be included in the analysis messages.

Figure 7-9 VOLSELMSG keyword before z/OS V1R12

## 7.8.1 z/OS V1R12 enhancements

The changes in z/OS V1R12 are for the SMS TRACE and VOLSELMSG facilities because they are enhanced to support different specifications on the shared parameters TYPE, ASID, and JOBNAME.

The SMS command SETSMS VOLSELMSG(ON) can be used to request summarized and detailed analysis messages on volume selection, if volume selection is not prematurely terminated by an error. These analysis messages can assist you to perform problem diagnosis on volume selection. See *z/OS MVS System Commands*, SA22-7627 for information about using the SETSMS VOLSELMSG(ON) command.

The keywords TYPE, ASID, and JOBNAME are used to specify the scope for SMS TRACE and VOLSELMSG facilities.

### SETSMS command

Use the SETSMS command when the Storage Management Subsystem (SMS) is active (running) to change a subset of SMS parameters from the console without changing the active IGDSMSxx member of SYS1.PARMLIB.

To display the setting of the parameters that are related to SMS volume selection analysis messages, enter:

**D SMS,VOLSELMSG**

```
The response to this command is as follows:  
IGD002I 13:58:46 DISPLAY SMS 789  
VOLSELMSG = (OFF,0) TYPE = ERROR JOBNAME = *  
ASID = * STEPNAME = *  
DSNAME = *  
TRACE = ON SIZE = 128K TYPE = ERROR  
JOBNAME = * ASID = *
```

**New parameter syntax specifications**

The values of TYPE, ASID, or JOBNAME parameters are set in an IGDSMSxx parmlib member or also using the SETSMS command. With z/OS V1R12, there is new syntax for the following parameters, as shown in Figure 7-10 and Figure 7-11 on page 150.

**TYPE(ERROR|ALL[(TRACE|T|VOLSELMSG|V) [,ALL|ERROR(TRACE|T|VOLSELMSG|V)])]**

- Specifies how you want to trace events (TRACE) and issue volume selection analysis messages (VOLSELMSG).

**ERROR**

Specify ERROR to trace error events and issue volume selection analysis messages (VOLSELMSG(ON)) for allocations that have failed.

**ALL**

Specify ALL to trace all events and issue volume selection analysis messages (VOLSELMSG(ON)) for all allocations.

**[(TRACE|T|VOLSELMSG|V) [,ALL|ERROR(TRACE|T|VOLSELMSG|V)]]**

These are optional sub-parameters. TRACE|T or VOLSELMSG|V associated with the first sub-parameter specifies whether the required value, ALL or ERROR, specified in the first sub-parameter applies to the TRACE or VOLSELMSG facility. The second sub-parameter is optional and can be used to specify another value and facility after the first sub-parameter is specified. When none of these optional sub-parameters are specified, the value specified in the first sub-parameter applies to both TRACE and VOLSELMSG. For example, if you want to set a TYPE value of ERROR for the SMS TRACE facility, and a TYPE value of ALL for the VOLSELMSG facility, you could code:

**TYPE(ERROR(TRACE),ALL(VOLSELMSG)).**

Figure 7-10 New syntax with z/OS V1R12 for the TYPE keyword

**ASID(asid|\*[(TRACE|T|VOLSELMSG|V) [,asid|\*(TRACE|T|VOLSELMSG|V)]])**

Specifies whether SMS is to limit tracing (**TRACE(ON)**) or issue volume selection messages (**VOLSELMSG(ON)**) to a specific address space (asid) or all address spaces **\***.

**asid|\***

Limit tracing and volume selection analysis messages for a certain address space, or all address spaces (**\***). You can enter up to 4 digits for the ASID keyword. If you leave off the leading zeros, they are inserted.

**[(TRACE|T|VOLSELMSG|V) [,asid|\*(TRACE|T|VOLSELMSG|V)]]**

These are optional sub-parameters. TRACE|T or VOLSELMSG|V associated with the first sub-parameter specifies whether the required value, asid or **\***, specified in the first sub-parameter applies to TRACE or VOLSELMSG facility. The second sub-parameter is optional and can be used to specify another value and facility after the first sub-parameter is specified.

When none of these optional sub-parameters are specified, the value specified in the first sub-parameter applies to both TRACE and VOLSELMSG.

For example, if you want to have ASID 0010 for the SMS TRACE facility, and ASID 0020 for the VOLSELMSG facility, you could code:

ASID(10(TRACE),20(VOLSELMSG)).

**Default:** **\***

**JOBNAME(jobname|\*[(TRACE|T|VOLSELMSG|V) [,jobname|\*(TRACE|T|VOLSELMSG|V)]])**

Specify whether SMS is to limit tracing (TRACE) and volume selection analysis messages (VOLSELMSG(ON)) for a certain job (jobname), or permit tracing and volume selection analysis messages on all jobs.

This keyword supports objects or tape libraries.

**jobname|\***

Limit tracing and volume selection analysis messages for a certain job, or all jobs (**\***). Specify **"\*"** to issue for all jobs.

**[(TRACE|T|VOLSELMSG|V) [,jobname|\*(TRACE|T|VOLSELMSG|V)]]**

These are optional sub-parameters. TRACE|T or VOLSELMSG|V associated with the first sub-parameter specifies whether the required value, jobname or **\***, specified in the first sub-parameter applies to TRACE or VOLSELMSG facility. The second sub-parameter is optional and can be used to specify another value and facility after the first sub-parameter is specified.

When none of these optional sub-parameters are specified, the value specified in the first sub-parameter applies to both TRACE and VOLSELMSG.

For example, if you want to have all jobs for the SMS TRACE facility, and a particular job, JOB111, for the VOLSELMSG facility, you could code:

JOBNAME(\*(TRACE),JOB111(VOLSELMSG)).

**Default:** **\***

Figure 7-11 New syntax with z/OS V1R12 for the ASID and JOBNAME keywords

## Parameter syntax rules

The parameters can be used as follows:

TYPE(ALL|ERROR[(T|V) [,ALL|ERROR(T|V)]])

ASID(asid|\*[(T|V) [,asid|\*(T|V)]])

JOBNAME(jobname|\*[(T|V) [,jobname|\*(T|V)]])

Each parameter is allowed to have up to 2 sets of subparameters, as follows:

T|TRACE or V|VOLSELMSG are valid facility name

When none of the optional subparameters is specified, the value specified in the first subparameter applies to both TRACE and VOLSELMSG as in previous releases, as follows:

**TYPE(ERROR)**

- Both TRACE and VOLSELMSG are limited to the ERROR event

**ASID(20)**

- Both TRACE and VOLSELMSG are limited to ASID 20

**JOBNAME(JOB111)**

- Both TRACE and VOLSELMSG are limited to job name JOB111.

### TRACE and VOLSELMSG parameter examples

The values for the TYPE, ASID, and JOBNAME parameters are set in an IGDSMSxx parmlib member or on a SETSMS command. The first optional subparameter, T | V, specifies whether the first set of subparameters applies to TRACE or VOLSELMSG, for example:

**TYPE(ERROR(T))**

- TRACE facility is limited to trace ERROR event

**ASID(20(V))**

- VOLSELMSG facility is limited to ASID 20.

**JOBNAME(JOB111(TRACE))**

- TRACE facility is limited to job name JOB111

The second set of subparameters is optional and can be used to specify another value and facility after the first set of subparameters is specified, for example:

**TYPE(ERROR(T),ALL(V))**

- TRACE facility is limited to trace ERROR event
- VOLSELMSG facility is allowed on ALL event

**ASID(20(VOLSELMSG),\*(TRACE))**

- VOLSELMSG facility is limited to ASID 20
- TRACE facility is allowed on all address spaces

**JOBNAME(JOB111(TRACE),JOB222(V))**

- TRACE facility is limited to job name JOB111
- VOLSELMSG facility is limited to job name JOB222

### Using the SETSMS command

You can use the SETSMS command to control how to issue the volume selection analysis messages (Figure 7-12). If you want to issue detailed analysis messages for all volumes on failure allocations, enter:

```
SETSMS VOLSELMSG(ON,ALL) TYPE(ERROR)
```

```
SETSMS VOLSELMSG(ON,ALL) TYPE(ERROR)
IEE712I SETSMS PROCESSING COMPLETE

DISPLAY SMS,VOLSELMSG
IGD002I 13:50:25 DISPLAY SMS 841
VOLSELMSG = (ON,ALL) TYPE = ERROR JOBNAME = *
ASID = * STEPNAME = *
DSNAME = *
TRACE = ON SIZE = 128K TYPE = ERROR
JOBNAME = * ASID = *
```

Figure 7-12 Using the SETSMS command with VOLSELMSG

## 7.9 Multitasking volume recovery from dump

With increased tape capacities and the cost of DASD, many installations are looking to tape for backup, archival, and disaster recovery solutions. DFSMSHsm allows you to better utilize the high capacity tapes with functions such as backup, migration, and dump. The DUMP function allows multiple volumes to be “stacked” onto a single dump tape.

DFSMSHsm functions that write to these tapes are multitasked. This allows for a smaller window when backing up or migrating data to tape. Functions such as data set recovery, volume recovery from backup, and recall are also multitasked. Volume recovery from dump, however, is not and with the increasing popularity of the fast replication function, the absence of this support is becoming apparent.

### Fast replication copy pool

A fast replication copy pool may contain hundreds of volumes and there is no support to recover a whole copy pool from tape with a single command. A FRRECOV command must be issued for each volume. When an installation wants to recover multiple volumes from a dump tape and has multiple tape drives to utilize, multiple DFSMSdss jobs can be invoked so that the jobs are multitasked. Although for a small amount of volumes this may be feasible, recovering many volumes can be a tedious task.

Prior to z/OS V1R12, a recovery operation from tape using the RECOVER command could only recover one volume from a dump backup copy at a time. In addition, a fast replication copy pool can contain hundreds of volumes, and previously there was no way to recover an entire copy pool version from tape with a single command. An FRRECOV command had to be issued for each volume. A copy pool version can be recovered from dump with a single command.

### 7.9.1 z/OS V1R12 support enhancements

With z/OS V1R12, DFSMSHsm now allows a user to define how many volume recovery from dump tasks may run concurrently. The actual number of effective tasks can be up to 64 and may be limited if restore and system resources are not available. To provide for better control of DFSMSdss cross memory support, the following enhancements are:

- ▶ The existing SETSYS DSSXMMODE parameter is now expanded.
- ▶ An additional mutually exclusive alternative to the existing Y and N will be to specify:  
BACKUP(Y|N), CDSBACKUP(Y|N), DUMP(Y|N), MIGRATION(Y|N), and RECOVERY(Y|N)  
to specify whether DFSMSdss cross-memory will be invoked for:  
Backup, CDS backup, dump, migration, and recovery functions, respectively.
- ▶ If the DSSXMMODE parameter is not specified on the SETSYS command during startup, then the DFSMSHsm default for all functions is not to start DFSMSdss in its own address space. Any functions not specified will default to N.
- ▶ If the DSSXMMODE parameter is specified without Y or N during startup, specify all of the functions for which DFSMSdss should be started in its own address space. Any functions not specified will default to N.

### 7.9.2 Using multitasking recovery

Using the multitasking recovery you can do the following:

- ▶ Concurrently recover up to 64 volumes from dump

- ▶ Recover a copy pool version from dump with a single command
- ▶ Recover partially dumped versions
- ▶ Dump and recover processing now using up to 256 KB blocks

As a result of these enhancements, the improvements are as follows:

- ▶ Recovery windows when recovering volumes from a dump copy are substantially reduced.  
Volume recovery from dump can be of two flavors: recoveries of volumes that were dumped via the BACKVOL command and from fast replication versions that were dumped. For volumes that were dumped via the BACKVOL command, a RECOVER command must be issued for each volume recovery. These commands will be queued up and processed concurrently based on the number of tasks that are specified in the new MAXDUMPRECOVERYTASKS(1-64) SETSYS keyword. For fast replication copy pools, a single FRR command specifying which version is to be recovered can multitask volume recovery from dump.
- ▶ Installations do not have to query their copy pools to get a list of volumes for individual FRR commands when recovering a copy pool from dump or separately utilize DFSMSdss. Installations can recover from an incomplete dump version if the PARTIALOK keyword is specified on the FRR command. This should be used only under certain circumstances.  
When you request a copy pool version recover from a partial dump version, the FRRECOV command will be failed with message ARC1806E RC68. If you wish to override and recover the partial dump version, the PARTIALOK keyword must be specified on the FRRECOV command. When the PARTIALOK keyword is used, the recovers will be attempted and DFSMSHsm will append PARTIALOK SPECIFIED to the end of the ARC1802I message. If any of the recovers fail, the function return code displayed in the ARC1802I message will be 8 or greater. If the PARTIALOK keyword is used unnecessarily on a full dump version, the keyword will be ignored and PARTIALOK SPECIFIED will not be included in the ARC1802I message text.
- ▶ A larger block size increases throughput.

**Note:** A copy pool version is a point-in-time backup, and recovering a partial copy pool version could result in incomplete or out-of-sync data. For example, user catalogs for the data sets in a copy pool are kept on source volumes. If a user catalog is recovered as part of a partial dump volume recovery, it could reflect a state of that catalog which does not reflect the current data set environment.

### 7.9.3 Copy pool enhancements

Since recovering an entire copy pool version from dump can take a long time, a failure during the recovery process can be costly. To reduce the recovery window after a failure, multitasking recovery from dump processing will keep track of which volumes were successfully recovered, and on subsequent recovery requests, will not recover volumes that have already been successfully recovered. A new keyword, RESUME(YES|NO), will be added to the FRR command so that you can prevent recover processing from resuming a previously failed recovery. The default will be to resume a partially recovered version.

A recovery of a copy pool that has already been partially recovered will only recover volumes that have not been recovered yet (default). This eliminates redundant recoveries. You can specify RESUME(NO) on the FRR command to recover all volumes.

DFSMSHsm will avoid unnecessary demounts and remounts of dump tapes after a volume has been recovered by scanning the dump volume recovery queue for other volumes that are on the same tape before requesting that the volume be demounted.

When a volume recovery request is found on the queue that needs the same dump volume that is currently mounted, DFSMSHsm will process that request before demounting the tape. This optimization reduces tape mounts and demounts.

More granularity with the DFSMSdss cross memory feature gives an installation more control over how their resources are used by DFSMSHsm.

### **Recovering all volumes in a copy pool**

This enhancement provides the ability for multiple RECOVER commands to be processed concurrently and provides the ability to use a single FRRECOV command to recover all volumes within a copy pool from DASD or tape. You can use the following command to recover all of the volumes from a copy pool:

```
FRRECOV COPYPOOL(cpname)
```

The COPYPOOL(cpname) keyword indicates that all volumes associated with the named copy pool are to be recovered.

You can specify the VERIFY keyword to verify that none of the volumes are in an unexpected FlashCopy relationship before starting the recovery. When the copy pool is defined allowing fast reverse restore, VERIFY(Y) indicates DFSMSHsm is to verify that the state of the FlashCopy relationships meet the fast reverse restore requirements. Otherwise, regular FlashCopy can be used for the copy pool or the FRRECOV command will be failed.

VERIFY(N) indicates that the user has verified that the recovery can proceed successfully. VERIFY(N) cannot be specified when the copy pool is defined allowing fast reverse restore unless a previous FRRECOV COPYPOOL operation has determined that the copy pool backup version is no longer eligible to use fast reverse restore. Regular fast replication recovery can be used instead. If one or more volumes fail, the FRRECOV command ends with a nonzero return code. If the copy pool was recovered without using fast reverse restore, volumes that fail recovery can be tried again using the TOVOLUME keyword.

If the copy pool was recovered using fast reverse restore and the recovery was incomplete, volumes that were not recovered can be tried again by issuing the FRRECOV COPYPOOL command again. The RESUME(YES) keyword can also be used to retry recovery of volumes that previously failed to recover.

## **7.9.4 SETSYS command enhancements**

To enable concurrent RECOVER, the FRRECOV command processing from dump tapes, use the new MAXDUMPRECOVERTASKS parameters on the SETSYS command. However, the actual number of effective tasks will be limited by the number of tape drives available. The new dump and recover processing is using up to 256 KB blocks.

### **MAXDUMPRECOVERTASKS(nn) parameter**

MAXDUMPRECOVERTASKS(nn) is an optional parameter specifying the maximum number of volume recovery from dump tasks DFSMSHsm can concurrently process. This parameter allows you to set the appropriate tasking level to recover individual volumes, based on the number of tape drives available.

- For nn, specify a decimal number from 1 to 64 to represent the number of volume recovery from dump tasks to be processed concurrently.



- ▶ To enable up to 64 dump tasks, specify:

```
SETSYS MAXDUMPRECOVERTASKS(64)
```

**Note:** Special care must be taken when considering dump classes and their dump stacking values. A higher dump stacking value requires fewer tapes; however, the higher the number of volumes dumped onto a single tape, the greater the likelihood that a single tape will be needed for multiple recovery tasks. If the specified number of concurrent tasks is greater than the number of tape drives in use, the maximum number of concurrent tasks will be limited.

To maximize throughput for fast replication volume recovery, use the SETSYS DSSXMMODE command to start DFSMSdss in a separate address space to process these requests. For more information about this address space and controlling it, see Chapter 60, “SETSYS command: Establishing or changing the values of DFSMSShsm control parameters,” in *z/OS DFSMSShsm Implementation and Customization Guide*, SC35-0418.

## 7.9.5 Fast replication recovery

For fast replication recovery of an entire copy pool, you can specify the FRRECOV command with the COPYPOOL parameter and copy pool name. By default, a fast replication recovery of a pool is attempted from DASD. When only a tape backup copy is available, recovery is attempted from tape. When both a DASD and tape backup copy are available, you can now use the FROMDUMP parameter to recover all volumes within a copy pool from the tape backup copy.

Because a copy pool version is a point-in-time backup, recovery from a partial dump could result in an incomplete recovery. To allow recovery of a copy pool from a dump, you can use the new PARTIALOK parameter on the FRRECOV COPYPOOL FROMDUMP command.

If a copy pool recovery request fails, by default, the request will automatically be resumed when a subsequent FRRECOV COPYVOL FROMDUMP command request is issued for the copy pool. The new RESUME parameter on the FRRECOV COPYPOOL FROMDUMP command allows you to control this function. Specify RESUME(N0) to prevent resuming a failed recovery. A full recovery will be attempted instead.

Furthermore, the QUERY ACTIVE, QUERY SETSYS, and CANCEL commands have been enhanced to support multitask volume recovery from dump, with no changes to the command syntax or input values.

For more information about recovering data sets or volumes using multitask volume recovery from dump, see “Specifying the maximum number of volume recovery from a dump tasks” in *z/OS DFSMSShsm Storage Administration*, SC35-0421.

**Note:** Installations in a mixed release environment are restricted from issuing the FRRECOV command from a lower release on a copy pool that has a partial recovery. See APAR OA30350.

## 7.10 DFSMS SMS health check enhancement

Currently, SMS does not have any health check to help SMS users perform routine check-up for installation configurations, report potential problems, and recommend solutions related to SMS.

With this enhancement in z/OS V1R12, SMS has a health check to monitor and alert the status of the COMMDS and ACDS data sets as follows:

- ▶ SMS CDS separate volumes health check

With the SMS CDS separate volumes health check, a system programmer can be warned when COMMDS and ACDS reside on the same volume to ease recovery in case of failure.

- ▶ SMS CDS REUSE option check

With the SMS CDS REUSE option health check, a system programmer can be warned when an ACDS or COMMDS is activated without the REUSE option to avoid running into space problems as a result of subsequent ACDS or COMMDS updates, or with IMPORT/EXPORT functions.

### Health check importance

This enhancement allows an installation to do the following:

- ▶ Verify that the ACDS and COMMDS are not allocated on the same volume.

If the ACDS and COMMDS are on the same volume, the health check will warn the system programmer about the discovery and recommend a reallocation of the data sets on different volumes.

**Note:** This prevents a possible single point of failure and eases the complexity of recovery in case of a failure.

- ▶ Verify that both the ACDS and COMMDS have the REUSE option

If the ACDS or COMMDS do not have the REUSE attribute, the health check warns the system programmer about the discovery and recommends changing the data set.

**Note:** This prevents running into space problems (SMS reason code 6068) as a result of subsequent ACDS updates or IMPORT/EXPORT functions.

## 7.10.1 New health checks in z/OS V1R12

The two new SMS checks in z/OS V1R12 are as follows:

- ▶ CHECK(IBMSMS, SMS\_CDS\_REUSE\_OPTION)

This check verifies that the active control data set (ACDS) and communications data set (COMMDS) are defined with the REUSE option.

- ▶ CHECK(IBMSMS, SMS\_CDS\_SEPARATE\_VOLUMES)

This check verifies that the active control data set (ACDS) and communications data set (COMMDS) are not residing on the same volume.

### Example usage of CDS

Figure 7-13 on page 157 shows a listing of IDCAMS of the ACDS and COMMDS not being defined with the REUSE option and being defined on the same volume.

```

CLUSTER ----- SYS1.STPPLEX.ACDS
  IN-CAT --- MCAT.BH5CAT
ASSOCIATIONS
  CLUSTER--SYS1.STPPLEX.ACDS
ATTRIBUTES
  KEYLEN-----0          AVGLRECL-----0          BUFSPACE-----
SHROPTNS(3,3)  RECOVERY    UNIQUE          NOERASE        LINEAR         N
  UNORDERED      NOREUSE    NONSPANNED
VOLUME
  VOLSER-----STPSY2    PHYREC-SIZE-----4096    HI-A-RBA-----
CLUSTER ----- SYS1.STPPLEX.COMMDS
  IN-CAT --- MCAT.BH5CAT
ASSOCIATIONS
  CLUSTER--SYS1.STPPLEX.COMMDS
ATTRIBUTES
  KEYLEN-----0          AVGLRECL-----0          BUFSPACE-----
SHROPTNS(3,3)  RECOVERY    UNIQUE          NOERASE        LINEAR         N
  UNORDERED      NOREUSE    NONSPANNED
VOLUME
  VOLSER-----STPSY2    PHYREC-SIZE-----4096    HI-A-RBA-----

```

Figure 7-13 Listing of IDCAMS ACDS and COMMDS

**REUSE option**

When the ACDS and COMMDS not are defined with the option REUSE, health checker issues the message shown in Figure 7-14.

```

HZS0002E CHECK(IBMSMS,SMS_CDS_REUSE_OPTION): 873
IGDH1011E CHECK(IBMSMS,SMS_CDS_REUSE_OPTION) DETECTED
ACDS (SYS1.STPPLEX.ACDS) AND COMMDS (SYS1.STPPLEX.COMMDS)
NOT DEFINED WITH THE REUSE OPTION.

Explanation: As a best practice, defining ACDS/COMMDS with the REUSE option
helps to avoid running into space problems (SMS reason code 6068) as result of
subsequent ACDS updates, or IMPORT/EXPORT functions.

System Programmer Response: Use ALTER command in IDCAMS to specify
the REUSE option for the control dataset. See DFSMS Access Method
Services for Catalogs for further details.

```

Figure 7-14 Health checker message for CDSs defined without the REUSE option

**Using the same volume**

When the ACDS and COMMDS reside on the same volume, health checker issues the message shown in Figure 7-15 on page 158.

```
HZS0002E CHECK(IBMSMS,SMS_CDS_SEPARATE_VOLUMES): 872
IGDH1001E CHECK(IBMSMS,SMS_CDS_SEPARATE_VOLUMES) DETECTED THE
ACDS (SYS1.STPPLEX.ACDS)
AND COMMDS (SYS1.STPPLEX.COMMDS)
ALLOCATED ON THE SAME VOLUME.
```

**Explanation:** As a best practice, an ACDS/COMMDS must reside on a volume, accessible from all systems in the SMS complex. To ease recovery in case of failure, the ACDS should reside on a different volume than the COMMDS. Also, you should allocate a spare ACDS on a different volume. The control data set (ACDS or COMMDS) must reside on a volume that is not reserved by other systems for a long period of time because the control data set (ACDS or COMMDS) must be available to access for SMS processing to continue.

**System Programmer Response:** Reallocate ACDS and COMMDS on different volumes.

Figure 7-15 Health checker message with CDSs on the same volume

## 7.11 IDCAMS DELETE PDS and PDSE

Before z/OS V1R12, with the IDCAMS **DELETE** command, only one data set member could be deleted from a partitioned data set (PDS or PDSE) at a time. Wildcards were not allowed for the member names. With z/OS V1R12, DFSMSdfp access method services (IDCAMs) has added a new wildcard to the **DELETE** command, which lets you delete all members of a PDS or PDSE. The **DELETE** command is enhanced to allow you to specify a single asterisk (\*) as the member name of a PDS or PDSE, as shown in Figure 7-16. This will delete all the members in that PDS or PDSE.

This new feature allows you to empty a partitioned data set with a single IDCAMS **DELETE** command, thus deleting all members. This removes the restriction of deleting only one member at a time. This should save administrative work in managing data sets.

```
//STEP1 EXEC PGM=IDCAMs,REGION=4096K
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
DELETE CESAR.PDS.DELETE.TEST(*)
DELETE CESAR.PDSE.DELETE.TEST(*)
```

Figure 7-16 JCL showing **DELETE** commands for a PDS and PDSE

The following new message, IDC0553I, is added with this support and is shown in Figure 7-17 on page 159:

```
IDC0553I ALL MEMBERS IN DATA SET dsname DELETED
```

```

DELETE CESAR.PDS.DELETE.TEST(*)
IDC0553I ALL MEMBERS IN DATA SET CESAR.PDS.DELETE.TEST  DELETED
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

DELETE CESAR.PDSE.DELETE.TEST(*)
IDC0553I ALL MEMBERS IN DATA SET CESAR.PDSE.DELETE.TEST  DELETED
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

```

Figure 7-17 Messages that all the members in a PDS and PDSE have been successfully deleted

## 7.12 DFSMSrmm enhancements for z/OS V1R12

The functional enhancements in z/OS V1R12 DFSMSrmm provide you with these benefits:

- ▶ DFSMSrmm operational enhancements
- ▶ DFSMSrmm inventory management and reporting enhancements
- ▶ DFSMSrmm scalability, performance
- ▶ DFSMSrmm reliability, availability, and serviceability (RAS)
- ▶ DFSMSrmm hardware support

## 7.13 DFSMSrmm operational enhancements

With z/OS V1R12, the operator can obtain DFSMSrmm status information by using any of the following methods:

- ▶ The F DFRMM,QUERY ACTIVE operator command when issued from an operator console at any time to display the status of DFSMSrmm, its local tasks, and request queues. The operator can also use the abbreviations of QUERY and ACTIVE, for example:

```
F DFRMM,Q ACT or F DFRMM,Q A.
```

- ▶ A TSO subcommand, RMM LISTCONTROL STATUS, from TSO or with any of the DFSMSrmm APIs when DFSMSrmm is active and able to process subsystem requests.

With the LISTCONTROL STATUS subcommand, there was no way to retrieve information about the DFSMS subsystem requests and task status except the MODIFY operator command. The new DFSMSrmm TSO LISTCONTROL STATUS subcommand can be used to request information about DFSMSrmm subsystem address space status, tasks, and queued requests. The information returned is very similar to the results of the operator QUERY ACTIVE command. The output of command RMM LISTCONTROL STATUS is shown in Figure 7-18 on page 160.

- ▶ You can use the CONTROL STATUS ISPF dialog (fast path command, selection 5.7.12 from the DFSMSrmm Command Menu - z/OS V1R12). You use the DFSMSrmm Command Menu to provide access to all DFSMSrmm function menus. From these menus you can use any DFSMSrmm function, provided you are authorized. You can then interactively display status and manage active tasks when DFSMSrmm is active and able to process subsystem requests.

```

Menu List Mode Functions Utilities Help
-----
                          ISPF Command Shell
Enter TSO or Workstation commands below:

===> rmm listcontrol status_____
_____
_____

Place cursor on choice and press enter to Retrieve command

=> _____
=> _____
=> _____
=> _____
=> _____
=> _____
DFSMSrmm status = ACTIVE      Journal = ENABLED  Server listener =
Local tasks      = 10          Server tasks   = 0
  Active         = 1           Active         = 0
  Held           = 0           Held           = 0
Queued requests = 0           New requests  = NOTHELD
  Nowait         = 0
  Catalog        = 0
Last RESERVE    = 10:04:04 - DEQ
Debug Setting   = DISABLED
Trace Levels    =
Active requests:
Function System Task Name Started Token S IP Status
-----
LC          TSU=CESAR 10:21:14 00500003

```

Figure 7-18 TSO subcommand RMM LISTCONTROL STATUS

### 7.13.1 CONTROL STATUS dialog option

From the DFSMSrmm ISPF dialog, the new CONTROL STATUS dialog option provides the operator and systems programmer a means to display and control task status interactively. You can use the CONTROL STATUS ISPF dialog (fast path command, selection 5.7.12 from the DFSMSrmm Command Menu - z/OS V1R12). Figure 7-19 on page 161 is displayed.

You use the DFSMSrmm Command Menu to provide access to all DFSMSrmm function menus. From these menus you can use any DFSMSrmm function, provided you are authorized. You can then interactively display status and manage active tasks when DFSMSrmm is active and able to process subsystem requests.

```

Panel Help
-----
EDGPC000                DFSMSrmm Control Information Menu
Option ==>>

0  OPTIONS      - Specify dialog options and defaults
1  CONTROL      - Display cds control record details
2  SYSTEM       - Display system options and defaults
3  SECURITY      - Display security classification rules
4  VLPOOLS      - Display volume pool definitions
5  MNTMSG       - Display mount message definitions
6  REJECT       - Display volumes to be rejected
7  ACTIONS      - Display volume moves and actions
8  LOCDEF       - Display location definitions
9  MEDINF       - Display media information definitions
10 PARTITION    - Display partition definitions
11 OPENRULE     - Display open rule definitions
12 STATUS      - Display status

```

Figure 7-19 DFSMS control information panel with the new option

When selecting Option 12 in Figure 7-19, the new panel in Figure 7-20 is shown with the information about DFSMSrmm status. Option 12 is new with z/OS V1R12.

```

Panel Help Scroll
-----
EDGPC000                DFSMSrmm Status                Row 1 to 1 of 1
Command ==>>                Scroll ==>> PAGE

DFSMSrmm status . : ACTIVE   Journal : ENABLED   Server listener :
Local tasks . . . : 10       Active : 1          Held : 0          Task Commands
Server Tasks . . . : 0       Active : 0          Held : 0          Active . .
Queued requests . : 0       Nowait : 0         Catalog : 0       HSKP . . .
Debug . . . . . : DISABLED  PDA trace level :
Last reserve . . . : 12:56:34 Outstanding : N     New held : N

The following commands are valid: C,H and R or 'ENTER' to refresh

S Function System  Task Name  Started  Token  S IP Status
-----
LC                TSU  ROGERS  12:56:34 00700002
***** Bottom of data *****

```

Figure 7-20 New panel with output of STATUS command

### DFSMSrmm dialog CLIST processing

The DFSMSrmm CLIST processing dialog now enables you to specify that a search results list is not to be created when generating a CLIST from the search panels. This might enable you to process all the resources in a shorter time and bypass any system memory size limitations when you select the CLIST Option to be YES in Figure 7-21 on page 162.

```

Panel  Help  Scroll
-----
EDGPT010                      DFSMSrmm Volume Search
Command ===>

Volume . . . . ITSO*          May be generic. Leave blank for all volumes
Owner . . . . . HAIMO        Owned by a specific user. Default is your userid
Job name . . . .                               May be generic
Limit . . . . .                               Limit search to first nnnn volumes
Media name . .                               Limit to a single media name
Vendor . . . . .                               Supplier of media
Pool prefix . .                               or to a particular pool
Status . . . . .                               Select volume status. Default is ALL

Dates          Start          End          Date, date range or relative value
Assigned . . . . .                               . .
Create . . . . .                               . .
Reference . . . . .                              . .
Read . . . . .                               . .
Write . . . . .                               . .
Changed . . . . .                              . .
Moved . . . . .                               . .

Actions . . . . .                               Specify one or more pending actions

Release
  Actions . . . . .                               Specify one or more release actions
  Options . . . . .                               Specify one or more release options

System use . . . . .                               Select system use. Default is ALL
Since . . . . .                               Volumes assigned since YYYY/DDD
Retention . . . . .                             Volumes retained up to YYYY/DDD
Original EXPDT . . . . .                         YES, NO, or a specific date YYYY/DDD
Hold . . . . .                               YES or NO
Clist . . . . . YES          YES to create a data set, or NO, or blank

Home . . . . .                               Limit to volumes with this home location name
Location . . . . .                             Limit to volumes by location. May be generic
In container . . . . .                         Stacked volser

Volume type . . . . .                           ( LOGICAL , PHYSICAL or STACKED )
MedInf . . . . .                               Media information name ( IBM or MEDINF defined)
Media type . . . . .                           Tape media type ( for example HPCT )
Label . . . . .                               Tape label standard ( for example SL )
  Current version . . . . .                     Label version number( for example 3 )
  Required version . . . . .                    Label version number( for example 4 )
VOL1 volser . . . . .                          Volser in the VOL1 label ( NONE or volser )
Density . . . . .                               Tape recording density

```

Figure 7-21 Panel volume SEARCH request



When CLIST YES is provided, as shown in Figure 7-21 on page 162, the DFSMSrmm CLIST Processing panel is displayed, as shown in Figure 7-22.

```

Panel  Help
-----
                                DFSMSrmm CLIST Processing
Command ===>

Enter optional prefix and suffix values

Prefix  . . . . .
Returned text depending on resource being searched
Suffix  . . . . .

Enter optional fully qualified or partial data set information for CLIST

Data set name  . . . .
Expected data set size                records

Extend existing CLIST                 YES, NO or blank

View search results   NO           YES, NO or blank

Press ENTER to CONTINUE, or END to RETURN.

```

Figure 7-22 Panel CLIST processing with the option view search results

**Note:** If you choose **VIEW search results**, in Figure 7-22, as NO (which is the default), the search results list is not displayed. If you choose YES, the search results list is displayed.

### 7.13.2 Auto-reply policy and DFSMSrmm

The z/OS auto-reply (AUTOR) facility provides replies to write-to-operator with reply (WTOR) messages in cases where there is no automation or when the operator is unaware of the outstanding request or is taking too long trying to determine what the response should be. When AUTOR is active, z/OS uses the auto-reply policies provided for the subset of the DFSMSrmm WTORS that are included in the default AUTOR00 parmlib member provided by z/OS. AUTOR00 also contains comments that provide the message text for each WTOR and the rule used to select the WTOR.

You can define your own AUTORxx parmlib member to customize the auto-reply policies for DFSMSrmm. You can override or supplement the policies in AUTOR00 to add more WTORS to be handled automatically and to change the replies for these WTORS already included in AUTOR00. DFSMSrmm provides these AUTORxx parmlib members, which you can use as-is or use as the base for your customization, as follows:

- AUTORRM** Includes, for all DFSMSrmm WTORS, an automated response suitable for production running.
- AUTORRP** Includes, for all DFSMSrmm WTORS, an automated response suitable for use when running DFSMSrmm in a mode other than OPMODE(PROTECT).

Figure 7-23 shows three examples of AUTOR messages related to DFSMSrmm.

```

/*-----*/
/* EDG0103D DFSMSrmm SUBSYSTEM INTERFACE IS INACTIVE - ENTER      */
/*           "IGNORE", "CANCEL" OR "RETRY"                          */
/*                                                                 */
/* Notes:                                                           */
/* This message is normal for starting RMM after IPL when running  */
/* parallel if you have not included INITRTN(EDGSSSI) in the IEFSSN */
/* parmlib entry for the rmm subsystem                             */
/* Reply RETRY to establish subsystem                               */
/*                                                                 */
/* Rule: 5                                                           */
/*                                                                 */
MSGID(EDG0103D) DELAY(1m) REPLY(RETRY)
/*-----*/
/* EDG0107A ENTER SUFFIX OF INITIALIZATION MEMBER OR "CANCEL"     */
/*                                                                 */
/* Notes:                                                           */
/* enter suffix of RMM initialization member appropriate to       */
/* operating mode.      ****                                       */
/*                                                                 */
/* Rule: 5                                                           */
/*                                                                 */
MSGID(EDG0107A) DELAY(1m) REPLY(P0)
/*-----*/
/* EDG0117D DFSMSrmm DYNAMIC INSTALLATION EXITS INITIALIZATION  */
/* FAILED - REPLY "IGNORE", "CANCEL" OR "RETRY"                  */
/*                                                                 */
/* Rule: 5                                                           */
/*                                                                 */
MSGID(EDG0117D) DELAY(1m) REPLY(IGNORE)

```

Figure 7-23 Example of messages for AUTOR support with DFSMSrmm

**Note:** See *z/OS DFSMSrmm Implementation and Customization Guide*, SC26-7405 for a complete list of all the messages and a complete description of the auto-reply support for DFSMSrmm.

## 7.14 DFSMSrmm inventory management and reporting

DFSMSrmm provides utilities to manage your inventory, create reports, maintain the DFSMSrmm control data set, and erase and initialize volumes. DFSMSrmm uses IKJTSOEV, if necessary, to establish a TSO environment for each of its batch utilities.

With z/OS V1R12, DFSMSrmm has enhanced the following inventory functions:

- ▶ Retention Limit Reporting for EXPDTDROP and VRSRETAIN
- ▶ Enhanced HOLD support

You can now set the volume HOLD attribute to prevent automatic expiration and to prevent use of the **RMM DELETEVOLUME** subcommand with RELEASE.

- ▶ **Copy Export Reporting**  
DFSMSrmm now provides a new EDGJCEXP sample job to report on copies of logical volumes that have been exported from TS7700 Virtualization Engine. The report consolidates point-in-time information from the copy export status file, the library, and DFSMSrmm to enable you to more easily identify tape data that has been copy exported.
- ▶ **Volume HOLD Reporting**  
DFSMSrmm now provides a new sample report, EDGGAHLD, to list all volumes where the Hold attribute is set.

## 7.14.1 Retention Limit Reporting for EXPDTRDOP and VRSRETAIN

DFSMSrmm provides the VERIFY function to perform a trial run of vital record processing and synchronize catalog processing so you can see the results of processing on a production run. The ACTIVITY file is optional except during VERIFY processing, when it is required so that you can analyze processing results before they are actually performed. The ACTIVITY file is a pre-allocated DASD data set, like the REPORT file. The ACTIVITY file is a variable blocked file with the record length set to the largest record created by DFSMSrmm. The block size is determined by the system.

### **z/OS V1R12 enhancements**

The ACTIVITY file now contains information about the volume-related changes DFSMSrmm makes to the control data set during inventory management.

The OPTION command operands have been updated to reflect enhanced retention limit reporting for EXPDTRDOP and VRSRETAIN and VRSDROP limit checking.

With z/OS V1R12, to aid in the analysis of the results of the EXPDTRDOP limit checking (when the action is not OFF) you can use the contents of the ACTIVITY file and extended records from the extract file. The EDGJACTP sample generates a detailed report and a summary report of expiration date retained volumes showing why they are set to pending release. See the contents of the EXPDTRDOP and EXPDTRDOPS files.

### **Activity file**

In z/OS V1R12, the ACTIVITY file is not intended to be a report, but to contain detailed information about changes made to data sets and volumes during DFSMSrmm inventory management processing. All data set record changes, but only a subset of volume changes, are reported in the ACTIVITY file:

- ▶ Newly assigned volumes that are retained only by a volume VRS or that are retained only because they are in a volume set and another volume in the set is VRS retained. This information is needed to aid analysis of the VRSRETAIN retention limit processing and the records are produced only when the VRSRETAIN action is not OFF.
- ▶ EXPDTRDOP retained volumes set to pending release because of the expiration date. This information is needed to aid analysis of the EXPDTRDOP retention limit processing and the records are produced only when the EXPDTRDOP action is not OFF, as shown in Figure 7-24 on page 166.

The ACTIVITY file contains detailed information about data set and volume related changes to the control data set made by DFSMSrmm during inventory management. The EDGJACTP sample JCL provides reports from a combination of the ACTIVITY file and the report extract file data set and volume records to help you analyze the results of processing. For the best retention limit reporting you need the ACTIVITY file and the extended records from the report

extract file (XREPTTEXT or REPTTEXT DD) from the same EDGHSKP run. This could be one of the following:

- ▶ VRSEL, VERIFY, RPTEXT
- ▶ VRSEL, RPTEXT
- ▶ VRSEL, EXPROC, RPTEXT
- ▶ EXPROC, RPTEXT

Other supported EDGHSKP parameters may also have been requested.

You can use the ACTRC\_DSN\_CHANGE field of the data set ACTIVITY record to identify the reason for the change. You can use the ACTRC\_VOL\_CHANGE field of the volume ACTIVITY record to identify the reason for the change. During processing, if an ACTIVITY file is allocated, DFSMSrmm writes information about changes in the data set and volume information, such as expiration, matching vital record specification, vital record status, retention date, and catalog status to the ACTIVITY file. If VERIFY processing is being run, or the retention limit checking action is FAIL and the limit is triggered, the changes are not actually made.

**Note:** During expiration processing, DFSMSrmm identifies volumes not required for vital records and not held by the HOLD attribute that are ready to expire by checking the expiration date.

### EXPDTDROP reports

To aid in the analysis of the results of the EXPDTDROP limit checking (when the action is not OFF) you can use the contents of the ACTIVITY file and extended records from the extract file. The EDGJACTP sample generates a detailed report and a summary report of expiration date retained volumes showing why they are set to pending release. See the contents of the EXPDROP and EXPDROPS files.

**Note:** If you want to quickly and easily identify which volumes have triggered the EXPDTDROP or VRSRETAIN threshold, you can use the sample ICETOOL job EDGJACTP that will produce detail and summary reports, as follows:

- ▶ EXPDROP and EXPDROPS
- ▶ VRSRETN and VRSRETNS

Figure 7-24 shows the updated sample report created by the EDGJACTP JCL. There is a new column, HOLD, that has been added to the EXPDTDROP report.

1EXPDT retained volumes subject to EXPDTDROP														10/14/09	13:35:42	- 1 -
Status: RELEASED																
VOLSER	VSEQ	DSNAME	JOBNAME	EXPRSN	ASSIGNED	EXPDT	SR	RETDATE	ACTIONS	LOCATION	HOME	DEST	RLS ACT	HOLD		
A06920	1			X	2009/10/14	2009/10/15	N			SHELF	SHELF		S	N		
A06921	1	RMUSER.TEST.DS21		X	2009/10/14	2009/10/15	N			SHELF	SHELF		S	N		
A06930	1			X	2009/10/14	2009/10/15	N			SHELF	SHELF		S	N		
A06931	1			X	2009/10/14	2009/10/15	N			SHELF	SHELF		S	N		
Volumes in this status: 4																
1EXPDT retained volumes subject to EXPDTDROP														10/14/09	13:35:42	- 2 -
Status: NOCHANGE																
VOLSER	VSEQ	DSNAME	JOBNAME	EXPRSN	ASSIGNED	EXPDT	SR	RETDATE	ACTIONS	LOCATION	HOME	DEST	RLS ACT	HOLD		
A00001	1				2009/10/14	2010/12/26	N			SHELF	SHELF		S	N		
A06910	1				2009/10/14	2009/10/15	N			SHELF	SHELF		S	Y		
A06911	1				2009/10/14	2009/10/15	N			SHELF	SHELF		S	Y		
A06933	1				2009/10/14	2009/01/01	N			SHELF	SHELF		S	Y		
Volumes in this status: 4																

Figure 7-24 Sample report of volumes subject to EXPDDROP limit checking

Figure 7-25 on page 167 shows a summary of EXPDT volumes for EXPDTDROP.

Status	VOLUME COUNT
NOCHANGE	2
RELEASED	2

Figure 7-25 EXPDTPDOP summary report

### VRSRETAIN report

To aid in the analysis of the results of the VRSRETAIN limit checking (when the action is not OFF) you can use the contents of the ACTIVITY file and extended records from the extract file. The EDGJACTP sample generates a detailed report and a summary report of newly assigned volumes showing how they are processed. See the contents of the VRSRETN and VRSRETNS files.

Figure 7-26 shows an example of the VRSRETAIN report (part 1). All volumes in the picture are newly assigned. The following volumes and data sets are VRS retained:

- ▶ On VOL1, data sets DSN11 and DSN13
- ▶ On VOL2, all data sets
- ▶ On VOL7, data set DSN72

**Note:** The widths of the DSNAME and PRIMARY VRS fields are adjusted so the REPORT can fit the figure.

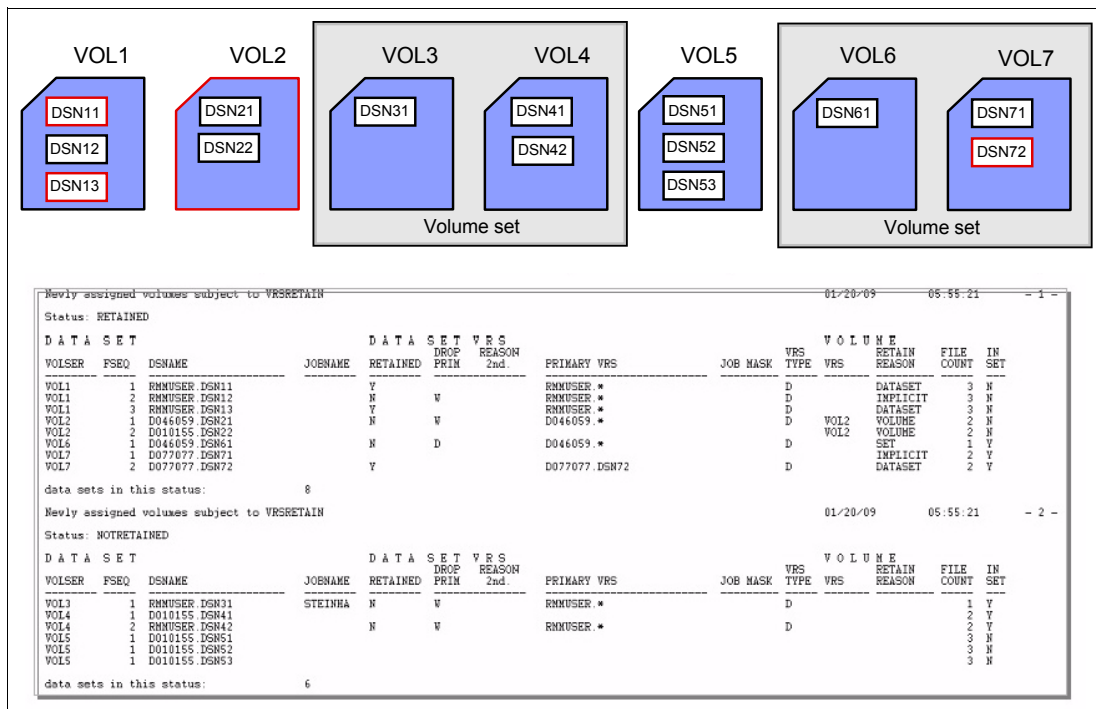


Figure 7-26 Report VRSRETAIN part 1

Figure 7-27 on page 168 shows the example of VRSRETAIN (part 2). The widths of the DSNAME and PRIMARY VRS fields are adjusted so the REPORT can fit the figure.

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						
VOLSER	FSEQ	DSNAME	JOBNAME	RETAINED	DROP PRIM	REASON 2nd.	PRIMARY VRS	JOB MASK	VRS TYPE	VRS	REASON	FILE COUNT	IN SET
VOL1	1	RHMUSER.DSN11		Y			RHMUSER.*		D		DATASET	3	N
VOL1	2	RHMUSER.DSN12		N	W		RHMUSER.*		D		IMPLICIT	3	N
VOL1	3	RHMUSER.DSN13		Y			RHMUSER.*		D		DATASET	3	N
VOL2	1	D046059.DSN21		N	W		D046059.*		D	VOL2	VOLUME	2	N
VOL2	2	D010155.DSN22							D	VOL2	VOLUME	2	N
VOL6	1	D046059.DSN61		N	D		D046059.*		D		SET	1	Y
VOL7	1	D077077.DSN71							D		IMPLICIT	2	Y
VOL7	2	D077077.DSN72		Y			D077077.DSN72		D		DATASET	2	Y
data sets in this status:				8									
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						
VOLSER	FSEQ	DSNAME	JOBNAME	RETAINED	DROP PRIM	REASON 2nd.	PRIMARY VRS	JOB MASK	VRS TYPE	VRS	REASON	FILE COUNT	IN SET
VOL3	1	RHMUSER.DSN31	STEINHA	N	W		RHMUSER.*		D			1	Y
VOL4	1	D010155.DSN41							D			2	Y
VOL4	2	RHMUSER.DSN42		N	W		RHMUSER.*		D			2	Y
VOL5	1	D010155.DSN51										3	N
VOL5	1	D010155.DSN52										3	N
VOL5	1	D010155.DSN53										3	N
data sets in this status:				6									

Summary of newly assigned volumes for VRSRETAIN										01/20/09	05:55:21	- 1 -	
Status	VOLUME COUNT												
RETAINED	4												
NOTRETAINED	3												

Figure 7-27 Report VRSRETAIN part 2

**Coexistence:** With APAR OA30881, this function is provided by rolling it down to z/OS V1R11 and z/OS V1r10.

## 7.15 DFSMSrmm scalability and performance

In z/OS V1R12, DFSMSrmm has new support for the following items:

- ▶ EAV support

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.

- ▶ Dynamic allocation support

In z/OS V1R12, BSAM, BPAM, QSAM, and OCE are enhanced to support the existing extended task input/output table (XTIOT), UCB nocapture, and DSAB-above-the-line options of dynamic allocation. Previously, VSAM and EXCP were the only access methods to support these dynamic allocation options. In V1R12 the EXCP support is enhanced and BSAM, BPAM, and QSAM support is provided.

- ▶ IPv6 support

DFSMSrmm supports the use of IP addresses that are compliant with either IPv4 or IPv6. To use IPv6, you must first configure z/OS Communications Server TCP/IP.

### 7.15.1 EAV support

All data sets used by DFSMSrmm can be eligible for allocation in the extended addressing space of an EAV. This includes the DFSMSrmm journal, which in previous releases was not

EAS eligible, and any dynamically allocated temporary files. For this support to work for the DFSMSrmm journal, it must not be shared with a z/OS release below z/OS V1R12. With z/OS V1R12 the DFSMSrmm journal is now EAS eligible. Any of the data sets used by or created by DFSMSrmm processing can be directed to EAS.

One exception for EAV support is the RMM CLIST data set when it is created automatically by **SEARCH** subcommand processing. You direct data sets to EAS by exploiting DC attributes, SMS ACS routines, and JCL keywords.

### **EATTR=OPT support**

For those data sets created dynamically by DFSMSrmm, the EATTR=OPT is specified so that you can use the system to decide where the data set is to be allocated. DFSMSrmm specifies EATTR=OPT on the dynamic allocation that creates the temporary data set.

### **DFSORT temporary sort files**

Temporary sort files created by DFSMSrmm for use by DFSORT are always large format EAS-eligible data sets when inventory management or EDGUTIL are used. This includes sort input/output files created by EDGHSKP and EDGUTIL processing (DFSORT V1R12 is required for this).

## **7.15.2 Dynamic allocation support**

This new support applies to dynamic allocation of DASD, tape, and dummy data sets, and cases where PATH= is coded. EXCP support, including the EXCPVR and XDAP macros, is also affected. These enhancements provide virtual storage constraint relief especially in the areas of DASD and tape support, and enable you to have more than about 3200 dynamically-allocated data sets. To exploit these enhancements, you need to set a new DEVSUPxx parmlib member option, and change the following programs:

1. Those that call dynamic allocation and want to exploit extended task input/output table (XTIOT), UCB nocapture, and DSAB-above-the-line options of dynamic allocation. They must check that DFAXTBAM is set ON before they exploit any of these dynamic allocation options. This bit signifies that the installation enables this function by setting the option in the DEVSUPxx parmlib member.
2. Those that open files that might have been dynamically allocated with XTIOT, UCB nocapture, or DSAB-above-the-line options of dynamic allocation. These programs need to either:
  - a. Have no dependency on the changes at all, and so need to specify a new DCBE macro LOC=ANY option.
  - b. Have other changes made in order to support XTIOT, uncaptured UCBs, and DSAB above the line. These changes *must* be made prior to using the new DCBE macro LOC=ANY option.

## **7.15.3 IPv6 support**

Coexistence of prior releases of DFSMSrmm using client and server systems is possible with z/OS V1R12 and later releases using client and server systems. DFSMSrmm prior to V1R12 release is an IPv4-enabled application. DFSMSrmm on V1R12 and later releases is an IPv6-enabled application that supports both IPv4 and IPv6 sockets. You can continue to use IPv4 on all systems or you can run a mixed environment with one or more V1R12 systems using IPv6 and other systems using IPv4. Once all systems are V1R12, you have the option of moving all systems to IPv6. In a mixed environment, dual-mode IP stacks are required.

Refer to *z/OS Communications Server: IPv6 Network and Application Design Guide*, SC31-8885 for more information.

**Note:** See *z/OS DFSMS Using the New Functions*, SC26-7473 for more information about EAS and EAV. All DFSMSrmm utilities and programs support NON\_VSAM\_XTIOT=YES options in the DEVSUPxx parmlib member.

### 7.15.4 DFSMSrmm reliability, availability, and serviceability (RAS)

DFSMSrmm now creates additional PDA trace records for processing outside the subsystem address space to enable improved diagnostics. Processing of PDA trace records created by DFSMSrmm on z/OS V1R12 or later releases using ARCPRPDO requires ARCPRPDO from z/OS V1R12 or later releases. Any attempt to process on a lower-level release results in an error message.

Figure 7-28 has an example of the new trace record for a job.

Where the possible fields are:

- ▶ D1 - Indicates a trace record for a job.
- ▶ E3 - Indicates a trace record for a TSU address space.
- ▶ E2 - Indicates a trace record for a STC.
- ▶ x'00' - indicates a trace record for the DFSMSrmm subsystem address space.
- ▶ 0240C4C6D9D4D4F1C1 - Indicates the 2-byte ASID and the 8-character jobname.

TIME	USECS	ID AS/TCB	MOD	LOGIC	CALLER	ARCPRPDO	LEVEL=	-OW1
96319	-----							
121050.389472		01 d18E15		OCEOV	ENTR	OCEXT		
	+R13ADDR=	+0 06447000					.....	
	ASID/JOB=	+0 0240C4C6 D9D4D4F1 C1					...DFRMM1.....	
	DATA=	+0 D3D6C3D2 60					LOCK-.....	

Figure 7-28 Sample formatted trace record

#### Improved shutdown processing

DFRMM shutdown now issues an additional message to list the job names of the address spaces preventing shutdown. DFSMSrmm subsystem interface processing now correctly detects that DFRMM is or has been stopped and fails incomplete and unprocessed requests for the reason DFSMSrmm is not active. The operator now can immediately see the users that caused the delay of a shutdown. Figure 7-29 on page 171 shows the new message issued by DFSMSrmm.



```

EDG0155I ADDRESS SPACE LIST BY JOBNAME:
      jobname jobname jobname jobname jobname
      NUMBER OF JOBNAME DELAYING SHUTDOWN =  jobname_count
Explanation: The DFSMSrmm subsystem is attempting to stop, but the shutdown
processing is delayed because there are tasks, on the current system, still
using DFSMSrmm resources. This message is preceded by EDG0154I.
In the message:
jobname Is the name of an address space delaying the DFSMSrmm shutdown. Up to 5
jobnames are listed in each of up to 6 lines of this multi-line message. A
maximum of 30 jobnames is listed. If more than 30 address spaces are delaying
DFSMSrmm shutdown the 30th jobname is displayed as ‘. . . . ‘
jobname_count
Is the count of all the address spaces delaying the DFSMSrmm shutdown not just
those listed.
System Action: The DFSMSrmm subsystem waits for the subsystem requests to be
completed.
Operator Response: If the reason for the delay is an outstanding WTOR, reply to
the outstanding WTOR for the address space holding the resource so the DFSMSrmm
function in that address space can complete. If you need to view the complete
list of address spaces at any time you can issue the “D
GRS,RES=(SYSZRMM,SHUTDOWN),DET,SCOPE=SYSTEM” command to determine which address
space is holding the resource SYSZRMM/SHUTDOWN.
Systems Programmer Response: None.
Source: DFSMSrmm
Detecting Module: EDGRCVR
Routing Codes: 1
Descriptor Codes: 11

```

Figure 7-29 Message EDG0155I

## 7.16 DFSMSrmm hardware support

DFSMSrmm now provides a new EDGJCEXP sample job to report on copies of logical volumes that have been exported from the TS7700 Virtualization Engine. The report consolidates point-in-time information from the copy export status file, the library, and DFSMSrmm to enable you to more easily identify tape data that has been copy exported and taken offsite for disaster recovery purposes. In prior releases there is no information regarding the exported data sets. The benefits of volume stacking, which places many logical volumes on a physical volume, are retained with this function. In addition, since the data being exported is a copy of the logical volume, the logical volume data remains accessible by the production host systems. Information about library and volume status can be obtained by using reports that the library provides.

### 7.16.1 Creating reports about data sets and volumes that are copy exported

EDGJCEXP provides a report on copies of logical volumes that have been exported from TS7700 Virtualization Engine. The report consolidates point-in-time information from the copy export status file, the library, and DFSMSrmm to help you identify tape data that has been copy exported.

## EDGJCEXP input and output

You can create the reports either from the export list file of up to three copy exports, or from the VOLUME MAP and PHYSICAL VOLUME STATUS POOL information created from the IBM Virtualization Engine TS7700 Series Bulk Volume Information Retrieval Function (BVIR). The information about stacked volumes and logical volume copies is retrieved from this input and merged with the information that the DFSMSrmm extract file X records has for the stacked and logical volumes. A current report extract containing extended records (type X) is required. You can use any date format and time zone when you create the extract file.

For information about creating the BVIR volume map or physical volume status pool map, see IBM Virtualization Engine TS7700 Series Bulk Volume Information Retrieval Function User's Guide Version 1.5 at:

<http://www-03.ibm.com/support/techdocs/atmsastr.nsf/WebIndex/WP101094>

## 7.16.2 Disaster recovery options

Copy export provides a new function that allows a copy of selected logical volumes written to the TS7700 to be removed and taken offsite for disaster recovery purposes. The benefits of volume stacking, which places many logical volumes on a physical volume, are retained with this function. In addition, since the data being exported is a copy of the logical volume, the logical volume data remains accessible by the production host systems. Information about library and volume status can be obtained by using reports that the library provides. This is shown in Figure 7-30.

### Export status file

During the execution of the copy export operation, the TS7700 appends records to the Export status file. On completion of the copy export operation, the Export Status file then provides a record with the details of the copy export operation.

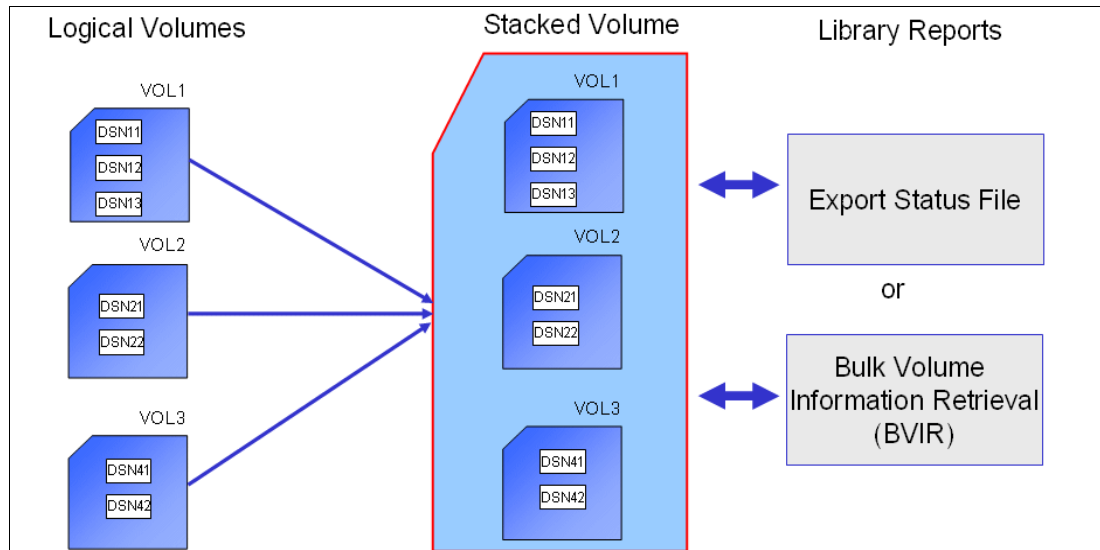


Figure 7-30 Merge information TS7700 and DFSMSrmm

## Bulk volume information retrieval (BVIR)

This facility uses an IBM standard labeled tape volume to both initiate a request for information and return the results. Several types of data can be requested, for example:

- ▶ VOLUME STATUS for a specific volume
- ▶ CACHE CONTENTS
- ▶ VOLUME MAP
- ▶ POINT IN TIME STATISTICS
- ▶ PHYSICAL VOLUME STATUS POOL xx

## Extract step JCL sample

Figure 7-31 shows sample JCL that contains an RMM Report Extract step at its beginning. Customize the EDGJCEXP sample JCL SET symbols to name the data sets to be used for input and output, and to select whether a copy export status file or BVIR output is used as input.

```
/**
// SET CEXP=1                ** Choose alternatively which
// SET BVIR=0                ** input shall be processed
/**
// SET EXTRACT=RMM.EXTRACT.FILE    ** customize pre-allocated
// SET MESSAGE=RMM.EXTRACT.MESSAGE ** files for RMM inventory m.
/**
// SET BVOLMAP=RMM.BVIR.VOLMAP      ** INPUT:
// SET BVOLSTA=RMM.BVIR.VOLSTAT     * customize with your input
// SET COEXP1=RMM.COPYEXP.FILE      * data set names
// SET COEXP2=NULLFILE             * (optional)
// SET COEXP3=NULLFILE             ** (optional)
/**
// SET REPDSN=RMM.REPORT.EXPDSN     ** OUTPUT:
// SET REPLVOL=RMM.REPORT.EXPLVOL   * customize with your output
// SET REPSVOL=RMM.REPORT.EXPSVOL   ** report data set names
/**
```

Figure 7-31 JCL to be tailored

**Note:** The input SET symbols are described in *z/OS DFSMSrmm Reporting*, SC26-7406.

## Copy export reports

Figure 7-32 on page 174 shows examples of the three types of copy export reports. The data columns in the copy export reports are presented in groups, which are presented in a sequence depending on the sort order, as follows:

- ▶ DATA SET INFO
- ▶ LOGICAL VOLUME INFO
- ▶ STACKED VOLUME INFO
- ▶ COPY EXPORT INFO

## EDGJCEXP reports

The sort order is different for each of the three reports created, as follows:

- ▶ For the data set report

The information is sorted by data set name, listing the most recent copies of a data set first.

- ▶ For the logical volume report

The information is sorted by ascending logical volume serial number, then by physical file sequence number, and the report starts a new page for each logical volume.

- ▶ For the stacked volume report

The information is sorted by stacked volume volser, by logical volume volser, and by physical file sequence number.

A new report page is used for each stacked volume.

```

1Copy Exported Data Sets          - 1 -      12/08/2000      03:30:21
based on Bulk Volume Information Retrieval data
DATA SET INFORMATION
DATA SET NAME          CREATE DATE      CREATE REC BLK RETENTION EXPIRATION PHYSICAL V LOGICAL VOLUME INFO
                        TIME   FM   SIZE  DATE      DATE      FILE SEQ R VOLSER VOLSEQ LOCATION DATE      STACKED VOLUME INFO
                        -----
BERNDS.EXPIRED.HYD060  2000/338  082750 F      80 2000/353 2000/341 1  Y HY0860 1  MRZ2 2000/341 A02039 ATL3484F MAZ1 Y 2020/001 Y 2000/338 083038
BERNDS.EXPIRED.HYD060  2000/337  150732 F      80 2000/352 2000/340 1  Y HY0860 1  MRZ2 2000/341 A02039 ATL3484F MAZ1 Y 2020/001 Y 2000/338 083038
BERNDS.MULTI.VOLUME.051  2000/338  082524 FB 80 2000/353 2000/341 1  Y HY0862 1  MRZ2 2000/341 A02039 ATL3484F MAZ1 Y 2020/001 Y 2000/338 083038
BERNDS.MULTI.VOLUME.051  2000/338  082524 FB 80 2000/353 2000/341 1  Y HY0861 1  MRZ2 2000/341 A02039 ATL3484F MAZ1 Y 2020/001 Y 2000/338 083038

1Copy Exported Data Sets By Logical Volume - 1 -      12/08/2000      03:30:22
based on Bulk Volume Information Retrieval data
Logical Volume Info: HYD061 1 MAZ2 2000/341
DATA SET INFORMATION
DATA SET NAME          CREATE DATE      CREATE REC BLK RETENTION EXPIRATION PHYSICAL V STACKED VOLUME INFO
                        TIME   FM   SIZE  DATE      DATE      FILE SEQ R VOLSER CURRENT DESTI IN RETENTION COPY EXPORT INFO
                        -----
BERNDS.MULTI.VOLUME.051  2000/338  082524 FB 80 2000/353 2000/341 1  Y A02039 ATL3484F MAZ1 Y 2020/001 Y 2000/338 083038
BERNDS.SEC14.HYD061     2000/338  082527 F      80 2000/353 2000/341 2  Y A02039 ATL3484F MAZ1 Y 2020/001 Y 2000/338 083038
BERNDS.SEC14.HYD061     2000/338  082630 F      80 2000/353 2000/341 3  Y A02039 ATL3484F MAZ1 Y 2020/001 Y 2000/338 083038
BERNDS.SEC14.HYD061     2000/338  082740 F      80 2000/353 2000/341 4  Y A02039 ATL3484F MAZ1 Y 2020/001 Y 2000/338 083038

1Copy Exported Data Sets By Stacked Volume - 1 -      12/08/2000      03:30:22
based on Bulk Volume Information Retrieval data
Stacked Volume Info: A02039 ATL3484F MAZ1 Y 2020/001 Y 2000/338 083038
LOGICAL VOLUME INFO
VOLSER VOLSEQ LOCATION DATE      DATA SET INFORMATION
                        -----
HYD061 1  MRZ2 2000/341  BERNDS.MULTI.VOLUME.051  2000/338  082524 FB 80 2000/353 2000/341 1  Y
HYD061 1  MRZ2 2000/341  BERNDS.SEC14.HYD061     2000/338  082527 F 80 2000/353 2000/341 2  Y
HYD061 1  MRZ2 2000/341  BERNDS.SEC14.HYD061     2000/338  082630 F 80 2000/353 2000/341 3  Y
HYD061 1  MRZ2 2000/341  BERNDS.SEC14.HYD061     2000/338  082740 F 80 2000/353 2000/341 4  Y
  
```

Figure 7-32 Sample report EDGJCEXP

## 7.17 DFSMSShsm space management performance

The target client environment for the DFSMSShsm space management performance improvements will be installations that currently are experiencing or are expected to experience performance issues due to vertical growth (increasing the number of data sets on individual disk volumes such as EAVs) and horizontal growth (increasing the number of data sets across more disk volumes). Some indications that an installation may be experiencing these performance problems can be primary space management and/or interval migration not finishing within their respective windows.

### Space management phases

Space management, which includes primary space management, interval migration, and command volume migration, will be performed in multitasking mode. A migration queue will be created and when in multitask mode, the implementation is as follows:

- ▶ One task performs VTOC scan, eligibility checking of extracted DSCBs, expiration, partial release, and reconnect of eligible data sets (phase 1). Additionally, this task places all data sets eligible in the extract list for extent reduction and/or migration on the Migration Queue (MQ).

- ▶ Another task (phase 2) processes MQ while the previously mentioned scanning VTOC task (phase 1) continues onto the next volume for SMS and next set of entries for non-SMS volumes.

In the top portion of Figure 7-33, the pre-V1R12 processing of phase 1 and phase 2 being a serial process is being presented. First phase 1 processing is performed and then phase 2 processing is performed. In the bottom portion of Figure 7-33, the V1R12 multitasking of phase 1 and phase 2 overlap is being shown. Notice that in the V1R12 graphic the phase 1 processing is a separate task from the actual data movement task (phase 2).

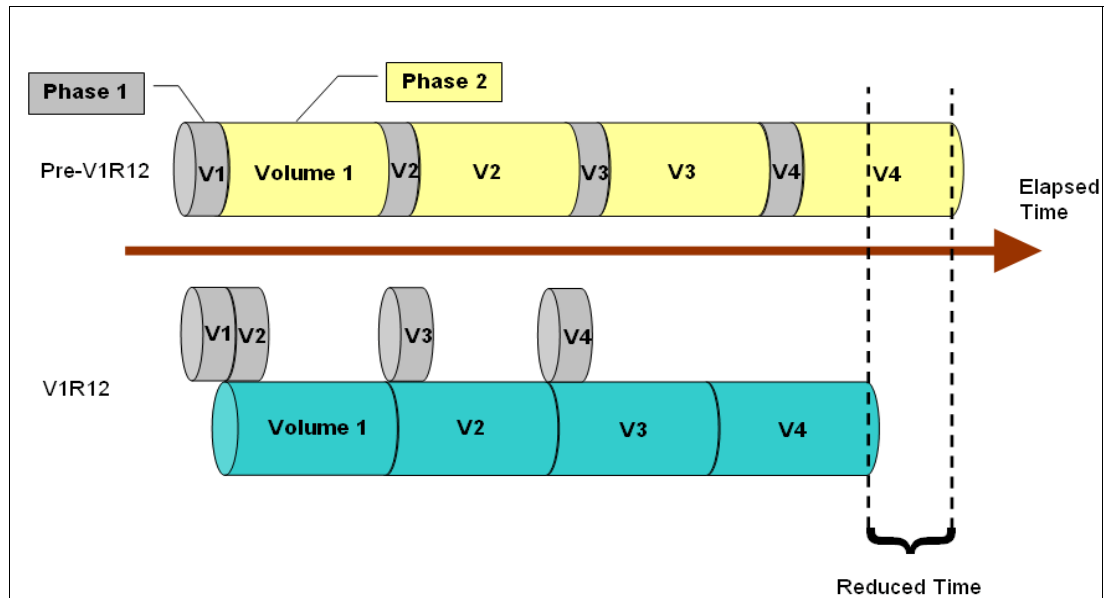


Figure 7-33 Space management phase 1 and 2

As the ratio of volumes-to-tasks increases, performance savings increase because DFSMSHsm is preprocessing for the next volume while performing data movement for the current volume, which reduces the overall processing time. Installations running DFSMSHsm's space management functions (primary space management, interval migration) should notice a reduction in the overall *time windows* used for these functions. This reduction in existing time windows allows for future data growth (vertically and horizontally).

## 7.18 DFSMS AMS DCOLLECT enhancements

**DCOLLECT** is an IDCAMS command to gather system information and write machine-readable records. As DFSMS grew from release to release, DCOLLECT did not keep up with all the line items supported in DFSMS. As a result, clients cannot query the setting of some DFSMS features. In z/OS V1R12, DFSMS access method services adds a number of attributes to the output produced by the DCOLLECT command. The new attributes include the following list. For details about the added attributes and how they appear in the DCOLLECT output record structure, see "Appendix F" in *z/OS DFSMS Access Method Services for Catalogs*, SC26-7394.

The following new attributes are added to the data class type records:

- ▶ VSAM SPEED and REUSE
- ▶ Tailored Compression

- ▶ CICSVR Forward Log
- ▶ RLS Greater Than 4 K Cache
- ▶ Block Size Limit
- ▶ Dynamic Volume Count
- ▶ RLS 64 Bits Virtual (RLS Above The Bar)
- ▶ Tape Performance Scaling
- ▶ Tape performance Segmentation
- ▶ System-managed Buffering
- ▶ Tape Encryption (Key-label, Key-code)
- ▶ CA Reclaim
- ▶ EATTR value

The following new attributes are added to the data set type records:

- ▶ Job-Name used to create the data set described by its format 8 DSCB
- ▶ Step-Name used to create the data set described by its format 8 DSCB
- ▶ Number of microseconds since midnight local time that the data set was created
- ▶ Larger fields for space amount

The following new attributes are added to the storage group type record:

- ▶ OAM Retention Protection
- ▶ OAM Deletion Protection



## DFSORT improvements

DFSORT is distributed on a tape that you can install using IBM SMP/E for z/OS. This tape contains functional modification identifiers (FMIDs) that identify DFSORT to SMP/E.

DFSORT provides a set of sample jobs that demonstrate techniques of interest to storage administrators and others who analyze data collected from products such as DFSMSHsm), DFSMSrmm, DCOLLECT, and SMF. These sample jobs can be found in the ICESTGEX member of the SICESAMP library after DFSORT is installed. You can also download these sample jobs from the DFSORT FTP site listed to show some of the many ways that DFSORT features such as ICETOOL and OUTFIL can be used to analyze data and generate reports.

In this chapter we introduce the DFSORT improvements in z/OS V1R12, especially with regard to availability, RAS, scalability, and performance:

- ▶ There is a new method for work space calculation when input data resides on VSAM data sets that were not properly closed. This improvement reduces user abends due to lack of work space for sorting.
- ▶ Now DFSORT issues diagnostic messages in more situations, avoiding frequent reruns for problem documentation.
- ▶ There are additional diagnostics for message ICE083A.
- ▶ Dynamic Allocation is improved.
- ▶ DFSORT now supports EVA cylinder-managed space eligible data sets.
- ▶ Memory object work space is now supported.

## 8.1 File size for improperly closed VSAM data sets

DFSORT needs a good file size estimate to properly allocate work space and successfully sort input records. DFSORT can automatically estimate the file size for disk input data sets and tape data sets managed by DFSMSrmm or a tape management system that uses the ICETPEX routine; see *z/OS DFSORT Installation and Customization*, SC26-7524.

When a VSAM data set is opened for output, by a program or utility, but not closed properly, the statistics associated with the data set are not updated. This may result in inaccurate file size information for the VSAM data set. When possible, DFSORT will attempt to calculate a more accurate file size for such an improperly closed VSAM fixed-length record data set. Former versions of DFSORT would fail with error messages ICE046A or ICE083A when the input VSAM data sets were not properly closed. DFSORT now uses an alternate file size determination method, using High Used Relative Byte Address (HURBA) to correctly calculate the needed work space.

In the case of detection of improperly closed input VSAM data sets, DFSORT now issues the new informative message ICE264I, instead of ICE255, as shown in Figure 8-1.

```
ICE264I ddname STATISTICS MAY be INCORRECT
Explanation: DFSORT could not be certain that the data set information provided
by Catalog was correct for this copy application. The catalog entry for this
data set indicated that the component was not closed properly and the
statistics for the data set are invalid.
System action: None.
Programmer response: None required. For more information regarding the cause of
the invalid statistics, or the steps necessary to correct the statistics, see
the Statistics Group" section of z/OS DFSMS Access Method Services for Catalogs,
SC26-7394.
```

Figure 8-1 New message ICE264I

**Note:** Message ICE255I was changed to be issued *only* for a SORT application. Any automated actions based on the ICE255I message should be evaluated; you may now want to base these actions on the ICE264I message for Copy applications.

## 8.2 Improved first failure data capture

In prior releases of DFSORT, diagnostic messages were not issued unless a SORTDIAG JCL statement was present, or installation option DIAGSIM was turned to YES. The default value for this installation parameter is NO. In z/OS V1R12, when an error message (ICExxxA) is issued and the message data set is available, DFSORT will always display ICE75xI, ICE8xxI, and ICE9xxI diagnostic messages, unless MSGPRT=NONE is in effect. This can improve first failure data capture and remove the need to rerun the application to display diagnostic messages.

You no longer need to code the SORTDIAG DD or DIAGSIM=YES to cause DFSORT to issue diagnostic messages when a critical error occurs (ICExxxA).



**Note:** Any automated actions based on the presence of diagnostic messages should be evaluated; these messages may now be present if an error message is issued.

## 8.3 Improved diagnostics

In prior DFSORT releases, not enough information was provided to diagnose why error message ICE083A (Resources Unavailable for Dynamic Allocation) was issued. In z/OS V1R12, DFSORT provides additional information in new messages ICE248I and ICE249I, when message ICE083A, ICE254I, or ICE258I is issued; see Figure 8-2. These new messages assist in determining the reason why DFSORT was unable to dynamically allocate all of the requested disk work space and what corrective actions may be taken:

**ICE248I DFSORT ATTEMPTED TO ALLOCATE xMB OF DISK WORK SPACE ON y WORK DATA SETS**

**Explanation:** Provides information on the total disk work space DFSORT attempted to allocate dynamically when it was unable to allocate all of the work space it requested. x indicates the total megabytes of disk work space DFSORT attempted to allocate. y indicates the total number of work data sets used for the allocations. x/y is the amount of free space that must be available on at least y volumes for DFSORT to allocate all of the work space it requested.

**System action:** None.

**Programmer response:** Verify that there are enough candidate volumes with the required free space to satisfy the work data set space requests. If necessary, increase the number of work data sets used for dynamic allocation to reduce the required disk space for each.

**ICE249I DFSORT SUCCESSFULLY ALLOCATED xMB OF DISK WORK SPACE ON y WORK DATA SETS**

**Explanation:** Provides information on the total disk work space DFSORT successfully allocated dynamically when it was unable to allocate all of the work space it requested. x indicates the total megabytes of disk work space DFSORT successfully allocated. y indicates the total number of work data sets used for the allocations.

**System action:** None.

**Programmer response:** Verify that there are enough candidate volumes with the required free space to satisfy the work data set space requests. If necessary, increase the number of work data sets used for dynamic allocation to reduce the required disk space for each.

Figure 8-2 New messages ICE248I and ICE249I

## 8.4 Dynamic allocation improvements

DFSORT's dynamic allocation of work data sets provided limited capability to react to unexpected increases in disk work space requirements, file sizes larger than expected, or central storage resources becoming constrained. In z/OS V1R12, DFSORT's dynamic allocation of work data sets has been enhanced to improve reliability in situations where the disk work space requirements are larger than expected. A new DYNAPCT installation and run-time option allows you to specify additional work data sets to be allocated with zero space. DFSORT only extends these data sets when necessary to complete a sort application. New message ICE278I is issued if these additional work data sets are used.

DFSORT's new shipped installation default of DYNAPCT=10 increases the number of dynamically allocated work data sets by 10%. These additional work data sets are allocated with zero space and only extended if required to complete a sort application. The availability of these additional work data sets can improve reliability in situations where the disk work space requirements are larger than expected. The DYNAPCT=n percentage can be changed at installation or run-time, if appropriate. If you want DFSORT to behave as it did in prior releases, you can set DYNAPCT=OLD.

## 8.5 XTIOU uncaptured UCBs and DSAB above 16 megabytes

In z/OS V1R12, a program that invokes DFSORT, ICETOOL, or ICEGENER can dynamically allocate input, output and work data sets using the options for XTIOU, uncaptured UCBs, and DSAB above 16-megabyte virtual. These data sets are supported to the extent that z/OS supports them. However, DFSORT cannot use the more efficient sort technique, Blockset, when this option is in use. Error message ICE189A was updated to inform that DFSORT's primary technique, Blockset, could not be used due to the reason indicated by reason code *rsn*, the meaning of which is documented in message ICE800I. Blockset was required to handle one of the situations that cannot be handled by DFSORT's secondary techniques, including this one.

A program dynamically allocated DFSORT input, output or work data sets using the options for XTIOU, uncaptured UCBs, or DSAB above 16-megabyte virtual.

**Note:** A program that invokes DFSORT, ICETOOL or ICEGENER should not allocate other data sets such as message, control, list, count, or symbol data sets using the options for XTIOU, uncaptured UCBs, and DSAB above 16-megabyte virtual. These data set options are not supported and the data set will not be recognized.

## 8.6 Extended address volumes

DFSORT supports EAS-eligible data set types on Extended Address Volumes to the extent that z/OS supports these data sets. With full track blocking, the maximum number of 1,048,576 tracks can be used for a single work data set allocated in the cylinder-managed space on an Extended Address Volume. The cylinder-managed space is space on the volume that is managed only in multicylinder units. Cylinder-managed space begins at cylinder address 65,520. Each data set occupies an integral multiple of multicylinder units. Space requests targeted for the cylinder-managed space are rounded up to the next multicylinder unit. The cylinder-managed space only exists on EAV volumes. Without full track blocking, less than 1,048,576 tracks may be used.

For more information about EAV, see *z/OS DFSMS Using Data Sets*, SC26-7410.

## 8.7 Memory object intermediate work space

DFSORT can now use memory object storage as intermediate work space or as an extension of main storage. A memory object is a data area in virtual storage that is allocated above the bar and backed by central storage. Using memory object storage as intermediate work space is the recommended and preferred choice, but can be disabled, if appropriate. A new MOWRK installation and run-time option allows you to specify whether memory object

storage can be used for intermediate work space or as an extension of main storage, as appropriate, or only as an extension of main storage. The MOSIZE installation default will still limit the total amount of memory object storage that can be used by a sort application.

MOWRK=YES, which is the default option, specifies that the memory object storage available to DFSORT for memory object sorting can be used as intermediate work space. NO specifies that memory object storage can only be used as an extension of main storage, as in the previous release of DFSORT.

**Note:** DFSORT's new shipped installation default of MOWRK=YES allows memory object storage to be used as intermediate work space or as an extension of main storage, as appropriate. MOWRK=YES can improve performance. If you want DFSORT to only use memory object storage as an extension of main storage, as in previous releases, you can set MOWRK=NO. The MOSIZE installation default will still limit the total amount of memory object storage that can be used by a sort application.

For a detailed description of the MOWRK parameter, see *z/OS DFSORT Installation and Customization*, SC26-7524.





## Service aids enhancements

MVS service aids are a group of tools to aid in detecting problems, gathering documentation needed to solve the causes, and communicate with support locations remote from where materials may have originally been collected.

This chapter describes the following service aids enhancements:

- ▶ SADMP support for EAV2
- ▶ SuperZap support for EAV3
- ▶ IPCS PDS enhancement
- ▶ IPCS SYSTRACE formatting enhancements
- ▶ SADMP ASID prioritization
- ▶ Console dump support
- ▶ Service Aids support for BAM XTLOT

## 9.1 SADMP support for EAV volumes

The first stage of EAV support was shipped in z/OS V1R10. EAV1 support allowed most VSAM objects to reside in cylinder-managed space on extended access volumes.

The second stage of EAV support in z/OS V1R11 adds the ability to place extended format sequential data sets in cylinder-managed space. EAV1 and EAV2 support were largely implemented by the DFSMS of z/OS.

SADMP provides a new function to fully support placement of dump data sets in cylinder-managed space on extended access volumes. This was introduced in z/OS V1R11 but the supporting IPCS functions were not enhanced. In z/OS V1R12 the REXX exec AMDSADDD and IPCS utility Option 6, SADMP DASD Dump Data Set Utility, are changed to provide support for SADMP data sets in cylinder-managed space.

SADMP data sets defined as extended format sequential data sets can now be allocated in cylinder-managed space and are fully supported by IPCS.

### 9.1.1 IPCS SADMP dump data set utility

The SADMP dump data set utility is accessed via IPCS Option 3.6, as shown in Figure 9-1 (Panelid BLSPPRIM), Figure 9-2 (Panelid BLSPUTIL), and Figure 9-3 on page 185, has been enhanced to support SADMP data sets in cylinder-managed space.

```
BLSPPRIM----- z/OS 01.12.00 IPCS PRIMARY OPTION MENU -----
OPTION ==>
                                *****
0  DEFAULTS   - Specify default dump and options      * USERID - ROGERS
1  BROWSE     - Browse dump data set                  * DATE   - 10/08/17
2  ANALYSIS   - Analyze dump contents                 * JULIAN - 10.229
3  UTILITY    - Perform utility functions           * TIME   - 13:14
4  INVENTORY  - Inventory of problem data             * PREFIX - ROGERS
5  SUBMIT     - Submit problem analysis job to batch  * TERMINAL- 3278
6  COMMAND    - Enter subcommand, CLIST or REXX exec * PF KEYS - 24
T  TUTORIAL   - Learn how to use the IPCS dialog      *****
X  EXIT       - Terminate using log and list defaults

Enter END command to terminate IPCS dialog
```

Figure 9-1 IPCS PRIMARY OPTION MENU panel

```
BLSPUTIL----- IPCS UTILITY MENU -----
OPTION ==>
                                *****
1  COPYDDIR - Copy dump directory data                * USERID - IPCSU1
2  COPYDUMP - Copy a dump data set                   * DATE   - 95/10/27
3  COPYTRC  - Copy trace data sets                   * JULIAN - 95.300
4  DSLIST   - Process list of data set names          * TIME   - 18:17
5  DAE      - Process DAE data                       * PREFIX - IPCSU1
6  SADMP    - SADMP dump data set utility           * TERMINAL- 3278T
                                * PF KEYS - 24
                                *****

Enter END command to terminate
```

Figure 9-2 IPCS UTILITY MENU panel

```

AMDSAPUT ----- SADMP DASD Dump Data Set Utility -----
Command ==>

Enter/verify parameters. Use ENTER to perform function, END to terminate

Function ==> R ( C - Clear, D - Define, R - Reallocate)
DSNAME ==>
Volume serial numbers: (1-32)
    1- 8 VOL001
    9-16
    17-24
    25-32
Unit ==> 3390 (3380, 3390, or 9345)
Cylinders ==> 500 (cylinders per volume)
DSNTYPE ==> B ( B - Basic, L - Large, E - ExtReq)
CATALOG ==> Y (Y or N)
EATTR ==> N ( N - No, O - Optional)

Optional SMS classes: (May be required by installation ACS routines)
StorClas ==> DataClas ==> MgmtClas ==>

```

Figure 9-3 SADMP dump data set utility

## DSNTYPE=EXTREQ

The panel now supports DSNTYPE=EXTREQ DASD dump data sets (extended-format data set (E)) with no operational changes required. The placement of the dump data set on an EAV relies upon the DSNTYPE and EATTR options and the BreakPointValue (BPV). EATTR indicates the extended attributes of the dump data set. EATTR=OPT indicates that extended attributes are optional for the dump data set. EATTR=NO indicates that extended attributes are not requested for the dump data set.

These are the expected results for the different combinations of EATTR and the size of the data set when DSNTYPE=EXTREQ is requested:

- ▶ DSNTYPE=EXTREQ,EATTR=OPT - Cylinders requested is more than BPV  
Result: Data set in cylinder-managed space (format 8 DSCB)
- ▶ DSNTYPE=EXTREQ,EATTR=OPT - Cylinder requested is less than BPV  
Result: Data set in track-managed space (Format 8 DSCB)
- ▶ DSNTYPE=EXTREQ,EATTR=NO  
Result: Data set in track-managed space (Format 1 DSCB)

**Note:** Use the EATTR parameter to indicate whether the data set can support extended attributes (format 8 and 9 DSCBs) or not. To create such data sets, you can include extended address volumes (EAVs) in specific storage groups or specify an EAV on the request or direct the allocation to an esoteric containing EAV devices.

**EATTR = OPT** - Extended attributes are optional. The data set can have extended attributes and reside in EAS. This is the default value for VSAM data sets.

**EATTR = NO** - No extended attributes. The data set cannot have extended attributes (format 8 and 9 DSCBs) or reside in EAS. This is the default value for non-VSAM data sets.

## 9.1.2 Using the AMDSADDD utility

The REXX utility AMDSADDD can be used to allocate and initialize the data set. Additional positional keywords were added to the AMDSADDD syntax to allow allocation of SADMP data sets in cylinder-managed space.

**Note:** The dump data set must be both allocated and initialized using the AMDSADDD REXX or IPCS SADMP dump data set utilities.

The DSNTYPE= option was extended to:

```
[EXTREQ|LARGE|BASIC]
```

**Note:** When DSNTYPE=EXTREQ is specified, EATTR=[OPTINO] should be specified. The default is EATTR=NO.

### New syntax for AMDSADDD

The new syntax is as follows:

```
AMDSADDD {DEFINE|CLEAR|REALLOC}  
volser{(data set name)}  
(type[, [STORCLAS] [, [DATACLAS] [, [MGMTCLAS]]]) [space]  
[YES|NO] [EXTREQ|LARGE|BASIC] {EATTR(OPT|NO)}
```

Or

```
AMDSADDD {DEFINE|CLEAR|REALLOC}  
(volumelist){(data set name)}  
(type[, [STORCLAS] [, [DATACLAS] [, [MGMTCLAS]]]) [space]  
[YES|NO] [EXTREQ|LARGE|BASIC] {EATTR(OPT|NO)}
```

**Note:** When specifying the REALLOC option for an existing multivolume data set, the same list of volumes must be specified as when the data set was originally allocated.

The example in Figure 9-4 on page 187 is taken from *z/OS MVS Diagnosis: Tools and Service Aids*, GA22-7589. It shows how the AMDSADDD utility can be used to allocate and initialize an extended format dump data set.



```

----- TSO COMMAND PROCESSOR -----
ENTER TSO COMMAND, CLIST, OR REXX EXEC BELOW:
====> exec 'sys1.sblsccli0(amsdadd)'
What function do you want?
Please enter DEFINE if you want to allocate a new dump data set
Please enter CLEAR if you want to clear an existing dump data set
Please enter REALLOC if you want to reallocate, clear an existing dump data set
Please enter QUIT if you want to leave this procedure
define
Please enter VOLSER or VOLSER(dump_data set_name) or (VOLLIST)
or (VOLLIST)(dump_data set_name)
(SADPK1,SADPK2) (SADMP.SAMPLE)
Please enter the device type for the dump data set
Device type choices are 3380 or 3390 or 9345
(An SMS STORAGE CLASS, DATA CLASS, AND MANAGEMENT CLASS
MAY ALSO BE SPECIFIED WITH THE DEVICE TYPE)
(3390,STORCLAS,DATACLAS,MGMTCLAS)
Please enter the number of cylinders (per volume)
400
Do you want the dump data set to be cataloged?
Please respond Y or N
y
Specify the DSNTYPE. Reply BASIC or LARGE or EXTREQ
EXTREQ
Specify the extended attributes for the dump data set. Reply OPT or NO
OPT
TIME-11:54:59 PM. CPU-00:00:00 SERVICE-58954 SESSION-00:07:25 AUGUST 1,2009
Note: Allocated space does not match requested amount
Amount allocated: 420
Amount requested: 400
Initializing output dump data set with a null record:
Dump data set has been successfully initialized
Results of the DEFINE request:
Dump data set Name : SADMP.SAMPLE
Volume : SADPK1 SADPK2
Device Type : 3390
Allocated Amount : 420 (per volume)

```

Figure 9-4 Using AMDSADDD to allocate and initialize an extended SADMP data set

### 9.1.3 Examples of running AMDSADDD in batch mode

The following examples show how to use JCL to allocate and initialize dump data sets. Because users cannot be prompted to enter values when invoking the AMDSADDD REXX utility in batch mode, you must specify all parameters in the order listed.

#### Example 1

This example shows how to use JCL to allocate and initialize the dump data set SADMP.DDS2 on VOL=SER=USRD53 with a size of 2653 cylinders. The BASIC type of data set is allocated because the dsntype parameter is not specified.

```

//SAMPLE JOB 'S3031,B7100003,S=C','BATCH EXAMPLE',RD=R,
// MSGLEVEL=(1,1),CLASS=E,NOTIFY=&SYSUID,MSGCLASS=H
//STEP1 EXEC PGM=IKJEFT01,REGION=64M

```

```
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
EXEC 'SYS1.SBLSCLI0.EXEC' 'DEFINE USRD53(SADMP.DDS2) 3390 2653 N'
/*
```

## Example 2

This example shows how to use JCL to allocate and initialize a dump data set named SADMP.DS on VOL=SER=USRDS1 with 2953 cylinders in the cylinder-managed space:

```
//SAMPLE JOB 'S3031,B707000,S=C', 'BATCH EXAMPLE', RD=R,
// MSGLEVEL=(1,1),CLASS=E,NOTIFY=&SYSUID,MSGCLASS=H
//STEP1 EXEC PGM=IKJEFT01,REGION=64M
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
EXEC 'SYS1.SBLSCLI0.EXEC'
'DEFINE USRDS1(SADMP.DS)(3390,storclas) 2953 Y EXTREQ OPT'
/*
```

## 9.2 Superzap support for EAV3

The AMASPZAP service aid, also called SPZAP or Superzap, dynamically updates or dumps programs and data sets. You can use AMASPZAP to inspect and modify instructions or data in any load module or program object in a program library, to dump a load module or program object in a program library, or to update the system status index in the directory entry for any load module or program object. Load modules can be updated in place; when a program object is updated using AMASPZAP, a new copy of the program object is created.

The first stage of EAV support (EAV1) was shipped in z/OS V1R10. EAV1 support allowed most VSAM objects to reside in cylinder-managed space on extended access volumes. The second stage of EAV support (EAV2) in z/OS V1R11 added the ability to place extended format sequential data sets in cylinder-managed space. Superzap has fully supported placement of SYSIN and SYSPRINT data sets in cylinder-managed space.

SYSLIB references to extended format sequential data sets other than program libraries were not supported in prior releases. The third stage of EAV support (EAV3) in z/OS V1R12 adds the ability to place additional non-VSAM data set types such as sequential (basic, large), partitioned (PDS & PDSE), and BDAM in cylinder-managed space. EAV3 support, like EAV1 and EAV2, is largely implemented by the DFSMS of z/OS.

### 9.2.1 Superzap control statements

In z/OS V1R12 Superzap control statements (commands) were updated to support basic and large sequential, PDS, and BDAM //SYSLIB DD data sets residing in cylinder-managed space. This provides you with the ability to place non-VSAM data set types such as sequential (basic, large), partitioned (PDS), and BDAM in cylinder-managed space.

The following Superzap control statements (commands) were changed to provide support for basic and large sequential, PDS, and BDAM //SYSLIB DD data sets residing in cylinder-managed space:

- ▶ ABSDUMPT CCCCcccHRR CCCCcccHRR
- ▶ CCHHR record address
- ▶ DUMPT member name

- ▶ NAME member name
- ▶ SETSSI ssi-info
- ▶ VER/REP address data

### **SYSLIB DD statement**

When the SYSLIB DD statement describes a data set placed in an extended address volume (EAV), the startaddr and stopaddr values must be specified in hexadecimal in the form CCCCcccHRR, where CCCCccc is referred to as a 28-bit cylinder address. The meanings of the codes are as follows:

- ▶ CCCC is the 16 low-order bits of the cylinder number.
- ▶ ccc is the 12 high-order bits of the cylinder number.
- ▶ H is the track number.
- ▶ RR is the record number.

## **9.3 IPCS PDS enhancement**

New function has been added to IPCS in z/OS V1R12 to support formatting of high-level reports into a repository in a scenario where responsibility for dump analysis may need to be passed from one analyst to another. Important information can be extracted from a dump and stored into a partitioned data set (PDS or PDSE). The source can be either SADMP or an SVC dump.

This allows the responsibility for dump analysis to be passed from one analyst to another, for example, from a customer systems programmer to IBM support personnel connecting to the customer system. The PDS is created once but can be shared among different teams and will help reduce the time spent on each customer problem.

### **Directing IPCS output**

A PDS must be allocated to DDNAME IPCSPDS to store output from the IPCS subcommand or REXX exec. The IPCS output is directed to DDNAME IPCSPDS via a new parameter, PDS or NOPDS, that was added to the SETDEF subcommand:

```
SETDEF PDS|NOPDS
```

The default is NOPDS.

The name of the PDS member will be equivalent to the name of the used IPCS subcommand. Table 9-1 shows the member name for various IPCS commands. For IPCS REXX EXECs the PDS member name is the REXX EXEC name.

*Table 9-1 IPCS subcommands and the equivalent PDS member name*

<b>IPCS subcommand</b>	<b>PDS member</b>
ANALYZE RESOURCE	ANALYZE
ASMK	ASMK
COMK	COMK
COPYCAPD	COPYCAPD
IOSK ALL VALIDATE	IOSK

IPCS subcommand	PDS member
STATUS CPU WORKSHEET	STATUS
SYSTRACE ALL TIME(LOCAL)	SYSTRACE
SYSTRACE TTCH(LIST)	SYSTRACE
VERBX MTRACE	MTRACE
VERBX SADMPMSG	SADMPMSG

### 9.3.1 Printing into a PDS

To get output into a partitioned data set (PDS or PDSE), enter the SETDEF subcommand with the PDS parameter. After you set the parameter PDS using the subcommand SETDEF PDS, messages are routed into a PDS. You can also change the defaults on the IPCS Default Values dialog panel (Option 0 — DEFAULTS) by modifying the following field, and as shown in Figure 9-5.

Message Routing ==> NOPRINT TERMINAL PDS

```

----- IPCS Default Values -----
Command ==>

You may change any of the defaults listed below. The defaults shown before
any changes are LOCAL. Change scope to GLOBAL to display global defaults.

Scope ==> LOCAL (LOCAL, GLOBAL, or BOTH)

If you change the Source default, IPCS will display the current default
Address Space for the new source and will ignore any data entered in
the Address Space field.

Source ==> NODSNAME
Address Space ==>
Message Routing ==> NOPRINT TERMINAL PDS
Message Control ==> CONFIRM VERIFY FLAG(WARNING)
Display Content ==> NOMACHINE REMARK REQUEST NOSTORAGE SYMBOL

Press ENTER to update defaults.

Use the END command to exit without an update.

```

Figure 9-5 IPCS default values with the new PDS option

#### Using the PDS value

Using the PDS value routes and transmits the output messages of the IPCS subcommand to the appropriate PDS member. The following rules for routing output apply:

- ▶ During the period between issuing SETDEF PDS and SETDEF NOPDS, the output messages of any IPCS subcommand are routed into a PDS member with a name equivalent to the name of the subcommand.
- ▶ If the same subcommand is used several times during an IPCS session, IPCS appends the new output to the previous output in the same member.

## Allocating the partitioned data set for routing messages

You must create the PDS before trying to output subcommand messages to it. The name of the partitioned data set can be included in the JCL used to start IPCS in batch mode by means of DDNAME IPCSPDS, for example:

```
//IPCSPDS DD DSN=IPCSU1.SESS7.PDS,DISP=SHR
```

**Note:** If IPCS is started using the TSO command IPCS in line mode or in full-screen mode, you can allocate DDNAME IPCSPDS with the TSO command ALLOCATE, for example:

```
ALLOC DDNAME(IPCSPDS) DSNAME('IPCSU1.SESS7.PDS') SHR
```

At the completion of the subcommand the data set for the PDS member will be closed. It will hold the report for further analysis. Any IPCS subcommand or REXX exec is eligible for PDS output. This support does not apply to subcommands issued from CLIST.

During the time of an IPCS session between setting SETDEF PDS and SETDEF NOPDS, the output messages of any IPCS subcommand will be routed to the PDS member.

**Note:** This support does not apply to subcommands issued from a CLIST.

This support is applied to IPCS subcommands executed in the following three modes of operation:

- ▶ Line mode on a terminal
- ▶ Full-screen mode on a terminal (the IPCS dialog)
- ▶ Batch mode using the terminal monitor program (TMP)

**Note:** The specification of PDSINOPDS is not available on subcommands but is being considered as a future enhancement.

### 9.3.2 IPCSPDS data set allocation attributes

The following commands give an example of using the new PDS function via the IPCS dialog, Panel 6 commands:

```
ALLOC DDNAME (IPCSPDS) DSNAME('MVSSVA.IPCSPDS.FCT.PDS') SHR  
ALLOC DDNAME(IPCSPRNT) DSNAME('MVSSVA.IPCSPRNT.FCT.PS') SHR  
SETDEF PRINT PDS TERMINAL
```

Figure 9-6 on page 192 shows a sample job using the new PDS function in batch.

```

//IPCSPRNT JOB   EEA,GO,MSGCLASS=A,MSGLEVEL=(1,1)
//L2REPORT EXEC  PGM=IKJEFT01,DYNAMNBR=20,REGION=3000K
//SYSPROC DD    DSN=SYS1.SBLSCLIO,DISP=SHR
//IPCSDDIR DD    DSN=SYS2.DDIR,DISP=SHR
//IPCSPDS DD    DSN=PMRXXXXX.YYY.ZZZ,DISP=SHR
//IPCSPRNT DD    SYSOUT=*
//IPCSTOC DD    SYSOUT=*
//SYSTSPRT DD    SYSOUT=*
//SYSUT1 DD     UNIT=SYSDA,SPACE=(CYL,(50,50))
//DUMP DD      DSN=DUMP.S00010,DISP=SHR
//INDD DD      DSN=DUMP.S00010,DISP=SHR
//SYSTSIN DD    *
      IPCSDDIR 'SYS2.DDIR'
      IPCS NOPARM
      SETDEF DDNAME(DUMP) LIST NOCONFIRM NOPRINT NOTERMINAL PDS
      SYSTRACE TTCH(LIST) TIME(LOCAL)
      SETDEF NOPDS
      END
/*

```

Figure 9-6 IPCS batch job to direct output to a PDS

## Allocation attributes

Table 9-2 gives the allocation attributes required for the IPCSPDS data set.

Table 9-2 Allocation attributes for print partitioned data sets

Attribute	IPCSPDS Partitioned Data Set
DDNAME or FILE	IPCSPDS
DISP	NEW, SHR or OLD
DSORG	PO
RECFM	VBA
LRECL	Minimum value: 83 Maximum value: 255 Default value: Value specified in session parameters or 137
BLKSIZE	Must be greater than or equal to LRECL + 4. If you specify none, IPCS uses 4 + (4 * LRECL) as BLKSIZE.
SPACE	You determine the space for the data sets.

**Recommendation:** Although either PDS or PDSE organization can be used, IBM recommends using PDSE (DSNTYPE=LIBRARY) for best performance and to avoid any fragmentation of the PDS space.

The FREE command can be used after termination of any IPCS subcommand to free the allocated partitioned data set:

```
FREE DDNAME(IPCSPDS)
```

### 9.3.3 Directing IPCS output to different mediums

Depending on which message routing parameters are in effect (PRINT, NOPRINT, PDS, NOPDS, TERMINAL, or NOTERMINAL) and depending in which mode (full-screen, line, or batch) you are using IPCS, the output can be directed to different mediums. Note that certain non-report type messages are always routed to the terminal or the SYSTSPRT data set.

Table 9-3 provides a summary of the output destination possibilities. It shows the destination of IPCS output based on message routing parameters and the IPCS mode (in-line, full-screen, or batch) that is used.

Table 9-3 Destination of IPCS output

Message routing parameters	IPCS online output	IPCS batch output
PRINT, TERMINAL, PDS	IPCSPRNT, terminal, IPCSPDS	IPCSPRNT, SYSTSPRT, IPCSPDS
PRINT, NOTERMINAL, PDS	IPCSPRNT, IPCSPDS	IPCSPRNT, IPCSPDS
PRINT, TERMINAL, NOPDS	IPCSPRNT, terminal	IPCSPRNT, SYSTSPRT
PRINT, NOTERMINAL, NOPDS	IPCSPRNT	IPCSPRNT
NOPRINT, TERMINAL, PDS	terminal, IPCSPDS	SYSTSPRT, IPCSPDS
NOPRINT, NOTERMINAL, PDS	IPCSPDS	IPCSPDS
NOPRINT, TERMINAL, NOPDS	terminal	SYSTSPRT
NOPRINT, NOTERMINAL, NOPDS	terminal	SYSTSPRT

### 9.3.4 New messages

These are the new messages introduced with the IPCS PDS enhancement:

**BLS21040I PDS routing rejected--FILE(IPCSPDS) not available**

Explanation: The user specified the PDS keyword on an IPCS subcommand. No partitioned data set is available for the remainder of this IPCS session.  
 System action: IPCS performs no partitioned data set routing.  
 User response: If the output of partitioned data set is required, allocate the partitioned data set to a suitably sized data set. Re-enter the subcommand requesting partitioned data set output. Otherwise, if the output of partitioned data set is not needed, proceed with the IPCS session.  
 Source: Interactive problem control system (IPCS)

**BLS21041I Unable to open PDS FILE(IPCSPDS)**

Explanation: Open processing for a PDS failed. The most probable reason is a failure to pre-allocate a file. IPCS could not open a partitioned data set with IPCSPDS dname.  
 System action: The PDS remains closed.  
 User response: Allocate a partitioned data set before starting the IPCS session or IPCS subcommand.  
 Source: Interactive problem control system (IPCS)

**BLS21042I PDS file is no longer available**

Explanation: The specified PDS file is no longer available for use. Probably, the partitioned data set is full.

System action: IPCS rejects this request and any subsequent requests for PDS output. The current transaction is cancelled.

User response: If PDS file output is required, perform the following:

1. End the IPCS session.
2. Allocate the PDS file to a suitably sized partitioned data set.
3. Start a new IPCS session.

Otherwise, if PDS file output is not needed, proceed with the IPCS session.

Source: Interactive problem control system (IPCS)

**BLS21047I PDS FILE(IPCSPDS) DCB parameters are not valid**

Explanation: IPCS processing encountered data control block (DCB) parameters that are not valid or are inconsistent while attempting to open the partitioned data set. Either:

- The logical record size (LRECL) was not within the allowable range.
- The blocksize was less than LRECL+4.

System action: The partitioned data set file remains closed.

User response: Determine which DCB parameters are incorrect. Correct them.

Source: Interactive problem control system (IPCS)

**BLS21048I PDS file page size is not valid**

Explanation: The specified size of the page for the partitioned data set is not within the allowed range.

System action: The partitioned data set remains closed.

User response: Correct the PDS file page size, which is specified within the active parmlib member.

Source: Interactive problem control system (IPCS)

**BLS21049I PDS FILE(IPCSPDS) is already open**

Explanation: You tried to open the partitioned data set, but the partitioned data set was already open with the IPCSPDS ddname.

System action: IPCS subcommand processing continues.

User response: None required. If IPCSPDS ddname is not the file that you want used, reallocate the partitioned data set for IPCSPDS ddname. Reenter the IPCS subcommand.

Source: Interactive problem control system (IPCS)

## 9.4 IPCS SYSTRACE formatting enhancements

Use the SYSTRACE subcommand to format system trace entries for all address spaces.

In z/OS V1R12 the IPCS SYSTRACE subcommand has been enhanced to:

- ▶ Allow sorting by processor.
- ▶ Generate a performance report based on data from the SYSTRACE output.

### 9.4.1 IPCS SYSTRACE SORTCPU

The SYSTRACE subcommand is used to format system trace entries for all address spaces running on the system. Units of work are dispatched on a processor and on servers with



many processors. The system trace entries on one processor are likely to be unrelated to work running on other processors.

Prior to R12, the SYSTRACE subcommand only showed system trace entries sorted by time merged with trace entries from all processors. When reviewing the system trace to debug a problem there is a need to group system trace entries by processor to analyze the flow of particular units of work.

With z/OS V1R12, this new function adds a SORTCPU option on the SYSTRACE subcommand with a time parameter and the number of system trace entries before and after the specified time. System trace entries are sorted by processor displaying the number of requested system trace entries before and after the specified time.

### **SORTCPU option implementation**

When the SORTCPU option is specified, IPCS displays trace entries for each processor separately in ascending order by processor address. N indicates the number of the trace entries before and after a specified time, which are displayed for each processor.

The syntax is:

```
SORTCPU[(mm/dd/yy, hh:mm:ss:dddddd,n) | (mm/dd/yy, hh:mm:ss:dddddd)]
```

- ▶ Up to n system trace entries before and up to n after system trace entries after the specified date/time will be formatted for each processor.
- ▶ If the number of entries is not specified, it will default to 10.
- ▶ If a date/time is not specified, then all system trace entries are formatted sorted by processor and then time.
- ▶ When a time is specified, the date is required.
- ▶ When SORTCPU is specified, a default of ALL (address spaces) is assumed and any other specification for filtering by ASID is incompatible, such as the CURRENT, ERROR, TCBERROR, ASIDLIST, JOBLIST, and JOBNAME keywords.
- ▶ The SORTCPU parameter is compatible with the following existing SYSTRACE parameters: TCB, WEB, CPU, TIME, EXCLUDE, TTCH, START, STOP, and ALL.

### **SORTCPU examples**

The following are examples of the usage of the SORTCPU parameter:

- ▶ Show all data in processor order:  

```
SYSTRACE ALL SORTCPU
```
- ▶ Show data in processor order, showing a default of 10 entries around 11 a.m. GMT:  

```
SYSTRACE ALL TIME(GMT) SORTCPU(12/30/09,11)
```
- ▶ Show data in processor order, showing 5 entries around 11:45:21:939233 a.m. local:  

```
SYSTRACE ALL TIME(LOCAL) SORTCPU(12/30/09,11:45:21:939233,5)
```

## **9.4.2 IPCS SYSTRACE PERFDATA**

A new report type parameter, PERFDATA, is added to SYSTRACE command to request summary information for the performance data report. The report will help resolve performance problems by identifying the large users of time as seen in the complete SYSTRACE output.

## Report type parameter with PERFDATA

The following statistics are shown if PERFDATA is specified:

- ▶ CPU usage summary
- ▶ CPU breakdown by ASID
- ▶ SRB breakdown by ASID
- ▶ With WHERE information (optional)
- ▶ TCB breakdown by ASID
- ▶ CLKC events with optional WHERE information
- ▶ Lock information
- ▶ SSCH to I/O times

Use these parameters to select the type of report. The PAERFDATA syntax is as follows:

```
PERFDATA ([SHOWTRC] [DOWHERE] [SIGCPU(sss.dddddd)])
```

**SHOWTRC** Requests that a system trace table be displayed in the output. If you do not specify this parameter, the default is to exclude the system trace table.

**DOWHERE** Requests the WHERE commands to be issued for PSWs within CLKC and SRB analysis sections of PERFDATA option output, in order to display the area in a dump in which these address spaces reside. This information may include the name of the load module, the name of a control block or the name of an area of storage containing the PSW address along with an offset. It is displayed on an extra field on the same row of a PERFDATA table as the PSW address.

**SIGCPU** SIGCPU(sss.dddddd) requests that CLKC analysis and WHERE analysis for SRB events should be bypassed for events with processor usage less than the specified time. If you do not specify this parameter, the default is SIGCPU(0.1).

**sss** - Represents seconds. You can specify one to three decimal digits.

**dddddd** - Represents decimal fractions of seconds. You can specify one to six decimal digits.

## IPCS SYSTRACE subcommand examples

The following examples show sections of the performance data report generated by the following IPCS subcommand:

```
SYSTRACE PERFDATA(DOWHERE)
```

Following are some examples of the SYSTRACE subcommand:

- ▶ Show all data in processor order:

```
SYSTRACE ALL SORTCPU
```

- ▶ Show data in processor order, showing a default of 10 entries around 11 a.m. GMT:

```
SYSTRACE ALL TIME(GMT) SORTCPU(12/30/09,11)
```

- ▶ Show data in processor order, showing 5 entries around 11:45:21:939233 a.m. local:

```
SYSTRACE ALL TIME(LOCAL) SORTCPU(12/30/09, 11:45:21.939233,5)
```

**Note:** When SORTCPU is specified, a default of ALL (address spaces) is assumed and any other specification for filtering by ASID is incompatible, such as the CURRENT, ERROR, TCBERROR, ASIDLIST, JOBLIST and JOBNAME keywords. The SORTCPU parameter is compatible with the following existing SYSTRACE parameters: TCB, WEB, CPU, TIME, EXCLUDE, TTCH, START, STOP and ALL.

### 9.4.3 IPCS SYSTEM TRACE panel

The IPCS SYSTEM TRACE panel shown in Figure 9-7 is accessed via Options 2.7.4 starting at the z/OS 01.12.00 IPCS PRIMARY OPTION MENU with Option 2 (Panelid BLSPPRIM). The final Option 4 panel has been updated to get the SYSTRACE performance data reports via Option U, as shown in Figure 9-8 on page 198 (Panelid BLSPPSTRC).

```

BLSPPRIM----- IPCS - SYSTEM TRACE -----
Option ===> U

ALL ASIDs      ===>
CURRENT ASID  ===> *   (default)
ERROR ASID    ===> *   (default)
TCBERROR ASID ===>
ASID list     ===>
Jobnames      ===>
EXCLUDE(BR)   ===>
EXCLUDE(MODE) ===>
TIME format   ===> LOCAL (HEX, GMT or LOCAL)
SORT by CPU   ===> (SHOW      ENTRIES BEFORE AND AFTER TIME
                    - MM/DD/YY,HH:MM:SS:DDDDDD)

SYSTRACE specification:

SYSTRACE CURRENT ERROR TIME(LOCAL)

S = start SYSTRACE, U = get SYSTRACE performance data report,
R = reset panel fields, END/PF3 = terminate SYSTRACE

```

Figure 9-7 IPCS system trace panel

Selecting Option **U**, Figure 9-8 on page 198 is displayed, which shows the IPCS SYSTRACE PERFORMANCE DATA REPORT panel (Panelid BLSPPSUMT).

```

BLSPSUMT----- IPCS - SYSTRACE PERFORMANCE DATA REPORT -----
SELECT OPTION ==>

Select options below or ENTER to use defaults.
- SHOWTRC
- DOWHERE
- SIGCPU      CPU TIME: ____ - Required if SIGCPU selected.

SYSTRACE specification:

SYSTRACE ALL PERFDATA()

SHOWTRC - Include system trace table before performance data report.
          Default is to exclude.
DOWHERE - Issue WHERE commands for SRB and CLKC PSWs.
          Default is not to issue WHERE commands.
SIGCPU  - CPUTIME to be considered when doing SRB and CLKC WHERE
          analysis. The default is SIGCPU(0.1)

ENTER = start SYSTRACE,
END/PF3 = return to SYSTRACE options.

```

*Figure 9-8 IPCS SYSTRACE performance data report panel*

### **Reports generated by the SYSTRACE PERFDATA subcommand**

Figure 9-9 on page 199, Figure 9-10 on page 200, Figure 9-11 on page 200, Figure 9-12 on page 201, and Figure 9-13 on page 201 are some sample reports that are added to the SYSTRACE command to request summary information for the performance data.

PERFDATA Analysis:

CPU#	Went from	To	Seconds	SRB Time	TCB Time
00	18:56:21.326874	18:56:26.608908	5.282034	0.006586	5.262213
03	18:56:21.327232	18:56:26.608933	5.281701	0.007093	5.269708
01	18:56:21.329194	18:56:26.608959	5.279765	0.012058	5.265242
02	18:56:21.332373	18:56:26.603688	5.271314	0.006677	5.260232
04	18:56:26.608908	18:56:26.608908	0.000000	0.000000	0.000000
05	18:56:26.608909	18:56:26.608909	0.000000	0.000000	0.000000
07	18:56:26.608927	18:56:26.608927	0.000000	0.000000	0.000000
06	18:56:26.608930	18:56:26.608930	0.000000	0.000000	0.000000
			21.114815	0.032416	21.057396

SRB time : 0.032416  
 TCB time : 21.057396  
 Idle time : 0.000000  
 CPU Overhead : 11.825359  
 -----  
 Total : 21.114815

Found 22 address spaces in SYSTRACE.  
 Found 48 SRB and SSRB PSWs in SYSTRACE.

Figure 9-9 CPU usage section

CPU breakdown by ASID:

ASID	Jobname	SRB Time	TCB Time	Total Time
0026	SUHIZA98	0.004660	5.257412	5.262073
001A	SUHIZA95	0.004588	5.258561	5.263150
0027	SUHIZA97	0.004664	5.256885	5.261549
0025	SUHIZA96	0.004631	5.250785	5.255417
000A	WLM	0.002302	0.018442	0.020745
0014	JES2MON	0.003854	0.003674	0.007529
0001	*MASTER*	0.000639	0.001464	0.002104
0013	IOSAS	0.000121	0.000412	0.000534
0006	XCFAS	0.000322	0.001075	0.001398
0022	RMF	0.000546	0.001115	0.001661
0008	SMSPDSE	0.000034	0.000124	0.000158
000D	DEVMAN	0.000033	0.000030	0.000063
0012	JES2	0.000080	0.000648	0.000728
0024	CATALOG	0.000012	0.000050	0.000063
0009	CONSOLE	0.000323	0.005744	0.006067
000B	ANTMAIN	0.000024	0.000121	0.000146
0020	VLF	0.000007	0.000018	0.000026
0010	JESXCF	0.000084	0.000129	0.000213
0023	RACF	0.000022	0.000045	0.000068
0021	VTAM	0.000078	0.000162	0.000241
001F	ZTTX	0.000145	0.000489	0.000634
0005	DUMPSRV	0.005238	0.000000	0.005238
		0.032416	21.057396	21.089812

(No. of CPUs in Systrace: 8)

Figure 9-10 CPU breakdown by ASID

SRB breakdown by ASID:

ASID	Jobname	SRB PSW	# of SRBs	Time
0026	SUHIZA98	070C0000 8119A728	704	0.002283
0026	SUHIZA98	070C3000 8119A73A	701	0.002368
0026	SUHIZA98	070C3000 80FF1D68	2	0.000008
				0.004660
-----				
001A	SUHIZA95	070C0000 8119A728	715	0.002189
001A	SUHIZA95	070C3000 8119A73A	714	0.002399
				0.004588

Figure 9-11 SRB breakdown by ASID

CLKC Events:					
ASID	Jobname	SRB/TCB	Clkc	PSW	Where processing (CPU usage for this
-----					
0026	SUHIZA98	005E6868	078D1000	800076B2	ASID(X'0026') 000076B2. SOAKERM+0562
0026	SUHIZA98	005E6868	078D1000	800076B2	(Same as above)
0026	SUHIZA98	005E6868	078D1000	800076B2	(Same as above)
0026	SUHIZA98	005E6868	078D1000	800076B2	(Same as above)
0026	SUHIZA98	005E6868	078D1000	800076B2	(Same as above)
0026	SUHIZA98	005E6868	078D1000	800076B2	(Same as above)
0026	SUHIZA98	005E6868	078D1000	800076B6	ASID(X'0026') 000076B6. SOAKERM+0566
0026	SUHIZA98	005E6868	078D1000	800076B6	(Same as above)
0026	SUHIZA98	005E6868	078D1000	800076B6	(Same as above)
0026	SUHIZA98	005E6868	078D1000	800076B6	(Same as above)

Figure 9-12 CLKC events including the WHERE output

SSCH to I/O times:			
Device	SSCH Issued	I/O Occurred	Duration
-----			
0982	18:56:25.976825	18:56:25.979629	0.002804
Device	SSCH Issued	I/O Occurred	Duration
-----			
03E0	18:56:26.600050	18:56:26.600221	0.000171
03E0	18:56:26.601601	18:56:26.601785	0.000184
03E0	18:56:26.605350	18:56:26.605746	0.000396
			-----
			0.000751
Events for 03E0 :		3	
Quickest I/O :		0.000171	
Slowest I/O :		0.000171	
Total :		0.000751	
Average :		0.000250	

Figure 9-13 SSCH to I/O times

## 9.5 SADMP ASID prioritization

It is possible to have an address space which is not in the default summary address space list and has a higher numeric number. It could be more important than some address spaces that have a lower numeric number and are swapped in. If these important address spaces are not dumped or truncated because the installation runs out of space during dump capture, it will cause difficulties in problem diagnosis.

New function has been added in z/OS V1R12 to allow installations to add address space(s) they consider important to an enhanced summary address space list. In addition, the existing summary address space list has been enhanced by using the COPYDUMP address space list. This allows important address spaces with high ASIDs to be dumped earlier, decreasing

the possibility that the address spaces are not dumped or truncated. This is an improvement for First Failure Data Capture.

### **SADMP processing**

Processing during SADMP data is ordered to capture the most important data first. SADMP captures storage in the following order:

1. Page frame table and its related structures
2. Real storage associated with the minimal address spaces
3. Real storage associated with the summary address spaces
4. Real storage associated with the swapped-in address spaces
5. In-use real storage
6. Page-out storage of minimal address spaces
7. Page-out storage of summary address spaces
8. Page-out storage of swapped-in address spaces
9. Storage of swapped-out address spaces
10. Available real storage

**Note:** ASID 1 to ASID 4 are always dumped during minimal address space processing. Prior to V1R12, primary system address spaces such as ANTMMAIN, CONSOLE, XCFAS, IOSAS, SMXC, WLW, CATALOG, GRS, and SMF were considered the summary address spaces. The summary address space list is hardcoded in module AMDSABSA.

### **Default summary address space list**

The default summary address space list has been changed to match the job name list of the EASYCOPY option of IPCS COPYDUMP. The EASYCOPY option of IPCS COPYDUMP uses the following job name list:

- ▶ ALLOCAS,ANTAS000,ANTMAIN,CATALOG,CONSOLE,DEVMAN,DUMPSRV,GRS,IEFSCHAS,IOSAS,IXGLOGR,JESXCF,JES2,JES3,OMVS,PCAUTH,RASP,SMSPDSE,SMSPDSE1,SMSVSAM,TRACE,WLM,XCFAS

## **9.5.1 ADDSUMM keyword of the AMDSADMP macro**

You can specify new dump tailoring options on the new ADDSUMM keyword of the AMDSADMP macro, or the operators who request the dump can specify additional options when they want additional address spaces to be dumped in prioritized order.

The new ADDSUMM keyword can be used to request additional address spaces via:

- ▶ The AMDSADMP macro by specifying ADDSUMM= ("addsumm-spec-list")
- ▶ The SADMP prompt where the new option ADDSUMM has been added

This allows you to specify address spaces that are to be added to the summary address space list and be dumped in ASID ascending order.

### **Specifying ADDSUMM via the AMDSADMP macro**

The syntax for specifying ADDSUMM=('addsumm-spec-list') via the AMDSADMP macro is:

```
addsumm-spec-list = ASID(address-space-list) [, ASID(address-space-list)] [,...]
```



```

address-space-list = asid[ dlm asid] | jobname[,...]

dlm = 'TO' | ':'

jobname = 'jjjjjjj'

asid = xxxx where xxxx is a hexadecimal number from X'1' to X'FFFF'.

```

Wildcard characters can be used to identify multiple jobnames. The valid wildcard characters are:

- ▶ \* - Zero or more characters, up to the maximum length of the string. An asterisk can start the string, end it, appear in the middle of the string, or appear in several places in the string. A single asterisk for the jobname indicates that all jobnames will match.
- ▶ ? - One character. One or more question marks can start the string, end it, appear in the middle of the string, or appear in several places in the string. A single question mark indicates all jobnames consisting of one character.

Examples for AMDSADMP macro:

```

ADDSUMM=('ASID(28),ASID('JOHNDOE')')
ADDSUMM=('ASID('DB*',40,46:48),ASID('MYJOB?')')

```

### Specifying ADDSUMM via the SADMP prompt

When taking a standalone dump, ADDSUMM 'addsumm-spec-list' is a new option in prompt mode. Additional address spaces can be specified at run-time. If PROMPT is coded on AMDSADMP, message AMD059D will be issued to prompt the operator to dump additional storage or address spaces. The AMD059D message has been updated to prompt the new keyword ADDSUMM.

```
AMD059D ENTER 'DUMP' OR 'SET' OR 'ADDSUMM' WITH OPTIONS, 'LIST' OR 'END'.
```

Any specifications made via the ADDSUMM prompt for AMD059D do not replace the existing ones but rather add new address spaces to the address space list. The operator can respond with ADDSUMM to specify those additional address spaces to be dumped as part of the summary phase:

```
ADDSUMM 'addsumm-spec-list'
```

Addsumm-spec-list is the same as described before. In this case the '=' after ADDSUMM is optional.

**Note:** AMD059D will be issued earlier than in releases prior to z/OS V1R12. It is issued after phase 1 rather than phase 5.

You can respond with a LIST command to display a current configuration of additional virtual areas to be dumped and an address space list to be dumped during a summary phase. The output of the LIST command has been modified:

- ▶ A default summary address space list is displayed.
- ▶ Specified additional summary address spaces are displayed.

Figure 9-14 on page 204 shows the output after LIST is specified in response to AMD059D.

```

AMD067I CURRENT DUMP OPTIONS: CSA ALSO LSQA,
SP(203:205,213:215,229:230,236:237,247:249) IN ASID(PHYSIN) ALSO SP(0:9) IN
ASID('JES2')
  ADDSUMM ASID('ALLOCAS','ANTAS000','ANTMAIN','CATALOG','CONSOLE','DEVMAN',
'DUMPSRV','GRS','IEFSCHAS','IOSAS','IXGLOGR','JESXCF','JES2','JES3','OMVS',
'SMSPDSE','SMSPDSE1','SMSVSAM','WLM','SMF','SMFXC','XCFAS')
  AMD082I WARNING: THE MINASID SPECIFICATION HAS BEEN SET TO 'PHYSIN'.
AMD059D ENTER 'DUMP' OR 'SET' OR 'ADDSUMM' WITH OPTIONS, 'LIST' OR 'END'.

```

Figure 9-14 Output after specifying LIST to AMD059D

### AMD059D prompt examples

Examples for AMD059D prompt:

```

ADDS ASID('IMS',48),ASID('REPORT',34:36)
ADDS ASID('CHART1')
ADDSUM AS('JOB*')

```

### SADMP ASID summary

Table 9-4 summarizes the differences in SADMP between z/OS V1R12 and prior releases.

Table 9-4 Summary table for SADMP ASID support

Prior to z/OS VR12	z/OS V1R12
	A new keyword ADDSUM= in AMDSADMP
AMD059D ENTER 'DUMP' OR 'SET' WITH OPTIONS, 'LIST' OR 'END'.	AMD059D ENTER 'DUMP' OR 'SET' OR 'ADDSUMM' WITH OPTIONS, 'LIST' OR 'END'.
	A new output for the LIST command (even if ADDSUMM= was not specified)
Default summary AS is hardcoded in AMDSABSA - ANTMAIN, CONSOLE, XCFAS, IOSAS, SMXC, WLM, CATALOG, GRS, SMF, ALLOCAS.	Default summary AS list is expanded and synchronized w/ IPCS EASYCOPY - ALLOCAS, ANTAS000, ANTMAIN, CATALOG, CONSOLE, DEVMAN, DUMPSRV, GRS, IEFSCHAS, IOSAS, IXGLOGR, JESXCF, JES2, JES3, OMVS, PCAUTH, RASP, SMSPDSE, SMSPDSE1, SMSVSAM, WLM, XCFAS.
A prompt AMD059D is issued after Phase 5.	A prompt AMD059D is issued after Phase 1.

## 9.6 Console dump support

SDUMP schedules SRBs to the ASIDs to be dumped while global storage is still being captured and global dump exits are being driven. The SRB sets the tasks of the address space non-dispatchable while global storage is being dumped elongating task non-dispatchability time. Statistics show that most of the time the tasks in an address space are non-dispatchable is while global dump exits are executing. For console dumps, it may not be as critical to get a consistent view of global storage and address space storage. It may be more important to reduce the dump's impact to the system.

## z/OS V1R12 support for console dumps

When console dumps are taken for performance or slowdown problems, the impact of stopping the rest of the address spaces can make the system slowdown in a sysplex worse. New function in z/OS V1R12 provides the option to defer the setting of tasks non-dispatchable when console dumps are taken. This reduces the amount of time tasks in the ASIDs being dumped are non-dispatchable, reducing the system impact.

### New dump option DEFERTND

In z/OS V1R12, the new dump option DEFERTND=(YES/NO) is provided on the CHNGDUMP and DUMP command REPLY, as follows:

```
CD SET,SDUMP,DEFERTND=YES
```

```
Id IEE094D SPECIFY OPERAND(S) FOR DUMP COMMAND
R id,DEFERTND=YES/NO,...
```

DEFERTND is also supported through the IEADMCxx parmlib member.

This allows an installation to specify whether SDUMP processing should delay the setting of the tasks non-dispatchable in the address spaces being dumped until after the global capture has completed.

### CHNGDUMP command syntax

The full syntax of the CHNGDUMP command is:

```
CD SET,{NODUMP }
      {OVER } {ADD }
      {SDUMP[(option[,option]...)] }
      [,Q={YES|NO}]
      [,TYPE={XMEM|XMEME}]
      [,BUFFERS={nnnnK|nnnM}]
      [,AUXMGMT={ON|OFF}]
      [,MAXSNDSP=sss]
      [DEFERTND={YES|NO}]
      [,MAXSPACE=xxxxxxxxxM]
      [,MSGTIME=yyyyy]
      [,SYSFAIL,STRLIST=(s-option[,s-option]...)]
      [,NODUMP|OVER|ADD]
      {{SYSABEND}[,SDATA=(option[,option]...)] [,NODUMP] }
      {{SYSUDUMP} |,PDATA=(option[,option]...)|,OVER } |,ADD
      {SYSDUMP[(option[,option]...)] [,NODUMP] } |,OVER |,ADD
      {ABDUMP,TIMENQ=yyyy}
```

The default is DEFERTND=NO.

In response to the DUMP command, the system prompts you with the following message for the dump options you want to specify:

```
* id IEE094D SPECIFY OPERAND(S) FOR DUMP COMMAND
```

You must use the REPLY command to respond to the IEE094D message. A new keyword, DEFERTND =Yes/No, is added to the DUMP options as follows:

```
R id,U
  or
R id[,ASID=(n[,n]...)] [,JOBNAME=(name[,name]...)] [,TSONAME=(name[,name]...)]
  [,DSPNAME=(dspname-entry[,dspname-entry]...)]
```

```
[,{PROBDESC|PROB|PD}=key-spec] [,REMOTE=(request[,request]...)]
[,SDATA[(option[,option]...)] [,STOR=(beg,end[,beg,end]...)]
[,STRLIST=(s-option[,s-option]...)] [,DEFERTND=YES/NO]
[,CONT|,END]
```

**Note:** The DUMP option specified in response to IEE094D overrides the CHNGDUMP option. These options only affect dumps initiated via the DUMP command.

The DISPLAY DUMP,OPTIONS command output message IEE857I is updated to include the DEFERTND= value.

Table 9-5 shows the rules that SDUMP uses to determine whether it will defer the settings of tasks non-dispatchable.

Table 9-5 Rules that SDUMP uses

DEFERTND= is specified in DUMP reply or in IEADMCxx as:	DEFERTND= is specified by CHNGDUMP command as:	Result of DEFERTND
YES	YES	YES
YES	NO	YES
NO	YES	NO
NO	NO	NO
not used	NO	NO
not used	YES	YES

## 9.7 Service aids support for BAM XTLOT

The TIOT resides in storage below 16 M and includes a 24-bit UCB address in the Device Entry for allocated devices. This requires the system to capture the UCB when the UCB resides above the 16 M line. The XTLOT resides above the 16 M storage and contains a 31-bit UCB address. This means that no explicit UCB capture is required.

BSAM, QSAM, BPAM, and OCE components of DFSMS support exploitation of XTLOT with uncaptured UCBs and DSABs above the 16 M line. The options of S99TIOEX, S99ACUCB, S99DSABA of dynamic allocation (DYNALLOC or SVC 99) have been supported for years but only by VSAM and media manager.

With z/OS V1R12 DFSMS allows BSAM, BPAM, QSAM, and EXCP to support XTLOT, NOCAPTURE, and DSAB above the line for dynamic allocation for DASD, tape, and dummy data sets. This provides VSCR benefits and allows support for more than about 3200 allocated data sets. Also see “Dynamic allocation support” on page 169.

With z/OS V1R12, new support allows SNAP, SYSABEND, SYSMDUMP, SYSUDUMP, SVC Dump, and Superzap to use XTLOT, NOCAPTURE, and DSAB above the line. This improves VSCR because less storage below the 16 M line is used.

### SNAP and SNAPX support

If the DD name for the SNAP dump has the XTLOT, UCB nocapture, or DSAB-above-the-line options of dynamic allocation, your DCB must point to a DCBE before you open the DCB. The

DCBE must have the LOC=ANY option, which can be coded even if the DD name does not have any of these dynamic allocation options. To point the DCB to the DCBE, code the DCBE operand on the DCB macro, or store into the DCBDCBE field if you also turn on the DCBH0 and DCBH1 bits.

**Note:** The DCBE can reside above the 16 MB line even if your program runs in 24-bit.

In order for the OPEN of the DCB to succeed, it is necessary for the NON\_VSAM\_XTIOT=YES option in the DEVSUPxx member of PARMLIB to be active.

For dynamic allocation, the XTIOT option bit is S99TIOEX, the UCB nocapture option bit is S99ACUCB, and the DSAB-above-the-line option bit is S99DSABA. These options are defined by the IEFZB4D0 macro.

For more information about the XTIOT option for dynamic allocation of the dump data set, see the S99PARMS programming interface in *MVS Data Area Volume 6*.

The following IPCS output shows extended TIOT entries in SDUMP, which was created by DYNALLOC with S99TIOEX, S99ACUCB, and S99DSABA on. It shows extended TIOT entries displayed via the IPCS command SUMMARY FORMAT ALL.

```
TIOT: 006DAFD0
      JOB..... IEESYSAS  STEP..... MAINASID

EXTENDED TIOT ENTRIES
      XTIOTPTR LN-STA DDNAME  TTR-STC  STB-UCB  ACTUCBP
      000EB138 14010100 SYSDUMP  0000DF00 90000000 00F01AC0
```





## z/OS 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 has been enhanced in z/OS V1R12 with new capabilities that let you print more output. This chapter describes the following enhancements:

- ▶ The IP PrintWay extended mode component of Infoprint Server can now print a greater number of data sets in the same print job (JES output group). This means that you can now submit jobs that create a large number of output data sets on the JES spool, up to the limit that JES allows.
- ▶ Infoprint Server now supports a greater number of jobs active in the system at one time, up to the maximum that JES currently allows.
- ▶ The Infoprint Server Line Printer Daemon (LPD) can now receive files larger than 2 GB. This means that you can use a TCP/IP LPR command or the Infoprint Server Port Monitor for Windows clients to print large files through Infoprint Server.
- ▶ Infoprint Server now gives higher priority to processing existing print jobs (spooling and printing jobs) versus receiving new print jobs. This change is expected to improve throughput of existing print jobs.
- ▶ Infoprint Central now lets authorized users see information about all the documents (output data sets) in a print job, up to a limit that the administrator can set.
- ▶ In addition, Infoprint Port Monitor V3 for Windows runs on additional Windows platforms and provides new function:
  - Runs on Windows Server 2003, Windows Server 2008, Windows 7, Windows Vista (Business Edition, Enterprise Edition), and Windows XP.
  - Supports Windows Terminal Services and Windows fast user switching.

## 10.1 IP PrintWay extended mode processing

The IP PrintWay extended mode component of Infoprint Server can now print a greater number of data sets in the same print job (JES output group). This means that you can now submit jobs that create a large number of output data sets on the JES spool, up to the limit that JES allows. Infoprint Server uses the JES spool as a temporary application print data repository. Print data may be received to the JES spool through any of the several print protocols supported by the Infoprint Server. Host applications make print data available for Infoprint Server processing by writing it directly to the JES spool.

With z/OS V1R12, this new support is designed to increase job and documents per job to maximum limits imposed by the job entry subsystem. This should provide scalable performance as the number of print documents per job increases. This support is available for the IP Printway extended mode processing shown in Figure 10-1.

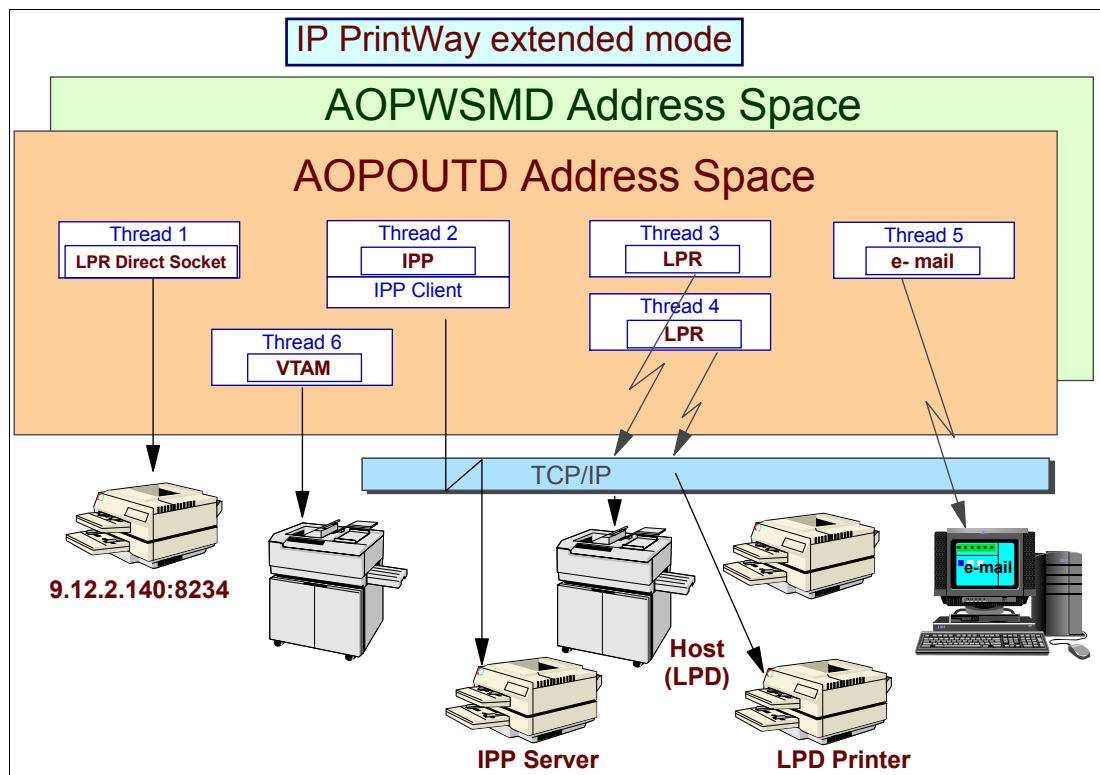


Figure 10-1 IP Printway extended mode processing

### Processing output groups

With automatic data set grouping with extended mode, you select how you want IP PrintWay to transmit data sets that are in the same JES output group. When you run IP PrintWay extended mode, select the automatic data set grouping (extended mode) field. Automatic data set grouping is the most efficient method of transmitting data sets to the printer. Also, it makes sure that data sets in the same output group print together.

**Note:** This support for processing of output groups only works in IP Printway extended mode and not in basic mode.



## Processing of large numbers of documents

Many installations have experienced bottlenecks when processing large numbers of documents during a short batch window. Typically printers are not busy but are waiting for print data to be spooled. Infoprint Server is busy de-spooling work from JES rather than keeping the printers at maximum throughput.

With z/OS V1R12, IP PrintWay raises the priority of the task that is spooling data to the output device and reduces the priority of the task that is de-spooling data from JES. This improves printer throughput and overall document processing time. This means that you can submit batch jobs that create a large number of output data sets on the JES spool, up to the limit that JES allows.

## 10.2 ANFUXRSP user exit

The ANFUXRSP response notification exit was only supported by Infoprint Server IP PrintWay basic mode. With z/OS V1R12, IP PrintWay extended mode is the recommended choice and is now supporting this exit. With this exit you can take actions based on the status of the current processing of the document.

**Note:** There are sample user exits in the SYS1.SAMPLIB. Any combination of user exits can be used, from none to all user exits. The samples provided in SYS1.SAMPLIB are written with the assumption that it is the only user exit being used. Be aware of the fields XTPWORK1 and XTPWORK2 in the interface ANFUEXTP, which is the common interface to all Infoprint user exits, and is used by all the user exits. Following are the Infoprint Server PrintWay user exits and the call sequence:

- ▶ Routing exit
- ▶ Begin Data Set exit
- ▶ Record exit
- ▶ End Data Set exit
- ▶ Response Notification exit
- ▶ SMF exit

### Installation considerations

You can use the ANFUXRSP Response notification user exit to write user-customized processing at the completion of document processing. The exit is called for both successful and unsuccessful transmissions. This exit provides the option of providing customized user exit processing based on the Response Notification code.

When entering the exit, it can send a message to either the operator's console or the IP PrintWay message-log data set, or both. The exit can also initiate actions outside of IP PrintWay by calling modules that are external to IP PrintWay. For example, if IP PrintWay transmission to a printer fails, the user response notification exit issues console messages to the operator to check the printer. If the IP PrintWay transmission completes successfully, the user response exit logs an audit record that the print completed.

**Note:** IP PrintWay calls the same response exit for all data set transmissions. You can install different response exits for different IP PrintWay FSSs.

## 10.3 LPD and API support for large files

In z/OS V1R12, the Infoprint Server LPD can receive files larger than 2 GB. To support such files, the Infoprint Server API can now return a document size value greater than 2 GB. These are the enhancements to the API:

- ▶ The DocumentInfo data structure has a new field, high\_size, to contain the high-order word of the document size.
- ▶ The existing GetJobInfo and EnumJobs API functions store the size of a document in two fields in the DocumentInfo data structure: size and high\_size.
- ▶ The new GetAPIVersionNumber function returns the API version number. In z/OS V1R12, the API version number is 1. You can use the existence of this function to determine that the DocumentInfo data structure contains the high\_size field.

**Note:** The DocumentInfo data structure contains information about a document. It is included in the JobInfo data structure. The status and format fields are described by the DocumentStatus and DocumentFormat enumerations.

### DocumentInfo - information about a document in a job

The DocumentInfo data structure contains information about a document. It is included in the JobInfo data structure. The status and format fields are described by the DocumentStatus and DocumentFormat enumerations.

```
typedef struct
{
    char status; /* Current document status */
    char format; /* Format of the document */
    char reserved[2];
    unsigned int size; /* low-order word of size in
octets of document on spool */
    char* name; /* Filename of document */
    unsigned int high_size; /* high-order word of size in
octets of document on spool */
} DocumentInfo;
```

The size of a document is stored in two fields: size and high\_size. You can calculate the total size and store it in the field total as follows:

```
unsigned long long total = (high_size << 31) + size;
```

The high\_size field is available only in API Version 1 and later versions. You can use the GetAPIVersionNumber function to obtain the API version number.

### GetAPIVersionNumber - return the API version number

The GetAPIVersionNumber function returns the version number of the API. Use this function if your program uses functions and data structures that are not available in all API versions. The format of this API is:

```
#include <aopapi.h>
int GetAPIVersionNumber();
```

**Note:** The GetAPIVersionNumber function is available in z/OS V1R12 and later releases. You can use the dlsym(0) function to determine whether the GetAPIVersionNumber function is in aopapi.dll. If the function is not found, the API is at Version 0.

### 10.3.1 Installation considerations

If you use the Infoprint Server API, do the following tasks. These tasks are required by all installations. Optional tasks are required only if the listed condition applies.

- ▶ Modify the program that uses the API in the following situations:
  - If the program that uses the API runs only on z/OS V1R12 and later releases, get the size of the document from these fields in DocumentInfo: size and high\_size.
  - If the program that uses the API runs on z/OS V1R12 and earlier releases, code these steps:
    - Use the dlsym() function to determine whether the GetAPIVersionNumber function is in aopapi.dll. For information about dlsym(), see *z/OS XL C/C++ Run-Time Library Reference*, SA22-7821.
    - If the GetAPIVersionNumber function is not found, the DocumentInfo structure is at API Version 0. Get the size of the document from this field in DocumentInfo: size.
    - If the GetAPIVersionNumber function is found, the DocumentInfo structure is at API Version 1 or later. Get the size of the document from both of these fields in DocumentInfo: size and high\_size.

**Note:** In V1R12, the Infoprint Server LPD can receive files larger than 2 GB. To support files larger than 2 GB, the Infoprint Server API can now return a document size value greater than 2 GB.

## 10.4 Printer Inventory for PSF enhancements

With z/OS V1R12 you can now specify new PSF printer attributes in the Printer Inventory when you create or modify a PSF functional subsystem application (FSA) definition for AFP Download Plus:

- ▶ The Infoprint Server ISPF panel for a PSF FSA definition for AFP Download Plus contains new fields.
- ▶ The Infoprint Server Printer Inventory Definition Utility (PIDU) lets you specify new attributes for AFP Download Plus.
- ▶ The Infoprint Server migration program (aopmig) supports the new attributes when it migrates PSF start-up procedures to the Printer Inventory.
- ▶ Infoprint Central displays the new attributes when you view information for an AFP Download Plus sender.

### 10.4.1 Using AFP Download Plus

To implement this product, you specify the new attributes when you create or modify PSF FSA definitions in the Printer Inventory for AFP Download Plus. To create definitions, you can use ISPF panels, the PIDU program, and the Infoprint Server migration program. To modify definitions, you can use ISPF panels or the PIDU program.

**Note:** For information about Infoprint Server ISPF panels and the PIDU program, see *z/OS Infoprint Server Operation and Administration*, S544-5745. Use Infoprint Central to work with AFP Download Plus senders.

## 10.5 Infoprint Central

Infoprint Central is a Web-based print management system primarily for help desk operators that is available beginning with z/OS V1R5. However, other authorized users or job submitters can also use it. Infoprint Central works with IP PrintWay extended mode. With Infoprint Central, you can:

- ▶ Work with print jobs
- ▶ Work with printers
- ▶ Work with NetSpool logical units (LUs)
- ▶ Display printer definitions
- ▶ Check system status

### Infoprint Central users

Infoprint Central provides functions for the following types of users:

- ▶ Help desk operators

Infoprint Central helps you respond to callers' questions about where print jobs have printed and why printers are not working. For example, you can find the status of print jobs, cancel print jobs, and move print jobs. Also, you can find the status of printers, restart printers, and redirect printers to alternate printers.

- ▶ Printer operators

Infoprint Central lets you work with printers. For example, you can start and stop printers, change the forms loaded in printers, and redirect all print jobs to a different printer.

- ▶ Job submitters

Infoprint Central lets you find out where and when your print job printed, see if a printer is busy, and find the name of a printer in your building.

### IP PrintWay extended mode

When you run IP PrintWay extended mode, Infoprint Central lets operators perform functions that cannot be done with other tools such as Infoprint Server ISPF panels, SDSF, or JES commands. For example, you can redirect printers to alternate printers, cancel print jobs that are currently printing on IP PrintWay printers, and see all messages in the common message log for print jobs and printers. Infoprint Central does not let you work with IP PrintWay basic mode printers or print jobs.

If you run PSF for z/OS, Infoprint Central lets operators see detailed information about print jobs submitted through Infoprint Server. For example, you can see information about print jobs that are no longer on the JES spool and all messages in the common message log for print jobs. If you customize PSF to use the Printer Inventory, you can also use Infoprint Central to work with PSF printers.

You must customize and start the z/OS HTTP Server to display Infoprint Central Web pages.

### 10.5.1 Infoprint Central enhancements

With z/OS V1R12, Infoprint Central now lets authorized users do these functions:

- ▶ Display information about all documents in a print job. For example, you can see the name and size of each document.

- ▶ Limit how many print jobs and documents Infoprint Central displays. The AOPLIMIT environment variable limits how many objects Infoprint Central displays. This is useful if you want to avoid timeouts that can occur when Infoprint Central displays a large number of objects. In z/OS V1R12, you can specifically limit the number of print jobs and documents that Infoprint Central displays in the AOPLIMIT\_JOBS and AOPLIMIT\_DOCS environment variables.

## 10.5.2 Installation considerations

To use the Infoprint Central enhancements, do the following required tasks:

- ▶ Specify the maximum number of jobs and documents Infoprint Central displays in the AOPLIMIT\_JOBS and AOPLIMIT\_DOCS environment variables in the z/OS HTTP Server environment variable file, httpd.envvars.

**Note:** This task is optional only if you want to specify a different limit for jobs or documents than for other objects that Infoprint Central displays.

- ▶ See information about documents in Infoprint Central, as follows:
  - In the left panel, select **Work with print jobs**, as shown in Figure 10-2 on page 216.
  - On the Infoprint Server Print Jobs panel, select **Infoprint Server Print Jobs** and search for a job.
  - Click the job ID for the print job.
  - On the Print Job Information panel, expand the Documents section.

### Infoprint Central Work with Print Jobs panel

The Work with Print Jobs panel shown in Figure 10-2 on page 216 lets you select the print jobs you want to work with. You can work with any print jobs that are currently on the JES spool. You can also see information about any Infoprint Server print jobs that have already finished processing.

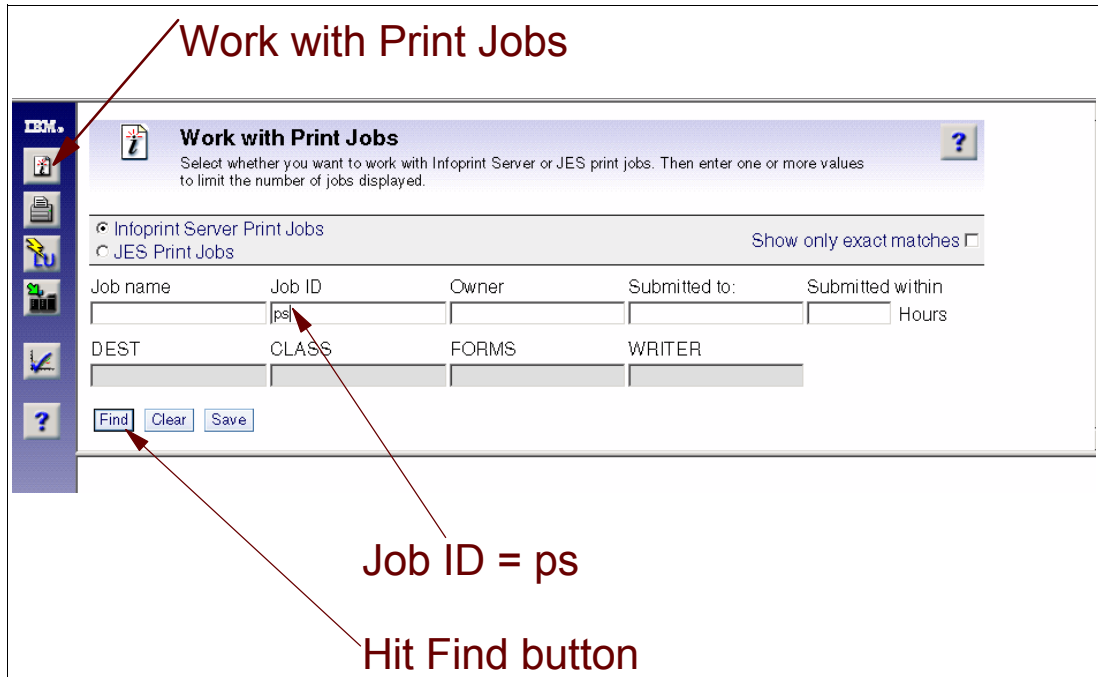


Figure 10-2 Infoprint Central Work with the Print Jobs panel

### Finding print jobs

Select the type of print jobs to display. Select either **Infoprint Server print jobs** or **JES print jobs**. The type you choose determines which print jobs are displayed, how much information you see about the print jobs, and which actions you can take on print jobs. The type you choose also changes which search fields are available on this panel.

### Infoprint Server print jobs

First select **Infoprint Server print jobs**. This option lets you:

- ▶ See more information about Infoprint Server print jobs, such as more detailed status.
- ▶ Take more actions on Infoprint Server print jobs, such as move a print job to another printer.
- ▶ See information, including messages, for Infoprint Server print jobs that have finished processing and are no longer in the JES spool.
- ▶ Search for print jobs using search values known only to Infoprint Server, such as the Windows logon name of the job submitter.

### JES print jobs

If the print job is not found, select **JES print jobs**. This option lets you:

- ▶ Work with print jobs that Infoprint Server does not process, such as print jobs submitted directly to a PSF-controlled printer using JCL.
- ▶ Work with print jobs that Infoprint Server has not received due to an error in job submission. For example, job submitters might fail to specify the correct CLASS value on their DD or OUTPUT JCL statements, so IP PrintWay does not select the print jobs from the JES spool.

## Find print job option 2

A separate option could be to enter a value in the “Submitted within” field to limit the number of jobs found. Do not enter a value if you want to find all possible print jobs. Use this field as follows:

Enter a value in at least one of the search fields. If you enter values in more than one field, only print jobs that match all values are displayed.

- ▶ Do not enter an asterisk (\*) or a question mark (?) as a wildcard symbol.
- ▶ If you enter only the first characters of a value in a search field instead of the full value, clear “Show only exact matches.”
- ▶ If you do not know the correct uppercase and lowercase letters for case-sensitive fields, clear “Show only exact matches.”

## Save button

Select **Save** to save these search values, which are displayed the next time you use this panel when you select **Find**.

## 10.6 Migration considerations

These changes require a reformatting of the internal inventory data base files. After migrating to z/OS V1R12, when you start Infoprint Server, it automatically reformats the Printer Inventory to the Version 2 format, which supports jobs with a large number of documents. It requires at least two times more than for the available space required to complete the step.

If the Version 2 Printer Inventory files exist after falling back to z/OS V1R11 or z/OS V1R10, remove them from the Infoprint Server base directory. Be careful not to remove any Version 2 files while running z/OS V1R12 because Infoprint Server on z/OS V1R12 requires Version 2 Printer Inventory files.

### Printer Inventory and base directories

Sufficient space is needed in the Infoprint Server base directory (default is `/var/Printsrv`) for the new reformatted database files initially created after migration to z/OS V1R12. The default configuration file is in `/etc/Printsrv/aopd.conf`. It contains the name of the base directory.

In z/OS V1R12, the format of the Infoprint Server Printer Inventory files has changed from Version 1 to Version 2. When you start Infoprint Server on z/OS V1R12 for the first time, Infoprint Server reformats the Version 1 Printer Inventory files and creates Version 2 Printer Inventory files. Both Version 1 and Version 2 Printer Inventory files exist in the Infoprint Server base directory.

The output from the reformat of the Printer Inventory is in the following file:

```
/var/Printsrv/aopreformat.log
```

The final message in the `aopreformat.log` file should be for a successful creation:

```
AOP150I The Printer Inventory has been reformatted. (program:aopd)
```

**Important note:** Infoprint Server on z/OS V1R12 uses the Version 2 Printer Inventory files. If you fall back to z/OS V1R11 or V1R10, Infoprint Server uses the Version 1 Printer Inventory files.

If you start Infoprint Server on z/OS V1R12 a second time after falling back to a previous z/OS release, Infoprint Server uses the existing Version 2 Printer Inventory files that it created the first time you started Infoprint Server on z/OS V1R12. It does not reformat the Version 1 Printer Inventory files again.

If you want Infoprint Server to reformat the Version 1 Printer Inventory files again, remove the Version 2 Printer Inventory files before you start Infoprint Server on z/OS V1R12. Because the Version 2 Printer Inventory files no longer exist, Infoprint Server reformats the Version 1 Printer Inventory files and creates a new set of Version 2 Printer Inventory files.

In most cases, you should remove the Version 2 Printer Inventory files if they exist. If you do not remove them, any changes that the administrator made to the Version 1 Printer Inventory on z/OS V1R11 or z/OS V1R10 are not in the Version 2 Printer Inventory. In addition, Infoprint Central on z/OS V1R12 cannot display historical information for jobs that Infoprint Server processed on z/OS V1R11 or z/OS V1R10.

## Infoprint Server health check

The new and previous release data base files will coexist in the same directory. A coexistence PTF is provided with APAR OA32093. The PTF will include a new migration health check to ensure sufficient space for the data base copy and conversion. You should run the new health check, INFOPRINT\_V2DB\_CHECK, for IBM Health Checker for z/OS. To run these health checks, Infoprint Server must be started.

### APAR OA32093

This APAR introduces two health checks for Infoprint Server. This PTF can be installed in installations that do not use the IBM Health Checker. Health check INFOPRINT\_V2DB\_CHECK provides information for clients who have migrated to z/OS V1R12 and then fallen back to an earlier z/OS release. This health check runs automatically the next time you start Infoprint Server and the Health Checker is running.

Health check ZOSMIGV1R12\_INFOPRINT\_INVSIZE helps you migrate to z/OS V1R12. It checks whether the Infoprint Server base directory has sufficient available space for the new Printer Inventory files that Infoprint Server creates when it starts on z/OS V1R12. You must activate this health check before running it.

**Note:** After installing this PTF, issue this console command to refresh the LNKLIST. It makes the AOPADDHC, AOPHCK, and AOPHCMSG modules in SYS1.LINKLIB callable:

```
F LLA,REFRESH
```

For information about changes to Infoprint Server in z/OS V1R12, see *z/OS Migration*, GA22-7499. For information about how to activate and run health checks, see *z/OS IBM Health Checker for z/OS: User's Guide*, SA22-7994.

## 10.6.1 Infoprint Port Monitor V3

The Infoprint Server Windows client consists of the Infoprint Port Monitor for Windows, which is software that runs on a Windows workstation. It lets you print files on z/OS printers just like you print files on local Windows printers. You can submit the files from any Windows



application that has a printing function. After the Infoprint Port Monitor is installed and configured on the Windows system, it automatically sends documents to the Print Interface component of Infoprint Server.

Version 3 of Infoprint Port Monitor for Windows runs on additional Windows systems and provides new function, as follows:

- ▶ Runs on Windows Server 2003, Windows Server 2008, Windows 7, Windows Vista (Business Edition, Enterprise Edition), and Windows XP.
- ▶ Supports Windows Terminal Services and Windows fast user switching.

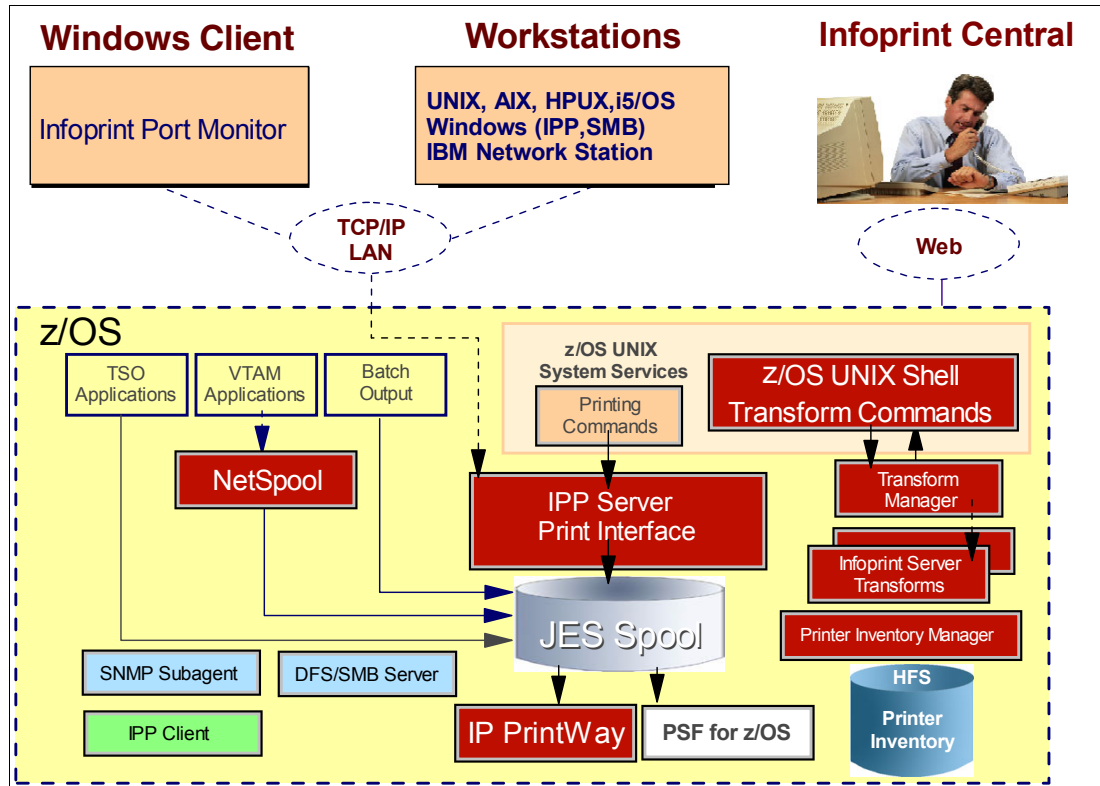


Figure 10-3 Infoprint Server components

**Note:** The Infoprint Port Monitor does not run on Windows 64-bit systems.





## TSO/E enhancements

TSO/E is a base element of z/OS. TSO/E allows users to interactively share computer time and resources. In general, TSO/E makes it easier for people with all levels of experience to interact with the MVS system.

TSO/E has advantages for a wide range of computer users. TSO/E users include system programmers, application programmers, information center administrators, information center users, TSO/E administrators, and others who access applications that run under TSO/E. Chapters of this document describe the major tasks that each of these users can perform using TSO/E.

This chapter describes the TSO/E enhancements of z/OS V1R12:

- ▶ Multiple TSO/E logons.
- ▶ The XTIO, nocapture UCB, and DSAB above-the-line support. Now the OPEN macro instruction and the I/O services support these functions for dynamically allocated non-VSAM data sets, in addition to the support provided by EXCP and VSAM access methods, on previous releases.
- ▶ Password special character support. TSO/E, starting with z/OS V1R12, does not check passwords for syntax check.

## 11.1 Multiple TSO/E logons

In z/OS V1R4, JES2 stopped preventing duplicate TSO/E logons. As a result, duplicate instances of a given user ID can log on at the same time to different systems, even in the same JES2 MAS, depending on the scope of the SYSIKJUA enqueue. If the enqueue has a scope of SYSTEM, multiple TSO/E logons are allowed. This configuration is supported, although users should note the following restrictions.

Restrictions:

Messages sent to the users who are logged on to more than one system at a time might not go to the expected instance of the user.

Logon reconnect will only allow users to reconnect to an instance of their user ID on the same system where they attempt to log on.

ISPF profile sharing or changes to default ISPF data set names might be needed to avoid errors in ISPF with multiple logons. See *z/OS Interactive System Productivity Facility (ISPF) Planning and Customizing*, GC34-4814 for more details about configuring ISPF to support multiple logons.

### Preventing multiple logons for the same user

If you want to prevent duplicate logons within a JES2 MAS, take the following steps:

1. Merge all existing versions of SYS1.UADS into a single version of the data set.
2. Modify the default RNLs by deleting the SYSDSN entry for SYS1.UADS from the SYSTEMS exclusion RNL. Then add SYSIKJUA as a generic qname in the SYSTEM inclusion RNL.

### JES2 TSO/E logon support

JES2 allows TSO/E users to log on once per system in a sysplex. However, TSO/E could not support it because of the SEND command. If a message is issued on system A to a user on system B and C, the MVS SEND function fails with an IKJ572I message and TSO/E SEND fails with an IKJ55072I message:

```
IKJ55072I USER(S) IBMUSER NOT LOGGED ON OR TERMINAL DISCONNECTED, MESSAGE  
CANCELLED
```

#### 11.1.1 z/OS V1R12 JES2 support

Starting with z/OS V1R12, JES2 officially supports multiple TSO/E logons of the same user ID in the MAS. Even though it was possible to log on to TSO/E multiple times in the same MAS before this release, JES2 did not officially support it.

With z/OS V1R12 JES2, the message is sent to the first instance of a user JESXCF finds. This allows TSO/E to support multiple logons for improved efficiency.

Users can now log on to TSO/E once per system in a sysplex if it is configured to allow this.

**Note:** This is not a good idea if there is a SYS1.UADS shared between systems. Notice that LOGON RECONNECT only works from the system a user is on. This reduces the need to manage multiple user IDs and passwords.

## Installation considerations

As an example of this support, user IDs KASPER, KASPER1, KASPER2, and KASPER3 allow for improved multitasking and operations by TSO/E users. JES2 has also improved notification processing by sending messages to the member where a job was submitted. If the target user ID is not logged on there, the old method is used.

Multiple logons are not supported on lower-level systems of JES2 due to the restrictions on SEND command processing that were described earlier. The SEND that is issued from a lower-level system will still fail as it did before. A SEND issued from a higher-level system will work for any target system but requires that the users could be logged on more than once anyway.

**Important:** In most cases, TSO/E users are stored in a RACF database, not in a SYS1.UADS data set. However, many installations retain a SYS1.UADS data set with a few users as a backup. Those installations in particular might want to keep separate versions of SYS1.UADS for each system in order to avoid serialization problems if the scope of SYSIKJUA is SYSTEM.

## 11.2 XTIO, nocapture UCB, and DSAB above the line support

In z/OS V1R12, BSAM, BPAM and QSAM are enhanced to support the existing XTIO, UCB nocapture, and DSAB above the 16 MB line options of dynamic allocation. Previously, VSAM and EXCP were the only access methods to support these dynamic allocation options. In z/OS V1R12 the EXCP support is enhanced and BSAM, BPAM and QSAM support is provided. This new support applies to dynamic allocation of DASD, tape, and dummy data sets, and cases where PATH= is coded. EXCP support, including the EXCPVR and XDAP macros, is also affected. These enhancements provide virtual storage constraint relief especially in the areas of DASD and tape support, and enable you to have more than about 3200 dynamically-allocated data sets.

TSO/E has been improved to fully support this enhancement. Authorized programs that allocate data sets, using SVC 99 (Dynaloc), running under TSO/E, can leave more storage below the 16 MB line for other uses. Using this facility, DB2 users that also use BSAM, BPAM, or QSAM data sets, may also have more storage for data set allocation. JCL allocation does not support the XTIO, UCB nocapture or DSAB above the 16 MB line options. Only dynamic allocation (SVC 99) supports these options. Although the NOCAPTURE dynamic allocation option does not have DD statement or TSO ALLOCATE option equivalents, the main non-VSAM access methods (BPAM, BSAM, and QSAM) do support the NOCAPTURE option of dynamic allocation (SVC 99).

### Activating TSO/E XTIO support

To exploit these enhancements, you need to set a new DEVSUPxx PARMLIB option, and change the following programs:

- ▶ Programs that call dynamic allocation and want to exploit XTIO, UCB nocapture, and DSAB above the 16 MB line options of dynamic allocation. They must check that DFAXTBAM is set ON before they exploit any of these dynamic allocation options. This bit signifies that the installation enables this function by setting the option in the DEVSUPxx member of PARMLIB.
- ▶ Programs that open files that might have been dynamically allocated with XTIO, UCB nocapture, or DSAB above the 16 MB line options of dynamic allocation. These programs need to:

- Have no dependency on the changes at all, and so need to specify a new DCBE macro LOC=ANY option
- Have other changes made in order to support XTIO, nocapture UCBs, and DSAB above the 16 MB line. These changes *must* be made prior to using the new DCBE macro LOC=ANY option.

### DEVSUPxx parmlib member option

Before using the XTIO option for non-VSAM data sets, you must activate this function, coding a new DEVSUPxx parmlib member parameter:

```
NON_VSAM_XTIO={ YES|NO }
```

The default value of this parameter is NO. This controls whether the access method OPEN macro instruction supports these three options of the data set dynamic allocation function. Set NO if you are concerned that some programs, including installed products, might not correctly handle these options. You can set YES if all programs that might process data sets that were dynamically allocated by other programs can handle these options. Setting YES, but not using these options, has no effect on virtual storage or performance. This parameter is described in Chapter 29, “DEVSUPxx (device support options)” in *z/OS MVS Initialization and Tuning Reference*, SA22-7592.

### Application programming

The DCBE macro instruction, in z/OS V1R12, has a new operand, as follows:

```
LOC={ANY|BELOW}
```

Specify this option, before issuing the OPEN or RDJFCB macro to a non-VSAM data set, if the XTIO, UCB NOCAPTURE, or DSAB above the 16 MB line options of dynamic allocation will be used. If you specify LOC=BELOW or do not code LOC=, you signify that the program does not support these three options of dynamic allocation. These are the S99TIOEX, S99ACUCB, and S99DSABA options.

By setting LOC=ANY, you signify that the program is either not affected by or that it allows for any of the following possibilities:

- ▶ The DCBTIO field (offset in TIO to an entry) might contain zeroes or contain a TIO offset.
- ▶ The DEBXDSAB field (address of DSAB) might point above the 16 MB line.
- ▶ The DSABTIO field might point to an XTIO or to a TIO entry.
- ▶ The UCB address field in the DEB might be four bytes or three bytes (test the DEB31UCB bit).
- ▶ The TIOEFSRT field might contain zeroes instead of a UCB address.

**Note:** IBM recommends that, regardless of dynamic allocation and the NON\_VSAM\_XTIO setting, you always specify DCBE LOC=ANY when either your program does not reference TIO, UCB, and DSAB, or that it is correctly modified to support the XTIO, UCB NOCAPTURE, or DSAB above the 16 MB line options.

### New message IEC133I and abend 113-4C

Once the S99TIOEX, S99ACUCB, and S99DSABA options of SVC99 are coded, during the dynamic allocation request, it becomes necessary that the application program also has a DCBE macro with the LOC=ANY parameter coded *and* NON\_VSAM\_XTIO=YES is specified in DEVSUPxx. If these conditions do not exist, one of the situations shown in Table 11-1 on page 225 can happen.

Table 11-1 OPEN macro instruction and message IEC133I

NON_VSAM_XTIOT=	DCBE LOC=	Result
NO or not coded	BELOW or not coded	OPEN return code 8, Message IEC133I, DCBOFOPN bit is off.
NO or not coded	ANY	ABEND 113-4C, messages IEC133I and IEC142I.
YES	BELOW or not coded	OPEN return code 8, Message IEC133I, DCBOFOPN bit is off.

If OPEN gives return code 8, there were attempts to read or write the data set that resulted in an abnormal end.

### Migration considerations

Programs that can run on previous releases of z/OS should not be updated to take advantage of this new support, or errors can result. Specifically, on lower-level systems, you can see the following error:

```
IEC133I ddname, OPEN FAILED FOR EXTENDED TIOT
```

### Additional information

More information about TSO/E XTIOT, nocapture UCB, and DSAB above the 16 MB line support can be found in the following manuals:

- ▶ *z/OS MVS Initialization and Tuning Reference, SA22-7592*
- ▶ *z/OS MVS Authorized Assembler Services Guide, SA22-7608*
- ▶ *z/OS DFSMS Using the New Functions, SC26-7473*
- ▶ *z/OS DFSMS Macro Instructions for Data Sets, SC26-7408*
- ▶ *z/OS DFSMS Using Data Sets, SC26-7410*
- ▶ *z/OS DFSMSdfp Advanced Services, SC26-7400*

## 11.3 Password special character support

In this section we discuss the support for additional special characters on TSO/E logon passwords. In previous z/OS versions, the TSO/E logon panel restricted passwords of eight or fewer characters to alphanumeric or national characters (a-z, A-Z, 0-9, @, #, \$). Some security products and protocols require passwords including additional special characters.

### Password syntax check

The z/OS V1R12 TSO/E logon panel does not perform password syntax checking as in previous releases. You can now control your logon TSO/E passwords using your own security product.

The behavior of the password prompter has changed when a password with the additional special character is typed.

Messages issued previous to the changes in z/OS V1R12:

```
IKJ56464I You have entered unacceptable characters in the highlighted field(s)
IKJ56465I Press PF1 or PF13 for help
```

The new messages messages are:

```
IKJ56414I NEW-PASSWORD IS INVALID FOR RACF
IKJ56429A REENTER -
```

### **Compatibility and migration considerations**

SPE APAR OA29028 rolls this support back to z/OS V1R11.

You should be aware that:

- ▶ RACF does not support, even in z/OS V1R12, passwords with special characters other than those listed above, which are allowed by this new function of the TSO/E logon panel.
- ▶ Users running multiple versions of z/OS, or multiple security products, must enforce compatibility through their own security products.
- ▶ TSO/E line mode logon is not affected by this new function, only the logon panel.

### **Additional information**

More information about TSO/E password special character support can be found in *z/OS TSO/E Customization*, SA22-7783.





## RMF enhancements

This chapter describes the changes to the Resource Monitoring Facility (RMF) in z/OS V1R12. The enhancements are as follows:

- ▶ RMF Postprocessor reports in XML format are implemented by specifying new ddnames in the job for the Postprocessor output. The DASD and WLMGL reports can be generated in XML format.
- ▶ SMF log streams are allowed as Postprocessor input. When generating a Postprocessor job, you can now specify SMF log streams as the SMF input data type as an alternative to either SMF data sets or data from the SMF buffers.
- ▶ New statistics on running or waiting work units in the CPU Activity report. Instead of in-ready statistics on address space level, the Postprocessor CPU Activity report now provides more granular statistics based on single work units that are running or ready to run. With more than one work unit running in an address space, the new in-ready distribution of work units provides a more detailed view of the processor demand than the in-ready distribution of address spaces. Also, the number of work units is presented per processor type (CP, zAAP, and zIIP). In addition, RMF provides new overview conditions for the Postprocessor based on SMF record 70-1 for the work unit statistics and processor wait dispatch rates.
- ▶ Information about nominal and effective processor capacity. The Monitor III CPC Capacity report, the Postprocessor CPU Activity report, and the Service Policy Page of the Postprocessor Workload Activity report are extended to provide information about the nominal and effective processor capacity.
- ▶ Enhanced Enterprise Disk Systems report and new overview conditions for error and performance counters. The ESS Rank Statistics section of the Postprocessor Enterprise Disk Systems (ESS) report is enhanced to indicate whether a Solid® State Drive is defined in a rank array or not. RMF Postprocessor reports are in XML format.
- ▶ Enhanced Crypto Hardware Activity report. RMF enhances the Postprocessor Crypto Hardware Activity report to provide measurements about 4096-Bit RSA operations.

## 12.1 RMF Postprocessor reports in XML format

The XML format is generated by specifying the following Postprocessor ddnames for the output data. With z/OS V1R12, there is a new ddname, XPXSRPTS, for combined sysplex-wide reports in XML format. There is one ddname for one data set to contain all sysplex reports for each interval included in the input data.

**Note:** There is no dynamic allocation of this ddname. You must define it explicitly if you want to get all reports in XML format into one data set or output class. If you define this ddname, no MFRnnnnn files are created. If you define this ddname and PPXSRPTS, no XML output in file XPXSRPTS is created.

In addition to the Overview report, the following reports can be obtained as XML output with z/OS V1R12:

- ▶ DEVICE Activity report
- ▶ WLMGL Workload Activity report

Any other reports requested on the REPORTS and/or SYSRPTS control statement are ignored.

The RMF Spreadsheet Reporter also provides a set of reports in XML format with z/OS V1R12. You can view these XML reports in a web browser.

**Note:** When you create XML reports, you determine which type of SMF data the Postprocessor should use: If you do not select any SMF Dump Data resource, then the Postprocessor automatically extracts the requested data from the RMF Sysplex Data Server's SMF buffer.

## 12.2 RMF support for SMF log stream

Starting with z/OS V1R9, SMF introduced log streams in addition to data sets for writing SMF records, via the z/OS System Logger. Log streams can be either Coupling Facility log streams, where data is stored in a Coupling Facility structure and then offloaded to DASD, or DASD-only log streams, where data is stored in local storage buffers and then offloaded to DASD. There can be multiple log streams active on a system, but also SMF data from several systems can be grouped into one log stream.

To access the data from the SMF log streams or the Offload DASD Log Data sets, the SMF log stream dump program IFASMF DL must be used. RMF Postprocessor reports can be generated with the ISPF Postprocessor interface. You can create Postprocessor reports based on data gathered as SMF records by Monitor I, Monitor II, and Monitor III.

**Note:** Because SMF produces VSAM data sets and the Postprocessor cannot process VSAM data sets, you must copy the SMF records into non-VSAM data sets. You should do this by using the IFASMF DP program for SMF data sets or the IFASMF DL program for SMF log streams.

If you want to process SMF records from SMF log streams, you have to replace the first step in the sample job using the IFASMF DL program instead of IFASMF DP for SMF data sets.

## 12.2.1 ISPF Postprocessor interface

The ISPF Postprocessor interface consists of a series of panels that are presented in sequence and allow to set Postprocessor options in an easy way. For example, the panels help you to specify the input data source: the RMF SMF BUFFER, SMF data sets or, new with RMF for z/OS V1R12, SMF log streams. The ISPF Postprocessor interface creates and submits a Postprocessor job based on the user input.

### Using the Postprocessor option

To invoke the ISPF postprocessor interface, enter RMF on ISPF Option 6. The ISPF RMF Performance Management menu is displayed, as shown in Figure 12-1.

```
ERBOPRM          RMF - Performance Management          z/OS V1R12 RMF
Selection ==> 1

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 V1R12 RMF

                    T TUTORIAL    X EXIT

RMF Home Page:    http://www.ibm.com/servers/eserver/zseries/rmf/

                    5694-A01 Copyright IBM Corp. 1994, 2010. All Rights Reserved
                    Licensed Materials - Property of IBM
```

Figure 12-1 ISPF RMF Primary menu

### Select Option 1 Postprocessor

Selecting Option 1 in Figure 12-1 displays Figure 12-2 on page 230. On the RMF Processor Setup panel of the ISPF Postprocessor interface you can now select SMFLOG (SMF log stream) as input data.

### RMF Postprocessor Setup panel

To view the SMFLOG, replace DATASET with SMFLOG and Enter; Figure 12-3 on page 230 is displayed.

```

ERBPPUT                                RMF - Postprocessor Setup
Command ==>

Input Data      ==> SMFLOG_  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   ==> _____

Edit generated JCL ==> YES      YES or NO

Job Statement Information:
==> //CESARP   JOB (ACCOUNT),'PGMRNAME',CLASS=A,REGION=32M
==> //*
==> //*
==> //*

Complete this panel and press ENTER to continue, or END to exit.
To return to RMF Primary Menu without saving input, enter CANCEL.

```

Figure 12-2 Select SMFLOG

### RMF Postprocessor Input panel

When you have selected SMFLOG as the Input data option, the RMF Postprocessor input panel shown in Figure 12-3 is displayed. It allows you to specify up to 14 SMF log streams containing the input data to be used for your Postprocessor report.

```

ERBPIIP                                RMF - Postprocessor Input
Command ==>

SMF Log Streams  ==> 'LOGSTR.SMF.SYSA.RMFDATA' _____
                  ==> 'LOGSTR.SMF.SYSB.RMFDATA' _____
                  ==> 'LOGSTR.SMF.SYSC.RMFDATA' _____
                  ==> _____
                  ==> _____
                  ==> _____
                  ==> _____
                  ==> _____
                  ==> _____
                  ==> _____
                  ==> _____
                  ==> _____
                  ==> _____
                  ==> _____

Sort Input Data  ==> NO      YES or NO

Press ENTER to continue. To return to previous panel, press END.
To return to the Postprocessor Setup Menu, enter CANCEL.

```

Figure 12-3 Input log stream data sets

## RMF Postprocessor JCL

Figure 12-4 is an extract of the RMF Postprocessor JCL that is generated by the ISPF Postprocessor interface, based on the user input that was specified in Figure 12-3 on page 230. The job calls the IFASMF DL program to access the SMF log streams that were entered in Figure 12-3 on page 230.

The LSNAME parameter specifies the log stream.

```
//CESARP JOB (ACCOUNT),'PGMRNAME',CLASS=A,REGION=32M
//*
//*
//*****
//*          CREATED VIA ISPF INTERFACE
//*          z/OS V1R12 RMF
//*****
//*
//*****
//*          RMF DATA EXTRACTING
//*****
//RMFDUMP EXEC PGM=IFASMF DL,REGION=0M
//OUTDD1 DD DISP=(NEW,PASS),UNIT=SYSDA,SPACE=(TRK,(200,200))
//SYSPRINT DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//SYSIN DD *
LSNAME(LOGSTR.SMF.SYSA.RMFDATA)
LSNAME(LOGSTR.SMF.SYSB.RMFDATA)
LSNAME(LOGSTR.SMF.SYSC.RMFDATA)
OUTDD(OUTDD1,TYPE(70:79,103,108))
```

Figure 12-4 Postprocessor JCL generated by ISPF

## 12.3 Statistics in the CPU Activity report

Instead of in-ready statistics on the address space level, the Postprocessor CPU Activity report now provides more granular statistics based on single work units that are running or waiting to run in z/OS V1R12. With more than one work unit running in an address space, the new in-ready distribution of work units provides a more detailed view of the processor demand than the in-ready distribution of address spaces.

Also, the number of work units is presented per processor type (CP, zAAP, and zIIP). In addition, RMF provides new overview conditions for the Postprocessor based on SMF record 70-1 for the work unit statistics and processor wait dispatch rates.

### System address space and work unit analysis

With z/OS V1R12, RMF in the System Address Space and Work Unit Analysis section of the CPU Activity report provides overall address space information and the minimum, maximum, and average numbers of running or ready-to-run work units. The number of SAMPLES taken during the RMF reporting interval is displayed at the top of this section.

The work unit statistics (MIN, MAX, and AVG) are provided per processor type, that is, per standard CPs, zAAPs, and zIIPs as part of the CPU Activity report, as shown in Figure 12-5 on page 232. The distribution shown on the right is now based on the number of work units (WEBs). It is displayed if data is collected with RMF V1R12. For earlier releases the former



```

RMF V1R12 CPC Capacity
Line 1 of 12
Command ==> Scroll ==> HALF

Samples: 59      System: Z201 Date: 11/28/09 Time: 13.16.00 Range: 60 Sec
Partition: Z2      2097 Model 703
CPC Capacity: 410 Weight % of Max: 74.2 4h Avg: 41 Group: CGRP0010
Capacity %: ***** WLM Capping %: 5.4 4h Max: 185 Limit: 100*
Image Capacity: 50

Partition --- MSU --- Cap Proc Logical Util % - Physical Util % -
              Def Act Def Num Effect Total LPAR Effect Total

+CP
TZ1          130 122 NO 4.3 11.2 12.5 0.4 3.4 3.8
Z1           150 89 NO 5.2 9.3 9.6 0.1 3.5 3.6
Z2           50 58 NO 2.1 11.5 12.8 0.2 1.7 1.9
Z3           N/A 0 YES 2.4 8.8 10.6 0.3 1.5 1.8
PHYSICAL
              0.1 0.1

+ICF
CF1          1 99.9 99.9 0.0 7.1 7.1
CF2          1 0.0 0.0 0.0 0.0 0.0
PHYSICAL
              0.1 0.1

```

Figure 12-6 CPC Capacity report

### CPU Activity report

New with z/OS V1R12 are the first three rows of the report that show the CPU, MODEL, and H/W MODEL.

```

CPU ACTIVITY
z/OS V1R12      SYSTEM ID S50      DATE 11/28/2009      INTERVAL 14.50.007      PAGE 1
RPT VERSION V1R12 RMF      TIME 16.45.00      CYCLE 1.000 SECONDS

CPU 2097 CPC CAPACITY 2092 SEQUENCE CODE 0000000000708B2
MODEL 739 CAPACITY % N/A HIPERDISPATCH=YES
H/W MODEL E40 CHANGE REASON=NONE

---CPU---      TIME %
NUM TYPE      ONLINE LPAR BUSY MVS BUSY PARKED LOG PROC --I/O INTERRUPTS--
SHARE %      RATE % VIA TPI

0 CP 100.00 00.06 100.0 0.00 100.0 HIGH 05.31 0.03
1 CP 100.00 00.60 100.0 0.00 100.0 HIGH 0.00 0.00
2 CP 100.00 00.50 99.07 0.00 100.0 HIGH 0.00 0.00
3 CP 100.00 00.50 99.07 0.00 100.0 HIGH 0.00 0.00
4 CP 100.00 00.50 99.00 0.00 100.0 HIGH 0.00 0.00
5 CP 100.00 78.17 100.0 0.00 70.3 MED 0.00 0.00
6 CP 100.00 78.10 100.0 0.00 70.3 MED 0.00 0.00

```

Figure 12-7 CPU Activity report with z/OS V1R12

**Note:** Logical processors can be parked, meaning they are not dispatched by z/OS. Some logical processors may have a low amount, or 0%, of physical processor share. These discretionary logical processors are not needed to allow the partition to consume the physical processor resource associated with its weight. These logical processors may be parked. In a parked state, discretionary processors do not dispatch work; they are in a long term wait state. They are parked when they are not needed to handle the partition's workload (not enough load) or are not useful because physical capacity does not exist for PR/SM™ to dispatch (no available time from other logical partitions).

### 12.4.1 WEB queue distribution

Current RMF reporting before z/OS V1R12 for processor delay is based on address spaces. It does not consider multiple work units (WEBs) within one address space. There is a need for

performance analysts to know the processor delay on WEB granularity. With z/OS V1R12, the RMF CPU Activity reporting is enhanced as discussed previously. The enhanced CPU Activity reporting helps installations to obtain information about the in-ready distribution based on WEBs and the number of work units per processor type, as shown in Figure 12-5 on page 232.

This information about processor delays for work units on a system can now be obtained about the minimum, maximum, and average number of work units on different processor types. This support is invoked by implementing the following functions:

- ▶ Calling the CPU Activity Report via Postprocessor
- ▶ Creating an Overview Report/Record with the new Overview conditions

The following new and changed output can be obtained using the following reports and SMF:

- ▶ Postprocessor CPU Activity report
- ▶ New Overview/Exception report records
- ▶ New SMF70 subtype 1 fields

### SMF type 70 records

SMF record type 70 subtype 1 (CPU activity) is changed, as shown in Figure 12-8. The ASID Data Area Section is extended starting at offset 684. With the RMF WEB queue distribution support, the ASID data area section of SMF70, subtype 1 is extended to hold the WEB queue distribution data.

SMF record type 70 subtype 1 – ASID Data Area Section				
Offsets	Name	Len	Format	Description
684 2AC	SMF70SRM	4	Binary	Number of SRM samples
688 2B0	SMF70CMN	4	Binary	Minimum number of work units for general purpose processors over interval.
692 2B4	SMF70CMM	4	Binary	Maximum number of work units for general purpose processors over interval.
696 2B8	SMF70CTT	4	Binary	Total number of work units for general purpose processors over interval.
700 2BC	SMF70DMN	4	Binary	Minimum number of work units for AAP processors over interval.
704 2C0	SMF70DMM	4	Binary	Maximum number of work units for AAP processors over interval.
708 2C4	SMF70DTT	4	Binary	Total number of work units for AAP processors over interval.
712 2C8	SMF70EMN	4	Binary	Minimum number of work units for IIP processors over interval.
716 2CC	SMF70EMM	4	Binary	Maximum number of work units for IIP processors over interval.
720 2D0	SMF70ETT	4	Binary	Total number of work units for IIP processors over interval.

Figure 12-8 SMF subtype 1 records for the ASID data area section

Fields SMF70U00 to SMF70U015 have the count of the number of Work Units based on the number N of processors being online and unparked when the sample was taken.



SMF record type 70 subtype 1 – ASID Data Area Section				
Offsets	Name	Len	Format	Description
Fields SMF70U00 to SMF70U15 count the number of Work Units based on the number N of processors being online and unparked when the sample was taken				
724 2D4	SMF70U00	4	Binary	Count of times the number of Work Units was less or equal N.
728 2D8	SMF70U01	4	Binary	Count of times the number of Work Units was N+1.
732 2DC	SMF70U02	4	Binary	Count of times the number of Work Units was N+2
736 2E0	SMF70U03	4	Binary	Count of times the number of Work Units was N+3.
740 2E4	SMF70U04	4	Binary	Count of times the number of Work Units was N+4 or N+5.
744 2E8	SMF70U05	4	Binary	Count of times the number of Work Units was N+6 to N+10.
748 2EC	SMF70U06	4	Binary	Count of times the number of Work Units was N+11 to N+15.
752 2F0	SMF70U07	4	Binary	Count of times the number of Work Units was N+16 to N+20.
756 2F4	SMF70U08	4	Binary	Count of times the number of Work Units was N+21 to N+30.
760 2F8	SMF70U09	4	Binary	Count of times the number of Work Units was N+31 to N+40.
764 2FC	SMF70U10	4	Binary	Count of times the number of Work Units was N+41 to N+60.
768 300	SMF70U11	4	Binary	Count of times the number of Work Units was N+61 to N+80.
772 304	SMF70U12	4	Binary	Count of times the number of Work Units was N+81 to N+100.
776 308	SMF70U13	4	Binary	Count of times the number of Work Units was N+101 to N+120.
780 30C	SMF70U14	4	Binary	Count of times the number of Work Units was N+121 to N+150.
784 310	SMF70U15	4	Binary	Count of times the number of Work Units was greater N+150.

Figure 12-9 SMF subtype 1 records for ASID data area section

## WEB queue distribution new Overview conditions

The WEB queue distribution support introduced new Overview conditions that can be used to create Overview/Exception reports or Overview records, as shown in Figure 12-10, Figure 12-11 on page 236, Figure 12-12 on page 236, and Figure 12-13 on page 237.

New Overview Conditions based on SMF record 70-1				
Condition	Condition Name	Qualifier	Source	Algorithm
Maximum number of work units for general purpose processors	MXWUCP	None	SMF70CMM	Value or comparison
Maximum number of work units for zAAPs	MXWUAAP	None	SMF70DMM	Value or comparison
Maximum number of work units for zIIPs	MXWUIIP	None	SMF70EMM	Value or comparison
Average number of work units for general purpose processors	AVGWUCP	None	SMF70CTT SMF70SRM	CTT / SRM
Average number of work units for zAAPs	AVGWUAAP	None	SMF70DTT SMF70SRM	DTT / SRM
Average number of work units for zIIPs	AVGWUIIP	None	SMF70ETT SMF70SRM	ETT / SRM

Figure 12-10 SMF type 70 records for Overview conditions

New Overview Conditions based on SMF record 70-1				
Condition	Condition Name	Qualifier	Source	Algorithm
Percentage of the reporting interval during which at least one work unit could not be dispatched	WCPU1	None	SMF70SRM	$(U01+U02+\dots+U15) / \text{SRM} \times 100$
Percentage of the reporting interval during which at least two work units could not be dispatched	WCPU2	None	SMF70SRM	$(U02+U03+\dots+U15) / \text{SRM} \times 100$
Percentage of the reporting interval during which at least three work units could not be dispatched	WCPU3	None	SMF70SRM	$(U03+U04+\dots+U15) / \text{SRM} \times 100$
Percentage of the reporting interval during which at least four work units could not be dispatched	WCPU4	None	SMF70SRM	$(U04+U05+\dots+U15) / \text{SRM} \times 100$
Percentage of the reporting interval during which more than five work units could not be dispatched	WCPU5	None	SMF70SRM	$(U05+U06+\dots+U15) / \text{SRM} \times 100$

Figure 12-11 SMF type 70 records for Overview conditions

New Overview Conditions based on SMF record 70-1				
Condition	Condition Name	Qualifier	Source	Algorithm
Percentage of the reporting interval during which more than ten work units could not be dispatched	WCPU10	None	SMF70SRM	$(U06+U07+\dots+U15) / \text{SRM} \times 100$
Percentage of the reporting interval during which more than 15 work units could not be dispatched	WCPU15	None	SMF70SRM	$(U07+U08+\dots+U15) / \text{SRM} \times 100$
Percentage of the reporting interval during which more than 20 work units could not be dispatched	WCPU29	None	SMF70SRM	$(U08+U09+\dots+U15) / \text{SRM} \times 100$
Percentage of the reporting interval during which more than 30 work units could not be dispatched	WCPU30	None	SMF70SRM	$(U09+U10+\dots+U15) / \text{SRM} \times 100$
Percentage of the reporting interval during which more than 40 work units could not be dispatched	WCPU40	None	SMF70SRM	$(U10+U11+\dots+U15) / \text{SRM} \times 100$

Figure 12-12 SMF type 70-1 records for Overview conditions

New Overview Conditions based on SMF record 70-1				
Condition	Condition Name	Qualifier	Source	Algorithm
Percentage of the reporting interval during which more than 60 work units could not be dispatched	WCPU60	None	SMF70SRM	$(U11+U12+\dots+U15) / SRM \times 100$
Percentage of the reporting interval during which more than 80 work units could not be dispatched	WCPU80	None	SMF70SRM	$(U12+U13+U14+U15) / SRM \times 100$
Percentage of the reporting interval during which more than 100 work units could not be dispatched	WCPU100	None	SMF70SRM	$(U13+U14+U15) / SRM \times 100$
Percentage of the reporting interval during which more than 120 work units could not be dispatched	WCPU120	None	SMF70SRM	$(U14+U15) / SRM \times 100$
Percentage of the reporting interval during which more than 150 work units could not be dispatched	WCPU150	None	SMF70SRM	$(U15) / SRM \times 100$

Figure 12-13 SMF type 70-1 records for Overview conditions

## 12.4.2 Mean time to wait (MTTW)

With z/OS V1R12, changes have been introduced to allow performance analysts to analyze LPAR overhead. This implementation provides new fields in the SMF70 subtype 1 CPU data section SMF70WTD. There is a new Overview condition to report, called the “mean time to wait.” This new Overview condition processing should help you to obtain information about the wait dispatch rate to analyze LPAR overhead.

A processor is dispatched several times during a certain time interval. This time is added up and divided by the number of times the processor is set into the wait state, as shown in Figure 12-14. The result is the average time during which the processor is dispatched and active - the “mean time to wait.”

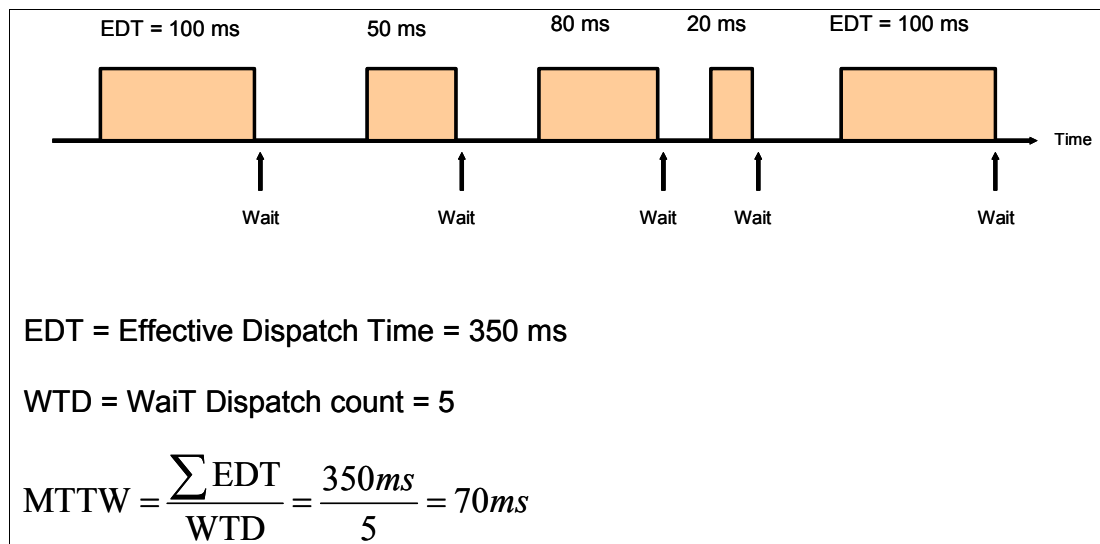


Figure 12-14 MTTW calculation

## SMF record type 70 subtype 1

SMF record type 70 subtype 1 (CPU activity) is changed in z/OS V1R12. The CPU Data Section is extended starting at offset 72, as shown in Figure 12-15. SMF70WTD contains the number of times a logical processor entered a wait (meaning the number of times a processor found no work to dispatch). A WTD of zero means there were no wait state dispatches in the interval. This would be the expected value for a parked logical processor (LCP).

Offsets	Name	Length	Format	Description
72 -48	SMF70WTD	8	Binary	Wait dispatch count for this CPU.

Figure 12-15 SMF record type 70 subtype 1 – CPU Data Section

### Overview conditions

New overview conditions have been implemented that can be used to create Overview/Exception reports or Overview records. SMF70WTD contains the number of times a logical processor entered a wait (meaning the number of times a processor found no work to dispatch). “Mean time to wait” is the *effective dispatch time* of an LPAR divided by the interval dispatch count.

The effective dispatch time is taken from the SMF70 record field SMF70EDT of a logical processor for the specified partition (logical processor effective dispatch time, in microseconds). The number of microseconds that were accumulated during the measurement interval (excluding LPAR management time), during which, a physical processor was assigned to this logical processor.

### New Overview conditions based on SMF record 70-1

The following new Overview condition fields in SMF type record 70-1 are:

- MTTWCP** Mean Time to Wait for CP processors in microseconds. The source fields are SMF70WTD and SMF70EDT. The algorithm is to divide LPAR effective dispatch time by the interval wait dispatch count.
- MTTWAAP** Mean Time to Wait for zAAP processors in microseconds. The source fields are SMF70WTD and SMF70EDT. The algorithm is to divide LPAR effective dispatch time by the interval wait dispatch count.
- MTTWIIP** Mean Time to Wait for zIIP processors in microseconds. The source fields are SMF70WTD and SMF70EDT. The algorithm is to divide LPAR effective dispatch time by the interval wait dispatch count.

**Note:** No MTTW for a specified processor is displayed in the Overview Report, if no processor data section exists because of the following conditions:

- ▶ The processor is parked for the entire interval.
- ▶ The processor is offline at the end of the interval and was not reconfigured during the interval.
- ▶ If no processor ID is specified, the average MTTW for all processors of the specified type is calculated.

The following example can be used to generate a Postprocessor Overview report for MTTW:

```
OVERVIEW(REPORT)
OVW(MTTWCPO(MTTWCP(0)))
OVW(MTTWCP1(MTTWCP(1)))
OVW(MTTWCP2(MTTWCP(2)))
```

## 12.5 Enhanced Enterprise Disk Systems report

The ESS Rank Statistics section of the Postprocessor Enterprise Disk Systems (ESS) report, shown in Figure 12-16, is enhanced to indicate whether a Solid State Drive is defined in a rank array or not, and is available for the IBM TotalStorage® DS family. In addition, RMF provides new overview conditions for the Postprocessor based on SMF records 74-5 and 74-8 to generate reports about link errors and performance counters.

**Note:** In the ARRAY section in Figure 12-16, if in the SSD column a Y is displayed, then there is at least one solid state drive in the rank array.

ESS RANK STATISTICS														PAGE	3
z/OS V1R12		SYSTEM ID VSL1				DATE 11/28/2000		INTERVAL 15.00.000							
		RPT VERSION V1R12 RMF				TIME 00.30.00		CYCLE 1.000 SECONDS							
SERIAL NUMBER 0000022300		TYPE-MODEL		2107-021		CDATE 00/14/2000		CTIME 00.30.00		CINT 15.00					
--EXTENT POOL--		----- READ OPERATIONS -----				----- WRITE OPERATIONS -----				---ARRAY---		MIN	RANK	RAID	
ID	TYPE	RRID	OPS /SEC	BYTES /OP	BYTES /SEC	RTIME /OP	OPS /SEC	BYTES /OP	BYTES /SEC	RTIME /OP	SSD NUM	WIDTH	RPM	CAP	TYPE
0004	FIBRE 1Gb	0004	164.3	181.3K	16.6M	15.0	90.0	102.1K	10.2M	14.3	Y	1	6	0	876G RAID 5
		000A	186.0	98.2K	18.4M	17.4	165.0	91.2K	15.1M	18.0	Y	1	6	0	876G RAID 5
		000C	175.0	96.2K	16.8M	16.4	163.2	77.3K	12.6M	15.0	Y	1	6	0	876G RAID 5
		POOL	526.2	98.6K	51.0M	16.6	420.0	90.2K	37.0M	16.0	Y	3	18	0	2628G RAID 5
000C	CND 1Gb	000E	143.6	110.3K	15.8M	16.0	145.2	110.5K	17.4M	17.5		1	6	0	876G RAID 10
		001A	99.2	120.8K	12.0M	15.8	123.1	85.4K	10.5M	15.3		1	7	0	1622G RAID 10
		001E	173.2	113.3K	10.6M	18.1	142.7	110.6K	15.8M	17.2		1	7	0	1622G RAID 5
		POOL	416.0	114.8K	47.8M	16.0	411.0	105.2K	43.2M	16.7		3	20	0	2028G MIXED

Figure 12-16 ESS Rank Statistics section of Postprocessor ESS report

## 12.6 Enhanced Crypto Hardware Activity report

The Crypto Hardware Activity report provides information about the activities in the various cryptographic hardware functions. Most of these functions can only be used through Cryptographic Support for z/OS (ICSF). ICSF is a standard component of z/OS. It provides cryptographic services in the z/OS environment.

RMF enhances the Postprocessor Crypto Hardware Activity report to provide measurements about 4096-Bit RSA operations. In addition, RMF provides new Overview conditions for the Postprocessor based on SMF record 70-2.

## 12.7 RMF SMF logstream support

Starting with z/OS V1R9, SMF introduced logstreams in addition to data sets for writing SMF records, via the z/OS System Logger. Logstreams can be either Coupling Facility log streams, where data is stored in a Coupling Facility structure and then offloaded to DASD, or DASD-only log streams, where data is stored in local storage buffers and then offloaded to DASD.

There can be multiple log streams active on a system. To access the data from the SMF logstreams or the offload DASD log data sets, the SMF logstream dump program IFASMF DL must be used.

The ISPF Postprocessor interface helps you to generate and submit a Postprocessor JCL to create RMF Postprocessor reports. The ISPF Postprocessor interface consists of a series of panels that are presented in sequence and allow to set Postprocessor options in an easy way. SMF can exploit the z/OS System Logger to write SMF logstreams instead of SMF dump data sets. Today the ISPF interface for the RMF Postprocessor supports only SMF dump data sets or the RMF SMF BUFFER as SMF data input source, but not SMF logstreams.

### z/OS V1R12 implementation

z/OS V1R12 is enhancing the ISPF Postprocessor interface to support the use of SMF logstreams as SMF data input. This support provides you the ability to now use the ISPF Postprocessor interface to create and submit RMF Postprocessor JCL to create RMF Postprocessor reports, which processes SMF logstreams as SMF data input. You can create Postprocessor reports based on data gathered as SMF records by Monitor I, Monitor II, and Monitor III.

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 following information:

- ▶ Input data source, such as the RMF SMF BUFFER
- ▶ SMF data sets
- ▶ SMF logstreams, new with RMF for z/OS V1R12

The ISPF Postprocessor interface creates and submits a Postprocessor job based on the user input.

**Note:** See “System Logger enhancements with z/OS V1R11” on page 88.



# Language Environment

Language Environment establishes a common run-time environment for all participating HLLs. It combines essential run-time services, such as routines for run-time message handling, condition handling, and storage management. All of these services are available through a set of interfaces that are consistent across programming languages. You may either call these interfaces yourself, or use language-specific services that call the interfaces. With Language Environment, you can use one run-time environment for your applications, regardless of the application's programming language or system resource needs.

This chapter describes the following enhancements to Language Environment in z/OS V1R12:

- ▶ C++ TR1 support
- ▶ Performance items
- ▶ Language Environment locale support including GWP 2008
- ▶ LE nonoverrideable PARMLIB capability
- ▶ Language environment realloc() optimization
- ▶ VSAM EA support for KSDS AIX for C/C++ in Language Environment
- ▶ Resolve year 2038 problem
- ▶ BSAM 64K tracks for LE record I/O
- ▶ DFSMS BAM XTLOT dependency on Language Environment

## 13.1 C++ TR1 support

In prior releases of z/OS C++ users were unable to utilize the ISO/IEC 9899:1999 Standard C (C99) namespace. Starting with z/OS V1R12, clients are now able to use the C Run-time Library to support Chapter 8 of the ISO/IEC TR 19768, C++ Library Extensions Technical Report (TR1).

That means clients can expect C++ function overloads to be provided so that C++ coders do not need to be concerned with the exact C signature of the C99 function. This behavior avoids C++ compiler errors.

This support is either invoked by defining the feature test macro `_TR1_C99` in the source file or `__IBMCPP_TR1__` as compiler option on the compile step.

### Feature test macros

Many of the symbols that are defined in headers are “protected” by a feature test macro. These “protected” symbols are invisible to the application unless you define the feature test macro with `#define`, using either of the following methods:

- ▶ In the source code before including any header files
- ▶ On the compilation command

### `_TR1_C99` test macro

This feature test macro is new with LE in z/OS V1R12. It exposes the C++ TR1 C99 name space as described in Chapter 8 of ISO/IEC DTR 19768: Draft Technical Report on C++ Library Extensions. When both the `_TR1_C99` and `_AIX_COMPATIBILITY` feature test macros are defined, the `_AIX_COMPATIBILITY` takes precedence. This affects `copysign()`, `scalbn()`, and the floating point classification functions. When both the `_TR1_C99` and `_FP_MODE_VARIABLE` feature test macros are defined, float overloads are not supported for the following functions: `atan2()`, `copysign()`, `fdim()`, `fma()`, `fmax()`, `fmin()`, `fmod()`, `frexp()`, `hypot()`, `ldexp()`, `modf()`, `nextafter()`, `nexttoward()`, `pow()`, `remainder()`, `remquo()`, `scalbn()`, and `scalbn()`. Also, when both the `_TR1_C99` and `_FP_MODE_VARIABLE` feature test macros are set, the long double overloads are not supported for `frexp()` and `ldexp()`.

**Note:** This feature test macro requires the use of the z/OS V1R10 z/OS XL C++ compiler or later. See *z/OS XL C/C++ Run-Time Library Reference*, SA22-7821 for additional information.

### Coding examples

Figure 13-1 shows an example of using the new functionality from the command line.

```
cxx -o testit -D__IBMCPP_TR1__ a.C
```

Figure 13-1 sample compiler call

In Figure 13-2 on page 243 you see an example of using this functionality inside the source code.



```
#define _TR1_C99
#include <inttypes.h>
intmax_t abs(intmax_t n);

#define _TR1_C99
#include <stdlib.h>
long long abs(long long n);

#define _TR1_C99
#include <math.h>
float acosh(float x);
long double acosh(long double x);
```

Figure 13-2 sample use in the source

### Overload example

The regular C signature for the `sin()` function is:

```
double sin(double);
```

The C++ code shown in Figure 13-3 will not compile unless `_TR1_C99` is defined.

```
#include <math.h>
int main()
{
    int x = 1;
    sin(x);
    return 0;
}
```

Figure 13-3 Sample C++ code

Without these overloads, you would get a compiler error if you tried to pass an “int” to the `sin()` function without first transforming it to a double. In Figure 13-4 on page 244 you can see the appropriate error messages when trying to compile without `_TR1_C99` being defined.

```

../TR1/> cxx -o test a.C
"./a.C", line 7.3: CCN5219 (S) The call to "sin" has no best match.
"./a.C", line 7.7: CCN6228 (I) Argument number 1 is an lvalue of type "int".
"./include/math.h", line 1337.28: CCN6202 (I) No candidate is better than "sin(long double)".
"./a.C", line 7.7: CCN6231 (I) The conversion from argument number 1 to "long double" uses "an lvalue-to-rvalue transformation" followed by "a floating point-integral conversion".
"./include/math.h", line 1336.23: CCN6202 (I) No candidate is better than "sin(double)".
"./a.C", line 7.7: CCN6231 (I) The conversion from argument number 1 to "double" uses "an lvalue-to-rvalue transformation" followed by "a floating point-integral conversion".
"./include/math.h", line 1335.22: CCN6202 (I) No candidate is better than "sin(float)".
"./a.C", line 7.7: CCN6231 (I) The conversion from argument number 1 to "float" uses "an lvalue-to-rvalue transformation" followed by "a floating point-integral conversion".
CCN0793(I) Compilation failed for file ./a.C. Object file not created.
FSUM3065 The COMPILE step ended with return code 12.
FSUM3017 Could not compile a.C. Correct the errors and try again.

```

Figure 13-4 Error messages during a compile

## 13.2 BSAM 64 K track support

In prior releases of z/OS sequential data sets could only use a maximum of 59 volumes. There is a limit on the number of tracks a volume can contain. Once the limit has been reached, there is no room for the data set to grow.

### Large-format sequential data sets

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. Beginning with z/OS V1R12, C/C++ applications now are able to exploit the functionality provided by DFSMSdfp. DFSMSdfp is able to allow XL C/C++ applications to deal with large-format sequential data sets greater than 65,535 tracks/volume opened for BSAM (seek) under record I/O.

Seek processing is obtained by omitting the `noseek` keyword on a `fopen()` or `freopen()` call. The support is invoked by using `fopen()` with “`type=record`” included in the mode string to open an existing large-format sequential data set.

**Note:** Allocation of a new large-format sequential data set can be accomplished by specifying the keyword `DSNTYPE=LARGE` on a JCL DD statement or using the dynamic allocation equivalent.

There is new support for the `fopen()` or the `freopen()` equivalent of the `DSNTYPE=LARGE` parameter. Once opened, other I/O functions can be used to process the stream.

## 13.3 Language Environment locale support

Beginning on January 1 2009, Slovakia adopted the Euro as its currency. The C Runtime-Library currently supports both `@euro` and `@preeuro` versions of the following Slovakian locales:

- ▶ `sk_SK.UTF-8@euro`

- ▶ sk\_SK.UTF-8@preeuro
- ▶ Sk\_SK.IBM-1153@euro
- ▶ Sk\_SK.IBM-1153@preeuro
- ▶ Sk\_SK.IBM-1165@euro
- ▶ Sk\_SK.IBM-1165@preeuro

However, in each case the default locale currently points to the @preeuro version of the locale, as follows:

- ▶ sk\_SK.UTF-8
- ▶ Sk\_SK.IBM-1153
- ▶ Sk\_SK.IBM-1165

You can use `setlocale()` to specify a locale `setlocale(LC_ALL,"sk_SK.UTF-8")` will now be a euro locale. `Setlocale(LC_ALL,"sk_SK.UTF-8@preeuro)` will allow the use of the previous default non-euro locale.

### Removal of conversion table source

In z/OSV1R9, the C/C++ Runtime Libraries `iconv()` family of functions started using Unicode Services to perform most of their character conversion services. At that time it was decided to continue to ship conversion table source for *all* `ucmap` and `genxlt` conversion tables through the z/OSV1R11 release. Beginning in z/OS V1R12, the C/C++ Runtime Library will no longer ship any `ucmap` source or `genxlt` source for character conversions now being performed by Unicode Services.

## 13.4 LE nonoverridable parmlib capability

For several years now, a method has been in place for controlling run-time option defaults at the system level using the `CEEPRMxx` parmlib member and the associated operator commands. This method can simplify customization of run-time options at your site. This is the recommended method for customizing run-time option defaults. Prior to z/OS V1R12, run-time option defaults that were required to be set as nonoverridable, could only be set as such using assembler language `CSECTs` and building `usermods`. LE has been recommending to its clients to use the parmlib support for setting run-time option defaults. Unfortunately, the parmlib support did not allow run-time options to be specified as nonoverridable.

In z/OS V1R12, parmlib support and the associated operator commands were extended to allow run-time options to be specified as nonoverridable (`NONOVR`). This support allows clients to completely move customization of run-time option defaults to the parmlib method. This eliminates the need for using the assembler language `CSECTs` and supports the LE strategic direction to remove the assembler language `CSECTs`. This support is exploited through the use of override attributes (`OVR` or `NONOVR`) specified for each run-time option. The override attribute can be specified in a `CEEPRMxx` parmlib member or on a `SETCEE` operator command. It is not required to use these attributes, so the existing syntax will continue to work.

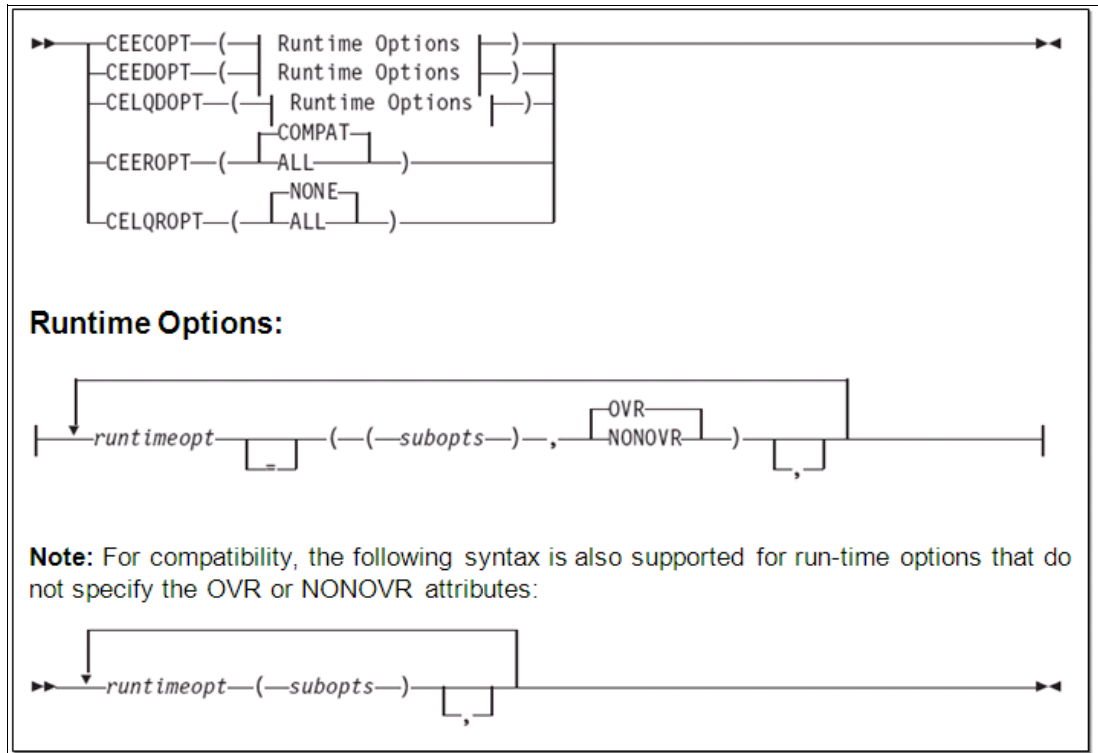


Figure 13-5 New syntax for the CEEPRMxx parmlib member

Figure 13-5 shows the full syntax for a CEEPRMxx parmlib member. There are some aspects of this syntax that are worth discussing:

- ▶ The current syntax without override attributes will continue to work.
- ▶ Any option that uses an override attribute must have two opening parenthesis before the the suboptions.
- ▶ The equal sign between the option and the two opening parentheses is not required, but it maps to the same syntax as the assembler language CSECTs.

In Figure 13-6 on page 247, you can see a partial example of CEEDOPT that ships in the CEEPRM00 parmlib member sample in the CEE.SCEESAMP data set. The CEEPRM00 parmlib member sample has been changed so that all run-time options use the new preferred syntax with the override attribute specified. CEEPRM00 ships with all options specified as overridable (OVR), just as the assembler language modules are shipped.

```

CEEDOPT(
    ABPERC=( (NONE),OVR),
    ABTERMENC=( (ABEND),OVR),
    AIXBLD=( (OFF),OVR),
    ALL31=( (ON),OVR),
    ANYHEAP=( (16K,8K,ANYWHERE,FREE),OVR),
    BELOWHEAP=( (8K,4K,FREE),OVR),
    CBLOPTS=( (ON),OVR),
    CBLPSHPOP=( (ON),OVR),
    CBLQDA=( (OFF),OVR),
    CEEDUMP=( (60,SYSOUT=*,FREE=END,SPIN=UNALLOC),OVR),
    CHECK=( (ON),OVR),
    .
    .
    .
    THREADHEAP=( (4K,4K,ANYWHERE,KEEP),OVR),
    THREADSTACK=( (OFF,4K,4K,ANYWHERE,KEEP,128K,128K),OVR),
    TRACE=( (OFF,4K,DUMP,LE=0),OVR),
    TRAP=( (ON,SPIE),OVR),
    UPSI=( (00000000),OVR),
    VCTRSAVE=( (OFF),OVR),
    XUFLOW=( (AUTO),OVR)

```

Figure 13-6 sample partial CEEPRMxx parmlib member

The equal sign in the syntax is not required, but is the preferred method for specifying run-time options with an override attribute. This syntax closely matches the syntax used in the assembler language modules. The old parmlib syntax RUNTIMEOPT(SUBOPTS) is still supported if the option does not specify an override attribute. All formats of the parmlib syntax can be specified within the same parmlib member.

**Restriction:** The OVR and NONOVR attributes are not supported for run-time options that use the runtimeopt | NOruntimeopt format without suboptions to turn the option on or off; for example, DEBUGINODEBUG. To use the override attribute with this type of option, specify it in the corresponding runtimeopt=((ON|OFF),OVR|NONOVR) format. For example, to turn off the DEBUG option, specify DEBUG=((OFF),OVR). Specifying one of these options with an OVR or NONOVR attribute without a suboption is not allowed. For example, specifying AIXBLD=((),OVR) results in an error and the option is ignored. The full list of options that are part of this restriction are as follows:

- ▶ AIXBLD
- ▶ DEBUG
- ▶ FILEHIST
- ▶ INQPCOPN
- ▶ OCSTATUS
- ▶ PC
- ▶ RTEREUS
- ▶ SIMVRD

### 13.4.1 Changing parmlib settings

If you specify the override attributes in a parmlib member, a CEE=xx parameter on the system IPL command, a CEE=xx parameter in the IEASYSy parmlib member, or a SET CEE=xx

operator command must be specified to pick up the changes. You can dynamically use this support through the use of a SETCEE operator command.

Figure 13-9 on page 249 shows the full syntax for the SETCEE operator command. This syntax is similar to the CEEPRMxx parmlib member syntax, and the following also applies:

- ▶ The current syntax without override attributes will continue to work.
- ▶ Any option that uses an override attribute must have two opening parentheses before the the suboptions.
- ▶ The equal sign between the option and the two opening parentheses is not required, but it maps to the same syntax as the assembler language CSECTs.

Figure 13-7 shows an example of the new syntax for the SETCEE command. In this case, it shows that setting the ALL31 option for CEEDOPT as nonoverridable.

```
SETCEE CEEDOPT,ALL31=((ON),NONOVR)
CEE3743I THE SETCEE COMMAND HAS COMPLETED.
```

Figure 13-7 SETCEE command example

Figure 13-8 shows the appropriate output for a SETCEE command in Figure 13-7. You can see the new option, nonoverridable, is displayed in addition to the CEE value.

```
D CEE,CEEDOPT
CEE3745I 09.21.45 DISPLAY CEEDOPT
NO MEMBERS SPECIFIED 572
LAST WHERE SET          OPTION
-----
--
SETCEE Non-overrideable    ALL31(ON)
```

Figure 13-8 D CEE command sample output

The SETCEE command syntax is shown in Figure 13-9 on page 249.

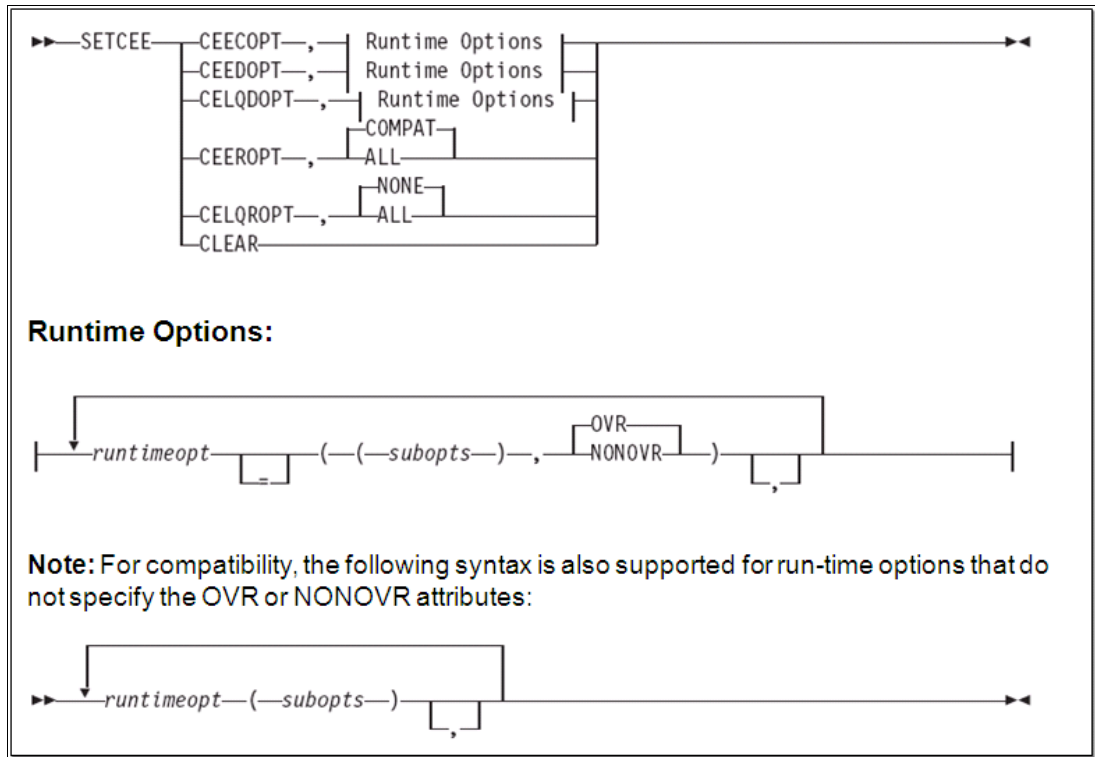


Figure 13-9 SETCEE command syntax

### New CEE messages

In order to reflect the changes, several new CEE messages were implemented. Figure 13-10 shows all the new messages regarding Language Environment.

Message ID	Message text
CEE3765I	The delimiter <i>delimiter</i> is not allowed from the current source.
CEE3766I	A required <i>delimiter</i> delimiter was not found.
CEE3767I	The required OVR or NONOVR attribute was not found.
CEE3768I	The system default for the run-time option <i>option</i> could not be overridden.
CEE3769I	The run-time option ENVAR cannot be marked as nonoverrideable.
CEE3770I	A required sub-option was not specified

Figure 13-10 New CEE messages

**Important:** In a future release, IBM plans to remove the capability to change the default Language Environment run-time option settings via SMP/E installable usermods. IBM recommends using the CEEPRMxx parmlib member to change the default Language Environment run-time options for the system.

## 13.5 realloc() optimization

As a string manipulation intensive application you need improved performance when using `realloc()` to grow the storage for the strings so that your application is not slowed down copying page after page of memory, and you need a mechanism for controlling that optimization.

In z/OS V1R12, IBM provides the new environment variable `_CEE_REALLOC_CONTROL` that accepts two parameters:

- ▶ The first parameter sets a lower boundary for particular storage sizes for which a `realloc()` or `CEECZST` call may be requested.
- ▶ The second provides a percentage to increase the requested storage size when the lower boundary is exceeded.

In this way, subsequent `realloc()` attempts may fit within the bounds of the previously requested storage and not require a new storage allocation and copy to occur. The syntax of the environment variable is:

```
_CEE_REALLOC_CONTROL=bound, percentage
```

If the variable is in effect, storage allocation is handled in the following way. For instance, if an application issues a `malloc()` to request storage and a subsequent `realloc()` to change the size of that storage allocation, this first parameter will determine whether the request will be increased with the intent that subsequent reallocations will not require that additional storage be obtained and data copied.

**Note:** The default is indicated by a percentage of 0, which will mean the feature is not in use.

### Example scenario

Assume in the following example that the variable is set to:

```
_CEE_REALLOC_CONTROL=100,20.
```

If a `realloc()` request is for greater than 100 bytes, increase that request by 20%, as follows:

1. A `malloc()` request for 100 bytes.
2. A `realloc()` request for 110 bytes. Greater than lower bounds, so increase request by 20%. The calculated amount of storage is now  $110 + 110 \cdot 0.20 = 132$  bytes. Buffer is copied to a new location.
3. A `realloc()` request for 120 bytes. Fits in currently allocated buffer, no copy needed.
4. A `realloc()` request for 130 bytes. Continues to fit in the currently allocated buffer, no copy needed.

With the support turned off in the above example, it would result in four separate allocations being performed and three copies of data to the new allocation. With the support turned on as above, two allocations will occur and one copy of data would result.

## 13.6 VSAM EA support for KSDS AIX for C/C++

Older releases of z/OS DFSMS have long supported VSAM data sets that can exceed 4 GB in size. The z/OS XL C/C++ Run-time Library has supported these data sets since z/OS V1R8, but it did not support KSDS alternate indexes. In z/OS V1R12, the z/OS XL C/C++



Run-time Library is updated to support processing of VSAM KSDS extended addressable (EA) data sets using alternate indexes. z/OS XL C/C++ applications can now read/write/position beyond the 4 GB boundary in VSAM KSDS data sets using alternate index keys.

Use `fopen()` or `freopen()` against an existing VSAM KSDS extended addressable data set using an alternate index pathname. Once opened, use other FILE functions such as `flocate()` to process the stream as you did before for non-KSDS data sets.

## 13.7 Resolve year 2038 problem

A `time_t` in C/C++ is a signed long (4 bytes) representing seconds since 00:00:00 UTC on January 1, 1970. The value will no longer be valid and will affect a series of C/C++ functions after 03:14:07 UTC on January 19, 2038.

**Note:** a `time_t` in the Language Environment AMODE 64 C/C++ run-time is also a signed long; however, it is 8 bytes long and is not affected by this problem or new support.

z/OS V1R12 Language Environment C/C++ run-time supports new typedefs, structures, and 31-bit functions that will support constructed calendar times up to and including the artificial limit of UTC 23:59:59 on December 31, 9999. You can now use new typedefs and functions to construct and manipulate calendar times past 2038.

The following functions are implemented by IBM to solve the year 2038 problem:

- ▶ `time64_t mktime64 (struct tm *timeptr)`
- ▶ `struct tm *localtime64 (const time64_t *timer)`
- ▶ `struct tm *localtime64_r(const time64_t *, struct tm *)`
- ▶ `char *asctime64(const struct tm *timeptr)`
- ▶ `char *asctime64_r(const struct tm *, char *)`
- ▶ `struct tm *gmtime64(const time64_t)`
- ▶ `struct tm *gmtime64_r(const time64_t *, struct tm *)`
- ▶ `char ctime64(const time64_t *)`
- ▶ `char ctime64_r(const time64_t *, char *)`
- ▶ `double difftime64(time64_t time2, time64_t time1)`
- ▶ `time64_t time64 ( time64_t * timer )` (see speaker note)
- ▶ `int gettimeofday64(struct timeval64 *, void *)`;
- ▶ `struct tm *getdate64(const char *)`;

These functions work similarly to the existing functions of the same names (without the 64) but take `time64_t` values rather than `time_t` values.

If you have applications that make use of `time_t` and functions to manipulate `time_t`, you have to consider updating them to use `time64_t` and the corresponding new functions.





## BCP allocation improvements

Device allocation is the assignment of input/output devices and volumes to job steps. Requests for device allocation come from data definition (DD) statements and dynamic device allocation requests.

There are two basic types of allocation: job step allocation and dynamic allocation. The two types allocate resources at different points in program processing. Job step allocation assigns resources to your program before your program runs, and dynamic allocation assigns resources to your program while it is running. The needs of your program determine which type of allocation you should use.

A temporary data set is a data set that you create and delete within a job. When you use DSNNAME for a temporary data set, code the name as two ampersands (&&) followed by a character string 1 to 8 characters in length.

In this chapter, the improvements in the allocation function of the BCP are discussed. The major improvements are:

- ▶ Memory-based data set ENQ management for dynamically allocated data sets (MEMDSENQMGMT)
- ▶ SMF type 30 suppress DD-based accounting
- ▶ Duplicate temporary data set name support

## 14.1 Memory-based data set ENQ management

The allocation function of z/OS is an evolution of an older function, designed when memory was very scarce and should be saved at the expense of processor usage. The data areas structure was kept as condensed as possible. The evolution of hardware, with 31-bit and 64-bit memory addressing schemas, has brought plenty of memory and made it more convenient to use instead of processor cycles. Memory use implied paging, which wasted a lot of processor and I/O resources.

On the other hand, workloads have changed with time to become more transactional and oriented to DBMS and OLTP. So, larger workloads in fewer system images, and thousands more data sets are allocated, when compared to batch workloads, which are limited to 255 data set definition statements (DDs). These changes in workloads have created control structures with thousands of elements and increasing complexity, thus motivating an evolution to a more efficient schema of data set allocation serialization control.

z/OS V1R12 introduces memory-based enqueue management for dynamically allocated data sets.

### 14.1.1 New MEMDSENQMGMT keyword

The memory-based data set ENQ management keyword, MEMDSENQMGMT, allows jobs and subsystems to use that management for dynamically allocated data sets. This parameter moves data set ENQ information created by allocation out of SWA and into efficient memory structures. This new feature is for jobs that allocate a large number of data sets, such as DB2.

**Note:** There is an option to suppress DD accounting in the SMF Type 30 records by a program request when using the S99DASUP flag on a DynAlloc request. You can suppress creation of SMF Type 30 EXCP section data on a per-DD basis.

#### ALLOCxx parmlib member

The new keyword in the ALLOCxx parmlib member is MEMDSENQMGMT, which specifies whether the MEMDSENQMGMT feature is available for exploitation by jobs and subsystems as follows:

MEMDSENQMGMT (ENABLE | DISABLE)

Where:

**ENABLE** Allows jobs and subsystems to use memory-based data set ENQ management for dynamically allocated data sets. Memory-based data set ENQ management is faster than the other option, SWA-based data set ENQ management, for jobs that allocate a large number of data sets. In addition to the above parmlib setting, a job or subsystem that is to use the new management system must enable the feature using the IEFDDSRV service, for example, IEFDDSRV MODIFY TYPE=FEATURE,DSENQMGMT=MEMORY.

**Note:** Specifying this option causes the job to be non-restartable through the check-point/restart function from this point on.

**DISABLE** Disables jobs and subsystems from using memory-based data set ENQ management for dynamically allocated data sets.

**Default:** DISABLE

## MEMDSENQMGMT activation

To activate memory-based data set ENQ management, you must set ENABLE in the ALLOCxx parmlib member, as shown in Figure 14-1. Once enabled, the function provides memory-based data set ENQ for the jobs or subsystems that do request it through the IEFDDSRV assembler macro instruction, as shown in Figure 14-2. This service requests that data set ENQs be managed in memory.

```
SYSTEM IEFBR14_DELMIGDS(LEGACY|NORECALL)
      TAPELIB_PREF(EQUAL|BYDEVICES)
      REMIND_INTV(xxx)
      TEMPDSFORMAT(UNIQUE|INCLUDELABEL)
      VERIFY_UNCAT(FAIL|TRACK|MSGTRACK)
      MEMDSENQMGMT(ENABLE|DISABLE)
```

Figure 14-1 MEMDSENQMGMT subparameter in the ALLOCxx parmlib member

## IEFDDSRV macro instruction

This macro is new with z/OS V1R12. When MODIFY and TYPE=FEATURE, as shown in Figure 14-2, are specified, this is a required parameter that indicates how allocation should manage ENQs on the data set name when you specify DSENQMGMT=MEMORY.

```
IEFDDSRV MODIFY
      ,TYPE=ALLOCATION Default: TYPE=ALLOCATION
      ,DDNAME=ddname ddname: RS-type address or register (2) - (12) ASM only.
      ,DSABPTR=dsabptr dsabptr: RS-type address or register (2) - (12) ASM only.
      ,NEWDDNAME=newddname newddname: RS-type address, or register (2)-(12).
      ,TYPE=FEATURE
      ,DSENQMGMT=NO_CHANGE
      ,DSENQMGMT=MEMORY
```

Figure 14-2 DSENQMGMT subparameter of the IEFDDSRV macro instruction

The options are as follows:

- DSENQMGMT=NO\_CHANGE** Requests that no change be made to the data set ENQ management.
- DSENQMGMT=MEMORY** Requests that data set ENQs be managed in memory. Specifying this option causes the job to be non-restartable through the check-point/restart function from this point on. The SYSTEM MEMDSENQMG MT keyword in the ALLOCxx parmlib member must be set to ENABLE to allow the job or subsystem to use memory-based data set ENQ management.

## Benefits of memory-based data set ENQ management

Heavy data set allocation users should expect significant reduction of processor time spent to manage serialization of access to data sets and observe a reduction of time to recovery after outages, when it becomes necessary to restart the job or subsystem. You should expect better performance from subsystems when many data sets are used, which should provide a better Mean Time To Recover from errors affecting subsystems like DB2. This should allow installations to consolidate a greater work load onto z/OS with less performance impact.

More information about memory-based ENQ management for dynamically allocated data sets can be obtained in the following manuals:

- ▶ *z/OS MVS Initialization and Tuning Reference, SA22-7592*
- ▶ *z/OS MVS Programming: Assembler Services Reference, Volume 2 (IAR-XCT), SA22-7607*

### Migration considerations

When exploiting this new feature, consider the following options:

- ▶ Programs exploiting the DD accounting suppression function will have less data in the SMF Type 30 EXCP section or Type 40 records.
- ▶ VSAM-related SMF record types are not affected.
- ▶ Programs should not request this function for non-VSAM data sets.

This new function requires only the addition of MEMDSENQMGMT(ENABLE) in the ALLOCxx parmlib member for exploitation.

**Note:** When using this function for DB2, an IPL is required. Also, it is preferable when you are running with DSMAX at 64 K or 100 K.

## 14.2 Suppressing SMF DD accounting

Starting with z/OS V1R12, it is possible to suppress DD accounting information on a per-DD basis for data sets in SMF records type 14, 30, and 40. Heavy dynamic allocation users, such as DB2, create high volumes of DD EXCP accounting data that may not be necessary all the time.

### Requesting accounting suppression

The suppression of the creation of DD accounting must be requested by an authorized program, through the IEFDDSRV macro instruction; see *z/OS MVS Authorized Assembler Services Guide, SA22-7608*. The S99DASUP bit of flag S99FLAG2 must be set on the request block (S99RB) to begin the suppression.

S99DASUP is used by authorized programs to suppress the DD-level accounting. Setting this bit can affect the SMF data created for the following:

- ▶ The EXCP section of SMF Record Type 30
- ▶ SMF Record Type 40
- ▶ SMF Record Type 14 for the fields SMF14NTR and SMF14NER

**Note:** This bit is only recommended for programs allocating VSAM data sets with generated DD names, or when the exploiting program has established that the usefulness of the SMF data is less than the benefit to system performance. Because the data is used by an installation and suppressed by the exploiting program, an external switch controlling the program's use of this bit is strongly recommended.

The SMF type 30 record has a new field, SMF30DAS, that shows the number of DDs that had their DD accounting information suppressed.

## 14.3 Duplicate temporary data set name support

With z/OS V1R12, duplicate temporary data set name support is enhanced. In previous z/OS versions, when two jobs, with the same jobname, using named (&&*mydsn*) temporary data sets, were scheduled in the same system, at the same time, the second would be canceled with a JCL error, due to a duplicate temporary data set name. This situation occurred mostly with jobs that passed temporary data sets from one step to another.

With z/OS V1R12, the behavior to generate data set names using the label or unique number can be controlled via the ALLOCxx parmlib member SYSTEM TEMPDSFORMAT setting. No changes to JCL will be needed.

The SETALLOC command now supports the new keyword, making it easy for clients to change the format of the generated temporary dsnames.

Only the naming convention of the temporary data set names is changed. There are no changes to the behavior. For example, all temporary data sets using the same input &&*mydsn* in the same job get the same data set name, as before. Similarly, all temporary data sets in the job referencing an earlier temporary data set will *not* see any change in behavior even though the referenced data set has a unique name.

### 14.3.1 Specifying unique temporary data set names

A new ALLOCxx parmlib member parameter, TEMPDSFORMAT, specifies how the system generates data set names for temporary data sets that include &&*label* as the specified data set name. This statement affects only the data sets that specify DSN=&&*mydsn*, but not the data sets that do not specify DSNNAME at all. This affects both JCL and dynamically allocated temporary data sets.

```
SYSTEM TEMPDSFORMAT(UNIQUE|INCLUDELABEL)
```

Where:

- |                     |  |
|---------------------|--|
| <b>UNIQUE</b>       | Indicates that when the system processes JCL that includes temporary data sets with DSN=&&LABEL, the generated data set name will be in the form "SYSyddd.Thhmmss.RA000.jjobname.Rggnnnnn", which does not include the && <i>label</i> specified in the JCL. All references to "&& <i>mydsn</i> " throughout the JCL correctly refer to the same data set. Using TEMPDSFORMAT=UNIQUE ensures that jobs with the same jobname running simultaneously do not create temporary data sets with the same names. See <i>z/OS MVS JCL Reference</i> , SA22-7597 for more information. |
| <b>INCLUDELABEL</b> | Indicates that when the system processes JCL that includes temporary data sets with DSN=&&LABEL, the generated data set name will include the && <i>label</i> specified in the JCL. See <i>z/OS MVS JCL Reference</i> for more information.  |

**Note:** When this parameter is specified and the Job Entry Subsystem (JES) allows multiple jobs with the same job name to execute at the same time, jobs with the same name, executing simultaneously, might fail with a duplicate data set name error.

**Default:** UNIQUE

## TEMPDSFORMAT(UNIQUE)

If you do not specify a data set name, or TEMPDSFORMAT(UNIQUE) is in effect, the full format of the temporary data set name is:

```
SYSydd.Thhmss.RA000.jjobname.Rggnnnn
```

Where:

- ▶ gg is 01, meaning in a sysplex:
  - For JCL allocations, the system identifier of the system that interpreted the job.
  - For dynamic allocations, the system identifier of the system on which the job executed.
- ▶ nnnnn is a number that is unique within a system.

## TEMPDSFORMAT(INCLUDELABEL)

If you do specify a data set name and TEMPDSFORMAT(INCLUDELABEL) is in effect, the full format of the temporary data set name is:

```
SYSydd.Thhmss.RA000.jjobname.dsetname.Hgg
```

Where:

- ▶ dsetname:
  - The 1 to 8 character DSNNAME coded following the two ampersands (&&).
- ▶ gg=01, in a sysplex:
  - For JCL allocations, the system identifier of the system that interpreted the job.
  - For dynamic allocations, the system identifier of the system on which the job executed.

## Using DSNNAME conditions

If you use DSNNAME, note that the system-generated qualified name for the temporary data set will not be unique under the following conditions:

- ▶ Multiple tasks or APPC transactions having identical jobnames executing at exactly the same time.
- ▶ The tasks or transactions contain DD statements with identical temporary data set names.

### Notes:

- ▶ To ensure that a temporary data set name is unique, do not code a temporary data set name. Allow the system to assign one. Only the job that creates a temporary data set has access to it to read and write data and to scratch the data set.
- ▶ In general, the system treats a single ampersand (&) followed by a character string of 1 to 8 characters as a symbolic parameter (See *Using System Symbols and JCL Symbols* in *z/OS MVS JCL Reference*, SA22-7597). However, if you code a data set name as a symbolic parameter (by coding DSNNAME=&xxxxxxx, and do not assign a value to or nullify the symbolic parameter, the system will process it as a temporary data set name.
- ▶ The SYSTEM TEMPDSFORMAT(UNIQUE|INCLUDELABEL) option in the parmlib member ALLOCxx enables allocation to use a more unique format for the data set name when DSN=&&mydsn is specified. The unique data set name allows jobs with the same data set names specified to run at the same time, without requiring the JCL programmer to remove the DSN=&&mydsn or to add data set name refer-back syntax. The system setting for this option may affect the data set name generated for a temporary data set.



### 14.3.2 TEMPDSFORMAT command options

The option set for temporary data set naming can be displayed with the MVS command D ALLOC,OPTIONS, as shown in Figure 14-3.

```
D ALLOC,OPTIONS
.....
VERIFY_VOL      POLICY:          YES
SYSTEM          IEFBR14_DELMIGDS: LEGACY
                TAPELIB_PREF:    EQUAL
                REMIND_INTV:     90
                VERIFY_UNCAT:    FAIL
                TEMPDSFORMAT:    UNIQUE
                MEMDSENQMGMT:    DISABLE
```

Figure 14-3 ALLOCxx parmlib member setting for SYSTEM with TEMPDSFORMAT

#### JCL temporary data set example

The result of running a two-step job passing a temporary data set named &&MYTEMP, with TEMPDSFORMAT=UNIQUE in effect, can be seen in Figure 14-4.

```
IEF236I ALLOC. FOR HEUSERJ STEP1
IGD100I D527 ALLOCATED TO DDNAME DD1      DATACLAS (      )
IEF142I HEUSERJ STEP1 - STEP WAS EXECUTED - COND CODE 0000
IEF285I  SYS10141.T143444.RA000.HEUSERJ.R0100590      PASSED
IEF285I  VOL SER NOS= BH5ST6.
IEF373I STEP/STEP1 /START 2010141.1434
IEF374I STEP/STEP1 /STOP 2010141.1434 CPU  OMIN 00.00SEC SRB
IEF236I ALLOC. FOR HEUSERJ STEP2
IEF237I D527 ALLOCATED TO DD2
IEF142I HEUSERJ STEP2 - STEP WAS EXECUTED - COND CODE 0000
IEF285I  SYS10141.T143444.RA000.HEUSERJ.R0100590      DELETED
IEF285I  VOL SER NOS= BH5ST6.
```

Figure 14-4 Example of UNIQUE temporary data set naming

#### Command to set the TEMPDSFORMAT parameter

The TEMPDSFORMAT parameter can be changed, either in the ALLOCxx parmlib member or by issuing the SETALLOC command, as shown in Figure 14-5.

```
SETALLOC SYSTEM,TEMPDSFORMAT=INCLUDELABEL
IEFA010I SETALLOC COMMAND SUCCESSFUL
TEMPDSFORMAT SET TO INCLUDELABEL.
```

Figure 14-5 Changing TEMPDSFORMAT with the SETALLOC command

Running the same job again, after issuing the command in Figure 14-5, resulted in the temporary data set names shown in Figure 14-6 on page 260, for the same files.

```

IEF236I ALLOC. FOR HEUSERP STEP1
IGD100I 8083 ALLOCATED TO DDNAME DD1      DATACLAS (      )
IEF142I HEUSERP STEP1 - STEP WAS EXECUTED - COND CODE 0000
IEF285I  SYS10141.T151047.RA000.HEUSERP.MYTEMP.H01  PASSED
IEF285I  VOL SER NOS= BH5ST2.
IEF373I STEP/STEP1  /START 2010141.1510
IEF374I STEP/STEP1  /STOP 2010141.1510 CPU    OMIN 00.00SEC SRB
IEF236I ALLOC. FOR HEUSERP STEP2
IEF237I 8083 ALLOCATED TO DD2
IEF142I HEUSERP STEP2 - STEP WAS EXECUTED - COND CODE 0000
IEF285I  SYS10141.T151047.RA000.HEUSERP.MYTEMP.H01  DELETED
IEF285I  VOL SER NOS= BH5ST2.

```

Figure 14-6 Example of INCLUDELABEL temporary data set naming

**Warning:** Prior to using this option, enforced by default, review any user-written procedure that depends on temporary data set name format.

### Additional information

More information about unique temporary data set names can be obtained in the following manuals:

- ▶ *z/OS MVS System Commands, SA22-7627*
- ▶ *z/OS MVS System Messages, Volume 8 (IEF-IGD), SA22-7638*
- ▶ *z/OS MVS JCL Reference, SA22-7597*
- ▶ *z/OS MVS Initialization and Tuning Reference, SA22-7592*
- ▶ *z/OS Migration, GA22-7499*



## XML System Services

XML allows you to tag data in a way that is similar to how you tag data when creating an HTML file. XML incorporates many of the successful features of HTML, but was also developed to address some of the limitations of HTML. XML tags may be user-defined, by either a DTD or a document written in the XML Schema language, that can be used for validation. In addition, namespaces can help ensure that you have unique tags for your XML document.

The syntax of XML has more restrictions than HTML, but this results in faster and cheaper browsing. The ability to create your own tagging structure gives you the power to categorize and structure data for both ease of retrieval and ease of display. XML is already being used for publishing, as well as for data storage and retrieval, data interchange between heterogeneous platforms, data transformations, and data displays. As these XML applications evolve and become more powerful, they may allow for single-source data retrieval and data display.

This chapter describes the enhancements to XML in z/OS V1R12, as follows:

- ▶ Restrict the root element name
- ▶ Dynamic schema
- ▶ Fragment parsing

## 15.1 Restrict root element name

When DB2 encounters an unresolved general entity, it always wants to stop. XML System Services has specific scenarios where it does not stop, which is in accordance with the XML specifications. With z/OS V1R12, IBM has enhanced XML System Services to add a new option that causes the non-compliant behavior. The new function is invoked by performing a parser control operation to pass in a list of allowable names prior to a parse.

In Table 15-1 you see the old behavior of the XML System Services parser in prior releases.

Table 15-1 Old parsing behavior

ENTITY resolved	DTD declared	Standalone	Behavior of the parser
No	Yes	No	The parser will continue and put out an additional record in the output stream and continue parsing (XEC_TOK_UNRESOLVED_REF).
No	Yes	Yes	The parser stops and flags an error (external DTD not considered if standalone = "yes").
No	No	No	The parser stops and indicates an error.
No	No	Yes	The parser stops and flags an error (external DTD not considered if standalone="yes").

### 15.1.1 z/OS V1R12 parsing

While performing a validating parse, the caller has the option to restrict the root element name to a list of one or more root name or namespace pairs. When selecting this option, validation is performed on the root name in the document being parsed.

**Note:** This option is only available for validating parses.

Table 15-2 shows the behavior of the parser when the new functions are enabled. In any case where an entity is resolved, the parse continues with no extra action taken. This allows the callers to perform a parser control operation where they can pass in a list of valid root element names. During validation, check that the name of the root element is in this list.

Table 15-2 New parsing behavior

ENTITY resolved	DTD declared	Standalone	Behavior of the parser
No	Yes	No	The parser stops and flags this as an error. This is the changed scenario.
No	Yes	Yes	The parser stops and flags an error (external DTD not considered if standalone = "yes").
No	No	No	The parser stops and indicates an error.
No	No	Yes	The parser stops and it is flagged an error (external DTD not considered if standalone="yes").

## Function enablement

To enable this option, the caller must perform a control call (`gxlpControl`) during the parse step with the control option `GXLHxec_CTL_RESTRICT_ROOT` prior to parsing the document to indicate that the root name is to be validated. The caller must also pass along a data area in the format of `GXLHXRR`, which contains the list of root names. Failing to do this will cause an error, resulting in the z/OS XML parser needing to be reset using `CTL_FIN`. The root names are specified by a local name (root name) and an optional URI for the root namespace. The strings passed to the control call (`gxlpControl`) must be in the encoding of the z/OS XML parser configured at initialization time.

The control call (`gxlpControl`) prepares the z/OS XML parser for a new document, but the current feature set is preserved. Subsequent resets, such as `CTL_FEAT`, will not change the current settings of the restrict root element control call. These settings will still apply when parsing subsequent documents.

## Steps to perform after parser initialization

The following need to be done after parser initialization, and before the first parse request to specify the list of allowable names:

- ▶ The caller passes the name list through a new data structure.
- ▶ `GXLHXRR` - defined in `gxlhctl.h`.
- ▶ Includes the following fields (among others):
  - The number of names in the list (`XRR_ENTRY_COUNT`)
  - A pointer to the list of names (`XRR_ENTRY_PTR`)
- ▶ Each name is made up of two components:
  - The local name
  - The namespace URI

The new function is invoked by performing a parser control operation to pass a list of allowable names prior to a parse. You can remove the restriction on the root element name by calling `gxlpControl()` with the control option `GXLHxec_CTL_RESTRICT_ROOT` and setting the `XRR_ENTRY_COUNT` value to 0 in the `GXLHXRR` data area. The information produced in the output buffer from the subsequent parse does not change when using this option.

The following is an example call sequence:

```
gxlpLoad
gxlpInit
gxlpControl(GXLHxec_CTL_LOAD_OSR)
gxlpControl(GXLHxec_CTL_RESTRICT_ROOT,GXLHXRR)
gxlpParse
gxlpTerminate
```

## 15.2 Dynamic schema

When parsing a document containing schema references, you generate and load an OSR. In order to make sure that the appropriate OSR is loaded, you must determine which schemas are referenced in the document and their locations. To this end, you can query the XML document for namespaces and schema locations.

## New feature flag

You can obtain information about the schema location by initializing the PIMA with the GXLHxec\_FEAT\_SCHEMA\_DISCOVERY feature. This will cause the z/OS XML parser to pause after parsing the start tag of the root element. The output buffer is then populated with records as if a normal parse was performed, with the following additional records: GXLHxec\_TOK\_ROOT\_ELEMENT and GXLHxec\_TOK\_SCHEMA\_LOCATION. GXLHxec\_TOK\_ROOT\_ELEMENT contains the root element name and GXLHxec\_TOK\_SCHEMA\_LOCATION contains the schema location information. The output buffer will not contain any start element, attribute value, or namespace declaration records. After the end of the start tag has been reached and all schema info records have been put out, the z/OS XML parser provides you an opportunity to load an OSR before the parse is continued. If the parse is continued, whichever OSR was loaded by a GXLHxec\_CTL\_LOAD\_OSR operation will be used to validate the document. If no OSR loading operation has been performed since parser initialization, the parser must be reset in order to parse again.

Parse up to the end of the start element tag for the root, and return any schema location information before continuing with a validating parse. XML System Services users do not have to manage schema information separately from the XML instance document in those cases where the location hint mechanism is used.

The new option may only be specified at parser initialization time. It cannot be specified on a parser control request. The parser performs a non-validating parse to the end of the root element start tag. It then returns schema location information if it is present and the caller loads the appropriate schema and makes another parse request. Afterwards, the parser switches to validating mode, and parsing continues to the end of the document.

A flag will be set in the output buffer header whenever an output record in that buffer contains a record that had a character replaced with the replacement character (a dash, by default). The flag will be set if the condition described is encountered (that is, this is a new function that is always included and cannot be set with a CTL call).

## 15.3 Fragment parsing

IBM introduced a new function in XML System Services called Fragment Parsing support. This function allows a caller to parse XML fragments instead of entire XML documents. This support is available by providing a new parser mode that allows parsing of a fragment as if it were an external parsed entity. Callers may validate selected portions of XML text, rather than a whole document. This can be much more efficient for callers concerned with only very small portions of very large documents. Figure 15-1 shows the general flow of parsing XML documents.

<code>gxlpLoad(...)</code>	load the validating parser module
<code>gxlpInit(...)</code>	initialize the parser for validation
<code>gxlpControl(GXLHxec_CTL_LOAD_OSR)</code>	load the required schema
<code>gxlpControl(GXLHxec_CTL_LOAD_FRAG_CONTEXT,FPATH)</code>	load the fragment context and path
<code>gxlpControl(GXLHxec_CTL_FRAGMENT_PARSE)</code>	enable fragment mode
<code>gxlpParse(...)</code>	parse the XML fragment
<code>gxlpControl(GXLHxec_CTL_FRAGMENT_PARSE)</code>	disable fragment mode
<code>gxlpTerminate(...)</code>	terminate the parse instance

Figure 15-1 Example call sequence for a validating fragment parse

**Restriction:** Validation in fragment parsing cannot satisfy all aspects of schema validation for an arbitrary input string because various aspects of schema validation refer to other aspects of the document. Although it may be possible to validate some aspects of the following schema constructs, in general they require the entire document to be available. The restriction falls into two broad categories: those things that cannot be validated reasonably and those things that can be validated in isolation but could possibly fail within the context of a document. The first category is avoided by requiring the client to ensure that the name of the element matches the element that it is replacing. The second category includes a long list of attributes and elements; see *z/OS XML System Services User's Guide and Reference*, SA23-1350.

### 15.3.1 Parsing XML document fragments with validation

Before beginning document fragment parsing, you must specify the fragment path. The namespace binding information is also required when there is namespace context associated with the fragment path. To load the fragment context, need to issue a `gxlpControl` call with the control option `XEC_CTL_LOAD_FRAG_CONTEXT`. This must occur before document fragment parsing is enabled.

The validating parser provides support for parsing XML document fragments. The W3C XML specification allows parsing of such document fragments as external parsed entities. The document fragment can be the value of a single attribute, as well as a single element and its descendants. It may be followed by comments and processing instructions. The parser must be provided ancestor and namespace context information to ensure proper validation.

**Note:** A new OSR with the extra Fragment Parsing Table information is required in order to parse a document fragment with validation. Pre-z/OS V1R12 OSRs cannot be used in this parsing environment. To load the OSR, you need to issue a control call with the control option `XEC_CTL_LOAD_OSR`. This must occur before document fragment parsing is enabled.

#### Enabling fragment parsing

To enable fragment parsing, issue a `gxlpControl` call with the control option `XEC_CTL_FRAGMENT_PARSE` and set the `XFP_FLAGS_FRAGMENT_MODE` bit to ON in the control data structure. This must occur prior to issuing the `gxlpParse` call with the XML document fragment loaded into the input buffer. The z/OS XML parser will perform regular parsing including well-formed checking and validation. However, the root element is not required. The XML declaration and Doctype Declaration are not allowed as part of the document fragment.

#### Fragment mode

Switch to fragment mode (`GXLXEC_CTL_FRAGMENT_PARSE`). One attribute value or element may be validated at a time. The element may contain nested descendants. The element may optionally be followed by miscellaneous text. The parser must be reset after completing a fragment parse. You may toggle between document and fragment mode as needed within a single parse instance. Some aspects of validation cannot be handled in fragment mode. When you finish parsing a document fragment, you must issue a `gxlpControl` call with the control option `XEC_CTL_FRAGMENT_PARSE` and set the `XFP_FLAGS_FRAGMENT_MODE` bit to OFF in the control data structure to notify the z/OS XML parser that fragment parsing has been disabled.

### 15.3.2 Enable offload to specialty engines

z/OS XML System Services provides the ability for parsing operations to be run on specialty processors: an IBM System z Application Assist Processor (zAAP) or an IBM System z10 Integrated Information Processor (zIIP). The z/OS XML parser, when running in TCB mode, is eligible to run on a zAAP, in environments in which one or more zAAPs are configured. The z/OS XML parser, when running in enclave SRB mode, is eligible to run on a zIIP processor, in environments where one or more zIIPs are configured. Ancillary z/OS XML System Services, such as the query service and the control service, as well as the StringID exit and memory management exits, are not eligible to run on specialty processors. Running of z/OS XML System Services parsing operations on a specialty processor occurs transparently to the calling application.





## z/OS UNIX System Services

The UNIX System Services element of z/OS 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 system interfaces on the z/OS operating system: an application program interface (API) and an interactive shell interface.

This chapter provides information about changed and updated functions for z/OS UNIX System Services and zFS, as follows:

- ▶ Exploit GRS identity for latches used by UNIX System Services
- ▶ BCP Health Checker exploits BPX.SUPERUSER
- ▶ BCP Health Checker USS\_HFS\_DETECTED
- ▶ z/OS UNIX mmap improvements
- ▶ z/OS UNIX Source Address Selection Support
- ▶ z/OS UNIX FILEDATA=RECORD limit for USS
- ▶ USS Shells and Utilities tsocmd
- ▶ USS support for sysplex-aware on a file system basis (OA29712)
- ▶ zFS sysplex-aware on a file system basis (OA29619)
- ▶ DFSMSdfp indirect volume serial for zFS data sets
- ▶ zFS usage of DFSMSdfp VSAM partial release on multiple volumes
- ▶ Remove file system support for zFS multfile system aggregates

## 16.1 GRS identity for latches used by z/OS UNIX

Global Resource Serialization latches that the UNIX System Services logical file system (LFS) uses are to provide serialization for file systems. The UNIX System Services LFS uses three levels of Global Resource Serialization latches to provide serialization for file systems:

### Mount latch

The mount latch provides serialization for operations involving the LFS and is the latch number two in the SYS.BPX.A000.FSLIT.FILESYS.LSN latch set. The mount latch is obtained exclusively:

- ▶ When a file system is mounted or unmounted.
- ▶ In a sysplex configuration, for operations such as file system moves, lost system recovery, system initialization, and reading from or writing to a couple data set.

Obtaining the mount latch exclusively ensures that only one of these activities is going on at the same time. Use the DISPLAY GRS,LATCH,CONTENTION command to look for mount latch contention.

### File system latch

There is a latch for each file system mounted. These latches are within the SYS.BPX.A000.FSLIT.FILESYS.LSN latch set. The file system latch is:

- ▶ Obtained exclusively every time that file system is unmounted, synchronized, exported or unexported by the server message block (SMB) server, moved, or recovered within a sysplex.
- ▶ Obtained in shared mode for the duration of any operation within the file system, such as reads from or writes to a file. This prevents the file system from being unmounted or moved, for example, while there is an operation in progress on a file within the file system. Use the DISPLAY GRS,LATCH,CONTENTION command to look for file system latch contention.

### File latch

There is a file latch associated with each active file or directory. A file latch can be obtained in either exclusive or shared mode, depending on the operation involved. For example, the file latch for a directory would be obtained in shared mode to read a name from the directory. But it would be obtained exclusively to write a name to the directory during a file create operation.

File latches are not used with the zSeries File System (zFS) physical file system because the zFS has its own file level serialization mechanisms. File latches are used with shared file systems, TFS, pipes, character special, and NFS client physical file systems. File latches are in a special group of latches with names in the form of SYS.BPX.A000.FSLIT.LSN.nn, where nn is a hexadecimal number. Use the DISPLAY GRS,LATCH,CONTENTION command to look for file latch contention.

### Latch numbers

The DISPLAY GRS commands identify latch numbers within a latch set that is in contention. For the most part, latch numbers cannot be related to any specific resource without taking and analyzing a dump. In z/OS V1R11, GRS added an interface that allows a latch exploiter to associate a name or latch identity with a latch number. Exploitation of that capability in UNIX System Services makes the DISPLAY GRS commands more useful for problem determination.

## 16.1.1 GRS latch identity used by z/OS UNIX

Here are three examples of latch identities:

**FS: <fs\_name>** If the LSETNAME is SYS.BPX.A000.FSLIT.FILESYS.LSN, the latch is used to serialize operations on the file system named in the latch identity string. If the LSETNAME is SYS.BPX.A000.FSLIT. QUIESCES.LSN, the latch is used to quiesce the file system named in the latch identity string.

**MOUNT** This latch is used by the file system to serialize operations such as file system mount, unmount, move, automount, and others.

**MessageQ ID=<msgid>** This latch is used when the system is traversing or modifying structures related to the message queue whose identifier is shown in the latch identity string. The value of *msgid* is in decimal.

The entire list of identities can be found in *z/OS MVS Diagnosis: Reference*, GA22-7588 in Chapter 20, Table 20-1. For more information about the DISPLAY GRS commands, see *z/OS MVS System Commands*, SA22-7627.

### Using D GRS in z/OS V1R11

In the previous release, the output from a DISPLAY GRS,ANALYZE command for a UNIX System Services latch contention is shown in Figure 16-1.

```
D GRS,ANALYZE,LATCH,WAITER
ISG374I 16.15.24 GRS ANALYSIS 734
LONG WAITER ANALYSIS: ENTIRE SYSTEM
----- LONG WAITER #1
WAITTIME JOBNAME E/S CASID LSETNAME/LATCHID
00:01:01 TCO *E* 000E SYS.BPX.A000.FSLIT.FILESYS.LSN
                20:(ID NOT SPECIFIED)
```

Figure 16-1 D GRS,ANALYZE,LATCH,WAITER in z/OS V1R11

## 16.1.2 Using GRS latch identities with z/OS V1R12

z/OS UNIX System Services uses GRS latches to serialize resources and operations. Once a contention occurs, the MVS DISPLAY GRS command can be issued to analyze what latch is held and the name of the job that holds it. In z/OS V1R12, UNIX System Services exploits using GRS identities for its latches. This enhances problem determination.

### Using D GRS in z/OS V1R12

In z/OS V1R12, with the new support, you can see a latch identity (LATCHID) together with a latch number within the latch set (LSETNAME), as shown in Figure 16-2 on page 270. This example shows the following:

- ▶ The latch set name (LSETNAME) is SYS.BPX.A000.FSLIT.FILESYS.LSN.
- ▶ The latch identity string (LATCHID) is FS: HOST12.AJAX.DIRECTORY, and the latch number is “20”.

```

D GRS,ANALYZE,LATCH,WAITER
ISG374I 16.15.24 GRS ANALYSIS 734
LONG WAITER ANALYSIS: ENTIRE SYSTEM
----- LONG WAITER #1
WAITTIME JOBNAME E/S CASID LSETNAME/LATCHID
00:01:01 TCO *E* 000E SYS.BPX.A000.FSLIT.FILESYS.LSN
                20:FS: HOST12.AJAX.DIRECTORY

```

Figure 16-2 D GRS,ANALYZE,LATCH,WAITER command in z/OS V1R12

## 16.2 Health Checker exploitation of BPX.SUPERUSER

In order to successfully run health checks that use z/OS UNIX System Services, Health Checker requires a user profile with the following information in the OMVS segment:

- ▶ USS superuser authority
- ▶ A home directory of HOME('/')
- ▶ A program of PROGRAM('/bin/sh')
- ▶ NOPASSWORD

Before z/OS V1R12, superuser authority was required to be given via a user profile with an OMVS UID being set to 0. With access to the BPX.SUPERUSER resource, the advantage of this method is that it might be more audit friendly, because you avoid having a user profile with UID(0) explicitly assigned to it.

**Note:** Once you start IBM Health Checker for z/OS with its associated user ID, changes you make to the UID for the user ID will not usually take effect until the IBM Health Checker for the z/OS address space is stopped and restarted. UID(0) user profiles are closely watched by auditors and might require extra explanation.

### 16.2.1 Using BPX.SUPERUSER resource in the FACILITY class

With z/OS V1R12, you can associate the Health Checker address space with a user profile that has the following definitions:

- ▶ READ access to the BPX.SUPERUSER resource in the FACILITY class
- ▶ A non-zero UID

At runtime, IBM Health Checker for z/OS dynamically switches to (and stays in) an effective UID(0) superuser authority using the defined BPX.SUPERUSER access.

**Note:** The Health Checker associated user profile does not appear on any UID(0) audit reports any longer. So, using a user profile with explicit UID(0) is worse than using the still powerful BPX.SUPERUSER access.

#### Examples for setting up the Health Checker user ID

The *z/OS IBM Health Checker for z/OS: User's Guide*, SA22-7994 provides details and example statements for associating the Health Checker address space with a UID(0). The user profile is shown in Figure 16-3 on page 271 or with a BPX.SUPERUSER user profile that is displayed in Figure 16-4 on page 271, and which is recommended for clients who do not like to give user IDs of zero.

```
ADDUSER hcsuperid OMVS(UID(0) HOME('/') PROGRAM('/bin/sh')) NOPASSWORD
RDEFINE STARTED HZSPROC.* STDATA(USER(hcsuperid))
```

Figure 16-3 Defining the Health Checker user ID as a superuser

Define the Health Checker user ID with a BPX.SUPERUSER FACILITY class profile in z/OS V1R12.

```
ADDUSER hcsuperid OMVS(UID(non-zero) HOME('/') PROGRAM('/bin/sh')) NOPASSWORD
PERMIT BPX.SUPERUSER CLASS(FACILITY) ID(hcsuperid) ACCESS(READ)
RDEFINE STARTED HZSPROC.* STDATA(USER(hcsuperid))
```

Figure 16-4 Defining the Health Checker user ID as a BPX.SUPERUSER

After startup of Health Checker, it switches its non-zero UID to 0 for its work, by using the authority given it with the BPX.SUPERUSER access. So it is a “dynamic” UID(0) address space.

**Important:** The Health Checker BPX.SUPERUSER functionality is available only in z/OS V1R12 and future releases. No function rollback to previous releases is planned. The two setup choices are transparent to all checks, in particular checks requiring USS, and do not require any changes.

## 16.3 Health Checker USS\_HFS\_DETECTED

Beginning with z/OS V1R5, HFS was no longer considered the strategic file system, and zFS became the strategic file system. This check will be used to highlight any HFS file systems that are still being used so that they can be migrated to zFS.

This new check verifies all file systems mounted and issues an exception message if any HFS file systems are found. The check only looks for HFS file systems that are owned on the system running the health check.

This check runs any time an HFS file system is successfully mounted, unless overridden by the RUN\_ON\_MOUNT=NO parameter. The check will also run any time a F BPXOINIT,FILESYS=REINIT command is issued.

The check does not run after the F OMVS,NEWROOT=xxx command is issued.

**Check reason:** HFS file systems are no longer the strategic file systems. All HFS file systems should be migrated to zFS. This check is a quick way to identify which HFS file systems are mounted on the system and a reminder to migrate to zFS.

This Health Checker routine is run with a setting of Low Severity once a day.

- ▶ The check will be scheduled to run every 24 hours.
- ▶ The check will run after a F BPXOINIT,FILESYS=REINIT command is issued. This can only be on the system where the command is issued.

### 16.3.1 USS\_HFS\_DETECTED parameters

There are two parameters that can be specified with this health check:

**RUN\_ON\_MOUNT** This indicates whether the check should run after the successful mount of an HFS file system. You can specify YES or NO.

DEFAULT: RUN\_ON\_MOUNT=YES

**HFS\_LIST** You can specify a list of HFS file systems that you wish to ignore for this check. A file system specified for this parameter will not cause the check to issue an exception message. This parameter is limited to 23 file systems and by the health check limit of 256 characters for a check's parameters. Following is an example for specifying this parameter:

HFS\_LIST=(OMVS.TEST1.HFS,OMVS.TEST2.HFS)

#### Defining the health check

The following shows keywords you can use to override check values on either a POLICY statement in the HZSPRMxx parmlib member or on a MODIFY command. This statement may be copied and modified to override the check defaults:

```
UPDATE
CHECK(IBMUS, USS_HFS_DETECTED)
SEVERITY(LOW)
INTERVAL(24:00)
PARM('RUN_ON_MOUNT=YES')
DATE(20090427)
REASON('your reason for making the update')
```

### 16.3.2 USS\_HFS\_DETECTED messages

Three specific messages can be displayed with this health check. The messages are used as follows:

- ▶ BPXH067I is displayed in the Health Checker report when everything is okay. This means that no mounted HFS file systems (except those in the HFS\_LIST) have been found.

BPXH067I No HFS file systems are mounted.

- ▶ BPXH069I is displayed in the Health Checker report if at least one HFS file system has been found. The message is followed by the list of the mounted HFS file systems.

BPXH069I The following HFS file systems were found

- ▶ BPXH068E is an exception message that is shown when errors are found. This means that message BPXH069I is displayed as well. This exception message is listed in the Health Checker report and is also displayed in the SYSLOG or OPERLOG.

BPXH068E One or more HFS file systems mounted.

### 16.3.3 USS\_HFS\_DETECTED examples

You can force running the check if RUN\_ON\_MOUNT=YES is specified, as shown in Figure 16-5 on page 273.

```
#> /usr/sbin/mount -f hering.testmnt.hfs test
#> df -v test | head -13 | tail -1
HFS, Read/Write, Device:197, ACLS=Y
#>
```

Figure 16-5 Mounting an HFS file system

This results in the messages shown in Figure 16-6 going to the SYSLOG.

```
HZS0001I CHECK(IBMUSS,USS_HFS_DETECTED): 235
BPXH068E One or more HFS file systems mounted.
```

Figure 16-6 USS\_HFS\_DETECTED messages in the syslog

Excerpts of the corresponding Health Checker report output are displayed in Figure 16-7.

```
CHECK(IBMUSS,USS_HFS_DETECTED)
START TIME: 05/14/2010 15:41:10.052735
CHECK DATE: 20090427 CHECK SEVERITY: LOW
CHECK PARM: RUN_ON_MOUNT=YES

BPXH003I z/OS UNIX System Services was initialized using OMVS=(2A),
where each 2-character item is a BPXPRMxx suffix.

BPXH069I The following HFS file systems were found:
-----
HERING.TESTMNT.HFS
OMVS.SC74.VAR
OMVS.SC74.ETC
OMVS.DB2V8.UK05586.HFS

* Low Severity Exception *

BPXH068E One or more HFS file systems mounted.

Explanation: The USS_HFS_DETECTED check found one or more active HFS
file systems on the current system.

System Action: The system continues processing.
... (17 lines have been removed)
Check Reason: HFS file systems are no longer the strategic file
system. All HFS file systems should be migrated to zFS.

END TIME: 05/14/2010 15:41:10.053512 STATUS: EXCEPTION-LOW
```

Figure 16-7 USS\_HFS\_DETECTED Health Checker output

You can also explicitly run this health check, as shown in Figure 16-8 on page 274.

```
F HZSPROC, RUN, CHECK=(IBMUSS, USS_HFS_DETECTED)
HZS0400I CHECK(IBMUSS, USS_HFS_DETECTED): 261
RUN PROCESSING HAS BEEN COMPLETED
HZS0001I CHECK(IBMUSS, USS_HFS_DETECTED): 262
BPXH068E One or more HFS file systems mounted.
```

Figure 16-8 Running health check USS\_HFS\_DETECTED using F HZSPROC

## 16.4 z/OS UNIX mmap improvements

A memory mapping (mmap) is a POSIX-compliant function that allows files, or any resource identified via a file descriptor, to be cached and shared in virtual memory.

mmap() is typically used by a single process to map a file into virtual memory using the same sort of logic used by DIV (Data In Virtual). Used in this manner, each page of the file requires three RSM control blocks (anchor block, user page, and kernel data space page). Each additional user sharing an mmap page of a file will consume an additional control block.

The `__MAP_MEGA` option of mmap() enables applications to map very large files without the system overhead in ESQA. If you have applications using the `__MAP_MEGA` option, you do not need to be concerned with the above calculations. If you are not using `__MAP_MEGA` and issue mmap(), you can estimate the ESQA usage just as you would for shared memory.

### z/OS V1R12 mmap improvements

In z/OS V1R12 two changes have been added to USS mmap() support:

- ▶ Allowing mmap() to an NFS Client file
- ▶ New message identifying the problem file causing an USS EC6 abend

### 16.4.1 USS mmap support for NFS Client files

Before z/OS V1R12, mmap() to an NFS Client file resulted in program error JrNotSupportedforRemoteFile.

In z/OS V1R12, file system interfaces were redesigned to allow all hierarchical file system types (HFS, NFS, TFS, and zFS) to be supported for mmap() using the same internals. Therefore, this now makes it possible that NFS files can be memory mapped.

**Note:** This support is transparent. Existing mmap and NFS externals still apply.

This means that a POSIX application currently using mmap() to memory map a file does not need to be changed when the file happens to be accessed via NFS.

### 16.4.2 Serviceability improvement

Unresolved mmap page faults resulted in a USS EC6 abend, MmapFileError. The abend information did not identify the file. Message BPXM122I in the hardcopy log now identifies it, as shown in Figure 16-9 on page 275.



```
BPXM122I MEMORY MAP ERROR FOR FILE SYSTEM fs PATH path INODE ino RETURN CODE=rc  
REASON CODE=rsn
```

Figure 16-9 BPXM122I message

## 16.5 z/OS UNIX source address selection support

No IPv6 Socket API exists that allows an AF\_INET6 socket descriptor to be bound to the “best” source address based on the selection rules for the provided destination IP address.

In z/OS V1R12, enhancements to the z/OS UNIX System Services IPv6 socket API functions are provided to support the RFC-5014 for source address selection.

The socket API will be particularly useful for IPv6 applications that want to choose between temporary and public addresses, and for mobile IPv6-aware applications that want to use the *care-of address* for communication. It also specifies socket options and flags for selecting cryptographically generated addresses (CGA) or non-CGA source addresses.

**Important:** The U.S. Government is expected to require products that it purchases to support the RFC 5014. Failure to do so may result in U.S. Government agencies no longer being able to purchase z/OS.

### 16.5.1 Using source address selection

Using source address selection support provides interfaces for an application to set up and retrieve information about its partner’s connection and security credentials.

- ▶ You can bind the best source address for the provided destination IP address based on the selection rules.
- ▶ You can validate the passed source address against a set of input source address selection preferences.
- ▶ You can retrieve information about the partner’s connection information and security credentials using a new IOCTL interface.

### 16.5.2 Binding to source address selection

Using the new support, the destination IP address is specified and requests that the socket be bound to the best source address for the provided destination IP address. The best source address is set by using preference bits set on doing `setsockopt()` with `Sock#IPV6_ADDR_PREFERENCES`.

**Restrictions:** There are some restrictions for the new support.

- ▶ The new function is not supported for IPv4. You will see return code `EOPNOTSUPP` and reason code `JrInCorrectSocketType`.
- ▶ RAW sockets are not supported.
- ▶ `IN6ADDR_ANY` destination address is not supported. You see return code `EINVAL` and reason code `JrInAddrAnyNotAllowed`.

The new IPv6 IPPROTO\_IPV6 level option values for setsockopt() and getsockopt() are listed in Figure 16-10.

Sock#IPv6_ADDR_PREFERENCES	EQU 32	
IPV6_PREFER_SRC_HOME	EQU x'00000001'	Prefer home address
IPV6_PREFER_SRC_COA	EQU x'00000002'	Prefer care of address
IPV6_PREFER_SRC_TMP	EQU x'00000004'	Prefer temporary address
IPV6_PREFER_SRC_PUBLIC	EQU x'00000008'	Prefer public address
IPV6_PREFER_SRC_CGA	EQU x'00000010'	Prefer Cryptographically - generated address
IPV6_PREFER_SRC_NONCGA	EQU x'00000020'	Prefer non-cryptographically gener'd address

Figure 16-10 New IPv6 IPPROTO\_IPV6 level option values for setsockopt () and getsockopt ()

**Note:** When the IPV6\_ADDR\_PREFERENCES is successfully set with setsockopt(), the option value given is used to specify the address preference for any connection initiation through the socket and all subsequent packets sent via that socket. If no option is set, the system selects a default value as per default address selection algorithm or by some other equivalent means.

The external application interface is provided with the following functions:

- ▶ BPX1BAS
- ▶ BPX4BAS
- ▶ bind2addrsel()

The return value, return code, and reason code are the same as with the bind() operation.

### 16.5.3 Validation of source address selection

The source address selection is validated against the set of source address selection preference flags. The external interfaces are as follows:

- ▶ inet6\_is\_srcaddr(IsSrcAddrIpAddr, IsSrcAddrFlags)
- ▶ BPX1PCT(PC#IsSrcAddr)

**Note:** There is no socket descriptor used. It is implemented as a vfs\_pfsctl operation.

The command value for this pfsctl service PC#IsSrcAddr is 'C0000018'x.

#### Assembler mapping for the PC#IsSrcAddr vfs\_pfsctl

The constant and mapping is provided in the BPXYCONS and BPXYSOCK external macros shown in Figure 16-11 and Figure 16-12 on page 277.

The constant PC#ISSRCADDR is found in BPXYCONS, shown in Figure 16-11.

* <pre>             inet6_is_srcaddr() function             PC#ISSRCADDR      EQU X'C0000018' inet6_is_srcaddr         </pre>
---

Figure 16-11 Constant PC#ISSRCADDR in BPXYCONS

The mapping is found in BPXYSOCK, shown in Figure 16-12 on page 277.

ISSRCADDR	DS 0F	
ISSRCADDRVER	DS XL1	Version. 1
ISSRCADDRVER1	EQU 1	Version value
	DS XL3	Reserved. Must be 0
ISSRCADDRIPADDR	DS CL(SOCK#LEN+SOCK_SIN6#LEN)	sockaddr_in6
ISSRCADDRFLAGS	DS F	Flags. See IPV6_PREFER_SRC_*
	DS 6F	Reserved. Must be 0
*		
ISSRCADDR#LEN	EQU *-ISSRCADDR	Length of ISSRCADDR area

Figure 16-12 PC#IsSrcAddr mapping in BPXYSOCK

### Installation considerations

If there is no AF\_INET6 NETWORK statement in BPXPRMxx, you receive return code x045A (EAFNOSUPPORT) and reason code x7313 (JrIPv6NotEnabled).

**EAFNOSUPPORT** The address family is not supported.

**JrIPv6NotEnabled** TCP/IP cannot process the IPv6 request because the TCP/IP stack is not currently IPv6 enabled.

If all AF\_INET6 TCP/IP stacks are down, you get return code x0070 (EAGAIN) and reason code x00B6 (JrPFSSuspend).

**EAGAIN** The resource is temporarily unavailable.

**JrPFSSuspend** The PFS is waiting to restart.

## 16.5.4 Set and Get partner information

Two new IOCTLS have been defined and added to the BPXYIOCC external macro, as shown in Figure 16-13.

* Connection type and security credentials on TCPIP sockets.		
* Refer to: Comm Svr: IP Programmer's Guide and References.		
SIOCGPARTNERINFO	EQU X'C000F612'	Get Info
SIOCSPARTNERINFO	EQU X'8004F613'	Set Optimization

Figure 16-13 Get and Set partner information IOCTLS

**SIOCGPARTNERINFO** This ioctl provides an interface for an application to retrieve information about its partner, including connection routing information, the user ID of the partner, or both.

**SIOCSPARTNERINFO** This ioctl provides an interface for an application to set up the environment that is required to retrieve the user ID of its partner using the SIOCGPARTNERINFO ioctl.

For more details, refer to *z/OS Communication Server: IP Programmer's Guide and Reference*, SC31-8885.

## 16.6 z/OS UNIX FILEDATA=RECORD support

In the past there were no z/OS UNIX commands that could set, display, copy, move, save or restore the record file format for z/OS UNIX files.

In z/OS V1R12, new support has been added to use **FILEDATA=RECORD** to do so.

- ▶ You can set the record file format for z/OS UNIX files.
- ▶ This is supported especially during copy processing.
- ▶ The record file format can now be displayed for z/OS UNIX files.
- ▶ Finally, you can save and restore the record file format for z/OS UNIX files.

**Note:** MVS applications using files in the z/OS UNIX file system can take advantage of record format files.

### 16.6.1 Using the record file format

Several changes have been added to external interfaces to support this. A new constant is defined in the external BPXYFTYP, and BPXYVFSI macros for the new file data type record. The architected constant value is 8.

#### **Update of mapping macro BPXYFTYP on using assembler**

BPXYFTYP is used to provide the values for the different file types. The new type FTFFRECORD has been added as shown in Figure 16-14 on page 279.

**Note:** Macro BPXYFTYP is composed only of EQUates.

For more information on BPXYFTYP, see Appendix B. Mapping macros—AMODE 31 in *z/OS UNIX System Services Programming: Assembler Callable Services Reference*, SA22-7803.

```

** BPXYFTYP: File type definitions
** Used By: FST MKD MKN OPN
FT_DIR            EQU 1      Directory File
FT_CHARSPEC      EQU 2      Character Special File
FT_REGFILE       EQU 3      Regular File
FT_FIFO          EQU 4      Named Pipe (FIFO) File
FT_SYMLINK       EQU 5      Symbolic link
*                EQU 6      Reserved for Block Special
FT_SOCKET        EQU 7      Socket File
*
** File format definitions (for chatter)          9
FTFFNA           EQU 0      Not specified          7
FTFFBINARY       EQU 1      Binary data
*                EQU 1      Text data delimiters:
FTFFNL           EQU 2      New Line
FTFFCR           EQU 3      Carrage Return
FTFFLF           EQU 4      Line Feed
FTFFCRLF         EQU 5      CR & LF
FTFFLF CR        EQU 6      LF & CR
FTFFCRNL         EQU 7      CR & NL
*
FTFFRECORD      EQU 8      Data consists of Records:
                          Records
** BPXYFTYP End

```

Figure 16-14 Mapping macro BPXYFTYP

### Update of header file BPXYVFSI on using C

BPXYVFSI describes the VFS (virtual file system interface) definitions. The new format, ATTR\_FFRECORD, has been added as shown in Figure 16-15 on page 280.

For more information about BPXYVFSI, see Appendix D. Interface structures for C language servers and clients in *z/OS UNIX System Services File System Interface Reference*, SA22-7808.

```

/*****START OF SPECIFICATIONS*****
*
*   $MAC (BPXYVFSI) COMP(SCPX4) PROD(FOM):
*
*01* MACRO NAME: BPXYVFSI
*
*01* DSECT NAME: N/A
*
*01* DESCRIPTIVE NAME: Virtual File System Interface Definition for C
...
        /*-----*/
        /* File Format Type Constants */
        /*-----$DAA*/
#define ATTR_FFNA      0 /* Not specified */
#define ATTR_FFBinary 1 /* Binary data */
/* Text data delimiters: */
#define ATTR_FFNL      2 /* New Line */
#define ATTR_FFCR      3 /* Carrage Return */
#define ATTR_FFLF      4 /* Line Feed */
#define ATTR_FFCRLF    5 /* CR & LF */
#define ATTR_FFLFCR    6 /* LF & CR */
#define ATTR_FFCRNL    7 /* CR & NL */
/* Data consists of Records: */
#define ATTR_FFRECORD 8 /* Records $DZA*/
...

```

Figure 16-15 Excerpts of header file BPXYVFSI

### z/OS UNIX REXX update

Use the new S\_FFRECORD variable for the record file data type. It is of data type decimal and has a value of 8. An example for exploiting this has been added to Figure 16-16 on page 281.

## 16.6.2 z/OS UNIX shell updates

Several commands and interfaces are updated to support using the new file format record.

### Setting and querying file format record

Use the **extattr** command to set the record file format for a z/OS UNIX file. Use the **1s** command with the option **-H** to display the z/OS UNIX file format, including new record values. Examples are shown in Figure 16-16 on page 281.

```

$> echo System $(sysvar SYSNAME) running $(uname -I) Version $(uname -Iv).$(uname -Ir)
System SC74 running z/OS Version 01.12.00
$> extattr -F REC record.file
$> ls -H record.file
-rw-r--r-- rec 1 HERING SYS1 17 Mai 10 17:38 record.file
$> rexx
SH> s "stat record.file st."
OMVS Return Value (retval) = 0
SH> say st.st_filefmt=s_ffrecord
1
SH> exit
$>

```

Figure 16-16 Using and displaying file format record

**Note:** The command `rexx` used in Figure 16-16 is an interactive REXX routine. Subcommand `s` is an abbreviation of Address `SYSCALL`.

### Setting or preserving file format record on using `cp`, `mv` or `pax`

Following we mention further commands to set or preserve a file format in general or using record specifically; refer to Figure 16-17.

- ▶ Use the command `cp` with option `-p` to preserve the record file format, or option `-F` to set the format.
- ▶ Use the `mv` command with option `-F` to set the record file format, or preserve the format value if `-F` is not specified.
- ▶ Use the `pax` command with option `-x pax` to preserve the record file format value, or use the value `ZOS.filefmt` to set the format as part of the save or restore operation.

```

$> cp -p record.file record.file2
$> ls -H record.file2
-rw-r--r-- rec 1 HERING SYS1 17 Mai 10 17:38 record.file2
$> rm record.file2
$> cp record.file record.file2
$> ls -H record.file2
-rw-r--r-- ---- 1 HERING SYS1 17 Mai 10 19:00 record.file2
$> cp -F rec record.file2 record.file3
$> ls -H record.file3
-rw-r--r-- rec 1 HERING SYS1 17 Mai 10 19:07 record.file3
$> mv record.file3 record.file2
$> ls -H record.file2
-rw-r--r-- rec 1 HERING SYS1 17 Mai 10 19:07 record.file2
$>

```

Figure 16-17 Setting or preserving file format record on using `cp` or `mv`

For more details, see *z/OS UNIX System Services Command Reference*, SA22-7802.

### 16.6.3 z/OS ISHELL update

The panel Display File Attributes shows the new record file format for z/OS UNIX files if set. This is demonstrated in Figure 16-18 on page 282.

```

File Directory Special_file Commands Help
-
  Edit Help
  -----
S |          Display File Attributes
a |
w | Pathname : /u/hering/record.file
n |          More:  - +
E | Link count . . . . . : 1
  | Set UID bit . . . . . : 0
a | Set GID bit . . . . . : 0
- | Sticky bit . . . . . : 0
- | Auditor audit . . . . : R=   W=   E=
- | User audit . . . . . : R= F  W= F  E= F
- | Device number . . . . : B3
- | Inode number . . . . . : 213
- | Major device . . . . . : 0
- | Minor device . . . . . : 0
- | File format . . . . . : RECORD
- | F1=Help      F3=Exit      F4=Name
- | F7=Backward  F8=Forward    F12=Cancel
- |
- | File  644  HERING  tcpip.boulder.info
- | File  644  HERING  tcpip.de.info
- | File  644  HERING  tcpip.emea.info
- | File  644  HERING  tcpip.info
- | Dir   755  HERING  test
- | Dir   755  HERING  TEST
- | File  644  HERING  test.ascii
- | Dir   755  HERING  test.dircache
Command ==>>
  F3=Exit      F4=Name      F5=Retrieve  F6=Keyshelp  F10=SwUID    F11=Command

```

Figure 16-18 ISHELL showing file format record if set

Using the Change File Format panel, record file format can be set for z/OS UNIX files. This is shown in Figure 16-19 on page 283.



```

File Directory Special_file Commands Help
-----
Edit Help
-----
S          Display File Attributes
a
w Pathname : /u/hering/record.file3
n
E File typ |          Change File Format
  Permissi |
  Access c | Change the file format and press Enter:
  File siz | 9  1. NA
  File own | 2. Binary
  Group ow | 3. NL
  Last mod | 4. CR
  Last cha | 5. LF
  Last acc | 6. CRLF
  Created  | 7. LFCR
  Link cou | 8. CRNL
  F1=Help  | 9. RECORD
  F7=Back  | F1=Help    F3=Exit    F6=Keyshelp
          | F12=Cancel
-----
File 644
File 644 HERING tcpip.boulder.info
File 644 HERING tcpip.de.info
File 644 HERING tcpip.emea.info
File 644 HERING tcpip.info
Dir 755 HERING test
Dir 755 HERING TEST
File 644 HERING test.ascii
Command ==>
F3=Exit   F4=Name   F5=Retrieve F6=Keyshelp F10=SwUID  F11=Command
-----

```

Figure 16-19 ISHELL setting file format record

### 16.6.4 Message update

The FOMF0301I message has been updated to show the new value REC for option -F. This is shown in Figure 16-20 on page 283.

```

$> echo System $(sysvar SYSNAME) running $(uname -I) Version $(uname -Iv).$(uname -Ir)
System SC74 running z/OS Version 01.12.00
$> extattr
FOMF0301I Usage: extattr [+a1ps] [-a1ps] [-F NA|BIN|NL|CR|LF|CRLF|LFCR|CRNL|REC] file
...
$>

```

Figure 16-20 Updated message FOMF0301I

### 16.6.5 Exploiting FILEDATA=RECORD with correct data contents

Record file data consists of records with prefixes. The record prefix contains the length of the record that follows. In Figure 16-21 on page 284 we show a sample job with JCL using FILEDATA=RECORD.

```

//TSOBATCH EXEC PGM=IKJEFT01
//SYSEXEC DD DSN=&SYSUID..ZFS.REXX.EXEC,DISP=SHR
//SYSTSPRT DD SYSOUT=*,LRECL=136,RECFM=VB
//FDRECORD DD PATH='/u/hering/record.data',FILEDATA=RECORD,RECFM=VB,
//          PATHOPTS=(OWRONLY,OCREAT,OTRUNC),
//          PATHMODE=(SIRUSR,SIWUSR,SIRGRP,SIROTH)
//SYSTSIN DD DATA,DLM=##
rexx
fdln.1 = Copies("a",10)
fdln.2 = Copies("b",5)
"EXECIO 2 DISKW FDRECORD (STEM FDLN. FINIS"
exit rc
##

```

Figure 16-21 JCL using FILEDATA=RECORD for USS PATH DD name

After running this job, the commands shown in Figure 16-22 demonstrate the results in USS.

```

$> ls -H record.data
-rw-r--r-- rec 1 HERING TTY 23 Mai 11 10:16 record.data
$> cat record.data | od -txLL
0000000000 0000000A81818181 8181818181810000
0000000020 0005828282828200
0000000027
$>

```

Figure 16-22 Display information and contents of file record.file

We see the following results:

- ▶ The file format record is set.
- ▶ Each record consists of a 4-byte length prefix followed by the line data.

### Using REXX with BPXWDYN() and EXECIO

By allocating USS files with Bpxwdyn(), for example, you can then use EXECIO to get or put data from or to such record files. An example is shown in Figure 16-23.

```

$> rexx
SH> call bpxwdyn "alloc dd(testrec) filedata(text) recfm(v,b) -
> path('/u/hering/record.data') pathopts(ordonly) msg(2)"; say result
0
SH> address mvs "EXECIO * DISKR TESTREC (STEM LN. FINIS"
SH> do i=1 to ln.0; say ln.i; end i
aaaaaaaaaa
bbbbbb
SH> call bpxwdyn "free dd(testrec)"
SH> exit
$>

```

Figure 16-23 Using Bpxwdyn() and EXECIO from REXX

### Support in OEDIT and OBROWSE

There is also support available for editing and browsing USS files.

**Note:** The new ISPF-based commands OEDIT and OBROWSE support the file format record.

If the file contents have not been created accordingly, an I/O error is received. This is shown in Figure 16-24 and Figure 16-25.

```
$> ls -H record.wrong
ls: FSUM6785 File or directory "record.wrong" is not found
$> echo "aaaaaaaaa\nbbbb" > record.wrong
$> extattr -F REC record.wrong
$> ls -H record.wrong
-rw-r--r-- rec 1 HERING TTY 17 Mai 11 11:09 record.wrong
$> cat record.wrong | od -txLL
0000000000 8181818181818181 8181158282828282
0000000020 1500000000000000
0000000021
$> oedit record.wrong
$>
```

Figure 16-24 Editing a standard USS file with file format record

The corresponding error messages are displayed in Figure 16-25.

```
/u/hering/record.wrong
ISRE904 I/O error occurred reading the edit data. Check data set block sizes.
```

Figure 16-25 Error messages on editing a standard USS file with file format record set

### 16.6.6 Migration and coexistence consideration

If a file with the record file format set is saved to a release prior to V1R12 (using pax), then the format will not be set as part of the restore operation on that earlier release.

## 16.7 z/OS UNIX dynamic socket limit for USS

In the past, once they were established, AF\_UNIX MAXSOCKETS parmlib value and Common Inet reserved port range (defined with the INADDRANYCOUNT parmlib value) could not be dynamically increased without a re-IPL or a shutdown and restart of z/OS UNIX. The maximum allowable number of Common Inet reserved ports was 4,000.

With z/OS V1R12, the following enhancements have been implemented:

- ▶ The maximum number of open AF\_UNIX sockets allowed is 10,000 and it is no longer required to specify MAXSOCKETS on the NETWORK statement for AF\_UNIX.
- ▶ The Common Inet reserved port range can now be dynamically increased, and the maximum number of reserved ports has been increased from 4,000 to 8,000.

**Important:** The new support avoids an IPL or an OMVS shutdown and restart to increase the Common Inet reserved port range or the AF\_UNIX maximum sockets. It also provides a larger maximum number of Common Inet reserved ports.

## 16.7.1 AF\_UNIX MAXSOCKETS

The value of MAXSOCKETS on the AF\_UNIX NETWORK statement in BPXPRMxx was the maximum number of open AF\_UNIX sockets allowed in the system. This was set at the time that the first AF\_UNIX NETWORK parmlib statement was processed, either during initialization, or on a SET OMVS=xx or SETOMVS RESET=xx command.

For AF\_UNIX, MAXSOCKETS could be set to a value up to 10,000, but once the NETWORK statement was processed, the MAXSOCKETS value could not be dynamically increased without a re-IPL or MODIFY OMVS,SHUTDOWN and MODIFY OMVS,RESTART.

### New function

The maximum number of open AF\_UNIX sockets allowed will now be 10,000 always. It is no longer required to specify MAXSOCKETS on the NETWORK statement for AF\_UNIX.

Specification of MAXSOCKETS is still allowed on the NETWORK parmlib statement for AF\_UNIX, but will be ignored. If it was specified with a value other than 10,000, the following informational message will be issued:

```
BPXT002I THE MAXSOCKETS VALUE FOR AF_UNIX HAS BEEN SET TO 10000
```

No message will be issued if MAXSOCKETS was not specified or if was specified as 10,000.

## 16.7.2 Common Inet reserved ports

The INADDRANYCOUNT value on the NETWORK parmlib statement for CINET (Common Inet) defined the number of reserved ports. It was set at the time that the first CINET NETWORK parmlib statement was processed, either during initialization, or on a SET OMVS=xx or SETOMVS RESET=xx command.

Port numbers are assigned from the reserved port range on port 0, INADDR\_ANY binds. It was not possible to increase the number of reserved ports dynamically. When all reserved ports were in use, future binds to port 0, INADDR\_ANY started failing with return and reason code set as follows:

```
errno=EADDRNOTAVAIL, errnojr=JrNoReservedPorts
```

Figure 16-26 shows a typical NETWORK statement for Common Inet.

```
NETWORK DOMAINNAME(AF_INET) DOMAINNUMBER(2) MAXSOCKETS(5000) TYPE(CINET)
INADDRANYPORT(5000) INADDRANYCOUNT(2000)
```

Figure 16-26 Typical NETWORK statement in a BPXPRMxx member

This establishes 2,000 reserved ports starting at port number 5,000.

### Dynamic increase of reserved ports

The support is invoked with the system command SET OMVS=xx or SETOMVS RESET=xx. In the command xx is the suffix of a BPXPRMxx parmlib member. This member needs to contain a NETWORK statement, as shown in Figure 16-27 on page 287, with INADDRANYCOUNT being greater than the current value.

```
NETWORK DOMAINNAME(AF_INET) DOMAINNUMBER(2) MAXSOCKETS(5000) TYPE(CINET)
INADDRANYPORT(5000) INADDRANYCOUNT(3000)
```

Figure 16-27 NETWORK statement in a BPXPRMxx member to increase INADDRANYCOUNT

This defines 3,000 reserved ports. The INADDRANYCOUNT can be increased any number of times up to the new maximum value of 8,000.

### New message

If the new value for INADDRANYCOUNT is smaller than the old one, an informational message is displayed.

**Note:** There is no support for decreasing INADDRANYCOUNT. If the value is less than the current value, the informational message BPXF202I is issued.

Message BPXF202I is shown in Figure 16-28.

```
BPXF202I DOMAIN AF_INET WAS NOT ACTIVATED FOR FILE SYSTEM 420
TYPE CINET. RETURN CODE = 00000079, REASON CODE = 12C60636
```

Figure 16-28 Message BPXF202I

The return code in message BPXF202I is EINVAL, the reason code is JrCannotDecrease.

## 16.8 Remove USS support for zFS multifile system aggregates

The zFS strategic direction is to completely remove its multifile system aggregate support.

In z/OS V1R12, z/OS UNIX is removing all existing support that displays or enables users to manipulate file systems in a multifile system aggregate in preparation for zFS to remove this support in a future release.

**Note:** z/OS UNIX is now well prepared for the situation when zFS makes the change.

### 16.8.1 Changes made in z/OS V1R12

Changes are mainly made to some messages and ISHELL panels.

#### Changed message BPXO045I

Figure 16-29 on page 288 shows the output of the command DISPLAY OMVS,FILE in a z/OS V1R11 system.

```

D OMVS,F,N=HERING.TEST.ZFS
BPX0045I 16.05.43 DISPLAY OMVS 354
OMVS      0011 ACTIVE          OMVS=(1A)
TYPENAME  DEVICE  -----STATUS-----  MODE  MOUNTED  LATCHES
ZFS       695 ACTIVE          RDWR  05/02/2010  L=189
        NAME=HERING.TEST.ZFS          21.47.51  Q=189
        PATH=/u/hering/test
AGGREGATE NAME=HERING.TEST.ZFS
OWNER=SC70      AUTOMOVE=Y CLIENT=N

```

Figure 16-29 DISPLAY OMVS,FILE in z/OS V1R11

In z/OS V1R12, this looks as shown in Figure 16-30.

```

D OMVS,F,N=HERING.TEST.ZFS
BPX0045I 16.09.43 DISPLAY OMVS 782
OMVS      0010 ACTIVE          OMVS=(RH)
TYPENAME  DEVICE  -----STATUS-----  MODE  MOUNTED  LATCHES
ZFS       196 ACTIVE          RDWR  05/13/2010  L=71
        NAME=HERING.TEST.ZFS          16.00.21  Q=0
        PATH=/u/hering/test
OWNER=SC74      AUTOMOVE=Y CLIENT=N

```

Figure 16-30 DISPLAY OMVS,FILE in z/OS V1R12

The aggregate line shown in red for z/OS V1R11 in Figure 16-31 has been removed in z/OS V1R12.

### Changed message BPXF035I

The output of the following command has been changed in z/OS V1R12 as well.

```
MODIFY BPX0INIT,FILESYS=DISPLAY,[ALL | FILESYSTEM=file_system_name]
```

Figure 16-31 shows an example as it is displayed in z/OS V1R11.

```

F BPX0INIT,FILESYS=DISPLAY,FILESYSTEM=HERING.TEST.ZFS
BPXM027I COMMAND ACCEPTED.
BPXF040I MODIFY BPX0INIT,FILESYS PROCESSING IS COMPLETE.
BPXF035I 2010/05/13 16.40.51 MODIFY BPX0INIT,FILESYS=DISPLAY 382
-----NAME-----  DEVICE  MODE
HERING.TEST.ZFS          695  RDWR
AGGREGATE NAME=HERING.TEST.ZFS
PATH=/u/hering/test
STATUS=ACTIVE          LOCAL STATUS=ACTIVE
OWNER=SC70      RECOVERY OWNER=SC70      AUTOMOVE=Y PFSMOVE=Y
TYPENAME=ZFS      MOUNTPPOINT DEVICE=      687
MOUNTPPOINT FILESYSTEM=OMVS.HERING.HFS
ENTRY FLAGS=90074000  FLAGS=40000000  LFSFLAGS=00000000
LOCAL FLAGS=40000000  LOCAL LFSFLAGS=20000000
ACTIVECHK =00000000  LFSFLAGS2      =D8000000

```

Figure 16-31 MODIFY BPX0INIT,FILESYS=DISPLAY in z/OS V1R11

In Figure 16-32 on page 289 the modified output in z/OS V1R12 is shown.

```
F BPXOINIT,FILESYS=DISPLAY,FILESYSTEM=HERING.TEST.ZFS
BPXM027I COMMAND ACCEPTED.
BPXF040I MODIFY BPXOINIT,FILESYS PROCESSING IS COMPLETE.
BPXF035I 2010/05/13 16.40.03 MODIFY BPXOINIT,FILESYS=DISPLAY 786
-----NAME-----
HERING.TEST.ZFS                                196  RDWR
  PATH=/u/hering/test
  STATUS=ACTIVE                                LOCAL STATUS=ACTIVE
  OWNER=SC74      RECOVERY OWNER=SC74      AUTOMOVE=Y  PFSMOVE=Y
  TYPENAME=ZFS    MOUNTPOINT DEVICE=      179
  MOUNTPOINT FILESYSTEM=HERING.ZFS
  ENTRY FLAGS=90074000  FLAGS=40000000  LFSFLAGS=00000000
  LOCAL FLAGS=40000000  LOCAL LFSFLAGS=22000000
  ACTIVECHK  =00000000  LFSFLAGS2    =D8000000
```

Figure 16-32 MODIFY BPXOINIT,FILESYS=DISPLAY in z/OS V1R12

Again, the aggregate line shown in red for z/OS V1R11 in Figure 16-31 on page 288 has been removed in z/OS V1R12.

### Interfaces and functions not changed

API interfaces and functions are not changed, as follows:

- ▶ BPX1GMN (w\_getmntent) still returns the zFS aggregate name pointed to by MNTENTAGGNAMEOFFSET.
- ▶ BPXEKDA continues to return the zFS aggregate name in ODMVAGGNAME.

## 16.8.2 ISHELL panel changes

Several ISHELL panels have been changed in z/OS V1R12 by removing information about the aggregate name or performing specific actions regarding aggregates. We provide here only the new panel views.

### ISHELL display of file system attributes

Figure 16-33 on page 290 shows the new information displayed for a zFS file system in z/OS V1R12.

```

Work with Mounted File Systems

S
U
  File System Attributes
  File system name:
  HERING.TEST.ZFS
  Mount point:
  /u/hering/test
  More:      +
  Status . . . . . : Available
  File system type . . . : ZFS
  Mount mode . . . . . : R/W
  Device number . . . . . : 196
  Type number . . . . . : 1
  DD name . . . . . :
  Block size . . . . . : 1024
  Total blocks . . . . . : 35280
  Available blocks . . . . : 49
  Blocks in use . . . . . : 35231
  F1=Help      F3=Exit      F4=Name      F6=Keyshelp
  F12=Cancel
  S
  S
HFS  SC74      No  /SC74/etc      OMVS.SC74.ETC
ZFS  SC74      No  /SC74/var/tdssrv1  OMVS.SC74.TDSSRV1.HFS
HFS  SC74      No  /SC74/var      OMVS.SC74.VAR
ZFS  SC74      No  /SC74/WebSphereTP1/V  OMVS.SC74.WAS61TP1.CONFIG.HFS
ZFS  SC74      Yes /Z1CRB1/usr/lpp/java  OMVS.ZOSR1C.Z1CRB1.JAVA31V5
ZFS  SC74      Yes /Z1CRB1/usr/lpp/java  OMVS.ZOSR1C.Z1CRB1.JAVA31V6
Command ==>
F3=Exit      F4=Name      F5=Retrieve  F6=Keyshelp  F10=SwUID    F11=Command

```

Figure 16-33 ISHELL display of file system attributes in z/OS V1R12

The lines with the aggregate name have been removed.

### ISHELL attached zFS aggregates

Figure 16-34 on page 291 shows the new ISHELL information displayed for all zFS aggregates that are attached in z/OS V1R12.



```

Attached zFS Aggregates                               Row 1 to 23 of 45

Select an aggregate with a line command.
A=Attributes  L=List file systems  E=Extend

S  Aggregate Name                                Free Space  Total Space
-  BBA.SBBAZFS.D080505                            149359      1728000
-  BB16174.SBBOHFS                                136969      2347200
-  BIN.ZFS                                          1101        2880
-  BIRD.HFS                                         7035        7200
-  CESAR.HFS                                        7034        7200
-  CFZADM.HFS                                       7031        7200
-  HAIMO.ZFS                                        6966        7200
-  HERING.TEST.ZFS                                 49          35280
-  HERING.ZFS                                       2404        87840
-  HEUSER.HFS                                       7035        7200
-  JES2.ZFS                                         7038        7200
-  LAFITTE.HFS                                      7034        7200
-  LUIZ.HFS                                         7035        7200
-  OMVS.DB2V9.D080325.HFS                          3047        7200
-  OMVS.PP.HFS                                      3432        3600
-  OMVS.SC74.TDSSRV1.HFS                           16538       18000
-  OMVS.SC74.WAS61TP1.CONFIG.HFS                   152415     699840
-  OMVS.ZOSR1C.Z1CRB1.JAVA31V5                     10486     283680
-  OMVS.ZOSR1C.Z1CRB1.JAVA31V6                     16263     408240
-  OMVS.ZOSR1C.Z1CRB1.JAVA64V5                     10086     268560
-  OMVS.ZOSR1C.Z1CRB1.JAVA64V6                     17004     455760
-  OMVS.ZOSR1C.Z1CRB1.ROOT                          93452     2260800
-  OMVS.ZOSR1C.Z1CRB1.SBBN7ZFS                     60949     1728000

Command ==>>> _____ Scroll ==>>> CSR
F1=Help      F3=Exit      F6=Keyshelp F12=Cancel

```

Figure 16-34 ISHELL attached zFS aggregates z/OS V1R12

All the options for attaching, detaching or creating an aggregate have been removed.

### ISHELL file system list

Figure 16-35 on page 292 shows the new information displayed for the zFS file systems in a zFS aggregate in z/OS V1R12.

```

Attached zFS Aggregates                                Row 1 to 23 of 45
S |-----|
A |           File System List                          Row 1 to 1 of 1
S |           Select a file system with a line command.
  | A=Attributes C=Clone D=Delete
- |
- | S   File System Name                                Space Used      Quota
- |   _  HERING.TEST.ZFS                                35086           35280
- |   ***** Bottom of data *****
- |
- | Command ==> _____ Scroll ==> CSR
- | F1=Help      F3=Exit      F6=Keyshelp  F12=Cancel
- |-----|
- | OMVS.ZOSR1C.Z1CRB1.JAVA64V5                          10086           268560
- | OMVS.ZOSR1C.Z1CRB1.JAVA64V6                          17004           455760
- | OMVS.ZOSR1C.Z1CRB1.ROOT                              93452           2260800
- | OMVS.ZOSR1C.Z1CRB1.SBBN7ZFS                          60949           1728000
- |-----|
- | Command ==> _____ Scroll ==> CSR
- | F1=Help      F3=Exit      F6=Keyshelp  F12=Cancel

```

Figure 16-35 ISHELL file system list z/OS V1R12

All the options to create, rename, or set the quota of a file system from this panel have been removed.

## 16.9 USS shell and utility `tsocmd`

The utility `tsocmd` was previously provided as a Tools and Toys item. In z/OS V1R12, a fully supported version is shipped in the base release.

The utility is very similar to the existing `tso` command and allows to run a TSO/E command from the shell, including authorized TSO/E commands.

**Important:** Command `tso` cannot run authorized TSO/E commands.

The new base release `tsocmd` is documented in *z/OS V1R12 Unix System Services Command Reference*.

### 16.9.1 Using `tsocmd`

The `tsocmd` utility can be called from a shell script or shell prompt. It runs a TSO/E command from the shell using the TSO/E terminal monitor program (TMP), which is IKJEFT01.

**Important:** When you run `tsocmd` from a UNIX process, the TSO/E TMP is run in a separate address space and the TSO/E commands that are issued do not affect the shell environment that the `tsocmd` is issued from.

## New tsocmd usage message

If **tsocmd** is entered with no command, a usage message appears, as shown in Figure 16-36. This is new and was not done in the former Tools and Toys version.

```
$> whence tsocmd.tat
/usr/local/bin/tsocmd.tat
$> tsocmd.tat

READY
END
$> whence tsocmd
/bin/tsocmd
$> tsocmd
Usage: tsocmd [tsocommand]
$>
```

Figure 16-36 New tsocmd usage message

In the sample commands shown in Figure 16-37, **tsocmd.tat** is the name chosen for the old Tools and Toys version located in directory `/usr/local/bin`.

## tsocmd behavior

If the issued TSO/E command fails, **tsocmd** will return an exit value between 1 and 255, depending on the failure. We show an example in Figure 16-37.

```
$> tso -t "invalid"; echo $?
invalid
FOMF0136I Command not found
255
$> tsocmd.tat "invalid"; echo $?
invalid
RC(12)
COMMAND INVALID NOT FOUND
READY
END
12
$> tsocmd "invalid"; echo $?
invalid
FSUMB451 tsocmd: TSO/E command "invalid" not found.
255
$>
```

Figure 16-37 Exit values provided with tsocmd

See *z/OS UNIX System Services Command Reference*, SA22-7802 for more information about possible failures.

The following environment variables can be set to help with TSO/E commands that require these settings. These are also currently supported on the **tso** command.

- ▶ SYSEXEC
- ▶ SYSPROC
- ▶ TSOALLOC
- ▶ TSOPROFILE

For more details about using these environment variables and the `tsocmd` utility in general, see Chapter 2, “Shell command descriptions” in *z/OS UNIX System Services Command Reference*, SA22-7802.

### Authorized TSO commands

Figure 16-38 shows an example of issuing an authorized TSO command.

```
$> tsocmd.tat "lu" | tail -3
lu
SECURITY-LABEL=NONE SPECIFIED
READY
END
$> tsocmd "lu" | tail -3
lu
CATEGORY-AUTHORIZATION
  NONE SPECIFIED
SECURITY-LABEL=NONE SPECIFIED
$>
```

Figure 16-38 Issuing an authorized TSO command

The last two lines of the Tools and Toys version have been removed in the new `tsocmd` utility.

**Note:** When using `tsocmd`, we suggest that the TSO/E command and its parameters be enclosed in quotes (“”) to avoid shell parsing.

### Differences between the old and new `tsocmd` utility

The new base release version of `tsocmd` differs from the Tools and Toys version in the following ways:

- ▶ Exit values are now consistently sent if the issued TSO/E command fails. Previously, in some cases a failure was not even indicated.
- ▶ Environment variables `tsoin` and `tsoout` are no longer supported. Instead `stdin` and `stdout` are used, as is done for most other shell commands.
- ▶ Environment variable `TSOPROFILE` is now supported to match the existing `tso` command support.

## 16.9.2 Installation and migration

There are some installation and migration actions to be taken into account when you want to use the new base `tsocmd` version:

- ▶ If the Tools and Toys version is not currently used or installed, then no migration actions are required.
- ▶ If the Tools and Toys version is currently used, determine whether the command is in `/bin` or some other directory, and determine whether you want to preserve the Tools and Toys version.
- ▶ If you want to preserve the Tools and Toys version, make sure that the command is not located in `/bin` prior to the installation of z/OS V1R12.
- ▶ If you do not want to preserve the Tools and Toys version, and it is located in `/bin`, then the installation of z/OS V1R12 will automatically replace the Tools and Toys version with the

new officially supported version. If the Tools and Toys version is not located in `/bin`, remove it from its current location.

- ▶ In case of not keeping it or if you are not using it directly from UNIX, you need to remove the Tools and Toys related information from the authorized load library where you have placed it, for example `hlq.TSOCMD.LOADLIB` or `SYS1.LINKLIB`. A description is available in the `tsocmd` Tools and Toys README documentation at this website:

<ftp://ftp.software.ibm.com/s390/zos/tools/tsocmd/tsocmd.readme.txt>

**Note:** A new version of `tsocmd` that matches the version shipped with z/OS V1R12 will be made available on the Tools and Toys website.

If you have multiple systems at different releases and you want to have the same version of the tool on all releases, you can download the new version to the earlier systems and replace the previous Tools and Toys version of `tsocmd` if appropriate. If you do this, you also need to clean up the authorized load library, as described.

## 16.10 USS support for sysplex-aware on a file system basis

Sysplex awareness could only be determined by a PFS on a system basis during its initialization. The new zFS SPE OA29619 requires the capability of determining R/W sysplex awareness on a file system basis. The zFS SPE is discussed in detail in “zFS sysplex-aware on a file system basis” on page 297.

- ▶ The USS SPE OA29712 is a prerequisite for the zFS SPE OA29619. However, the USS SPE does not require the zFS SPE. It has no prerequisites or corequisites.
- ▶ This USS SPE is for z/OS V1R11 only.
- ▶ There are no USS toleration PTFs for any other releases.

### 16.10.1 Minor background explanation

Here we provide some minor hints about mount behavior in a USS sysplex sharing environment.

- ▶ In case of a sysplex-aware mount, all the catchup mounts on the non-owner systems are directed to the native PFS and the file systems are locally mounted on all systems.
- ▶ If the mount is sysplex-unaware, the catchup mounts are directed to the Cross System PFS (XPFS) and USS function-ships to the owner.
- ▶ After a successful `vn_lookup`, the name and output vnode pointer (`VnodPtr`) are stored in the lookaside cache (LLA) on the parent directory vnode. This avoids a `vn_lookup` call to the PFS on the next lookup of the same name.

**Note:** If the file is deleted or renamed by any system, all systems must remove the entry from their cache.

### 16.10.2 USS SPE OA29712

In order to be able to handle the new functions introduced for zFS, USS needs to handle and maintain additional PFS flags and change function processing in several situations. If the SPE is applied, the new LFS version is 1.11.10; refer to Table 16-1 on page 296.

**Note:** Modification level 10 indicates that the z/OS V1R11 system has applied the USS SPE.

Systems with the USS SPE check whether the owner system has the USS SPE active to determine and check for the additional PFS flags.

*Table 16-1 USS support for sysplex-aware on a file system basis*

z/OS release (USS release)	PTF number	APAR number
z/OS V1R11 (760)	UA53117	OA29712

### Changes in USS

USS functions affected are those that issue `vfs_mount`:

- ▶ Mount processing
- ▶ Catchup mount processing
- ▶ Remount processing with mode-switch or same-mode
- ▶ Move processing
- ▶ Dead system takeover
- ▶ Blackhole takeover
- ▶ Blackhole catchup mount processing
- ▶ PFS termination and restart processing
- ▶ Mounting a new root or processing setup of an alternate root

### 16.10.3 Information about F BPXOINIT display command

The output of commands `F BPXOINIT,FILESYS=DISPLAY,FILESYSTEM=` provides diagnostic information that especially shows the local and global state of sysplex-awareness.

**Note:** We do not go down to single bits here, but you find more details in 16.12.4, “zFS sysplex-aware on a file system basis” on page 310 because these necessary new zFS functions explained there were the reason for having to provide the changes in the USS implementation.

With the USS SPE applied, the new LFSFLAGS2 flags shown in the example of Figure 16-39 on page 297 are typical for the case of an R/W mounted sysplex-aware zFS file system. In case of an R/W mounted HFS these bits could be shown as `x00003000`, for example.

```

$> cn f bpxoinit,filesys=display | grep $(sysvar SYSNAME)
SC70      1. 11. 10 VERIFIED                NONE
$> zfsowner hering.test.zfs
zFS Owner   : SC70      - Aggregate read-only=N, compat-mode=Y, sysplex-aware=Y
$> cn f bpxoinit,filesys=display,filesystem=hering.test.zfs
F BPXOINIT,FILESYS=DISPLAY,FILESYSTEM=HERING.TEST.ZFS
BPXM027I COMMAND ACCEPTED.
IEF196I IEF285I  SYS1.LINKLIB                KEPT
IEF196I IEF285I  VOL SER NOS= Z1BRC1.
BPXF040I MODIFY BPXOINIT,FILESYS PROCESSING IS COMPLETE.
BPXF035I 2010/05/18 18.16.25 MODIFY BPXOINIT,FILESYS=DISPLAY
-----NAME-----
HERING.TEST.ZFS                695 RDWR
AGGREGATE NAME=HERING.TEST.ZFS
PATH=/u/hering/test
STATUS=ACTIVE                  LOCAL STATUS=ACTIVE
OWNER=SC70                     RECOVERY OWNER=SC70     AUTOMOVE=Y PFSMOVE=Y
TYPENAME=ZFS                   MOUNTPOINT DEVICE=      687
MOUNTPOINT FILESYSTEM=OMVS.HERING.HFS
ENTRY FLAGS=90074000  FLAGS=40000000  LFSFLAGS=00000000
LOCAL FLAGS=40000000  LOCAL LFSFLAGS=20000000
ACTIVECHK  =00000000  LFSFLAGS2   =D8000000
$>

```

Figure 16-39 Diagnostic information for an R/W mounted sysplex-aware zFS file system

## 16.10.4 Message BPXF221I

With the base code of z/OS V1R11, whenever a non-owning system is running sysplex=on and the owner is sysplex=off, a catchup mount to the zFS PFS locally is always attempted. However, if the owner is running sysplex=off, the mount fails, message BPXF221I, as shown in Figure 16-40, is issued and the mount is directed to XPFS.

```

BPXF221I FILE SYSTEM name FAILED TO MOUNT LOCALLY RETURN CODE =xxxxxxx,
REASON CODE = yyyyyyy. THE FILE SYSTEM IS ACCESSIBLE ON THIS
SYSTEM THROUGH A MOUNT ON A REMOTE SYSTEM.

```

Figure 16-40 Message BPXF221I

With this USS SPE applied, the owner sets a flag showing to be sysplex-unaware, and no local catchup mount is attempted from a non-owning system. Therefore, fewer BPXF221I messages should be issued.

**Note:** The message can still be issued if a local mount fails, due to no DASD connectivity or some other mount failure.

## 16.11 zFS sysplex-aware on a file system basis

Beginning with z/OS V1R11, zFS introduced the ability to enable zFS read-write file systems to be sysplex-aware. This means that USS sends all file requests directly down to the local zFS physical file system (PFS). It does not “function ship” the requests. The local zFS either

sends the request to the owning zFS system or it satisfies the request from the local zFS cache. In many cases, this improves the path length over the function shipping model. The request and data flow are shown in Figure 16-42 on page 303.

### 16.11.1 zFS APAR OA29619

You could do this in one of the following two ways:

- ▶ You can choose individual file systems to be sysplex-aware. This requires zFS APAR OA29619, which is for both z/OS V1R11 and z/OS V1R12.
- ▶ You can choose to have all zFS file systems be sysplex-aware.

With the new support provided in APAR OA29619, several goals have been achieved by changing the support provided by z/OS V1R11, as follows:

- ▶ An environment has been provided where performance problems caused by mixed environments do not occur.
- ▶ The new possibilities allow the usage of zFS sysplex-aware file systems with servers that cannot tolerate sysplex-aware file systems, such as the z/OS SMB server and Fast Response Cache Accelerator support of the IBM HTTP Server in the same shared file system environment.
- ▶ It gives more granular control and flexibility over the decision to make zFS file systems sysplex-aware.

**Important:** Before you can run zFS sysplex-aware on a file system basis, you must have z/OS V1R11 zFS APAR OA29619 and z/OS V1R11 UNIX APAR OA29712 installed and active on all of your z/OS V1R11 systems. In addition, conditioning APAR OA29786 must be installed and active on your V1R10 systems. Finally, if you use the SMB server, APAR OA31112 must also be installed.

### zFS APAR OA29712

The zFS APAR OA29619 provides the capability of setting read-write sysplex-awareness on a file system (mount) basis, which requires the USS support provided with APAR OA29712.

With APAR OA29712, USS provides support for allowing a PFS to set the read-write sysplex-awareness of a file system during a mount. Prior to this support, the sysplex-awareness of a PFS that was set during PFS initialization would apply to all file systems mounted on that system for that PFS. This affects all flows where USS sends a `vfs_mount` to the PFS, which includes the following types of actions:

- ▶ Mount

A local mount means that z/OS UNIX issues a successful mount to the local PFS, which in this case is zFS. z/OS UNIX does this when either the file system is mounted sysplex-aware for that mode (read-write or read-only) or the system is the z/OS UNIX owner. When a file system is locally mounted on the system, z/OS UNIX does not function ship requests to the z/OS UNIX owning system.

- ▶ Catchup mount

When a file system mount is successful on a system in a shared file system environment, z/OS UNIX automatically issues a corresponding local mount, the catch-up mount, to every other system's PFS that is running sysplex-aware for that mode (read-write or read-only) when zFS is running `sysplex=on`, or for each zFS file system that is mounted `RWSHARE` when zFS is running `sysplex=filesystems`. If the corresponding local mount is successful, z/OS UNIX does not function ship from that system to the z/OS UNIX owning



system when that file system is accessed. Rather, the file request is sent directly to the local PFS. This is sometimes referred to as Client=N. If the corresponding local mount is unsuccessful (for instance, DASD is not accessible from that system), z/OS UNIX function ships requests to the z/OS UNIX owning system when that file system is accessed (message BPXF221I might be issued). This is sometimes referred to as Client=Y.

- ▶ Remount

Remount allows you to change a mounted file system from read-only to read-write or from read-write to read-only without affecting lower mounted file systems.

- ▶ Dead system recovery

When a system leaves the sysplex, use the recovery options for the mounted file systems.

- ▶ Unowned file system recovery or catchup

This can occur, for example, when a physical I/O path from another system to the volume where the file system resides is not available. As a result, the file system becomes unowned; if this happens, you will see message BPXF213E. This is true if the file system is mounted either read/write or read-only. The file system still exists in the file system hierarchy so that any dependent file systems that are owned by another system are still usable. However, all file operations for the unowned file system will fail until a new owner is established. The shared file system support will continue to attempt recovery of AUTOMOVE file systems on all systems in the sysplex that are enabled for shared file system. If a subsequent recovery attempt succeeds, the file system moves from the unowned to the active state.

- ▶ Newroot or altroot

Beginning with z/OS V1R11, in a sysplex configuration, the alternate sysplex root file system is a hot standby for the sysplex root file system that is used to replace the current sysplex root file system when the sysplex root file system becomes unowned. The alternate sysplex root file system is established by using the ALTROOT statement in the BPXPRMxx parmlib member during OMVS initialization or by using the SET OMVS command.

- ▶ PFS termination or restart

Move file systems to any system using the AUTOMOVE settings.

## 16.11.2 Background information about previous zFS behavior

First, it is necessary to provide some information about how zFS processed file systems in the past before z/OS V1R11 and with the base support in z/OS V1R11.

### 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.

### z/OS UNIX file system owner

The file system owner is the system to which file requests are forwarded. Having the appropriate owner is important for performance. We use the term z/OS UNIX file system owner to mean 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. We call this owner 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. The local zFS PFS forwards the request to the zFS file system owner, if necessary. The z/OS UNIX file system owner can be different from the zFS file system owner.

**Note:** 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.

### **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.

### **Local mount**

A local mount means that z/OS UNIX issues a successful mount to the local PFS, which in this case is zFS. z/OS UNIX does this when either the file system is mounted sysplex-aware for that mode (read-write or read-only) or the system is the z/OS UNIX owner. When a file system is locally mounted on the system, z/OS UNIX does not function ship requests to the z/OS UNIX owning system.

### **Non-sysplex aware (sysplex-unaware)**

A file system is non-sysplex aware (or sysplex-unaware) if the PFS 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. Other systems' access 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.

**Note:** If you decide to run zFS non-sysplex aware, shared file system support works as in prior releases to z/OS V1R11. Ensure that you do not specify `sysplex=on` or `sysplex=filesys` in your zFS IOEFSPRM configuration options file. See "New and older important zFS configuration options" on page 307.

### **Sysplex-aware PFS**

A physical file system (PFS), for example zFS, is sysplex-aware or non-sysplex aware for a particular mount mode (read-only or read-write) in a shared file system environment. When it is sysplex-aware, it means that the PFS is capable of handling a local mount on the system that is not the z/OS UNIX owning system. The PFS that is sysplex-aware can avoid z/OS UNIX function shipping for that mode. All physical file systems are always sysplex-aware for read-only mounts. HFS is always non-sysplex aware for read-write mounts and always results in z/OS UNIX function shipping from systems that are not the z/OS UNIX owning system. zFS can be sysplex-aware or non-sysplex aware for read-write mounts. The default for zFS is to be non-sysplex aware (zFS IOEFSPRM option `sysplex=off`) and therefore z/OS UNIX function shipping occurs from systems that are not the z/OS UNIX owning system. Beginning with

z/OS V1R11, zFS can be sysplex-aware for read-write mounted file systems. zFS can be configured (zFS IOEFSPRM option `sysplex=on`) to treat all zFS read-write mounted file systems owned on that system as sysplex-aware file systems or it can be configured (zFS IOEFSPRM option `sysplex=filesys`, which is new with APAR OA29619) to allow some zFS read-write mounted file systems owned on that system to be sysplex-aware file systems and some to be non-sysplex aware file systems. zFS must be running sysplex-aware for read-write in order to allow a zFS read-write file system to be mounted as sysplex-aware.

### **Sysplex-aware file system**

A file system can be mounted sysplex-aware or non-sysplex aware. When a file system is mounted sysplex-aware, it means that the file system is locally mounted on every system (when the PFS is capable of handling a local mount on every system, that is, the PFS is running sysplex-aware) and therefore, file requests are handled by the local PFS. All read-only mounted file systems are always mounted sysplex-aware. HFS read-write mounted file systems are always mounted non-sysplex aware. This means that file requests from non-z/OS UNIX owning systems are always function shipped by z/OS UNIX to the z/OS UNIX owning system where the file system is locally mounted and the I/O is actually done. Beginning with z/OS V1R11, zFS read-write mounted file systems can be mounted sysplex-aware when zFS is configured as sysplex-aware (zFS IOEFSPRM option `sysplex=on` or zFS IOEFSPRM option `sysplex=filesys`).

### **16.11.3 zFS R/W mounted file system being sysplex-unaware**

Figure 16-41 on page 302 shows an example for a R/W mounted zFS file system in z/OS V1R10 or older or in z/OS V1R11 when zFS has been started with `sysplex=off`. One system is the owning system, SY2, and the other systems are “clients”. The file system is only locally mounted on the owning system, but also externally mounted and available on all systems.

Applications running on the owning system (SY2) access the file system locally and without any XCF communications.

But applications that run on the other systems (SY1 and SY3), access the file system through the use of z/OS UNIX function shipping with the XPFS and using XCF communications. That is, the request is forwarded by the z/OS UNIX on that system, SY1 or SY3, to the z/OS UNIX running on the owning system SY2, which then uses the local zFS.

The response goes back along the same path. So, access to the file system from systems other than the owner, and they are called client systems, involves XCF communications.

**Note:** This makes it important to have the z/OS UNIX owning system be the system that is doing the most accesses. This file system is non-sysplex aware.

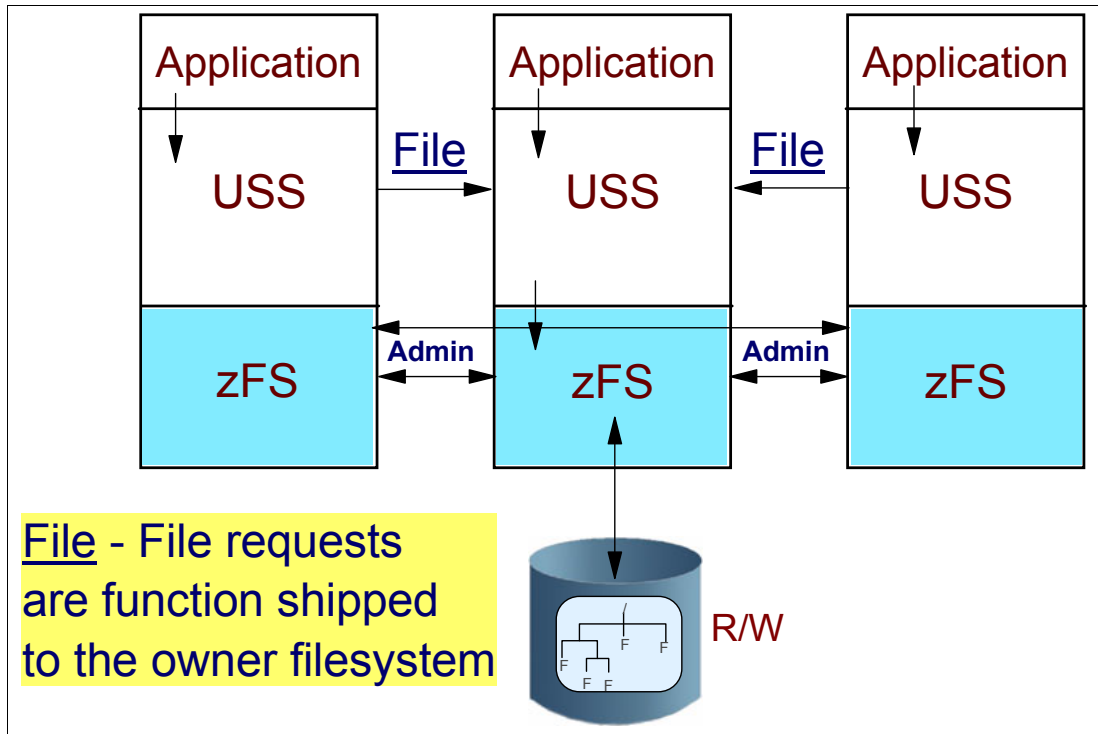


Figure 16-41 zFS R/W mounted file system being sysplex-unaware

### Running all systems with zFS sysplex-unaware

If you decide to run zFS non-sysplex aware, shared file system support works as in prior releases. Ensure that you do not specify `sysplex=on` or `sysplex=filesys` in your zFS IOEFSPRM configuration options file, as shown in “Setting up the zFS parameters” on page 308.

### 16.11.4 zFS being sysplex-aware in z/OS V1R11 for R/W mounts

Beginning with z/OS V1R11, zFS is sysplex-aware for file systems mounted read-write. This means that USS sends all file requests directly down to the local zFS physical file system (PFS). It does not “function ship” the requests. The local zFS either sends the request to the owning zFS system or it satisfies the request from the local zFS cache. In many cases, this improves the path length over the function shipping model. The request and data flow are shown in Figure 16-42 on page 303.

#### z/OS V1R11 supports `sysplex=on`

With z/OS V1R11, the zFS PFS allows a file system to be locally accessed on all systems in a sysplex for a particular mode, and then the PFS is sysplex aware for that mode. Therefore, performance can be improved for zFS in the USS sysplex shared file system mode by zFS becoming sysplex-aware for R/W mounted file systems. This is a new specification that is specified with a new option as follows:

```
sysplex=on
```

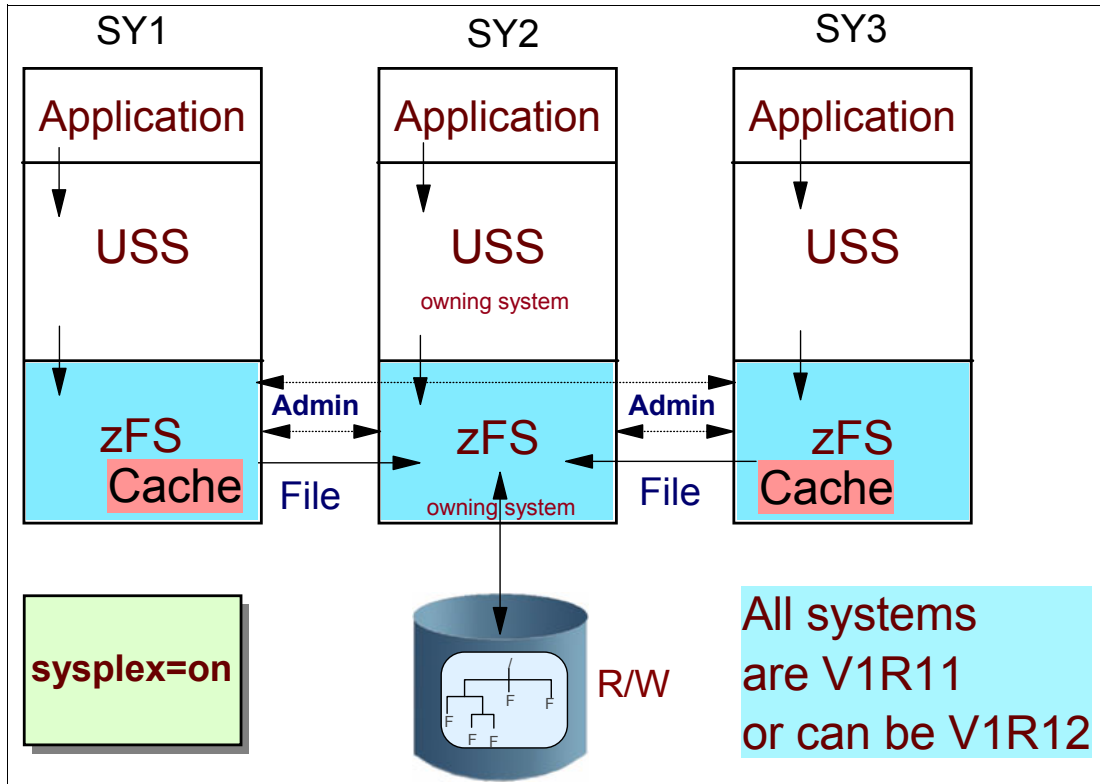


Figure 16-42 R/W mounted zFS sysplex-aware with zFS R11 started sysplex=on

**Note:** There is a new option available with APAR OA29619 for z/OS V1R11 and z/OS V1R12 systems, `sysplex=filesystem`; see “Setting up the zFS parameters” on page 308.

This zFS sysplex file system support improves the response time for zFS file system requests:

- ▶ zFS becomes sysplex-aware for zFS read-write file systems.
- ▶ zFS uses less XCF communications as a new client caching is introduced.

### z/OS V1R11 systems with `sysplex=on`

Figure 16-42 shows the z/OS V1R11 zFS sysplex-aware for read-write on a system basis support. When zFS runs sysplex-aware for read-write on all systems, a read-write mounted 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. The 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.

**Note:** If this was all that zFS did, it would be essentially the same as the z/OS UNIX function shipping shown in Figure 16-42.

However, each zFS client system has a local cache where it keeps the most recently read file system information. So, in many cases (when the data is still in the cache), zFS can avoid the zFS function shipping (and the XCF communications) and satisfy the request locally.

**Note:** Any zFS read-write file system owned on SY2 is sysplex-aware.

### zFS command output in z/OS V1R11 with zFS base support

Figure 16-43 shows zFS and MVS system commands with information about zFS aggregates.

```

$> zfsadm aggrinfo hering.test.zfs -long
HERING.TEST.ZFS (R/W COMP): 451 K free out of total 2880
version 1.4
auditfid E2C2D6E7 F1C3009C 0000
sysplex-aware
          52 free 8k blocks;          35 free 1K fragments
        112 K log file;              16 K filesystem table
          8 K bitmap file

$> cn f zfs,query,file
F ZFS,QUERY<FILE
IOEZ00438I Starting Query Command FILESETS.
File System Name                Aggr #  Flg  Operations
-----
...
HERING.TEST.ZFS                 83    AMS      3
...
IOEZ00025I zFS kernel: MODIFY command - QUERY,FILE completed successfully.
$> zfsowner hering.test.zfs
zFS Owner      : SC70      - Aggregate read-only=N, compat-mode=Y, sysplex-aware=Y
$>

```

Figure 16-43 zFS command output in z/OS V1R11 with zFS base support

All these commands provide the information that zFS aggregate HERING.TEST.ZFS is mounted sysplex-aware.

**Note:** For REXX commands **cn** and **zfsowner**, shown in Figure 16-43, see *z/OS Distributed File Service zSeries File System Implementation z/OS V1R11*, SG24-6580 (or later) for a detailed description.

### zFS API output in z/OS V1R11 with zFS base support

The zFS Application Programming Interface (API) command LIST AGGREGATE STATUS (Version 2) indicates when an aggregate is sysplex-aware:

```
AGGR_STATUS, as_flags2, 0x40 (AS_SYSPLEXAWARE)
```

**Note:** The LIST AGGREGATE STATUS subcommand call is an aggregate operation that returns information about a specified attached aggregate on this system. Version 2 returns additional flags and fields.

See *z/OS Distributed File Service zSeries File System Administration*, SC24-5989 for more information.

**Note:** This `as_flags2` bit shows the zFS aggregate sysplex-aware state independent of the zFS release and service level used.

The `zfsowner` utility uses this bit to decide whether a R/W mounted zFS aggregate is sysplex-aware.

### **zFS mixed environment**

Figure 16-44 on page 306 shows a zFS mixed environment with systems that are `sysplex=off` and `sysplex=on`.

With SY1, running `sysplex=off`, there are three problems with this mixed environment, as follows:

- ▶ Since zFS is running `sysplex=off`, SY1 cannot present requests for that file system to the local zFS. This is unused for that file system. This means that z/OS UNIX must forward requests to the z/OS UNIX owner, which is SY2 in this case.
- ▶ Furthermore, the z/OS UNIX directory lookup cache is not active on SY1 because the z/OS UNIX owner system SY2 is a `sysplex=on` system. The loss of this z/OS UNIX directory cache can increase the rate of transmissions to the z/OS UNIX owner SY2 for pathname resolution within the file system.
- ▶ Since the z/OS UNIX owner SY2 and the zFS owner SY3 are different, z/OS UNIX on SY1 forwards requests to the z/OS UNIX owner SY2. Then, z/OS UNIX calls zFS on system SY2. Since the zFS owner is on SY3, if the request is a write request or a read that has a cache-miss, a message is sent to SY3 by zFS to resolve the request. This means that two messages are sent in the sysplex for a user application request from SY1 instead of 1 or 0.

Thus a mixed environment (mixed environment with `sysplex=off` and `sysplex=on` systems) can lead to increased transmissions from `sysplex=off` members and a double transmit, both of which cannot occur in a non-mixed environment.

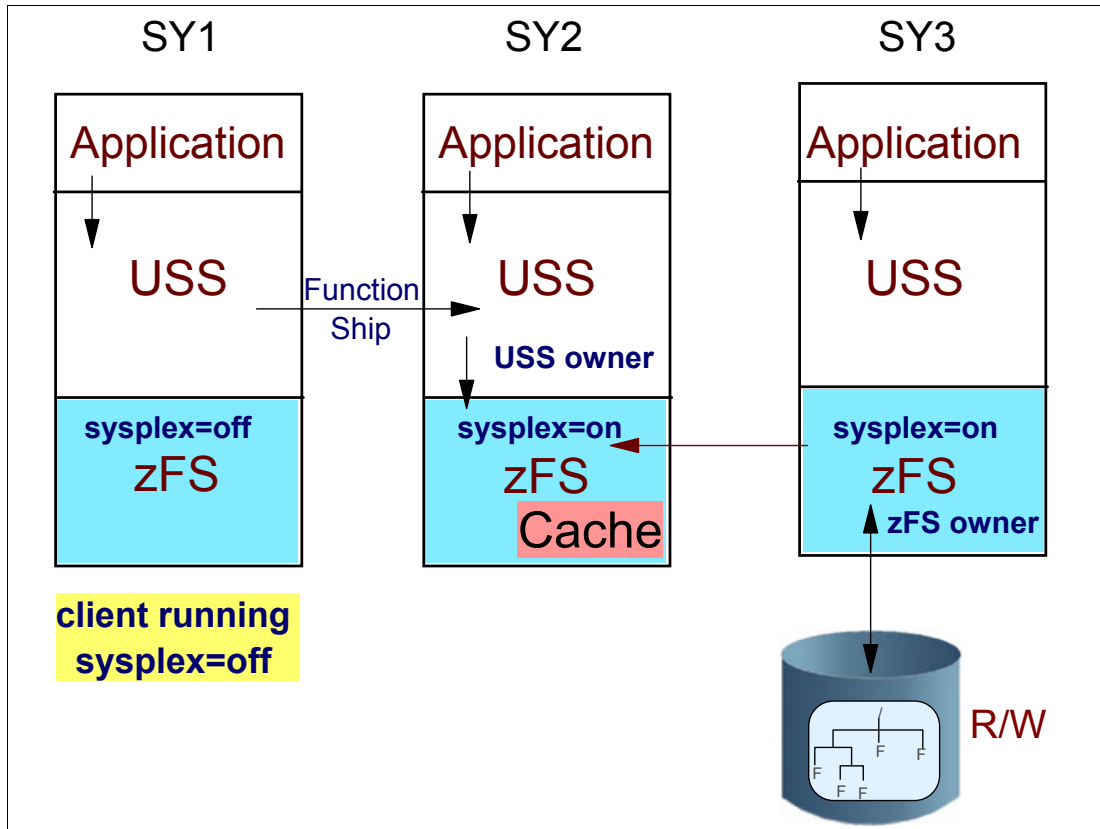


Figure 16-44 zFS mixed environment

## 16.12 New zFS support with APAR OA29619

APAR OA29619 provides a new function that allows you to specify which zFS read-write file systems are to be made sysplex-aware. To enable zFS sysplex-aware on a file system basis, specify `sysplex=filesys` in the IOEFSPRM configuration file. Leave the `sysplex_filesys_sharemode` with its default of `norwshare`. You can specify it in a shared IOEFSPRM configuration file and each system picks up the specification in a rolling IPL. The `sysplex` option is ignored by previous releases.

**Note:** Any z/OS V1R9 or V1R10 systems in the shared file system environment must have zFS APAR OA29786 active before you can have zFS `sysplex=filesys` active on any z/OS V1R11 system.

A physical file system (PFS), for example zFS, is sysplex-aware or non-sysplex aware for a particular mount mode (read-only or read-write) in a shared file system environment. When it is sysplex-aware, it means that the PFS is capable of handling a local mount on the system that is not the z/OS UNIX owning system. The PFS that is sysplex-aware can avoid z/OS UNIX function shipping for that mode. All physical file systems are always sysplex-aware for read-only mounts.

HFS is always non-sysplex aware for read-write mounts and always results in z/OS UNIX function shipping from systems that are not the z/OS UNIX owning system. zFS can be sysplex-aware or non-sysplex aware for read-write mounts. The default for zFS is to be



non-sysplex aware (zFS IOEFSPRM option sysplex=off) and therefore z/OS UNIX function shipping occurs from systems that are not the z/OS UNIX owning system.

## APAR OA29619

Beginning with z/OS V1R11 and with z/OS V1R12, zFS can be sysplex-aware for read-write mounted file systems because with APAR OA29619, zFS provides new support for being sysplex-aware on a file system basis; refer to Table 16-2.

Table 16-2 zFS sysplex-aware on a file system basis

z/OS release (zFS release)	PTF number	APAR number
z/OS V1R11 (3B0)	UA52930	OA29619

APAR OA29619 was introduced to enable zFS sysplex-aware on a file system basis, by specifying sysplex=filesys in the IOEFSPRM configuration file. Any z/OS V1R9 or V1R10 systems in the shared file system environment must have zFS APAR OA29786 active before you can have zFS sysplex=filesys active on any z/OS V1R11 system.

### 16.12.1 New and older important zFS configuration options

Following are new configuration options that can be defined either in the IOEPRMxx parmliib member or in the IOEFSPRM DD member in the ZFS procedure. These parameters are changed as indicated by z/OS V1R11 and with APAR OA29619.

- sysplex={on | off | filesys}** Specifies whether zFS should run sysplex-aware and if so, whether zFS is sysplex-aware on a file system basis (sysplex=filesys) or sysplex-aware on a system basis (sysplex=on). When sysplex=off, zFS does not automatically move zFS ownership of the aggregate. (Changed with APAR OA29619, filesys is new, see “Setting up the zFS parameters” on page 308).  
**Default Value:** off  
**Expected Value:** off, filesys, or on if BPXPRMxx specifies SYSPLEX(YES). Off if BPXPRMxx does not specify SYSPLEX(YES).  
**Example:** sysplex=on
- sysplex\_filesys\_sharemode** Specifies the default for the mount PARM for a zFS read-write file system mounted on a sysplex=filesys system. (New with APAR OA29619, see “Setting up the zFS parameters” on page 308).  
**Default Value:** norwshare when running zFS as sysplex=filesys or <no value> otherwise  
**Expected Value:** rwshare or norwshare  
**Example:** sysplex\_filesys\_sharemode=rwshare
- token\_cache\_size** Specifies the maximum number of tokens in the server token manager cache to use for cache consistency between zFS members. The default value is double the number of vnode\_cache\_size (new with z/OS V1R11).
- file\_threads = 40** The file\_threads configuration option determines how many threads are in the pool to handle requests from remote clients (new with z/OS V1R11).

**client\_cache\_size=128M**

The client\_cache\_size configuration option specifies the size of the user data cache for client access. The user\_cache\_size is used for the user data cache for server access (new with z/OS V1R11).

**client\_reply\_storage = 10M**

The client\_reply\_storage configuration option specifies how much storage is used for sysplex server replies (new with z/OS V1R11).

**Note:** With APAR OA29619, the default value is changed from 40 M to 10 M.

**recovery\_max\_storage=256M**

Indicates the maximum amount of zFS address space storage to use for concurrent log recovery during multiple concurrent aggregate mounts (new with z/OS V1R11).

**sysplex\_admin\_level**

Specifies the zFS XCF communication interface level that zFS is running as sysplex-aware. This is only valid for systems running zFS V1R9 or V1R10 with the proper APARs applied. The sysplex\_admin\_level option is ignored in z/OS V1R11 because zFS runs at sysplex\_admin\_level 3 on z/OS V1R11. One (1) indicates that zFS running on z/OS V1R9 or V1R10 is in preconditioning state and is using interface level 1. zFS uses the existing XCF protocol but also supports enough of the new XCF protocol for zFS interface level 2 to run properly. Two (2) indicates that zFS running on z/OS V1R9 or V1R10 is in toleration mode and is using interface level 2. zFS uses the new XCF protocol and is able to communicate with other zFS instances running at interface level 1 to display aggregate information for aggregates owned by zFS interface level 1 systems. At level 2, zFS is able to communicate with other zFS instances running at z/OS V1R11 (level 3). The value must be 2 on all members of the sysplex in order to bring z/OS V1R11 zFS into the shared file system environment.

**Default Value:** 3 in z/OS V1R11 and V1R12. 1 in z/OS V1R9 and V1R10.

**Expected Value:** In z/OS V1R11, this option is ignored. In z/OS V1R9 and V1R10, 1 or 2.

**Example:** sysplex\_admin\_level=2

**Note:** The sysplex\_admin\_level option is ignored in z/OS V1R11 because zFS runs at sysplex\_admin\_level 3 on z/OS V1R11.

## 16.12.2 Setting up the zFS parameters

There is a new value for the IOEPRMxx or IOEFSPRM sysplex option, filesys. This option is for z/OS V1R11 and z/OS V1R12 systems, as follows:

```
sysplex=off | on | filesys
```

Where:

- sysplex=on** zFS can be configured `sysplex=on` to treat all zFS read-write mounted file systems owned on that system as sysplex-aware file systems.
- sysplex=filesystems** zFS can be configured `sysplex=filesystems` to allow some zFS read-write mounted file systems owned on that system to be sysplex-aware file systems and some to be non-sysplex aware file systems.

zFS must be running sysplex-aware for read-write in order to allow a zFS read-write file system to be mounted as sysplex-aware. Specifying `sysplex=filesystems` allows individual control of which zFS read-write file systems are sysplex-aware and which are not (in a shared file system environment).

**Note:** A z/OS V1R10 system requires APAR OA29786.

### 16.12.3 Using `sysplex=filesystems`

Whether the PFS is running sysplex-aware on a file system basis (referred to as `filesystems`), or sysplex-aware on a system basis (referred to as `file`), or not sysplex-aware (referred to as `admin-only`), and the zFS XCF protocol level (normally 3 for zFS on z/OS V1R11) when running in a shared file system environment, when `filesystems` is indicated, the default mount PARM (`NORSHARE` or `RWSHARE`) also displays. See “Expected Value: `rwshare` or `norwshare`” on page 307.

When running with `sysplex=filesystems`, a new mount parameter can be used to specify whether a file system is mounted sysplex-aware:

`RWSHARE` | `NORSHARE`

When `sysplex=filesystems`, the new `IOEPRMxx` or `IOEFSPRM` option controls the default mount parameter:

`sysplex_filesys_sharemode=[ norwshare | rwshare ]`

When you run `sysplex=filesystems`, the zFS PFS runs sysplex-aware, but each zFS file system is mounted non-sysplex aware (by default). zFS is enabled to allow zFS read-write file systems to be sysplex-aware but it must be explicitly requested on a file system basis. Note that although it is possible to change the `sysplex_filesys_sharemode` from its default of `norwshare` to `rwshare` at IPL using the `IOEFSPRM` option or dynamically using the `zfsadm config` command, this is not recommended while migrating from `sysplex=off` to `sysplex=filesystems` because it can cause performance problems.

**Note:** The new `sysplex=filesystems` is documented in a refresh of *z/OS Distributed File Service zFS Administration*, SC24-5989.

#### zFS command output

You can determine whether zFS on a particular system is running sysplex-aware or not with the `F ZFS,QUERY,LEVEL` operator command. If the command indicates `sysplex(file)` or `sysplex(filesys)`, zFS is sysplex-aware. If the command indicates `sysplex(admin-only)`, zFS is not sysplex-aware.

**Note:** In z/OS V1R11 and above in a shared file system environment, when zFS is running with `sysplex=on` in the IOEFSPRM configuration file, the `sysplex` level is (file), and the interface level is (3).

When zFS is running `sysplex=filesys` in the IOEFSPRM configuration file, the `sysplex` level is (filesys,norwshare) or (filesys,rwshare) depending on the `sysplex_filesys_sharemode`, and the interface is (3). Otherwise, the zFS `sysplex` level is (admin-only) and the interface is (3).

Figure 16-45 shows zFS `sysplex` status information data.

```
/* From the OMVS shell .....
$> zfsadm configquery -sysplex_state
IOEZ00317I The value for configuration option -sysplex_state is 3.
$> zfsadm configquery -syslevel
IOEZ00644I The value for configuration option -syslevel is:
zFS kernel: z/OS      zSeries File System
Version 01.11.00 Service Level 0A29619 - HZFS3B0.
Created on Wed Jan 13 09:39:20 EST 2010.
sysplex(filesys,norwshare) interface(3)

/* MVS command.....
f zfs,query,level
IOEZ00639I zFS kernel: z/OS      zSeries File System
Version 01.11.00 Service Level 0A29619 - HZFS3B0.
Created on Wed Jan 13 09:39:20 EST 2010.
sysplex(filesys,norwshare) interface(3)
IOEZ00025I zFS kernel: MODIFY command - QUERY,LEVEL completed successfully.
```

Figure 16-45 zFS `sysplex` status queries

### zFS API output

Figure 16-46 shows the output of the API example for `syslevel`, which is the same as what the `zfsadm configquery -syslevel` command provides.

```
$> zfs_query_syslevel
zFS kernel: z/OS      zSeries File System
Version 01.11.00 Service Level 0A29619 - HZFS3B0.
Created on Wed Jan 13 09:39:20 EST 2010.
sysplex(filesys,norwshare) interface(3)
$>
```

Figure 16-46 Displaying the zFS `syslevel` information using APIs

## 16.12.4 zFS `sysplex`-aware on a file system basis

With z/OS V1R12 and V1R11, to enable zFS `sysplex`-aware on a file system basis, specify `sysplex=filesys` in the IOEFSPRM configuration file. Leave the `sysplex_filesys_sharemode` with its default of `norwshare`. You can specify it in a shared IOEFSPRM configuration file and each system picks up the specification in a rolling IPL. The `sysplex` option is ignored by previous releases. Any V1R10 systems in the shared file system environment must have zFS

APAR OA29786 active before you can have zFS `sysplex=filesystems` active on any z/OS V1R11 and z/OS V1R12 systems.

If you decide that you want some zFS read-write file systems to be sysplex-aware, you can run zFS sysplex-aware on a file system basis. Roll this support through all your sysplex members (this assumes all your members are at z/OS V1R11 with the V1R11 APAR OA29619 applied). You cannot change from zFS non-sysplex aware to zFS sysplex-aware on a file system basis dynamically. After changing IOEFSPRM, you must perform an IPL or a restart of zFS. IPL is the recommended method because in many ways it is less disruptive than a restart of zFS, which can cause zFS file systems to become unmounted.

### **Mixed sysplex example - sysplex-aware on file system basis**

Figure 16-47 on page 312 shows three systems with zFS sysplex-aware on a file system basis (`sysplex=filesystems`).

After you have `sysplex=filesystems` active on all your systems, you can consider which zFS read-write file systems you want to be sysplex-aware. Good candidates are zFS read-write file systems that are accessed from multiple systems or are mounted with AUTOMOVE and might be moved by z/OS UNIX (as a result of a shutdown or IPL) to systems that do not necessarily do the most accesses.

Figure 16-47 on page 312 shows two zFS read-write file systems on a sysplex running zFS sysplex-aware on file system basis on all members. One file system is mounted NORWSHARE and the other is mounted RWSHARE. They are both z/OS UNIX owned by system SY2.

The norwshare file system is a non-sysplex aware file system. It is only locally mounted on the z/OS UNIX owner and requests from z/OS UNIX clients are function shipped to the z/OS UNIX owner by z/OS UNIX. A `df -v` command for the norwshare file system FS1 from SY1 would display `Client=Y`, as shown in Figure 16-48 on page 314.

The other file system shown in Figure 16-47 on page 312 is mounted rwshare. It is a sysplex-aware file system and locally mounted on all systems and z/OS UNIX never function ships requests to the z/OS UNIX owner. A `df -v` command for the rwshare file system FS2 from SY1 would display `Client=N`. Figure 16-49 on page 314 demonstrates this situation.

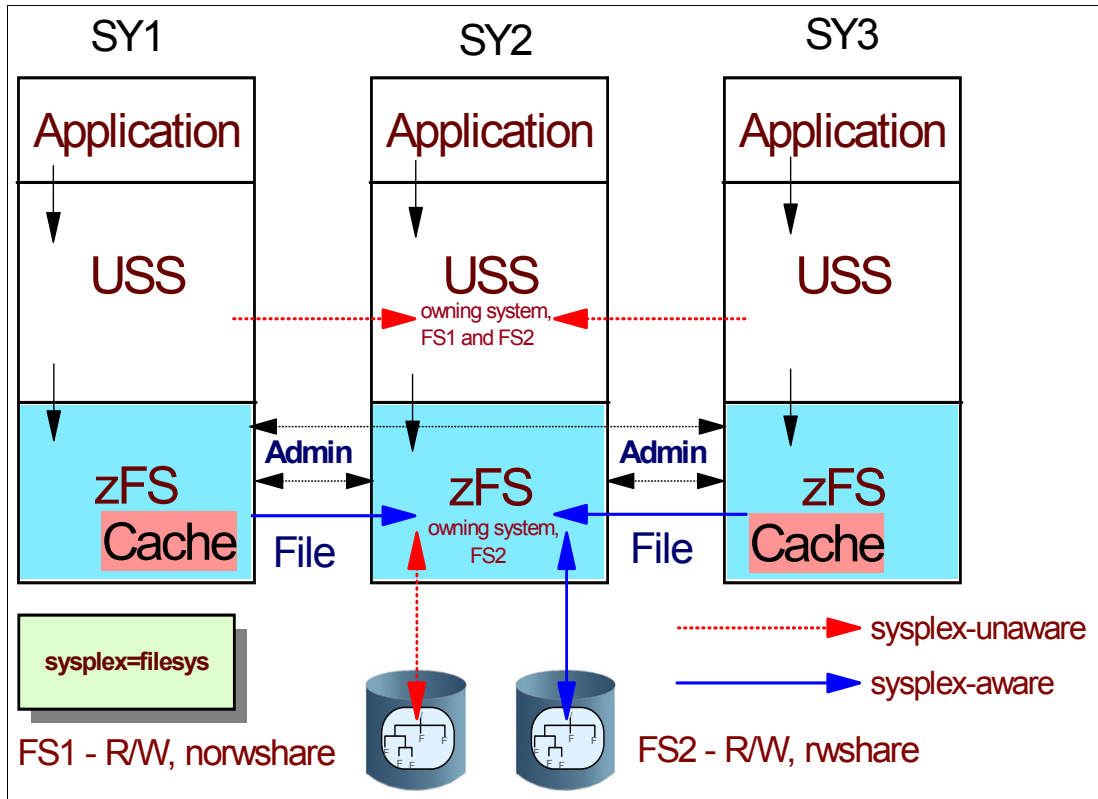


Figure 16-47 zFS sysplex-aware on a file system basis

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 mount parameters rwshare and norwshare.

**Note:** MOUNT commands for file systems in Figure 16-47.

- For file system FS2

```
MOUNT FILESYSTEM('OMVS.PRIV2.COMPAT.AGGR001') TYPE(ZFS)
MODE(RDWR) MOUNTPOINT('/usr/mountpt2') PARM('RWSHARE')
```

- For file system FS1

```
MOUNT FILESYSTEM('OMVS.PRIV1.COMPAT.AGGR001') TYPE(ZFS)
MODE(RDWR) MOUNTPOINT('/usr/mountpt1') PARM('NORWSHARE')
```

### 16.12.5 Running zFS sysplex-aware considerations

If you have (at least) APAR OA29619 and you intend to run zFS sysplex-aware on a file system basis (sysplex=filesys), you must ensure that you have (at least) APAR OA29786 on your prior releases (z/OS V1R10 if z/OS V1R12 is active) and z/OS V1R9 if z/OS V1R11 is your highest level of z/OS.

**Important:** In a shared file system environment, a zFS read-write sysplex-aware file system, in general, cannot be explicitly moved by z/OS UNIX to a prior release system or to a z/OS V1R11 system that is running zFS non-sysplex aware. In the special case that the zFS file system is sysplex-aware and every other system is doing z/OS UNIX function shipping of requests to the owning system, the zFS read-write file system can be explicitly moved, in most cases.

In a shared file system environment, when a zFS file system is mounted read-write and is z/OS UNIX owned on a prior release (or is z/OS UNIX owned on a z/OS V1R11 system that is running zFS non-sysplex aware), a BPXF221I message similar to the following example may be issued on each system that receives a local catch-up mount:

```
BPXF221I FILE SYSTEM PLEX.JMS.AGGR006.LDS0006 FAILED TO MOUNT LOCALLY.  
RETURN CODE = 00000079, REASON CODE = EF0969A8  
THE FILE SYSTEM IS ACCESSIBLE ON THIS SYSTEM THROUGH A MOUNT ON A REMOTE  
SYSTEM.
```

In this case, z/OS UNIX does forward file requests from the z/OS V1R11 system to the z/OS UNIX owning system.

## Read-only file systems

zFS read-only mounted file systems are not affected by this support. However, if you remount a read-only file system to read-write (using the **chmount** command or the TSO/E UNMOUNT REMOUNT command), this is treated like a primary mount on the current z/OS UNIX owning system and MOUNT parameters (such as RWSHARE or NORWSHARE) or MOUNT defaults (such as the current `sysplex_filesys_sharemode` setting on that system) take effect when it is mounted read-write. When you remount back to read-only, those mount options are irrelevant again.

**Note:** These MOUNT parameters and MOUNT defaults do not take effect when a remount to the same mode is run.

Do not make any zFS read-write file systems sysplex-aware until you have all systems in the shared file system environment at z/OS V1R11 with `sysplex=filesys` active. To make a zFS read-write file system sysplex-aware when running `sysplex=filesys` on all systems, you must unmount the file system, specify RWSHARE as a mount PARM and then mount the file system. The following TSO/E example shows a zFS mount PARM of RWSHARE:

```
MOUNT FILESYSTEM('OMVS.PRIV.COMPAT.AGGR001') TYPE(ZFS)  
MOUNTPOINT('/etc/mountpt') PARM('RWSHARE') MODE(RO)
```

Normally, if you are going to run any zFS read-write file systems sysplex-aware, you would run zFS sysplex-aware on a file system basis. This gives you more flexibility and control in determining which zFS file systems should be sysplex-aware for read-write. It also allows you to change whether a zFS read-write file system is sysplex-aware or not simply by unmounting and mounting it with a different mount PARM. However, if you want to give zFS complete control over which system actually does the I/O to the DASD for any zFS read-write file system and you do not want to concern yourself with which system is the z/OS UNIX owning system and you are not using any servers that are restricted from supporting zFS sysplex-aware, then you can run all your zFS read-write file systems sysplex-aware.

## Interface level

In z/OS V1R11 and above in a shared file system environment, when zFS is running with `sysplex=on` in the IOEFSPRM configuration file, the sysplex level is (file), and the interface

level is (3). When zFS is running sysplex=filesys in the IOEFSPRM configuration file, the sysplex level is (filesys,norwshare) or (filesys,rwshare) depending on the sysplex\_filesys\_sharemode and the interface is (3). Otherwise, the zFS sysplex level is (admin-only) and the interface is (3), as shown in Figure 16-48.

```

$> zfsadm configquery -syslevel | grep sysplex
sysplex(filesys,norwshare) interface(3)
$> sudo /usr/sbin/mount -f hering.test.zfs test
$> zfsowner hering.test.zfs
zFS Owner      : SC70      - Aggregate read-only=N, sysplex-aware=N
$> cn SC65 @usscmd "'df -v test | grep Client='"
@USSCMD 'df -v test | grep Client='
File System Owner : SC70      Automove=Y      Client=Y
$> cn SC65 @usscmd "'rxdowner -d test | grep local-client'"
@USSCMD 'rxdowner -d test | grep local-client'
Local Sysname: SC65      - File System local-client=Y

```

Figure 16-48 zFS aggregate mounted R/W sysplex-unaware

Some explanation of the commands used in Figure 16-49:

- ▶ **sudo** is a utility running just one command in super-user mode.
- ▶ **rxdowner** is a hardlink of the other utility, **zfsowner**.
- ▶ **cn** is a utility to run MVS system commands.
- ▶ **SC65** is an abbreviation for specifying ROUTE SC65.
- ▶ **USSCMD** is a SYSREXX running UNIX commands.

The sample REXX procedures are available in softcopy on the Internet from the Redbooks web server at:

<ftp://www.redbooks.ibm.com/redbooks/SG246580/>

Note that the SG part of SG246580 must be uppercase.

Alternatively, you can go to:

<http://www.redbooks.ibm.com>

Select **Redbooks Online**, and then **Additional Materials**.

```

$> sudo /usr/sbin/unmount test
$> sudo /usr/sbin/mount -f HERING.TEST.ZFS -t zFS -o rwshare test
$> zfsowner hering.test.zfs
zFS Owner      : SC70      - Aggregate read-only=N, sysplex-aware=Y
$> cn SC65 @usscmd "'df -v test | grep Client='"
@USSCMD 'df -v test | grep Client='
File System Owner : SC70      Automove=Y      Client=N
$> cn SC65 @usscmd "'rxdowner -d test | grep local-client'"
@USSCMD 'rxdowner -d test | grep local-client'
Local Sysname: SC65      - File System local-client=N
$>

```

Figure 16-49 zFS aggregate mounted R/W sysplex-aware

### Automatic movement of file systems

Figure 16-50 on page 315 shows again the three systems with zFS sysplex-aware on a file system basis (sysplex=filesys). However, this time the sysplex-aware zFS has two different



owners. Beginning with z/OS V1R11, as part of the supporting read-write mounted file systems that are accessed as sysplex-aware, zFS automatically moves zFS ownership of a zFS file system to the system that has more read-write activity coming from it compared to the total amount from all systems.

This is because zFS can move its ownership of zFS read-write sysplex-aware file systems dynamically, based on system usage among zFS systems where the file systems are locally mounted. If one sysplex member is a better fit for ownership, because more requests to the file system are being done there than at any other sysplex member, then zFS dynamically moves the zFS ownership to that system.

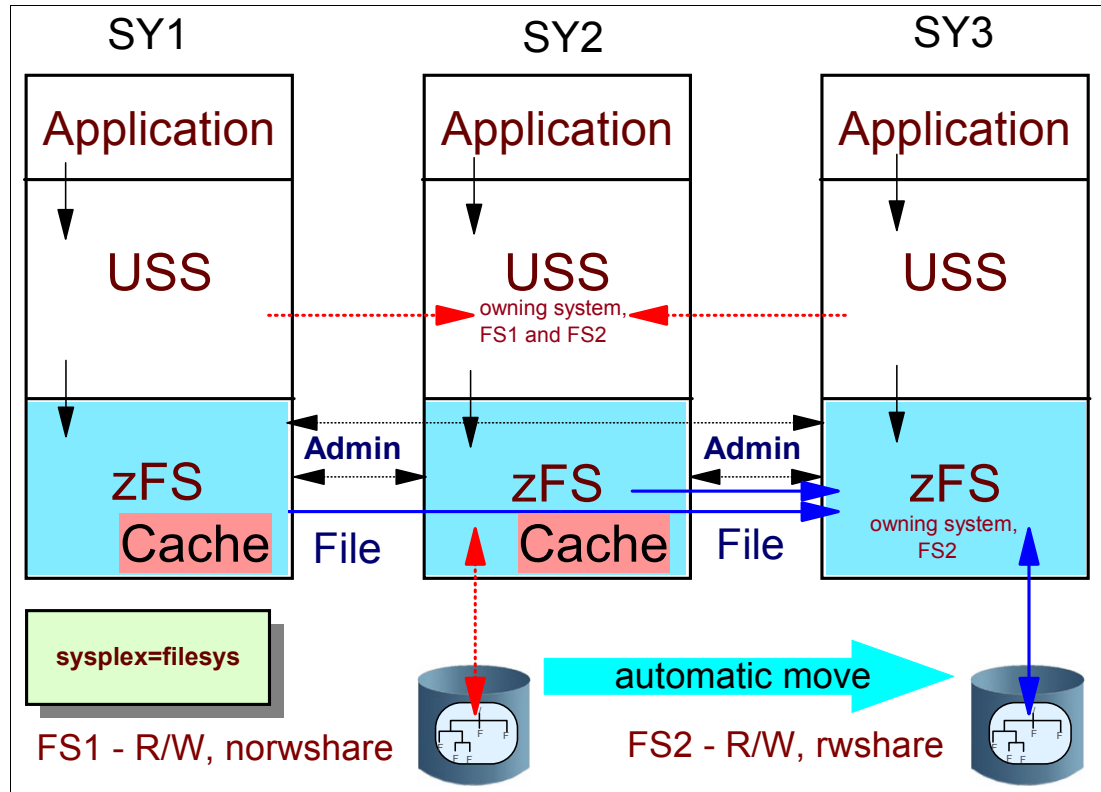


Figure 16-50 zFS sysplex-aware on a file system basis with a zFS having two different owners

### Other ownership movement

zFS will also move ownership when the owning system is shut down, or an abnormal outage occurs on the zFS owning system.

Thus the z/OS UNIX owner and the zFS owner can be two entirely different systems, depending on the sequence of events. This is normal, and for a sysplex that is using the zFS `sysplex=filesystems` support on all sysplex members this should have no negative effects.

zFS does not have any external commands to move zFS ownership between systems. It does provide query commands to show the zFS owner, such as the `zfsadm l saggr` command, the `pfctl()` APIs or the utilities `rxdowner` and `zfsowner` as shown in Figure 16-51 on page 316.

```

$> df -v test | grep Client=
File System Owner : SC70      Automove=Y      Client=N
$> zfsadm lsaggr | grep HERING.TEST.ZFS
HERING.TEST.ZFS              SC64      R/W
$> rxlsaggr | grep HERING.TEST.ZFS
HERING.TEST.ZFS              SC64      R/W
$> rxdowner -d test | grep local-client
Local Sysname: SC70      - File System local-client=N
$> zfsowner hering.test.zfs
zFS Owner      : SC64      - Aggregate read-only=N, sysplex-aware=Y
$>

```

Figure 16-51 zFS aggregate mounted R/W sysplex-aware and having two different owners

Note the following statement:

**Important:** A zFS ownership movement based on performance criteria is non-disruptive.

## 16.12.6 Benefits of and recommendations for the new support

The new support has several advantages over the zFS base functions in z/OS V1R11, as follows:

- ▶ The new support eliminates performance problems that can occur when running in a mixed environment.
- ▶ It allows better coexistence with servers that cannot handle zFS read-write sysplex-aware file systems.
- ▶ z/OS SMB server and Fast Response Cache Accelerator support of the IBM HTTP server.

The SMB server cannot export any zFS read-write file systems that are sysplex-aware. To export zFS read-write file systems using the SMB server, the file system must be non-sysplex aware. Ensure that you have SMB APAR OA31112 for full support.

When the IBM HTTP Server Version 5.3 for z/OS is configured to use Fast Response Cache Accelerator, static web pages are cached to improve performance. Part of this support uses register file interest, a function of z/OS UNIX, to get notified if the files representing the web pages are changed.

**Important:** Running zFS sysplex-aware on a file system basis, using `sysplex=filesystems`, is the preferred and recommended option for a shared file system environment.

- ▶ If you are currently running zFS with a setting `sysplex=off`:
  - Roll all your systems to `sysplex=filesystems`.
  - Schedule an unmount and mount of zFS file systems you want to be sysplex-aware and specify `RWSHARE` on the mount command.
- ▶ If you are currently running zFS with option `sysplex=on`:
  - Roll all your systems to `sysplex=filesystems` with `sysplex_filesys_sharemode=rwshare`.

Schedule an unmount and new mount of zFS file systems that you do not want to be sysplex-aware and specify `NORWSHARE` on the mount command.

## zFS R/W sysplex-aware test

Here we demonstrate that with the current implementation the sysplex-aware state of a file system can be changed only when unmounting and mounting it again or doing an unmount remount for reading and again an unmount remount for write mode. This is shown in Figure 16-52.

```
$> zfsadm configquery -syslevel | grep sysplex
sysplex(filesys,norwshare) interface(3)
$> rxdowner -d test

MP Directory : /u/hering/test
File System : HERING.TEST.ZFS
PFS Type     : ZFS
Local Sysname: SC70      - File System local-client=N
USS Owner    : SC70      - File System read-only=N
zFS Owner    : SC70      - Aggregate read-only=N, sysplex-aware=N

$> sudo zfsadm config -sysplex_filesys_sharemode rwshare
IOEZ00300I Successfully set -sysplex_filesys_sharemode to rwshare
$> sudo /usr/sbin/chmount -s test
$> zfsowner hering.test.zfs
zFS Owner    : SC70      - Aggregate read-only=N, sysplex-aware=N
$> sudo /usr/sbin/chmount -r test
$> sudo /usr/sbin/chmount -w test
$> sudo zfsadm config -sysplex_filesys_sharemode norwshare
IOEZ00300I Successfully set -sysplex_filesys_sharemode to norwshare
$> zfsowner hering.test.zfs
zFS Owner    : SC70      - Aggregate read-only=N, sysplex-aware=Y
$>
```

Figure 16-52 zFS R/W sysplex-aware file system state kept until doing a major remount

## zFS toleration in z/OS V1R9 and V1R10

zFS R9 and R10 require APAR OA29786 before the sysplex=filesys function can be enabled in zFS of z/OS V1R11; see Table 16-3.

Table 16-3 zFS Toleration in z/OS V1R9 and V1R10

z/OS release (zFS release)	PTF number	APAR number
z/OS V1R9 (390)	UA50338	OA29786
z/OS V1R10 (3A0)	UA50336	OA29786

- ▶ Any z/OS V1R9 or V1R10 systems in the shared file system environment must have zFS APAR OA29786 active before you can have zFS sysplex=filesys active on any z/OS V1R11 system, as shown in Figure 16-49 on page 314.
- ▶ The zFS mount parameters RWSHARE and NORWSHARE are ignored in z/OS V1R9 and V1R10 and the mount is allowed to succeed.

## z/OS UNIX APAR OA29712

As mentioned in “USS support for sysplex-aware on a file system basis” on page 295, the USS SPE OA29712 is a prerequisite for the zFS SPE OA29619.

## z/OS SMB APAR OA31112

Before this new support, the z/OS SMB server does not export a zFS read-write mounted file system that is on a system running zFS sysplex-aware. SMB checks whether the zFS `sysplex_state` is 2 (this is `sysplex=on`) or greater (`sysplex_state` 3 is `sysplex=filesys`).

With the SMB new function APAR OA31112, if the local zFS is not running `sysplex=on`, the SMB server checks the sysplex-aware state of the file system. If the file system is not sysplex-aware, then the SMB server attempts to export it.

Table 16-4 SMB support for zFS sysplex-aware on a file system basis

z/OS release (SMB release)	PTF number	APAR number
z/OS V1R11 (3B0)	UA52995 (PE)	OA31112
z/OS V1R11 (3B0)	UA53952	OA32648

**Note:** APAR OA32648 solves the problem with the PTF for APAR OA31112; see Table 16-4.

## 16.12.7 z/OS UNIX directory caching display tool

This tool displays whether z/OS UNIX directory caching is active. It is important for helping to resolve performance issues when z/OS UNIX directory caching is lost due to file system movement. This normally only happens when the shared file system environment is mixed.

You can get the tool from:

<ftp://ftp.software.ibm.com/s390/zos/tools/wjsip/wjsipndc.txt>

**Note:** Be sure to convert it from ISO8859-1 to IBM-1047 or IBM-037 when placing the file into your z/OS system.

Exceptions are listed for file systems when the system is a z/OS UNIX client and while no caching is active, as shown in Figure 16-53.

```
$> sudo wjsipndc
zFS file systems with a z/OS directory cache exception
HFS.ZOSR1B.Z1BRC1.ROOT
OMVS.DB2V9.SDSN5HFS.D081006
IMS11B.JMK1106.HFS

If a zFS file system was once read-write and sysplex aware but
is no longer sysplex aware due to moving ownership to a zFS
sysplex=off system or migrating back to sysplex=off from
sysplex=on, the z/OS UNIX System Services directory cache
will be disabled for that file system. It is possible
performance of that file system can be improved by unmounting
that file system and mounting it back.
```

Figure 16-53 Running the z/OS UNIX directory caching display tool

Note the following statement:

**Important:** Note that once a file system's z/OS UNIX directory cache is disabled, it does not become enabled again unless you do an explicit unmount and mount.

### z/OS directory caching for a zFS being sysplex-unaware

There are performance implications with running in a mixed environment. In some cases, your performance might be worse than running all zFS file systems non-sysplex aware. One reason for this is because z/OS UNIX only performs caching of directory information on z/OS UNIX clients when the read-write file system is non-sysplex aware. If the read-write file system is sysplex-aware, z/OS UNIX does not perform directory caching. Any z/OS UNIX client lookup request goes to the z/OS UNIX owning system through XCF communications rather than potentially being satisfied in the local z/OS UNIX directory cache; refer to Figure 16-54.

```
$> sudo zfsadm config -sysplex_filesys_sharemode rwshare -system SC65
IOEZ00300I Successfully set -sysplex_filesys_sharemode to rwshare
$> sudo /usr/sbin/mount -f HERING.TEST.ZFS -t zfs -d SC65 test
$> zfsowner hering.test.zfs
zFS Owner      : SC65      - Aggregate read-only=N, sysplex-aware=Y
$> rxdowner -d test | grep local-client=
Local Sysname: SC70      - File System local-client=N
$> sudo /usr/sbin/chmount -r test
$> sudo zfsadm config -sysplex_filesys_sharemode norwshare -system SC65
IOEZ00300I Successfully set -sysplex_filesys_sharemode to norwshare
$> sudo /usr/sbin/chmount -w test
$> zfsowner hering.test.zfs
zFS Owner      : SC65      - Aggregate read-only=N, sysplex-aware=N
$> rxdowner -d test | grep local-client=
Local Sysname: SC70      - File System local-client=Y
$> sudo wjsipndc
zFS file systems with a z/OS directory cache exception
HERING.TEST.ZFS
OMVS.DB2V9.SDSN5HFS.D081006
```

Figure 16-54 Loss of z/OS directory caching for a zFS being sysplex-unaware

Note the following statement:

**Note:** To make a zFS read-write file system sysplex-aware when running `sysplex=filesys` on all systems, you must unmount the file system, specify `RWSHARE` as a mount `PARM` and then mount the file system, as shown in Figure 16-49 on page 314.

## 16.12.8 File system monitoring tool FSMON

This `wjsfsmon` utility is mentioned in APAR OA29619. It can help you determine which systems are accessing your zFS read-write file systems and whether your zFS read-write file systems are accessed from multiple systems.

See the z/OS UNIX Tools and Toys website to download the tool and its documentation:

<http://www.ibm.com/servers/eserver/zseries/zos/unix/tools>

You can also directly go to the UNIX Tools site:

<http://www-03.ibm.com/systems/z/os/zos/features/unix/bpxalty2.html>

or retrieve it from:

<ftp://ftp.software.ibm.com/s390/zos/tools/wjsfsmon/wjsfsmon.txt>

**Note:** The documentation is available as a PDF. The tool itself is available in ASCII. Be sure to convert it from ISO8859-1 to IBM-1047 or IBM-037 when placing the file to your z/OS system.

## 16.13 DFSMSdfp indirect volume serial for zFS data sets

When z/OS system programmers have HFS file systems, they typically migrate them from one release to another using an IBM-recommended cloning process to copy the system residence volumes.

**Note:** The cloning of non-VSAM data sets such as HFS uses the extended indirect volume serial support that uses system symbols.

With zFS, IBM introduced new inhibitors, as follows:

- ▶ zFS file systems are VSAM-based using Linear Data Sets (LDS).
- ▶ VSAM data sets need to be cataloged using a real volume serial.

**Restriction:** This brings up cloning and maintenance issues that prevent the migration of USS version root file systems.

### 16.13.1 Cloning zFS file systems

z/OS V1R12 DFSMS provides the extended indirect volume serial support that can be used to clone zFS file systems. A single catalog entry (for zFS VSAM LDS data sets) can represent different volumes on various systems.

**Important:** This support is the solution for several marketing requirements (like MR0424065959) asking for extended indirect volume serial support for cloning of zFS file systems.

#### Detailed specification and requirements

Here we provide the requirements that need to be met to allow a zFS aggregate to have an extended indirect volume serial:

- ▶ The zFS data set needs to be a single volume VSAM linear data set (LDS).
- ▶ A system symbol has been defined in the systems involved pointing to a system-specific volume that the zFS LDS is located on.
- ▶ On a DEFINE RECATALOG command, the VOLUMES parameter can have a special form referred to as indirect volume serial by using that system symbol. It can only be used with the RECATALOG parameter.
- ▶ This new function can be used for both SMS and non-SMS managed zFS data sets on a DEFINE RECATALOG command of the data set.

**Note:** This results in the system dynamically resolving the volume serial to the system volume serial. There are two special forms of the VOLUMES parameter that can be provided, and they are referred to as the indirect volume serial forms. They result in the system dynamically resolving the volume serial to the system residence (or its logical extension) serial number when the catalog entry is retrieved. It is not resolved when the DEFINE NONVSAM is processed. This allows you to later change the volume serial number(s) of the system residence volume (or its logical extensions) without having to recatalog the non-VSAM data sets on those volumes.

## 16.13.2 Enabling the support

Some specific actions are needed to enable this support for a specific zFS data set name in your environment.

### Provide a single volume zFS LDS

For example, you can define and format it as shown in Figure 16-55.

```
$> zfsadm define omvs.zfs2.clone -storageclass openmvs -cylinders 1 0
IOEZ00248I VSAM linear dataset omvs.zfs2.clone successfully created.
$> sudo zfsadm format omvs.zfs2.clone -compat
IOEZ00077I HFS-compatibility aggregate OMVS.ZFS2.CLONE has been successfully
created
$> tsocmd "listcat ent('omvs.zfs2.clone') volume" 2>/dev/null | grep VOLSER \
> | awk -F'- ' '{print $13}' | awk '{print $1}'
BH50E2
$>
```

Figure 16-55 Defining and formatting a single volume zFS aggregate

### Defining a system symbol

A system symbol needs to exist or must be defined pointing to the volume serial, for example by using a definition in an IEASYSMxx parmlib member. A sample is shown in Figure 16-56.

```
SYSDEF SYMDEF(&ZFSCCL.= 'BH50E2')
```

Figure 16-56 System symbol pointing to a volume serial

**Note:** You can also dynamically add a system symbol using the IEASYMUP utility.

For more information about IEASYMUP, see Appendix B. IEASYMUP in *z/OS Planned Outage Avoidance Checklist*, SG24-7328.

### Deleting the catalog entry of the zFS LDS

Run an IDCAMS or TSO DELETE NOSCRATCH command against the data set, as shown in Figure 16-57.

```
$> tsocmd "delete 'omvs.zfs2.clone' noscratch"
delete 'omvs.zfs2.clone' noscratch
ENTRY (D) OMVS.ZFS2.CLONE.DATA DELETED
ENTRY (C) OMVS.ZFS2.CLONE DELETED
$>
```

Figure 16-57 DELETE NOSCRATCH

This deletes the catalog entry of the data set.

### Recataloging the zFS LDS

Run an IDCAMS or TSO DEFINE RECATALOG using the system symbol (which is the indirect volume serial) against the data set. This is shown in Figure 16-58 on page 322.

```
$> tsocmd "define cluster (name('omvs.zfs2.clone') volumes(&ZFSC) \
> linear recatalog)"
define cluster (name('omvs.zfs2.clone') volumes(&ZFSC) linear recatalog)
DATA ALLOCATION STATUS FOR VOLUME &ZFSC IS 0
$>
```

Figure 16-58 DEFINE RECATALOG

**Note:** This defines the data set with the system symbol.

For more details about how to set up an indirect volume serial, see *z/OS V1R12 DFSMS Access Method Services for Catalogs*.

### 16.13.3 Cloning of zFS aggregates

Setting the value of the system symbol to the desired values on each system achieves a single catalog entry (for zFS VSAM LDS data sets) that represents different volumes on these systems.

**Note:** System programmers can now use this to clone volumes, if this level of DFSMS is used.

### 16.13.4 Example of cloning a zFS aggregate

Following is a simple example of cloning using the example zFS aggregate OMVS.ZFS2.CLONE. This is normally not the way cloning will be used; usually the complete volume is copied. This is just done to demonstrate the principles used with zFS.

- ▶ Remove the catalog entry again as shown in Figure 16-57 on page 322.
- ▶ Then define a zFS aggregate with the same name on another system and provide a system symbol pointing to the new volume in that system. This is shown in Figure 16-59.



```

$> echo This is system $(sysvar SYSNAME).
This is system SC75.
$> zfsadm define omvs.zfs2.clone -storageclass openmvs -cylinders 2 0
IOEZ00248I VSAM linear dataset omvs.zfs2.clone successfully created.
$> tsocmd "listcat ent('omvs.zfs2.clone') volume" 2>/dev/null | grep VOLSER \
> | awk -F'-' '{print $13}' | awk '{print $1}'
BH50E1
$> tsocmd "call 'hering.tso.load(ieasymup)' 'ZFSC=BH50E1'"
call 'hering.tso.load(ieasymup)' 'ZFSC=BH50E1'
$> sysvar ZFSC
BH50E1
$>

```

Figure 16-59 Defining the target zFS with the same name on another system

Note the following information about IEASYMUP.

**Note:** In order to be able to use IEASYMUP successfully you must ensure the following:

- ▶ The library containing IEASYMUP must be APF-authorized.
- ▶ When using IEASYMUP from TSO it must be defined in the AUTHTSF names list of your TSO IKJTSOxx parmlib member.
- ▶ IEASYMUP is provided on an as-is basis, as are all samples in SYS1.SAMPLIB.

Instead of using the TSO interface, you can also use a job as shown in Figure 16-60.

```

//SYMBATCH EXEC PGM=IEASYMUP,PARM='ZFSC=BH50E1'
//STEPLIB DD DSN=&SYSUID..TSO.LOAD,DISP=SHR

```

Figure 16-60 Defining the system symbol dynamically using a job

Afterwards we mounted and worked with the zFS file system in the source system. This is shown in Figure 16-61.

```

$> echo This is system $(sysvar SYSNAME).
This is system SC74.
$> sudo /usr/sbin/mount -f omvs.zfs2.clone zfs2.clone
$> touch zfs2.clone/sc74.file
$> ls -l zfs2.clone/sc74.file
zfs2.clone/sc74.file
$> sudo /usr/sbin/unmount zfs2.clone
$>

```

Figure 16-61 Working with the zFS file system in the source system

## Activating the clone processing

First unload the source zFS aggregate using IDCAMS REPRO and restore it to the target system.

Then unload the zFS aggregate on system SC74. The JCL is shown in Figure 16-62.

```

/* -----
// SET ZFSAGGR=OMVS.ZFS2.CLONE          <=== zFS aggrname
// SET ZFSUNLOD=HERING.OMVS.ZFS2.CLONE.ULD  <=== Unloaded file
/* -----
//UNLOAD EXEC PGM=IDCAMS
//ZFSAGGR DD DSN=&ZFSAGGR.,DISP=SHR
//ZFSUNLOD DD DSN=&ZFSUNLOD.,DISP=(NEW,CATLG,DELETE),LRECL=4096,
// BLKSIZE=20480,RECFM=FB,UNIT=SYSDA,SPACE=(CYL,(1,1),RLSE)
//SYSIN DD DATA,DLM=##
      REPRO INFILE(ZFSAGGR) OUTFILE(ZFSUNLOD)
##
//SYSPRINT DD SYSOUT=*

```

Figure 16-62 Unload zFS aggregate using IDCAMS REPRO in system SC74

Then restore the zFS aggregate on system SC75. The JCL is shown in Figure 16-63.

```

/* -----
// SET ZFSAGGR=OMVS.ZFS2.CLONE          <=== Target zFS
// SET ZFSUNLOD=HERING.OMVS.ZFS2.CLONE.ULD  <=== Unloaded file
/* -----
//RESTORE EXEC PGM=IDCAMS
//ZFSAGGR DD DSN=&ZFSAGGR.,DISP=OLD
//ZFSUNLOD DD DSN=&ZFSUNLOD.,DISP=(SHR,DELETE,KEEP)
//SYSIN DD DATA,DLM=##
      REPRO INFILE(ZFSUNLOD) OUTFILE(ZFSAGGR)
##
//SYSPRINT DD SYSOUT=*

```

Figure 16-63 Restore zFS aggregate using IDCAMS REPRO in system SC75

Finally there are two different data sets on different volumes. This is shown in Figure 16-64.

```

$> echo This is system $(sysvar SYSNAME).
This is system SC74.
$> sudo /usr/sbin/mount -f omvs.zfs2.clone -d SC75 zfs2.clone
$> mv zfs2.clone/sc74.file zfs2.clone/sc75.file
$> ls -l zfs2.clone | grep sc7
sc75.file
$> sudo /usr/sbin/chmount -d SC74 zfs2.clone
$> ls -l zfs2.clone | grep sc7
sc74.file
$>

```

Figure 16-64 An example of having two different data sets on different volumes

This also shows that you should be careful if this is not done for real cloning of system residence volumes.

## 16.14 zFS usage of VSAM partial release on multiple volumes

With partial release, CLOSE releases the unused space only on the current volume except for VSAM extended format data sets or for a striped data set that has a stripe count of greater than one. For VSAM extended format data sets, partial release at CLOSE releases unused space from multiple volumes as follows:

- ▶ In non-EAV data sets, the starting point for freeing space is at a CA boundary; all space after the last used CA boundary is freed in a partial release, up to the high allocated RBA.
- ▶ In EAV data sets, the starting point for freeing space is an MCU boundary; all space after the last used MCU boundary is freed up to the high allocated RBA.
- ▶ Partial release does not release space on later volumes that can contain data either from a prior writing or due to guaranteed space.
- ▶ With a system-managed data set, this has no effect on a later use of DISP=MOD but it does mean that the space on the later volumes can be there due to guaranteed space allocation.

### Partial release

Partial release is used to release unused space from the end of an extended format data set and is specified through an SMS management class or by the JCL RLSE subparameter. For data sets whose ending high used RBA is in track-managed space, all space after the high used RBA is released on a CA boundary up to the high allocated RBA. If the high used RBA is not on a CA boundary, the high used amount is rounded to the next CA boundary. For data sets whose ending high used RBA is in cylinder-managed space, all space after the high used RBA is released on an MCU boundary up to the high allocated RBA. If the high used RBA is not on an MCU boundary, the high used amount is rounded to the next MCU boundary.

### Partial release restrictions

The following restrictions apply for partial release:

- ▶ Partial release processing is supported only for extended format data sets.
- ▶ Only the data component of the VSAM cluster is eligible for partial release.
- ▶ Alternate indexes opened for path or upgrade processing are not eligible for partial release. The data component of an alternate index when opened as cluster could be eligible for partial release.
- ▶ Partial release processing is not supported for temporary close.
- ▶ Partial release processing is not supported for data sets defined with guaranteed space.

For extended format data sets, partial release can release unused space across multiple volumes, from the high used RBA (or next CA/MCU boundary) to the high allocated RBA. Partial release requires that the primary volumes in the data set be in ascending order by RBA. For example, the first volume could have RBAs 1 to 1000, the next 1001 to 2000, and so on. If the primary volumes appear out of order, partial release issues an error message and releases no space.





## z/OS UNIX-related applications

The Network File System (NFS) is a distributed file system that enables you to access UNIX files and directories that are located on remote computers as if they were local. NFS is independent of machine types, operating systems, and network architectures.

IBM Ported Tools for z/OS is a collection of several OpenSource products such as OpenSSH, Perl, PHP, bzip2, or cURL that have been ported to be used with z/OS just as other IBM program products.

OpenSSH provides secure encryption for both remote login and file transfer. It consists of several utilities such as ssh, a z/OS client program for logging into a z/OS shell, scp for copying files between networks, sftp for file transfers over an encrypted ssh transport, and sshd, a daemon program for ssh that listens for connections from clients.

The new version of the OpenSSH program product can be installed on z/OS V1R10 and later.

This chapter provides new information about z/OS UNIX System Services-related applications with z/OS V1R12:

- ▶ Network file system services
- ▶ OpenSSH

## 17.1 Network File System (NFS)

With z/OS V1R12, the following enhancements have been made to NFS:

- ▶ z/OS UNIX System Services mmap support for an NFS client
- ▶ NFS server new SMF records for file operations
- ▶ NFS server cache monitoring and reporting
- ▶ NFS server password phrase support for the MVSlogin client utility
- ▶ The nfsstat shell program displays the accounting statistics of all the z/OS NFS servers.

## 17.2 z/OS UNIX System Services mmap support for NFS client

Before z/OS V1R12, mmap() to an NFS Client file resulted in program error “JrNotSupportedforRemoteFile”.

With z/OS V1R12, file system interfaces were redesigned to allow all hierarchical file system types (HFS, NFS, TFS, and zFS) to be supported for mmap() using the same internals. Therefore, this now allows that NFS files can be memory mapped.

**Note:** This support is transparent. Existing mmap and NFS externals still apply.

This means that a POSIX application currently using mmap() to memory map a file does not need to be changed when the file happens to be accessed via NFS.

## 17.3 NFS server new SMF records for file operations

Prior to the support in this new release, there is insufficient information when running with Security(None) or Security(EXPORTS) to determine who created, removed, or renamed an object on an NFS-mounted file system, either z/OS UNIX or MVS.

The NFS server did not provide SMF recording for auditing and security reasons. An audit trail was required specifying the user who created, removed, or renamed the file on the NFS-mounted file system. In addition, the NFS server did not generate SMF records for the creation, deletion, and renaming of MVS data sets and members.

A new SMF record type 42 subtype 26 is used to provide additional SMF recording for every create, remove, and rename NFS file system operation of the z/OS NFS server in V1R12.

Record type 42 subtype 26 is a record that is written when a client creates, removes, or renames the file objects on the NFS-mounted file system. It provides the NFS client's information, the type of operation (create, remove, rename) and object-descriptive information (depending on file system type: MVS or z/OS UNIX). For z/OS UNIX objects, the file system name, device number, object name, FID, and parent FID information are saved. For MVS objects, the volume name, full data set name, and member name (if appropriate) are saved.

**Note:** This new support provides needed audit capability and enhanced serviceability. Due to the performance impact, the “path” will not be saved for z/OS UNIX objects.

## Using the new support

There are two new levels for the NFS SMF site attribute:

**audit** Enables the SMF recording of the create, remove, and rename of a file

**all** Enables all the NFS server SMF recording types.

The **showattr** command has been enhanced to display these new levels. The SMF data collection can be controlled in the following ways:

- ▶ By the SMF site attribute:

```
smf(none | all | userfile | <subtype list> [,on|off])
```

- ▶ By the operand SMF of the MODIFY operator command:

```
MODIFY mvsnfs,smf=(none | all | <subtype list> [,on|off])
```

## Explanation of level specification

The specified levels are described as following:

**none** No SMF records are to be produced.

**all** All SMF NFS type 42 records are to be produced.

**userfile** Both the user session and file usage SMF records are to be produced.

Alternately, a list of levels that are delimited by commas can be specified. At least one of the levels (user, file, or audit) must be specified for a subtype list, and the remaining levels are optional, as follows:

**file** File usage SMF records (subtype 7) are to be produced.

**user** User session SMF records (subtype 8) are to be produced.

**audit** File creation, removal, and rename records (subtype 26) are to be produced.

There is an optional switch support to enable or disable the activation of the SMF records:

**off** Activation of SMF records collection can be done manually by using the **modify** command.

**on** Activates the SMF record collection.

The SMF settings for the NFS SMF record 42 and the new subtype 26 can also be specified in the SYS1.PARMLIB(SMFPRMxx) member. The new operand SMF for the MODIFY operator command is provided to enable changing the SMF site attribute settings for the SMF record type 42 without requiring a server restart.

## Examples

NFS server site attribute values activating the new SMF recording:

```
SMF(AUDIT,ON)  
SMF(ALL,ON)
```

Operator commands activating the new SMF recording:

```
MODIFY mvsnfs,SMF=AUDIT,ON  
MODIFY mvsnfs,SMF=ALL,ON
```

## Compatibility support for previous NFS releases

APAR OA31407 is required for prior releases to accommodate the changed syntax of the SMF site attribute and to ignore the newly defined additional values in the SMF site attribute.

## 17.4 NFS server cache monitoring and reporting

In the past there was no mechanism in NFS server code to alert at a console about impending legacy data buffer shortage problems. So there was the possibility of data buffers reaching a critical level and generating buffer shortage errors.

This problem has been solved in the z/OS V1R12 NFS version:

- ▶ New console messages warn about z/OS Network File System server legacy data buffer shortage and relief.

This addresses requirement MR0929084455.

**Note:** System automation routines can be configured to act upon these warning messages.

- ▶ There is the ability to track current legacy data buffer utilization and to change the maximum legacy data buffer pool size and warning thresholds from the console.
- ▶ It is now easier to tune z/OS NFS server attributes for the current workload.

### Changing NFS buffer size values

There are new console commands available:

```
MODIFY nfsserver, bufhigh=(xx,yy)
MODIFY nfsserver, bufhigh
MODIFY nfsserver, bufferusage
```

The first command is used to change current bufhigh values.

- xx** This is a new value attribute of the high water mark of data buffers in bytes, Kbytes, or Mbytes. The new value must not be less than the current value.

The upper boundary of the value depends on the amount of free space in the job region. If the new value is omitted in the command, the current value of the first argument is not changed.

- yy** This is the warning threshold in percent of the first value for printing a data buffer utilization alert message. The new value must be within the range 50 to 90, or 0. Use zero to switch off the cache monitoring and reporting mechanism.

**Note:** If the specified value for **xx** is less than the current value, then the command is failed and an error message (GFSA386E) is displayed.

### New console messages

There are several new console messages:

```
GFSA383I (mvsnfs) Network File System server data buffer utilization has
reached yyy% of xxxxxx MB
GFSA384I (mvsnfs) Resource shortage for Network File System server data
buffers has been relieved
GFSA385I (mvsnfs) Current data buffer utilization is xxxx MB
GFSA386E (mvsnfs) Command error: ERROR DESCRIPTION
GFSA387I (mvsnfs) The current attribute settings for bufhigh are ( XX MB, YY% )
```



## Using the new functions

The bufhigh server processing attribute is now extended to consist of two values:

```
bufhigh ( xx , yy )
```

In this specification the values xx and yy have the following meaning, being consistent with the values being specified on the MODIFY command:

- xx** Sets the high water mark (in bytes) of the data buffer.
- yy** Sets the watermark in percent of xx for printing a data buffer utilization alert message.

**Note:** With the following specification rules:

- ▶ Any invalid value of yy is converted to 80 percent.
- ▶ The default value of yy is 80 percent.
- ▶ If yy is omitted, the default value of yy is used.

## Examples using bufhigh()

The default value for bufhigh server site alerting threshold attribute is 80% of the maximum legacy data buffer pool size. Therefore, the following two settings in the server attributes data set are equal:

```
Bufhigh ( 100 M )  
Bufhigh ( 100 M, 80 )
```

This means that the console operator sees message GFSA383I on the system console if current legacy data buffer utilization is 80 MB:

```
GFSA383I (nfsserver) Network File System Server data buffer utilization has  
reached 80% of 100 MB
```

Message GFSA384I is seen if it falls below the alerting threshold value:

```
GFSA384I (nfsserver) Resource shortage for Network File System server data  
buffers has been relieved
```

The legacy data buffer monitoring could be switched off in the server attribute file:

```
bufhigh ( xx, 0 )
```

## Examples using bufhigh() with a MODIFY command

The legacy data buffer monitoring could be switched off from a console:

```
F mvsnfs, bufhigh=( , 0)
```

Use the following commands to modify current values from the console:

- ▶ To increase the maximum legacy data buffer pool size to 150 MB and change the alerting threshold to 60%, use the following command:

```
F nfsserver, bufhigh=( 150 M, 60 )
```

- ▶ To change only the first value, use:

```
F nfsserver, bufhigh=( 200 M )
```

- ▶ To change only the second value, use:

```
F nfsserver, bufhigh=( , 85 )
```

**Note:** From the console, the maximum legacy data buffer pool size can only be increased.

To check current legacy data buffer utilization on the console, use the command shown in Figure 17-1.

```
F nfsserver, bufferusage
GFS385I (nfsserver) Current data buffer utilization is .xxx MB
```

Figure 17-1 Checking current legacy data buffer utilization

To check the current bufhigh settings, use the command shown in Figure 17-2.

```
F nfsserver, bufhigh
GFS387I (nfsserver) The current attribute settings for bufhigh are ( XXX MB, YY % )
```

Figure 17-2 Checking current bufhigh settings

### Error message when changing bufhigh()

In case of any error in a bufhigh change command, an error message is printed as shown in Figure 17-3.

```
F nfsserver, bufhigh=( , 9999)
GFS386E (nfsserver) Command error: the second argument is out of range
```

Figure 17-3 Wrong bufhigh change command

In this message, the detailed error message can be one of the following:

- ▶ The first argument is out of range.
- ▶ The second argument is out of range.
- ▶ Wrong syntax.
- ▶ Invalid or omitted value.

### Compatibility support for previous NFS releases

The toleration APAR OA31407 is required for a prior release to accommodate the changed syntax of the bufhigh site attribute (AlertPercent) and ignore the newly defined option in the bufhigh site attribute.

## 17.5 Password phrase support for the MVSlogin client utility

In System Authorization Facility (SAF) or SAF and EXPORT (SAFEXP) mode, clients are required to issue an **mvslogin** command with the RACF user ID and password. Starting with z/OS V1R8 RACF has supported password phrase as a better security alternative than a password.

**Note:** Password phrase can be 9 - 100 characters. When ICHPWX11 is not present, the password phrase must be 14 - 100 characters. Contact your system programmer to find out whether your installation uses the new-password-phrase exit (ICHPWX11).

Now with V1R12 the NFS server also supports password phrases and allows a much longer value with a larger character set than the old 8-byte passwords that were the limit of NFS in previous releases.

- ▶ A password phrase can be defined for a RACF user in addition to the user's password.
- ▶ The user can authenticate with the defined password phrase when using the `mvslogin` command.
- ▶ The user can change from the current password phrase to a new password phrase using the `mvslogin` command.
- ▶ The user can authenticate with either password or password phrase.

**Note:** This provides increased security.

## Using the new support

To authenticate with a password phrase, the user can issue the `mvslogin` command with the password phrase enclosed in single quotation marks ('):

```
mvslogin -P 'passphrase' hostname(mvs user ID)
```

**Where:** `-P 'passphrase'` - No prompt for your z/OS password phrase, is a text string of a minimum of 9 to 14 characters (depending on whether or not ICHPWX11 is installed) to a maximum of 100 characters. This enables you to automate your MVS login. See *z/OS Security Server RACF Command Language Reference, SA22-7687* for more information on the z/OS password phrase and its syntax rules.

The message shown in Figure 17-4 is posted on the user console when the user tries to change from the current password to a new password phrase and vice versa, which is not supported by RACF.

```
GFS954I Host w42k returned error 129: The new password phrase cannot be set
when specifying a password, nor can a new password be set when specifying a
password phrase. Try again.
```

Figure 17-4 Message GFS954I with error 129

To minimize the impact to the user interface of the `mvslogin` command, the same operands (`-P`, `-n`, `-np` and so on) are used for both passwords and password phrases.

**Note:** There are some special cases:

- ▶ For a password phrase that has single quotation mark ('), enclose it with a double quote ("").
- ▶ Some characters, such as `!`, `\`, `/` and `$`, have special meanings to UNIX shells. If those characters are used as part of the password phrase, users need to follow certain rules to escape their special meanings.

## Installation considerations

The following steps need to be done to enable password phrase support:

- ▶ Modify `sys1.parmlib(IKJTS0xx)` to add the passphrase activation statement:  
LOGON PASSPHRASE(ON)
- ▶ ReIPL the system for the change to take effect or issue this command:  
parm lib update(xx)
- ▶ Make sure that using passphrases is on by looking in the output list after issuing:  
parm lib list

- ▶ Define an initial password phrase for the RACF user:  
ALTUSER USERX PHRASE('password phrase') NOEXPIRED
- ▶ Download the client utility code and run `makefile` to build the executable program.

## 17.6 NFS Server displays the accounting statistics

The `nfsstat` shell program, located in `/usr/lpp/NFS`, displays the accounting statistics of all the z/OS NFS servers within an LPAR along with the accounting statistics of the z/OS NFS client.

The accounting statistics are:

- ▶ The number of received Remote Procedure Calls (RPC)
- ▶ The number of the received Network File System (NFS) Version 2, or Version 3 procedures, or Version 4 operations.

The accounting statistics provide insight into the interactions between clients and servers. Before V1R12, `nfsstat` displayed the z/OS NFS Client RPC and NFS accounting statistics.

**Note:** Before V1R12, the `nfsstat` shell program:

- Did not provide any z/OS NFS server statistics.
- Was unaware of workloads presented by other NFS clients to the z/OS NFS servers.

The `nfsstat` program on other platforms provides both Server and Client RPC and NFS statistics, NFSv2 and NFSv3 procedures, and NFSv4 operations. With V1R12, NFS provides accounting statistics of all NFS servers in an LPAR.

**Note:** This provides awareness of NFS clients and servers processing.

### Options for `nfsstat` in previous releases

An old version of the `nfsstat` program interfaces with the started z/OS NFS client to report its statistics through the following options:

- c** Reports client RPC and NFS statistics
- r** Reports client RPC statistics
- n** Reports client NFS statistics
- z** Reports client RPC and NFS statistics and reset to zero
- d** Flushes log, switches log, resets log
- v** Reports z/OS NFS client service and maintenance level

Issuing `nfsstat` without any option is equivalent to using `nfsstat -c`.

### New and enhanced `nfsstat` options

The `nfsstat` program still interfaces with the started z/OS NFS client, and it additionally interfaces with all the started z/OS NFS servers to report the server statistics through the new and enhanced options:

- s** Reports server RPC and NFS statistics (new option)
- r** Reports server and client RPC statistics (enhanced option)

- n** Reports server and client NFS statistics (enhanced option)
- z** Reports server and client RPC and NFS statistics and reset to zero
- 4** Reports server and client NFSv4 statistics (new option)
- 3** Reports server and client NFSv3 statistics (new option)
- 2** Reports server and client NFSv2 statistics (new option)

Issuing `nfsstat` without any option is equivalent to using `nfsstat -cs`, reporting RPC and NFS statistics of all started z/OS NFS servers and the started z/OS NFS client.

Other options (`-d`, `-v` and `-m`) related to the z/OS NFS client information are not affected.

### Output of `nfsstat` commands

The general statistics output has the following data:

- ▶ The headline GFSC857I identifying the client or server with the started procedure name:
  - GFSC857I z/OS Network File System Server (nfsserver)
  - GFSC857I z/OS Network File System Client
- ▶ The RPC statistics, if selected
- ▶ The NFSv2 statistics, if selected
- ▶ The NFSv3 statistics, if selected
- ▶ The NFSv4 statistics, if selected
- ▶ Each counter is a 4-byte unsigned integer.

### Displaying RPC statistics

To display z/OS NFS client RPC statistics, use:

```
/usr/lpp/NFS/nfsstat -cr
```

To display all z/OS NFS server RPC statistics, use:

```
/usr/lpp/NFS/nfsstat -sr
```

### Displaying NFS V3 statistics

To display z/OS NFS client NFSv3 statistics, use:

```
/usr/lpp/NFS/nfsstat -c3
```

To display all z/OS NFS server NFSv3 statistics, use:

```
/usr/lpp/NFS/nfsstat -s3
```

### Displaying NFSv4 statistics

To display z/OS NFS client NFSv4 statistics, use:

```
/usr/lpp/NFS/nfsstat -c4
```

To display all z/OS NFS server NFSv4 statistics, use:

```
/usr/lpp/NFS/nfsstat -s4
```

### Resetting statistics

To display z/OS NFS client statistics, then reset them, use:

```
/usr/lpp/NFS/nfsstat -cz
```

To display all z/OS NFS server statistics and then reset them, use:

```
/usr/lpp/NFS/nfsstat -sz
```

## Migration and coexistence considerations

Note the following migration information:

- ▶ The options `-c`, `-r`, `-n` and so on of previous `nfsstat` versions are still valid.
- ▶ It may be necessary to add the `-c` option to limit the display to the z/OS NFS client statistics:
  - Listing z/OS NFS server and client RPC statistics:  
`nfsstat -r`
  - Listing z/OS NFS client RPC statistics:  
`nfsstat -cr`
  - Listing z/OS NFS server RPC statistics:  
`nfsstat -sr`
- ▶ Coexistence of the `/usr/lpp/NFS/nfsstat` program is not provided:
  - The new `nfsstat` options `-s`, `-2`, `-3` and `-4` are not applicable with the `nfsstat` versions in the older releases.

## 17.7 IBM Ported Tools for z/OS: OpenSSH

With z/OS V1R12, the following enhancements have been made to OpenSSH:

- ▶ Upgrade OpenSSH to version 5.0p1
- ▶ Provide SMF logging for OpenSSH
- ▶ Provide RACF keyring support in OpenSSH

### 17.7.1 Upgrade OpenSSH to version 5.0p1

The new IBM Ported Tools version of z/OS OpenSSH V1R2 is an upgrade to OpenSSH 5.0p1, OpenSSL 0.9.8k, and zlib 1.2.3.

**Note:** OpenSSH provides secure encryption for both remote login and file transfer. OpenSSL is an open source implementation of the Secure Sockets Layer (SSL) and Transport Layer Security (TLS) protocols. OpenSSH uses zlib, a data compression library.

- ▶ It addresses various functional, performance, and security requirements.
- ▶ It is recompiled with XPLINK to improve overall performance.

For more details about z/OS OpenSSH V1R2 and how to specify specific settings, see *IBM Ported Tools for z/OS: OpenSSH User's Guide*.

#### Configurable timeout value for `ssh-rand-helper`

The new `_ZOS_SSH_PRNG_CMDS_TIMEOUT` environment variable provides improved `ssh-rand-helper` support on heavily loaded systems.

This addresses the following requirements:

- OR0605084836** OpenSSH should externalize the ssh-rand-helper timeout value.
- MR0426073731** There is the need to be able to adjust the timeout value used by ssh-rand-helper.
- MR1212063936** OpenSSH should provide a configuration option for ssh-rand-helper timeout.

### OpenSSH message support improvement

OpenSSH provides a new `_ZOS_OPENSSH_MSGCAT` environment variable and all error-related messages are now documented. This enables quicker identification of problems.

The following requirements have been addressed:

- MR1107085910** The OpenSSH message catalog file is not automatically picked up by default NLSPATH as shipped in `/samples/etc/profile`.
- MR110308412** OpenSSH should document all its messages.
- MR1119092155** Ported Tools should document all messages that can be displayed with proper message identifiers.

### Improved support for shared UIDs

z/OS OpenSSH had the need to improve support for users that share a UID. The current MVS identity is used now to determine the user name and initial working directory.

**Note:** This improves ssh, ssh-add, ssh-keygen, ssh-rand-helper, and sshd functionality for users that share a UID.

The following requirements have been addressed:

- MR1103065717** OpenSSH should support a parameter for the sshd home directory.
- MR0120054254** OpenSSH should support a parameter for the sshd home directory.
- MR0327072218** OpenSSH should support a parameter to specify the location of sshd home directory.

### Connection sharing support

OpenSSH provides new `ssh` command line options `-M`, `-0` and `-S`. Furthermore, there are new `ssh_config` keywords `ControlMaster` and `ControlPath`.

Figure 17-5 on page 338 shows Illustration #1: Connection sharing support.

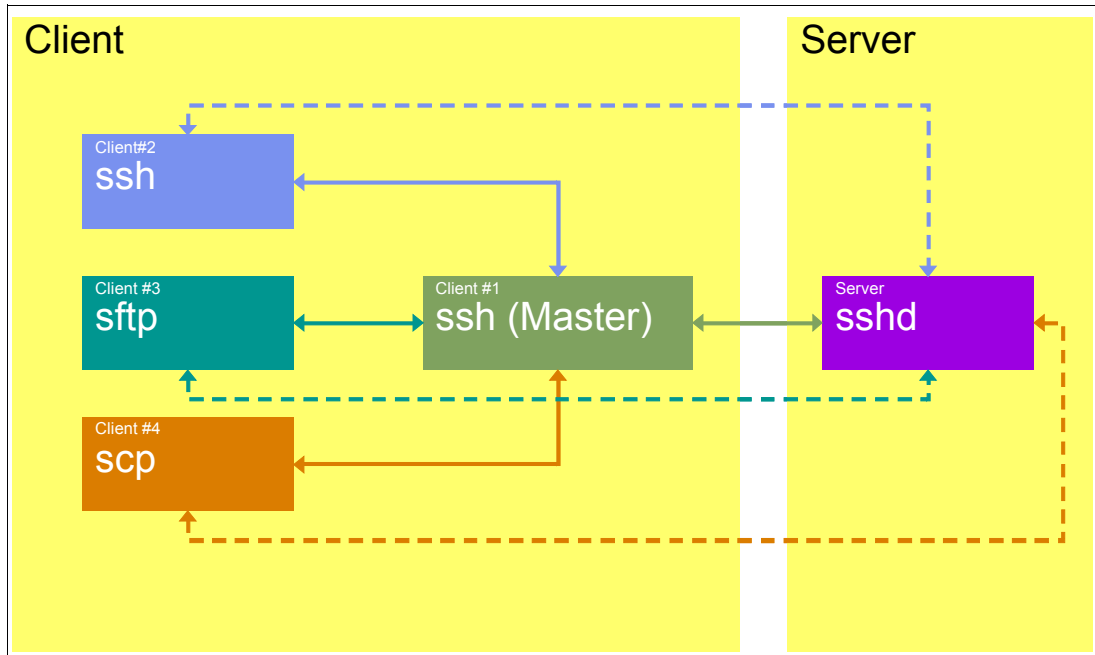


Figure 17-5 Illustration: Connection sharing support

- ▶ Client #1 establishes an authenticated connection to the server. The `ControlMaster` and `ControlPath` `ssh_config` keywords are used for this:
 

```
ssh -oControlMaster=yes -oControlPath=~/.ssh/mycontrolpath myserver
```
- ▶ Client #2 uses the already authenticated connection over Client #1 (solid line) to connect to the server instead of establishing a new authenticated connection itself (dashed line). The `ControlPath` `ssh_config` keyword is used for this:
 

```
ssh -oControlPath=~/.ssh/mycontrolpath myserver
```
- ▶ Client #3 does the same as Client #2 by using `sftp` instead of `ssh`:
 

```
sftp -oControlPath=~/.ssh/mycontrolpath myserver
```
- ▶ Client #4 does the same as Client #2 by using `scp` instead of `ssh`:
 

```
scp -oControlPath=~/.ssh/mycontrolpath a myserver:b
```

**Note:** This improves performance of connections with a short lifespan. These connections can avoid the overhead of the authentication process.

Only a single authentication is done for all connections. The `ControlMaster` `ssh_config` keyword documents additional options for security.

**Important:** You should be aware of the security issues of this. Also, be sure to use a secure `ControlPath`.

### Restricted environment support

OpenSSH provides new `sshd_config` `ChrootDirectory`, `ForceCommand` and `Match` keywords. Furthermore, there is a new “internal-sftp” support. For more information, see the description of the `sshd_config` `ForceCommand` and `Subsystem` keywords.

Figure 17-6 on page 339 shows Illustration #2: Restricted environment support.



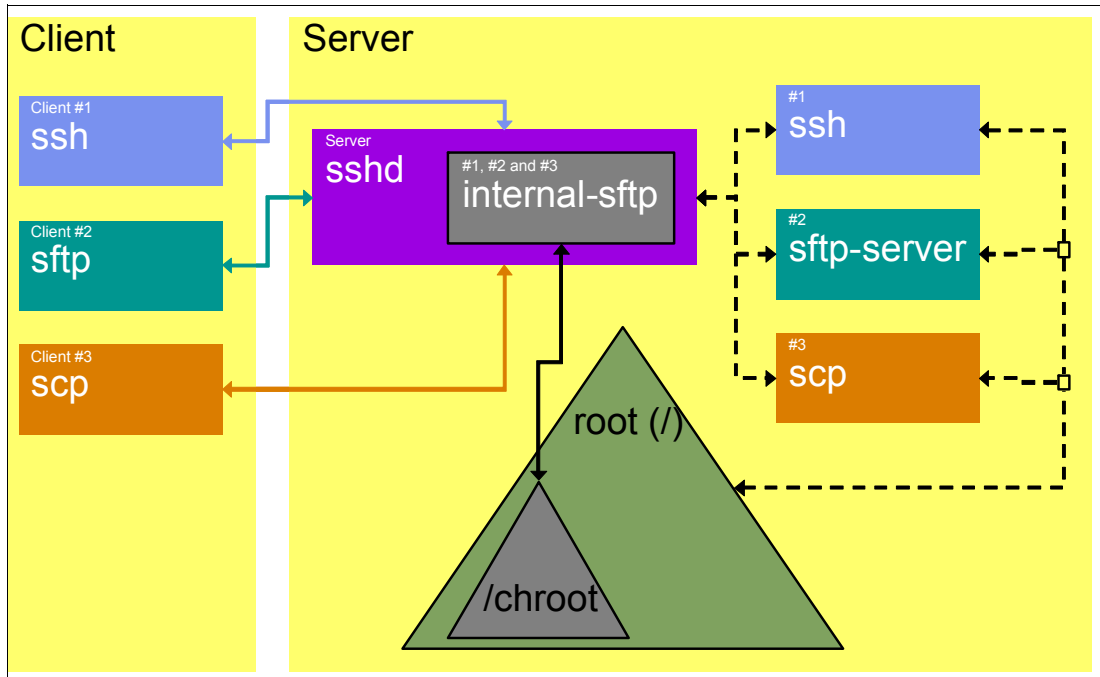


Figure 17-6 Illustration #2: Restricted environment support

If the server sets the `sshd_config` keywords as shown in Figure 17-7, this forces all users in the `SFTPUSEERS` group to doing an “internal-sftp”. They can only work with the file system data under the changed root directory `/changed.root` as set in the `ChrootDirectory` statement.

```
Match group SFTPUSEERS
ChrootDirectory /changed.root
ForceCommand internal-sftp
```

Figure 17-7 Excerpts of an SSHD configuration file

### Hashed hostname and address support

Beginning with the release of OpenSSH Version 4.0, a new configuration directive was introduced, shown in Figure 17-8.

```
HashKnownHosts yes
```

Figure 17-8 OpenSSH HashKnownHosts configuration

When you set this option in your `ssh_config` file, `ssh` will start recording a one-way cryptographic hash of the hostname and IP address rather than recording them in clear text.

Later, when initiating subsequent `ssh` sessions, your `ssh` client hashes the hostname you type and looks it up in the `known_hosts` file to find the previously recorded `ssh` host key for the remote server. You will effectively still be able to automatically verify `ssh` host keys, but attackers will not be able to harvest hostnames and IP addresses from the `known_hosts` file.

Figure 17-9 on page 340 shows Illustration #3: Hashed hostnames and addresses.

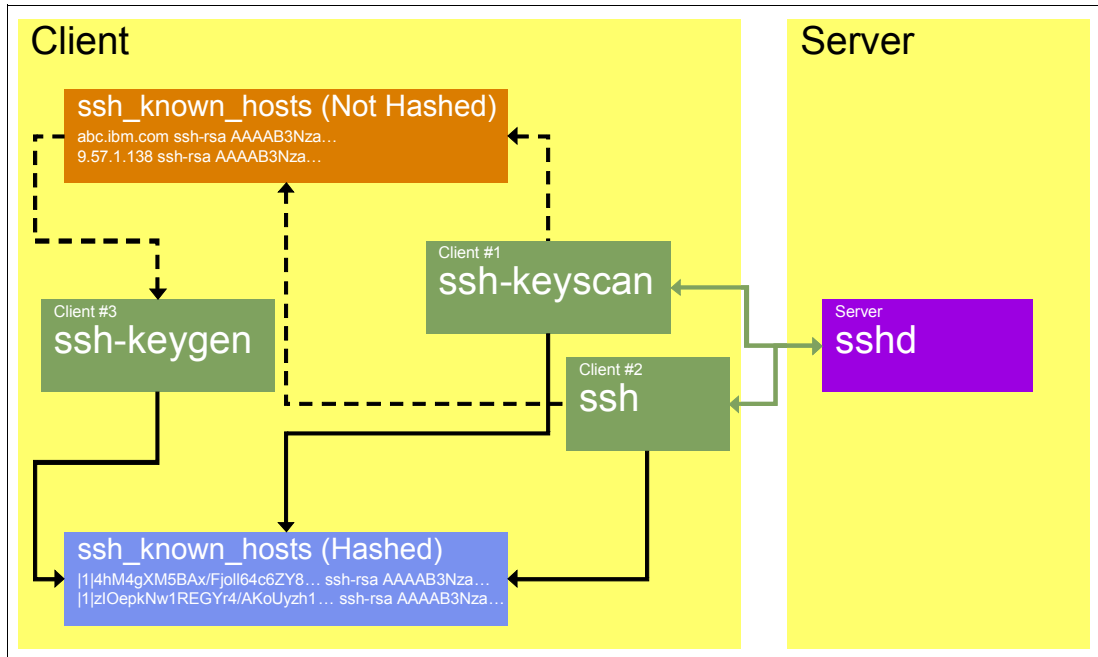


Figure 17-9 Illustration #3: Hashed hostnames and addresses

The solid lines indicate that hostnames and addresses are hashed. The dashed lines indicate that hostnames and addresses are not hashed.

### z/OS OpenSSH hash support

In supporting this, OpenSSH provides new options for several commands and a new file format:

- ▶ New `ssh-keyscan -H` command-line option
- ▶ New `ssh-keygen -F, -H` and `-R` command-line options
- ▶ New `ssh_known_hosts` file format support for hashed hostnames and addresses
- ▶ New `ssh_config HashKnownHosts` keyword

The new support can be used as follows:

- ▶ Using `ssh-keyscan` with the new `-H` command line option results in hashing the hostnames and addresses in the output. The output can then be redirected to an `ssh_known_hosts` file.
- ▶ When using `ssh`, the new `ssh_config HashKnownHosts` keyword causes it to hash the hostnames and addresses added to the `~/.ssh/known_hosts` file.
- ▶ Using `ssh-keygen` with the new `-H` command line option results in hashing the hostnames and addresses in the `ssh_known_hosts` file. The “unhashed” file is renamed as `*.old`. Figure 17-10 shows this command.

```
#> ssh-keygen -H -f /etc/ssh/ssh_known_hosts
#>
```

Figure 17-10 Hashing the system-wide `ssh_known_hosts` file

**Important:** Do not forget to delete `/etc/ssh/ssh_known_hosts.old`.

## 17.7.2 SMF logging for OpenSSH

z/OS OpenSSH now audits file transfers and login failures:

- ▶ SMF records are generated for both client and server file transfers.
- ▶ SMF records are generated for login failures.
- ▶ A new SMF server transfer completion record has been introduced:  
Type 119 - Subtype 96
- ▶ A new SMF client transfer completion record has been introduced:  
Type 119 - Subtype 97
- ▶ A new SMF login failure record has been introduced:  
Type 119 - Subtype 98

This support addresses a lot of requirements that had been opened.

### SMF records auditing OpenSSH activity

OpenSSH now creates SMF records for auditing OpenSSH activity:

- ▶ New SMF records are customized for z/OS OpenSSH.
- ▶ Support has been added for SMF record exits IEFU83 or IEFU84.

**Note:** This implementation does not require permitting all OpenSSH users to the BPX.SMF facility class profile.

### Using the new support

OpenSSH now supports client transfer completion records (SMF type 119, subtype 97):

- ▶ OpenSSH scp and sftp write client transfer completion records.
- ▶ This is enabled via the ClientSMF keyword in the new `zos_ssh_config` configuration file:  
`ClientSMF TYPE119_U83`

Also, server transfer completion records are supported (SMF type 119, subtype 96):

- ▶ OpenSSH sftp-server, scp and sshd write server transfer completion records via “internal-sftp”.
- ▶ This is enabled via the ServerSMF keyword in the `zos_sshd_config` configuration file:  
`ServerSMF TYPE119_U83`

Recording of login failures is done as follows (SMF type 119, subtype 98):

- ▶ OpenSSH sshd writes login failure records.
- ▶ This is enabled via the ServerSMF keyword in the `zos_sshd_config` configuration file:  
`ServerSMF TYPE119_U83`

Figure 17-11 on page 342 shows Illustration #4: SMF records audit OpenSSH activity.

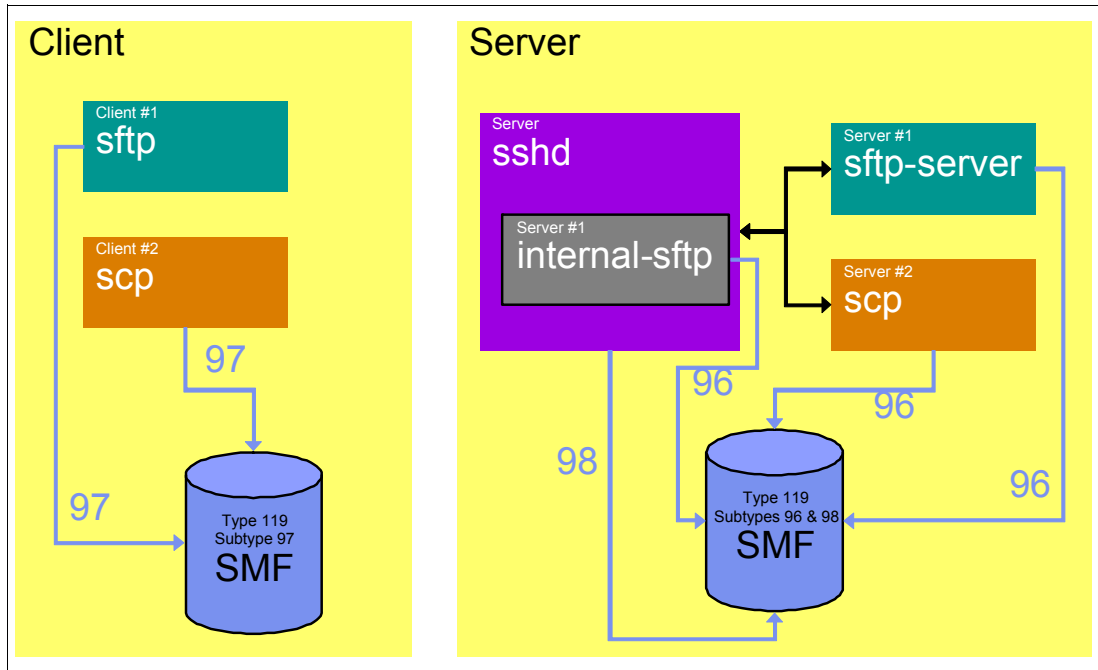


Figure 17-11 Illustration #4: SMF records audit OpenSSH activity

### 17.7.3 Providing RACF key ring support in OpenSSH

OpenSSH keys can now be stored in a digital certificate connected to an SAF key ring. New features are available for ssh, scp, sftp, ssh-add, ssh-keygen, and sshd to get keys from an SAF key ring.

- ▶ SAF control of OpenSSH keys is available for SSH protocol Version 2.
- ▶ Server authentication when keys are stored in key rings is supported.
- ▶ User authentication is supported when keys are stored in key rings
- ▶ Even mixing key storage, key rings, and UNIX files, is supported.
- ▶ Real or virtual key rings are supported.
- ▶ Additional features are available, for example expired certificate and signing.

**Restriction:** If you are using SSH protocol Version 1, you cannot use SAF key rings to hold your keys. You must use UNIX files to hold RSA keys used for SSH protocol Version 1.

The following requirements have been addressed:

- |                     |  |
|---------------------|--|
| <b>MR0913046227</b> | OpenSSH needs RACF key ring support.   |
| <b>MR0627075732</b> | For OpenSSH, the association between a user ID and the public key should be controlled by RACF.  |
| <b>MR0505065826</b> | OpenSSH should support x.509 certificates.   |
| <b>MR1020045932</b> | Ported Tools should have X509 Digital Certificate support, preferably using the SAF interface so that the certificates can be managed in the security product. |

MR0128051745

OpenSSH should have X509 Digital Certificate support using the SAF interface so that the certificates can be managed in the security product.

### Using RACF key rings

OpenSSH has included the support in several utilities used in various situations:

- ▶ OpenSSH supports server authentication with key rings:
  - This is ssh enabled via the new `zos-key-ring-label` option for the `ssh_known_hosts` file format.
  - This is sshd enabled via the new `HostKeyRingLabel` keyword in the new `zos_sshd_config` configuration file.
- ▶ OpenSSH supports user authentication with key rings:
  - This is ssh enabled via the new `IdentityKeyRingLabel` keyword in the new `zos_user_ssh_config` configuration file.
  - This is ssh-add enabled via the new `_ZOS_SSH_KEY_RING` and `_ZOS_SSH_KEY_RING_LABEL` environment variables.
  - This is ssh-keygen enabled via the new `_ZOS_SSH_KEY_RING_LABEL` environment variable.
  - This is sshd enabled via the new `zos-key-ring-label` option for the `authorized_keys` file format.
- ▶ OpenSSH also supports mixed keys: you can have keys in key rings together with keys in UNIX files.

### Server authentication with key rings

Figure 17-12 shows Illustration #5: Server authentication with key rings.

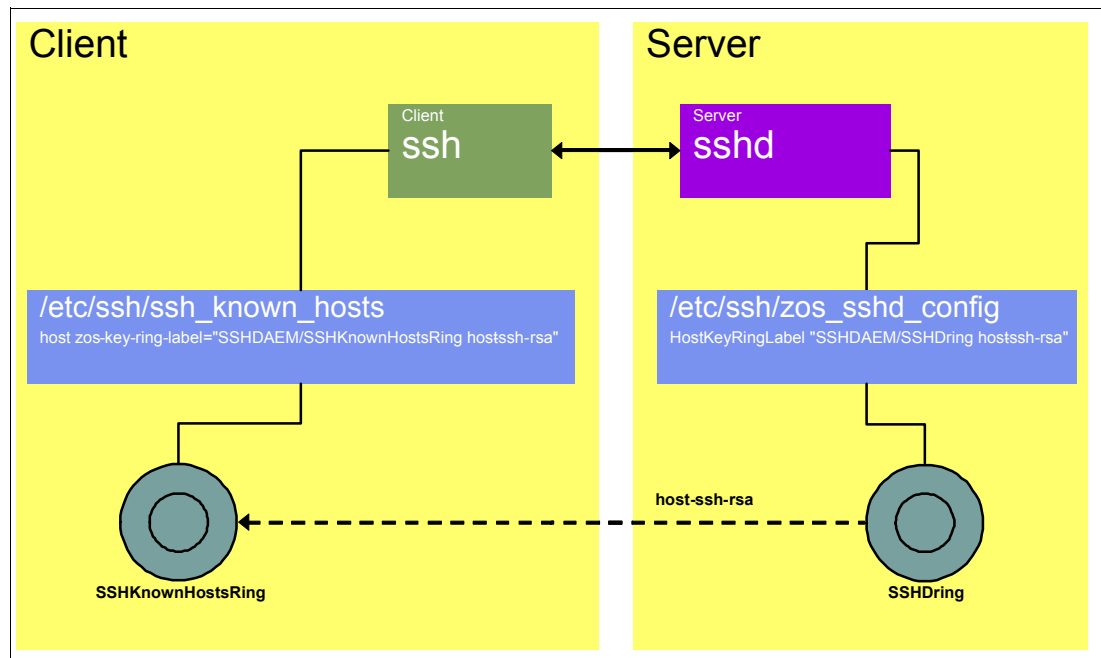


Figure 17-12 Illustration #5: Server authentication with key rings

We now show how to set up and use this.

- ▶ Server authentication setup when keys are stored in key rings:
  - a. On the server: Create the host key ring (SSHDring).
  - b. On the server: Generate a host certificate with public and private keys (host-ssh-rsa).
  - c. On the server: Connect the host certificate to the host key ring.
  - d. On the server: Add the HostKeyRingLabel line to the /etc/ssh/zos\_sshd\_config file.
  - e. On the server: Export the host certificate in DER format without the private key.
  - f. On the client: Create the known hosts key ring (SSHKnownHostsRing).
  - g. On the client: FTP the host certificate from the server and add the exported host certificate.
  - h. On the client: Connect the host certificate to the known hosts key ring.
  - i. On the client: Add the zos-key-ring-label line to the /etc/ssh/ssh\_known\_hosts file.

**Note:** See “Steps for setting up server authentication when keys are stored in key rings” in *IBM Ported Tools for z/OS User's Guide* for more details.

- ▶ Server authentication:
  - a. sshd gets the RSA host key during startup from the SSHDring key ring.
  - b. ssh gets the public RSA host key during server authentication from the SSHKnownHostsRing key ring.
  - c. Server authentication continues unaffected by where the keys came from, either a UNIX file or an SAF key ring.

### User authentication with key rings

Figure 17-13 shows Illustration #6: User authentication with key rings.

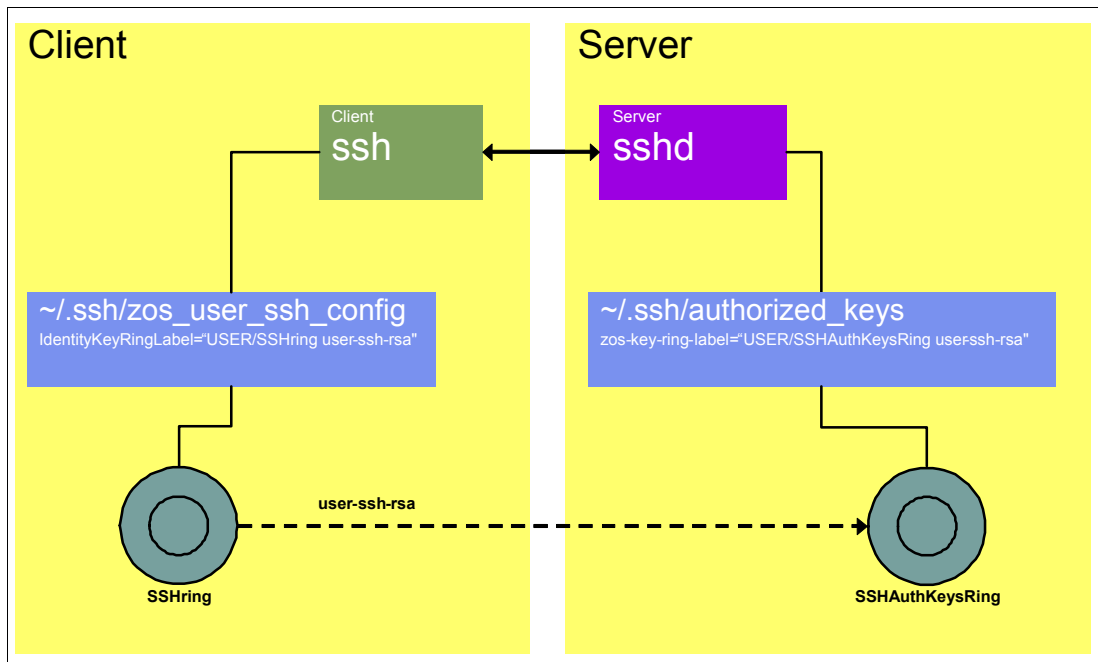


Figure 17-13 Illustration #6: User authentication with key rings

The following list shows how to set up and use this.

- ▶ User authentication setup when keys are stored in key rings:
  - d. On the client: Create the user key ring (SSHring).
  - e. On the client: Generate a user certificate with public and private keys (user-ssh-rsa).
  - f. On the client: Connect the user certificate to the user key ring.
  - g. On the client: Add the `IdentityKeyRingLabel` line to the `~/.ssh/zos_user_ssh_config` file.
  - h. On the client: Export the user certificate in DER format without the private key.
  - i. On the server: Create the authorized keys key ring (SSHAuthKeysRing).
  - j. On the server: FTP the user certificate from the client and add the exported user certificate.
  - k. On the server: Connect the user certificate to the authorized keys key ring.
  - l. On the server: Add the `zos-key-ring-label` line to the `~/.ssh/authorized_keys` file.

**Note:** See “Steps for setting up user authentication when using key rings to store keys” in *IBM Ported Tools for z/OS User's Guide* for more details.

- ▶ User authentication:
  - a. ssh gets the RSA user key during user authentication from the SSHring key ring.
  - b. sshd gets the public RSA user key during user authentication from the SSHAuthKeysRing key ring.
  - c. User authentication continues unaffected by where the keys came from, either a UNIX file or an SAF key ring.

## 17.7.4 OpenSSH migration and coexistence considerations

Here we show a large number of changes that need to be acknowledged, although most installations will not be affected by these migration actions most of which are the result of changes to the open source code.

All the descriptions of migration actions and coexistence considerations are assuming that the migration is done from z/OS OpenSSH V1R1 at level OpenSSH 3.8.1p1.

## 17.7.5 Incorrect owner or permissions of the ssh configuration file

On using a configuration file (by default `~/.ssh/config`), ssh now issues an error message and exits if the configuration file is not owned by the user or if the file is writable by the group owner or by other users.

**Migration action:** Correct the settings according to the new requirements.

**What is affected:** ssh commands

## 17.7.6 Relative path name on starting the sshd daemon

In OpenSSH before V1R2, the sshd daemon could be started using a relative path name (for example, `./sshd`). Now a full path name must be used instead.

**Migration action:** Change the startup process to use the full path name instead of a relative path name.

**What is affected:** sshd commands

### 17.7.7 Using the sftp command with the -b option and further interaction

When the **sftp** command is run with the **-b** option, the **-oBatchMode=yes** argument is now passed to the **ssh** command by default. This is a problem if entering a password or passphrase, or host key prompts are required during authentication.

**Migration action:** Run the **sftp** command with **-oBatchMode=no** as the first option.

**What is affected:** The **sftp -b** command line option

### 17.7.8 z/OS OpenSSH V1R2 is now an XPLINK application

Beginning in V1R2, z/OS OpenSSH is an XPLINK application. Extra Performance Linkage (XPLINK) is a type of call linkage that can improve performance in an environment of frequent calls between small functions.

**Migration action:** To set up the XPLINK environment (that is, to initialize the resources necessary to run an XPLINK application), do the following:

- ▶ Put the Language Environment run-time library SCEERUN2 in the LNKLST member of SYS1.PARMLIB.
- ▶ Put the XPLINK modules in SCEERUN2 in the dynamic LPA.

**What is affected:** All OpenSSH commands

### 17.7.9 Message numbers with OpenSSH error messages

In OpenSSH before V1R2, to associate message numbers (for example, FOTSnnnn) with OpenSSH error messages, the NLSPATH environment variable had to include the following path: `/usr/lib/nls/msg/%L/%N.cat`. Starting in Version 1 Release 2, message numbers for IBM Ported Tools for z/OS: OpenSSH are associated with OpenSSH error messages by default.

**Migration action:** Set environment variable `_ZOS_OPENSSH_MSGCAT="NONE"` before running an OpenSSH command. If you have previously modified the NLSPATH environment variable, you do not need to make any changes to it.

**What is affected:** All OpenSSH commands

### 17.7.10 Using an address containing delimiter characters

In OpenSSH before V1R2, addresses containing a colon (:) character could be parsed using the forward slash (/) character and vice versa. Now addresses containing delimiter characters (: or /) must be enclosed in square brackets.

**Migration action:** Enclose the address in square brackets.

**What is affected:** **ssh -L** and **-R** command-line options, **ssh\_config LocalForward** and **RemoteForward** keywords and the **permit open authorized\_keys** file format option



### 17.7.11 Using the default cipher list

In OpenSSH before V1R2, the default cipher list did not contain arcfour128 and arcfour256. Now the default cipher list contains arcfour128 and arcfour256. The order was also changed to prefer ciphers that are not susceptible to security vulnerability CVE-2008-5161.

If you used the previous default list and do not want to allow the new ciphers or the new order of the preferred ciphers, take the following migration actions.

**Migration action:** Specify the previous default list.

**What is affected:** `ssh -c` command-line option, `ssh_config` Ciphers keyword and `sshd_config` Ciphers keyword

### 17.7.12 Default MACs list

In OpenSSH before V1R2, the default MACs list did not contain `umac64@openssh.com`. Now it contains `umac64@openssh.com`.

**Migration action:** Specify the previous default list.

**What is affected:** `ssh -m` command-line option, `ssh_config` MACs keyword, and `sshd_config` MACs keyword

### 17.7.13 Port forwarding

In OpenSSH before V1R2, the `AllowTcpForwarding` default value was "yes". Now it is "no". This default was changed to reduce exposure to a vulnerability reported as CVE-2004-1653.

**Migration action:** Set `AllowTcpForwarding` to "yes".

**What is affected:** The `sshd_config` `AllowTcpForwarding` keyword

### 17.7.14 New minimum RekeyLimit value

In OpenSSH before V1R2, the `RekeyLimit` minimum value was 0. Now the minimum value is 16. A migration action is needed if you use a `RekeyLimit` value that is less than 16.

**Migration action:** Change the value so that the `RekeyLimit` value is greater than or equal to 16.

**What is affected:** The `ssh_config` `RekeyLimit` keyword

### 17.7.15 New minimum RSA key size

In OpenSSH before V1R2, the minimum RSA key size on the `ssh-keygen -b` option was 512 bits and the default was 1024 bits. Now the minimum RSA key size is 768 bits and the default is 2048 bits. The maximum remains 32,768 bits.

A migration action is needed if you are using `ssh-keygen` to generate RSA keys with a size that is less than 768 bits.

**Migration action:** Use `ssh-keygen` to generate new RSA keys based on the new size requirements.

**What is affected:** The `ssh-keygen -b` command line option

### 17.7.16 New DSA key size

In OpenSSH before V1R2, the DSA key size on the `ssh-keygen -b` option was allowed to be between 512 and 32,768 bits. Now the DSA key size must be 1024 bits.

A migration action is needed if you are using `ssh-keygen` to generate DSA keys with a size that is not equal to 1024 bits.

**Migration action:** Use `ssh-keygen` to generate new DSA keys based on the new size requirements.

**What is affected:** The `ssh-keygen -b` command line option

### 17.7.17 New shell environment variable for ProxyCommand

Instead of running `ProxyCommand` with `/bin/sh`, the user's shell as set in the `SHELL` environment variable is used.

A migration action is needed if you want to use a shell other than `/bin/sh`.

**Migration action:** Make sure that `ProxyCommand` conforms to your shell's syntax.

**What is affected:** The `ssh_config ProxyCommand` keyword

### 17.7.18 New behavior for specifying a file name on key generation

In OpenSSH before V1R2, if the file name was not specified, a prompt for the file name was issued. Now the default file names for RSA and DSA keys are used instead.

**Migration action:** Specify the file name on the `ssh-keygen` command.

**What is affected:** The `ssh-keygen -r` command line option

### 17.7.19 Long file names

Instead of truncating a long file name at 1023 characters, a message is issued.

**Migration action:** None.

**What is affected:** The `ssh-keygen -f` command line option

### 17.7.20 New behavior on key generation

In OpenSSH before V1R2, if `ssh-keygen` was issued without the `-d` or `-t` options, a message was issued. Now RSA is used as the default key type.

**Migration action:** None. Previously successful `ssh-keygen` commands will continue to be successful.

**What is affected:** The `ssh-keygen` command

## 17.7.21 Coexistence of old and new OpenSSH versions

In a sysplex environment, some systems might share the same configuration. They might also share the `ssh_known_hosts` or `authorized_keys` files. With different versions this might lead to problems because an old version does not support the new features.

When a newer version of the SSH client is trying to connect to a previous version of the `sshd` daemon, connection might not be established due to incompatibility of the new configuration options.

Options to avoid on sharing the files:

- ▶ **ssh\_config**: `ssh -F` command line option
- ▶ **sshd\_config**: `sshd -f` command line options
- ▶ **ssh\_known\_hosts**: `ssh_config` `GlobalKnownHostsFile` or `UserKnownHostsFile` keywords
- ▶ **authorized\_keys**: `sshd_config` `AuthorizedKeysFile` keyword

## 17.7.22 Installation of OpenSSH V1R2

Here we provide some information about installation of this new version of OpenSSH.

- ▶ The new release with FMID HOS1120 installs over the previous release with FMID HOS1110.
- ▶ You can no longer order the previous release once the new release is going GA (general availability).
- ▶ The new release is supported on z/OS V1R10 and later.
- ▶ For z/OS V1R10 and z/OS V1R11, the PTFs for APARs PK86329 and OA29401 are required:
  - PK86329: New function - Allow to use the new `__smf_record2()` C/C++ function in order to invoke a user exit IEFU83.
    - PTF for z/OS V1R10: UK51121
    - PTF for z/OS V1R11: UK51122
  - OA29401: New function - Allow a BPX1SMF caller to request an IEFU83 exit.
    - PTF for z/OS V1R10: UA50531
    - PTF for z/OS V1R11: UA50532

**Note:** SMP/E can be used to ensure that the PTFs are applied.

- ▶ Same ZONE: SMP/E will fail the installation if the PTFs are not applied unless the failure is explicitly bypassed.
  - ▶ Separate ZONE: SMP/E will successfully install OpenSSH. It is recommended to run a cross-zone report to ensure that the PTFs are applied.
- ▶ Important: extended attributes settings are set during install processing.
    - `sshd`, `scp`, `sftp`, and `sftp-server` must have the `APF-authorized` extended attribute set.
    - `ssh` and `ssh-keysign` must have the `noshareas` extended attribute set.
    - `sshd` must have the `noshareas` extended attribute set.
    - `sshd` must have the `program control` extended attribute set.
  - ▶ It is recommended to look to the “What you need to verify before using OpenSSH” section in the new *IBM Ported Tools for z/OS User's Guide* for details about installation steps.
  - ▶ `Xvfb` has been split off from OpenSSH. It is now documented in a separate book and has its own FMID.

**Note:** Xvfb is an X server that can run on machines with no display hardware and no physical input devices. It emulates a dumb frame buffer using virtual memory.



## BCP supervisor updates

This chapter discusses new BCP supervisor functions for z/OS V1R11. The following new functions and enhancements are explained:

- ▶ DISPLAY IPLINFO for zAAPzIIP
- ▶ Discretionary work timeslice
- ▶ MTTR instrumentation
- ▶ RTM enhancements
- ▶ Miscellaneous updates
  - D SYMBOLS
  - LCCA/PCCA health checks

## 18.1 zAAP on zIIP support

This chapter describes some changes to the ZAAPZIIP support that determine whether the system can run zAAP processor-eligible work on zIIP processors when no zAAP processors are installed on the machine. When you specify ZAAPZIIP=YES, the system can do that. When you specify ZAAPZIIP=NO, the system cannot do that.

### 18.1.1 zAAP on zIIP implementation with z/OS V1R12

IBM made support available for z/OS V1R11 with a new capability that can enable System z Application Assist Processor (zAAP) eligible workloads to run on System z Integrated Information Processors (zIIPs). This capability allowed customers to run zIIP and zAAP eligible workloads together on one type of specialty engine, the zIIP.

This capability is available with z/OS V1R11, z/OS V1R9, and V1R10 with a PTF for APAR OA27495 and is available on IBM System z9 and System z10 servers. Some restrictions apply.

#### System z specialty engines

Specialty engines are processors that can help users expand the use of the mainframe for new workloads while helping to lower their total cost of ownership. Since their introduction, IBM has continued to enhance specialty engine functionality and lower the cost per unit of work performed on the engines. Current System z specialty engines include:

- ▶ System z Application Assist Processor (zAAP)
- ▶ System z Integrated Information Processor (zIIP)
- ▶ The Integrated Facility for Linux (IFL)
- ▶ Internal Coupling Facility (ICF)
- ▶ System Assist Processor (SAP)
- ▶ zAAP on zIIP capability – value to clients

#### Using zAAP on zIIP considerations

This new capability is intended for two types of clients:

- ▶ Clients without enough zAAP- or zIIP-eligible workload to justify a specialty engine today.

Some clients want to invest in a System z specialty engine, but cannot justify doing so because there are not enough specialty engine-eligible workloads to justify the cost of either type of specialty engine. With this new capability, the combined eligible workloads may make the acquisition of a zIIP cost effective. This will also open up the possibility of adding new System z workloads in the future that can take advantage of zIIP functionality. This future workload could take the form of traditional zIIP-eligible workloads or Java and/or XML parsing workloads normally done on a zAAP.

- 5. Clients that currently have only zIIPs installed

Some clients have investments in zIIP processors, and some zAAP-eligible workloads, but not enough to justify the addition of a zAAP. With the zAAP on zIIP capability, the addition of eligible workloads to the traditional zIIP-eligible workloads can maximize the existing investment of the zIIP and avoid buying a zAAP.

In both instances, the zAAP on zIIP capability can help optimize utilization of server resources and simplify systems management by reducing the need to plan for and manage multiple types of specialty engines.

## Restrictions for zAAP on zIIP support

This new capability is not available if zAAPs are already installed on the server, as follows:

- ▶ The capability ships default-enabled with z/OS V1R11. If a client has zAAPs on the server and zAAP on zIIP capability is enabled, then z/OS will not honor zAAP on zIIP and workloads will be dispatched as normal on zAAP and zIIP engines.
- ▶ IBM will not enable zIIP-eligible workloads to run on zAAPs. This zAAP on zIIP capability is one way only. It is for enabling zAAP-eligible workloads to run on zIIPs. zIIP-eligible workloads can never run on zAAPs.
- ▶ This new capability does not remove the requirement to purchase and maintain one or more general purpose processors for every zIIP processor on the server.

## Z/OS V1R11 support

The IEASYSxx parmlib member uses the ZAAPZIIP (or ZZ as its alias) parameter to determine whether the system can run zAAP processor-eligible work on zIIP processors when no zAAP processors are installed on the machine.

**ZAAPZIIP=NO** When you specify ZAAPZIIP=NO, the system cannot run the zAAP processor-eligible work on zIIP processors when no zAAP processors are installed on the system.

**ZAAPZIIP=YES** When you specify ZAAPZIIP=YES, the system can run zAAP processor-eligible work on zIIP processors when no zAAP processors are installed on the machine.

Default Value: YES

**Note:** The default of ZAAPZIIP is NO in z/OS V1R9 and z/OS V1R10; and the default is YES as of z/OS V1R11 and later.

You can use the ZAAPZIIP (or ZZ as its alias) parameter to request the system to run zAAP processor-eligible work on zIIP processors when certain conditions are met.

The zAAP on zIIP function is active only when all of the following conditions are true:

- ▶ ZAAPZIIP=YES is specified or defaulted.
- ▶ There are (or could be, through dynamic processor addition) zIIPs defined to this LPAR.
- ▶ No zAAPs are defined to this LPAR, whether online, offline, or in the reserved state.
- ▶ No zAAPs in the configured state are installed on the machine. There could be zAAPs in the standby or reserved state.

This LPAR is allowed to find out about the presence of zAAPs on the entire machine. This security protocol is controlled by the Global Performance Data Control setting, described in *Processor Resource/Systems Manager Planning Guide, IBM System z10, SB10-7153*.

**Note:** The default of ZAAPZIIP is NO in z/OS V1R9 and z/OS V1R10; and the default is YES as of V1R11. Only the alias ZZ is valid in z/OS V1R9 and z/OS V1R10.

## 18.1.2 The DISPLAY IPLINFO command in z/OS V1R12

Before z/OS V1R12, users had difficulty verifying whether zAAP on zIIP support was enabled on their installations. The DISPLAY IPLINFO command is now enhanced to give information about the zAAP on zIIP state.

Figure 18-1 shows an example of this support to display the ZAAPZIIP value specified at IPL.

```
DISPLAY IPLINFO,ZAAPZIIP  
IEE255I SYSTEM PARAMETER 'ZAAPZIIP': YES
```

Yes - means you specified ZAAPZIIP=YES, and the system can run zAAP processor eligible work on zIIP processors when no zAAP processors are installed on the machine.

Figure 18-1 D IPLINFO for the zAAP on zIIP setting

In providing this support, the DISPLAY IPLINFO command is enhanced to display the value specified for any IPL parameter. Figure 18-2 shows the DISPLAY IPLINFO command being used to display the PROGxx parmlib members used at IPL.

```
DISPLAY IPLINFO,PROG  
IEE255I SYSTEM PARAMETER 'PROG': (AI,00,AA)
```

Figure 18-2 D IPLINFO for the PROG setting

When displaying the ZAAPZIIP status, the DISPLAY IPLINFO command (Figure 18-3) accepts the parameter STATE. This displays the ZAAPZIIP state of ACTIVE or INACTIVE and gives the reason if ZAAPZIIP is INACTIVE.

```
DISPLAY IPLINFO,ZAAPZIIP,STATE  
IEE256I ZAAPZIIP STATE: INACTIVE - ZAAP(S) DEFINED TO THIS LPAR
```

Figure 18-3 D IPLINFO command

The IEE256I message displays the state of the zAAP on zIIP function. The message text will indicate the following information:

**ACTIVE:** zAAP on zIIP is active.

**INACTIVE** ZAAPZIIP SYSTEM PARAMETER IS NO.

The zAAP on zIIP function is not active because the system parameter ZAAPZIIP=NO was specified meaning that no zIIPs are defined to this LPAR and the dynamic processor addition function is not enabled, so no zIIPs can be added to this LPAR after IPL.

**INACTIVE** NO ZIIPS DEFINED TO THIS LPAR.

The zAAP on zIIP function is not active because there are no zIIPs defined to this LPAR and the dynamic processor addition function is not enabled, so no zIIPs can be added to this LPAR after IPL.

**INACTIVE** ZAAP(S) DEFINED TO THIS LPAR.

The zAAP on zIIP function is not active because one or more zAAPs are defined to this LPAR. The zAAPs might be online or offline or in the reserved state.

**INACTIVE** ZAAP(S) INSTALLED ON THE MACHINE.

The zAAP on zIIP function is not active because one or more zAAPs in the configured state are installed on the machine (not necessarily defined to this LPAR). The presence of installed zAAPs in the standby or reserved state does not affect the state of the zAAP on zIIP function.



## INACTIVE

GLOBAL MACHINE DATA IS NOT AVAILABLE TO THIS LPAR.

The zAAP on zIIP function is not active because this LPAR is not allowed to find out about the presence of zAAPs on the entire machine. This security protocol is controlled by the Global Performance Data Control setting, described in *System z10 Processor Resource/Systems Manager Planning Guide*, SB10-7153.

**Note:** If INACTIVE is shown in the message output, in summary it is because of one of the following reasons:

- ▶ It has a zAAP installed on this machine.
- ▶ The ZAAPZIIP system parameter is set to NO.
- ▶ There are no zIIPs defined to this LPAR.
- ▶ zAAPs defined to this LPAR and are not installed but just defined.
- ▶ Global machine data is not available to this LPAR.

For additional related materials for quick reference, see *z/OS MVS System Commands*, SA22-7627 and *z/OS MVS Initialization and Tuning Reference*, SA22-7592.

### 18.1.3 IEAOPTxx parmlib member considerations

The following IEAOPTxx parmlib members have been updated in z/OS V1R12:

**CCCAWMT** Specifies whether to activate or deactivate Alternate Wait Management (AWM). If AWM is active, SRM and LPAR cooperate to reduce low utilization effects and overhead.

In an LPAR, some n-way environments with little work appear to require more capacity than expected because of the time spent waking up idle logical and physical processors to compete for individual pieces of work. If AWM is active, SRM and LPAR will reduce this unproductive use of processor so that capacity planning is more accurate and processor overhead is reduced.

With z/OS V1R12: With the value range for HIPERDISPATCH=NO (the default value), any value between 1 and 499,999 makes AWM active, and any value between 500,000 and 1,000,000 makes AWM inactive. AWM is active or inactive only for general CPs, zAAPs, and zIIPs.

For HIPERDISPATCH=YES, the valid range is 1,600 to 3,200. Any other value will be reset to the default value of 3,200. AWM is always active and cannot be turned off.

**Default Value:** AWM is active. For HIPERDISPATCH=NO, the default is 12,000 (12 ms). For HIPERDISPATCH=YES, the default is 3,200 (3.2 ms).

**Note:** When more than 64 logical processors are defined for the LPAR, the message IRA865I is issued and HIPERDISPATCH is forced.

**CCCSIGUR** Specifies the minimum mean-time-to-wait threshold value for heavy processor users. This constant is used to determine the range of mean-time-to-wait values that are assigned to each of the ten mean-time-to-wait dispatching priorities.

The specified real-time value is adjusted by relative processor speed to become SRM time to insure consistent SRM control across various processors.

The syntax is: CCCSIGUR=option

**Value Range:** 0-32,767 milliseconds

**Default Value:** 45

### **MANAGENONENCLAVEWORK**

Specifies whether non-enclave transaction work of queue servers and enclave servers is to be managed or not. The syntax is:

MANAGENONENCLAVEWORK=option

Where option is YES or NO.

### **MANAGENONENCLAVEWORK=YES**

Indicates that SRM is to manage the non-enclave transaction work towards the first service class period of the address space performance goal. Based on this expanded performance management, be sure to verify the performance goals for the service class of the address spaces that process the enclave work. For a detailed description, see the chapter “Performance Management of Address Spaces with Enclaves” in *z/OS MVS Planning: Workload Management, SA22-7602*.

### **MANAGENONENCLAVEWORK=NO**

Indicates that you expect no work consuming significant processor service to be running in the address space outside of an enclave. The processor consumption of work running outside of enclaves is not included when workload management assesses the impact of processor adjustments for the enclave work. This performance management is available for z/OS V1R12 and later releases.

**Note:** In order to complete a MANAGENONENCLAVEWORK status change from YES to NO or NO to YES for an active system, SRM issues a policy refresh for that active system. Workload Manager indicates the policy refresh for this status change with message IWM065I.

**Values:** YES or NO

**Default Value:** NO

**Example:** MANAGENONENCLAVEWORK=YES

In the example, SRM is to manage the non-enclave work of enclave servers and queue servers.

**RTPIFACTOR** Specifies how much the server performance index (PI) should affect the server routing weights returned by WLM services IWM4SRSC and IWMSRSRS with FUNCTION=SPECIFIC, as follows:

- ▶ When RTPIFACTOR is 0, the server weight is independent from the server PI.
- ▶ When RTPIFACTOR is 100 and the server PI is larger than 1, the server weight is divided by the server PI.
- ▶ When RTPIFACTOR is between 1 and 99, it results in a corresponding intermediate influence of the server PI on the server weight returned by WLM.

**Value Range:** 0-100

**Default Value:** 100

**Note:** The PROJECTCPU keyword specifies whether to activate or deactivate the projection of how work could be offloaded from regular CPs to special assist processors like the System z Application Assist Processor (zAAP) and the System z Integrated Information Processor (zIIP). In z/OS V1R12, if the installation requires ZIIP statistics but there is no ZIIP defined (reserved or actual) on the LPAR, then PROJECTCPU=YES is required regardless of the hardware model.

There are new value ranges for the ZAAPAWMT and ZIIPAWMT keywords in the IEAOPTxx parmlib member, as follows:

**ZAAPAWMT** **Value Range:** For HIPERDISPATCH=NO (the default value), the valid range is 1 to 499,999 microseconds. Any value between 1 and 499,999 makes AWM active. To inactivate AWM, set CCCAWMT to any value between 500,000 and 1,000,000.

For HIPERDISPATCH=YES, the valid range is 1,600 to 499,999 microseconds. Any other value will be reset to the default value of 3,200. AWM is always active and cannot be turned off.

**Default Value:** AWM is active. For HIPERDISPATCH=NO, the default is 12,000 (12 ms), and for HIPERDISPATCH=YES, the default is 3,200 (3.2 ms).

**ZIIPAWMT** **Value Range:** For HIPERDISPATCH=NO (the default value), the valid range is between 1 and 499,999 microseconds. Any value between 1 to 499,999 makes AWM active. To inactivate AWM, set CCCAWMT to any value between 500,000 and 1,000,000.

For HIPERDISPATCH=YES, the valid range is 1,600 to 499,999 microseconds. Any other value will be reset to the default value of 3,200. AWM is always active and cannot be turned off.

**Default Value:** AWM is active. For HIPERDISPATCH=NO, the default is 12,000 (12 ms), and for HIPERDISPATCH=YES, the default is 3,200 (3.2 ms).

As in previous releases, the ZAAPAWMT and ZIIPAWMT parameters internally affect the frequency with which the specialty engines will check the need for help. If help is required, the zAAP or zIIP processor signals a waiting zAAP or zIIP to help. When all zAAP or zIIP processors are busy and IFAHONORPRIORITY/IIPHONORPRIORITY=YES, the zAAP or zIIP processors ask for help from the standard processors. All available specialty engines (that is, all zAAPs or all zIIPs) must be busy before help is asked of the standard processors.

Reducing the value specified for ZAAPAWMT or ZIIPAWMT causes the specialty engines to request help after being busy for a shorter period of time. If IFAHONORPRIORITY or IIPHONORPRIORITY is set to YES, help is provided to one CP at a time, in the priority order of zAAP or zIIP processor-eligible work, non-zAAP or non-zIIP processor-eligible work. Reducing the ZAAPAWMT or ZIIPAWMT value too much can cause the standard processors to run an excessive amount of zAAP or zIIP processor-eligible workload, which might result in lower priority non-zAAP/zIIP processor-eligible work to be delayed. Conversely, increasing the value specified for ZAAPAWMT or ZIIPAWMT causes the specialty engines to request help only after being busy for a longer period of time, which might delay the standard processors from providing help when it is necessary.

## 18.2 Discretionary work timeslice

In this section we describe the ability to provide additional processor resources for processor-intensive discretionary workloads. z/OS V1R12 provides a new way to reduce the

overhead caused by cache misses due to context switching among equal dispatching priority units of work, improving system throughput.

## 18.2.1 Discretionary workload

Discretionary workload receives processor resources when there is no higher priority workload to dispatch, and has the following characteristics:

- ▶ Executes without other specific business performance goals associated with it, such as response time or velocity oriented goals.
- ▶ Must be in the last performance period when defined within a service class with multiple performance periods.
- ▶ Runs using system resources not required to meet the goals of other work.
- ▶ Uses the mean-time-to-wait (MTTW) dispatching priorities (192 to 201).

Workload Manager is constantly evaluating the processor demand for each discretionary unit of work in the MTTW group. Higher MTTW values indicate high I/O activity. For these units of work, WLM assigns dispatching priorities within the high end of the range 192 to 201 (X'CO' to X'C9'). On the other hand, the lower MTTW values indicate high demand for processor resources. For these units of work, WLM assigns dispatching priorities in the lower end of the mean-time-to-wait range.

The idea behind this priorities attribution schema, based on processor resource consumption, is to improve system throughput, because high I/O intensive units of work can quickly have access to processor resources and return to long wait periods. However, as the context switching rate increases, due to work unit preemption, cache misses tend to also become higher, reducing the throughput of the system. So, a mechanism for reducing the rate of switching the control of processor resources, among processor-intense work units, becomes necessary.

## 18.3 Customized time slices

z/OS V1R12 introduces a new IEAOPTxx parmlib member, the TIMESLICES parameter, to assign additional time slices of processor control to discretionary units of work. The CCCSIGUR parameter has returned, having been removed from the documentation in z/OS V1R7, to attribute processor-intensive characteristics to work units. To implement customized time slices for discretionary processor-intensive workloads, it is necessary to define a TIMESLICES member with a parameter value greater than 1, and change the default value of CCCSIGUR, if you desire another value.

Detailed information about these parameters can be found in *z/OS MVS Initialization and Tuning Reference*, SA22-7592.

### 18.3.1 The TIMESLICES parameter

The new IEAOPTxx parmlib member parameter, TIMESLICES, specifies the number of time slices that a processor-intensive address space or enclave with a discretionary goal should be given before a dispatchable unit of equal importance is dispatched. Increasing this parameter might increase the processor delay for some processor-intensive work, but decrease the number of context switches between equal priority work, and therefore increase the throughput of the system. This parameter only affects discretionary work that is

processor-intensive as determined by significant mean-time-to-wait (MTTW); see the parameter CCCSIGUR.

The syntax is:

```
TIMESLICES=value  
Value range: 1-255  
Default value: 1
```

### 18.3.2 Mean-time-to-wait threshold

Specifies the minimum mean-time-to-wait threshold value for heavy processor users. This constant is used to determine the range of mean-time-to-wait values that are assigned to each of the ten mean-time-to-wait dispatching priorities. The specified real-time value is adjusted by relative processor speed to become SRM time to insure consistent SRM control across various processors.

The syntax is:

```
CCCSIGUR=value  
Value range: 0-32767 milliseconds  
Default value: 45
```

### 18.3.3 Expected benefits

The result of using additional processor time slices to reduce context switching among discretionary processor-intensive work units depends mainly on the values specified on TIMESLICES, as can be seen in the benchmark in Figure 18-4, with more threads than CPUs, very low pause times, and a high number of processor-intensive discretionary work units. TIMESLICES=64 showed 33% throughput improvement at 7 threads per JVM. TIMESLICES=255 showed only a 2% improvement over TIMESLICES=64.

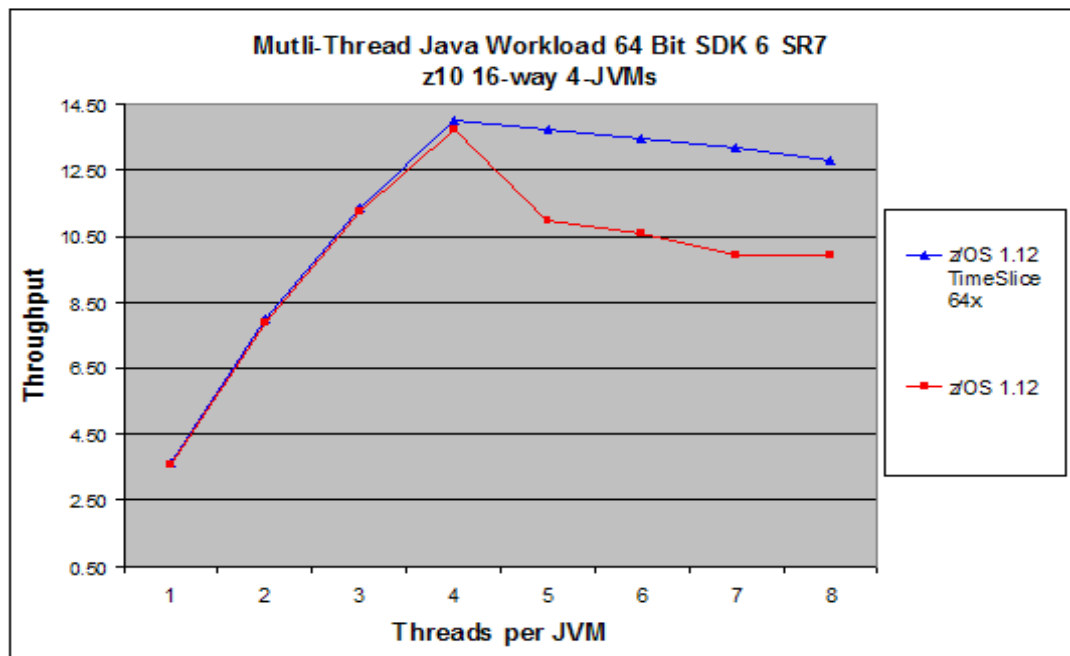


Figure 18-4 Throughput: TIMESLICES=1 (default) vs. TIMESLICES=64

## 18.4 MTTR instrumentation

Mean Time To Recovery (MTTR) is the time it takes to get a system back to running its normal production workload after a failure. If a system fails with a wait state, then MTTR is the time it takes to gather diagnostics (for example, a SADUMP) and to then IPL the system and start up all of the needed subsystems and applications. If the system is experiencing problems and is brought down with an orderly shutdown, then the MTTR includes the shutdown time as well as the re-IPL and start-up time.

Software developers need a way to gain insight into how the code behaves in various configurations. The MTTR trace is a new tool that will help developers gather information to help them do that. The information gathered through MTTR traces can be used to determine whether changes need to be made to improve MTTR.

The approach to understanding and making improvements to the code using the MTTR trace is envisioned as an iterative approach. Trace points are added, information is gathered, code changes are made, and then new information is gathered to determine whether the changes are effective. While this might seem rather straightforward and mundane, it actually turns out that making improvements is difficult work. The MTTR trace will help to ease that effort by providing a consistent approach across all components and products that use it.

### 18.4.1 MTTR process overview

Using the MTTR trace involves using the new macro IEATEDS and the new REXX exec IEAVFTED. The following steps need to be taken to add code that writes data to the trace table and then to analyze the trace table data:

1. Invoke macro IEATEDS REQUEST=REGISTER

The developer adds code to invoke IEATEDS for a REGISTER request, usually in a module that does the initialization for the component. The number of trace entries is specified in the parameter MaxEvents, and this is used to determine the size of the trace table. The table does not wrap once it is filled, so MaxEvents should be a number sufficient to get the needed entries. The storage for the trace table will be in above-the-bar common storage and so MaxEvents should not be overspecified. The report can help determine a good value because it will indicate the number of entries in the table and the number that overflowed if the table became full.

2. Invoke macro IEATEDS REQUEST=RECORD

The developer adds code to invoke IEATEDS for a RECORD request that will trace an event. The developer will specify information to help identify the point in the code, including the module name and model level. The service will also gather some additional information, including the offset in the module, jobname, TCB address, HASN, PASN, OUCB information, and the current TOD (Time Of Day) clock value. The developer can also specify up to 16 bytes of data that can help provide some context for the event (for example, a loop counter, an index value, or a return code value). The trace type and trace thread are used to help organize the trace data by providing a way to indicate the start, middle, and end events for a series of trace points, and tying it all together with a user-defined trace thread value (which can be any unique value, such as TOD or a sequence number).

3. Invoke REXX exec IEAVFTED

The customer runs the IEAVFTED REXX exec to gather the trace data and write a Timed Events Data report to a data set. IEAVFTED can be run on a live system at any time to get a report, and the trace tables are not cleared when this is done so that any additional activity can be obtained later by running IEAVFTED again. IEAVFTED can also be run in

IPCS to gather the trace data from a dump. Also, the Timed Event Data report will contain two sections: a human-readable section and a section that can be used as input to a spread sheet program where the data can be sorted and analyzed. Note that in addition to the Timed Event Data created with IEATEDS, the IPLSTATS data will also appear in the output of the human-readable and spreadsheet data.

The IEATEDS macro and the *z/OS MVS Programming: Assembler Services Reference, Volume 2 (IAR-XCT)*, SA22-7607 contain detailed descriptions on all of the IEATEDS keywords, the IEAVFTED keywords, a full explanation of the report fields, and several example invocations of both IEATEDS and IEAVFTED.

## 18.4.2 The IEATEDS macro

The IEATEDS macro can be used only by authorized programs.

The IEATEDS macro environment:

- ▶ Supervisor state, any key
- ▶ Task or SRB mode
- ▶ Any PASN, SASN, or HASN
- ▶ AMODE 31
- ▶ Primary or AR mode
- ▶ Enabled
- ▶ May hold a local lock, or local lock and CMS lock

The IEATEDS macro requires the invoker to pass in a 2048-byte work area. IHATEDS provides constants and types that can be used to help declare the work area, TedToken, and Event Thread, and also includes the constants for the return and reason codes.

## 18.4.3 The IEAVFTED REXX exec

The IEAVFTED REXX exec can be used to get a Timed Event Data report in TSO on a live system, or in IPCS to get the report from a dump. The dump must contain CSA since the trace tables reside in above-the-bar common.

- ▶ IEAVFTED must reside in a FIXED LRECL 80 data set.
- ▶ IEAVFTED requires REXX LIBR BASE MVS FMID HWJ9140.
- ▶ Environment:
  - TSO on live system
  - IPCS for a dump with CSA
- ▶ Can write to either:
  - Pre-allocated LRECL 512 variable blocked data set
  - z/OS UNIX file
- ▶ Can filter on component name using the Component keyword that helps reduce the size of the Timed Event Data report. The component name filter is the 32-character component name specified during the IEATEDS REGISTER request.

## Invoking IEAVFTED via TSO

The following are examples of using the IEAVFTED REXX exec via TSO. It will:

- ▶ Write a Timed Event Data report to an LRECL 512, RECFM V or VB pre-allocated data set and will contain data for all components that have registered using IEATEDS:

```
IEAVFTED DA('USERID.MTTR.REPORT1')
```

- ▶ Write a Timed Event Data report to an LRECL 512, RECFM V or VB pre-allocated data set and will contain data only for the component named GRS:

```
IEAVFTED DA('USERID.MTTR.REPORTS(REPORT2)') COMPONENT(GRS)
```

- ▶ Write a Timed Event Data report to a z/OS UNIX file and will contain data for all components that have registered using IEATEDS:

```
EAVFTED PATH('/usr/zed_data/System_SY1')
```

**Note:** The z/OS UNIX file name is case sensitive.

## Invoking IEAVFTED via IPCS

To invoke the IEAVFTED REXX exec under IPCS, the following steps need to be taken. An LRECL 255, RECFM VB data set is first created, which is the size that IPCS needs it to be. The TSO ALLOC command allocates the data set to IPCSPRINT. The IP ALTLIB command indicates to IPCS the name of the data set containing the IEAVFTED REXX exec. The IP SETDEF PRINT NOTERM command directs the IPCS output to the printer instead of the screen. The IP IEAVFTED creates the Timed Event Data report, and the IP CLOSE and IP SETDEF statement complete the process.

- ▶ Create an output DSN (LRECL of 255, RECFM of VB)
- ▶ TSO ALLOC F(IPCSPRINT) DA(DSN) SHR REUS
- ▶ IP ALTLIB ACTIVATE APPL(EXEC) DA('SYS1.SBLSCLI0')
- ▶ IP SETDEF PRINT NOTERM
- ▶ IP IEAVFTED
- ▶ IP CLOSE PRINT
- ▶ IP SETDEF TERM NOPRINT
- ▶ Data set DSN now contains the Timed Event Data Report

A second IEAVFTED invocation is optional and is only required if the spreadsheet data is needed. It does not produce a new report but instead converts the LRECL 255 data set to a 512 LRECL data set to allow the spreadsheet data to be sent via FTP.

- ▶ IEAVFTED IPCSDA('input\_data\_set\_name') DA('output\_data\_set')

## IEAVFTED spreadsheet

Figure 18-5 on page 363 shows an example of the Timed Event Data loaded into a spreadsheet.

- ▶ The Event Time and Date columns display the time and date that the IEATEDS RECORD request was made.
- ▶ The Thread EBCDIC column displays the value in EBCDIC that was specified on the IEATEDS RECORD request as the EventThread, which is useful in associating a series of events that use the same EventThread.
- ▶ The Type column displays the value specified on the IEATEDS RECORD request as the EventType. This is used to indicate a relative time position within a series of events associated with the same EventThread.



- ▶ The Description column displays the value specified on the IEATEDS RECORD request as the Description.
- ▶ The Component column displays the value specified on the IEATEDS REGISTER request as the CompName.
- ▶ The IPL Start Delta column displays the elapsed time from the start of the IP to this event.
- ▶ The Thread Start Event Delta column displays the elapsed time from the start of the first IEATEDS RECORD request that specified an EventType of START and with the same EventThread as this event.
- ▶ The T.E.D. Registration Delta column displays the elapsed time from the time that the IEATEDS REGISTER request was made to the time of this event.
- ▶ The Thread Prior Event Delta column displays the elapsed time from a prior IEATEDS RECORD request with the same EventThread.

Event Time	Date	Event Thread	Thread EBCDIC	Type	Description	Component	IPL Start Delta	Thread Start Event Delta	T.E.D. Registration Delta	Thread Prior Event Delta
00:00:00.000000	3/15/10 16:01:23 15 Mar 2010	Report HBB7>Sysplex: PLEX1 System: SY1 FMachine: 4381-FF			IPL Star					
00:00:00.000000	3/15/10	Total Timed Event Data Table Storage: 00072A70								
15:51:39.783511	3/15/10	T.E.D. Table Register	TheProdu	*	Size: 00002C60 Requested Max: 64 Resultant Max: 64 Current #: 3 Overflow #: 0					
15:51:39.783516	3/15/10	E2C1D4D7D3C54040	*SAMPLE	*	Start A sample of a TED entry	THEPRODUCT	2973.41933	0	0.000005	0.000004
15:51:39.783521	3/15/10	E2C1D4D7D3C54040	*SAMPLE	*	Mid Before doing XYZ	THEPRODUCT	2973.41933	0.000004	0.00001	0.000004
15:51:39.783525	3/15/10	E2C1D4D7D3C54040	*SAMPLE	*	End After doing XYZ	THEPRODUCT	2973.41934	0.000009	0.000014	0.000004

Figure 18-5 IEAVFTED spreadsheet

**Note:** In addition to the Timed Event Data created with IEATEDS, the IPLSTATS data will also appear in the output of the human-readable and spreadsheet data.

### 18.4.4 Software dependencies

MTTR instrumentation is installed as part of the BCP and so no special steps are needed. However, there are two things to consider:

- ▶ IEAVFTED must reside in a FIXED LREC 80 data set. You may wish to move the IEAVTED REXX exec to another data set. That is possible as long as the data set has an LRECL or 80 and a RECFM of F or FB.
- ▶ The MTTR Trace service, which is invoked by macro IEATEDS, obtains above-the-bar storage using IARV64 and IARST64 services, and so these services must be available. This will not be a problem for clients since they do not have code that runs before the system is IPLed, but it could be a problem for developers that have code in the IPL path.

IEAVFTED is a compiled REXX exec that uses functions in the REXX LIBR BASE MVS FMID HWJ9140.

**Note:** The REXX Alternate Runtime Library z/OS Base HWJ9143 will not work.

Any authorized application can use MTTR Instrumentation, though its intended purpose is to allow developers (IBM, vendor, and client) to understand their code to help make it run more efficiently during IPL and shutdown to help improve MTTR. The current exploiters are Allocation, XCF/XES, GRS, and JES2.

## 18.5 RTM enhancements

This section describes the enhancements to the Recovery Termination Manager (RTM) component in z/OS V1R12 and their benefits. Following is the list of updates to RTM. All are intended to improve system reliability, availability, and serviceability (RAS).

- ▶ ESPIE percolation to RTM
- ▶ SPIEOVERRIDE for ESTAEX
- ▶ SDWALOC31 for FRRs
- ▶ RESMGR RLXPC (reusable LX PC)
- ▶ IEAARR DYNSTORAGE=NOTAVAIL
- ▶ Time Of Error Locks for ESTAE-type recovery in the SDWA
- ▶ SDWANMFS - Not my fault summary
- ▶ New TCB flag - TCBENDNG
- ▶ New TCB flag - TCBEndingAbnormally
- ▶ New STCB field - STCBCMPC
- ▶ TCB address in formatted logrec record

### 18.5.1 ESPIE percolation to RTM

Traditionally, ESPIE exits have been required to resume processing by retrying to the program which was interrupted by the program exception. However, instead of resuming, you may want your ESPIE exit to “give up” and pass the program exception to RTM so that regular abend processing can take place.

New function in z/OS V1R12 allows ESPIE exits to percolate to RTM. Percolation is to ESTAE-type recovery and the program exception is converted by RTM to an abend just as it would have been if no ESPIE exit had been established.

**Note:** ESPIE exits cannot percolate to a previous SPIE or ESPIE exit because only one SPIE/ESPIE exit can be established at a time.

When an ESPIE exit percolates, it is not deactivated. ESPIE percolation is requested by setting the new EPIEperc bit before the ESPIE exit returns to the system.

**Note:** The EPIEperc bit is defined in data area IHAEPIC.

An example of a situation where an ESPIE exit might choose to “give up” would be if it determines that the program exception occurred unexpectedly outside of the bounds of the function that it was providing recovery for. This ability is not provided for the older SPIE exit function.

An example of RTM converting an exception to an abend is when an Operation Exception Program Interrupt Code (PIC) 1 becomes ABEND 0C1.

An ESPIE exit that percolates may wish to also deactivate itself or restore a previous SPIE or ESPIE environment. ESPIE RESET should not be issued from an exit routine because doing so may cause the EPIE data area that they are working with to be freed. RTM now provides another way via the EPIE interface. The new EPIERSET bit can be turned on to request that the system deactivate the ESPIE exit.

EPIERSET is logically equivalent to issuing ESPIE RESET. As with ESPIE RESET, the previously-established SPIE or ESPIE exit can be restored by placing its token into the new EPIERTOK field. This ability is not restricted to ESPIE exits that percolate. Resuming ESPIE exits may also request deactivation this way.

## 18.5.2 SPIEVERRIDE for ESTAEX

Your ESTAEX exit may not receive control if your program was called by a program that has a SPIE or ESPIE exit active. This is a problem that service routines occasionally run into.

SPIEVERRIDE=YES is a new ESTAEX parameter that specifies that your ESTAEX exit should receive control instead of any SPIE or ESPIE exit. No SPIE or ESPIE exit will receive control as long as this ESTAEX exit remains established.

ESTAEX SPIEVERRIDE=YES remains active if other programs are called. If the ESTAEX exit for which ESTAEX SPIEVERRIDE=YES was specified percolates, percolation occurs to any older ESTAE-type recovery and, if necessary, the task will be terminated.

**Note:** ESTAEX SPIEVERRIDE=YES is available only to authorized programs.

The old way of dealing with this concern was for a called program to establish an ESPIE exit of its own, specifying no program interrupts. The drawbacks of this method include a requirement to restore the caller's SPIE/ESPIE environment if one existed and the overhead involved in invoking the ESPIE service, possibly unnecessarily.

## 18.5.3 SDWALOC31 for FRRs

If RTM cannot obtain an SDWA from common storage for an FRR, that FRR is skipped. The recovery provided by an FRR can be critical and a system outage can result due to a skipped FRR.

FRRs established in AMODE 31 receive an SDWA below the 16 MB line, which makes them vulnerable to being skipped during an SQA shortage. This cannot be changed compatibly.

z/OS V1R12 introduces a new SDWALOC31=YES parameter to SETFRR, which indicates that an AMODE 31 FRR can tolerate an SDWA above the 16 MB line. SDWALOC31 for FRRs answers requirement MR1002093459.

SETFRR SDWALOC31=YES should be used whenever possible for AMODE 31 FRRs.

**Note:** Programs compiled with SETFRR SDWALOC31=YES may be used on older levels of z/OS because the parameter will be ignored.

A LOGREC record is written when an FRR is skipped.

The SDWALOC31 parameter is ignored when SETFRR is issued in AMODE 24 or AMODE 64. FRRs established in AMODE 64 are always expected to tolerate an SDWA above the 16 MB line.

#### 18.5.4 RESMGR RLXPC (reusable LX PC)

The RESMGR service is used to establish an End Of Task (EOT) or End Of Memory (EOM) resource manager, which may be called via LINK, BRANCH, or PC. This is specified with the RESMGR ROUTINE= parameter.

z/OS V1R12 allows the support of reusable LX PC routines via a new RESMGR parameter:

```
ROUTINE=(RLXPC, 1xseq/pcnum)
```

1xseq/pcnum is an 8-byte area; the first 4 bytes contain the LX sequence number and the second 4 bytes contain the PC number.

RESMGR RLXPC answers requirement MR1201093132.

#### 18.5.5 IEAARR DYNSTORAGE=NOTAVAIL

Users of IEAARR may wish to invoke the service from a module that does not have dynamic storage, but the current implementation requires storage for parameters.

z/OS V1R12 adds a parameter, DYNSTORAGE=NOTAVAIL, to IEAARR, which allows IEAARR to be used without dynamic storage.

A new set of non-pointer parameters are defined to be used when IEAARR DYNSTORAGE=NOTAVAIL is specified. These have similar names to their equivalent pointer parameters except that PTR is omitted. For example, instead of specifying PARAMPTR=xxxxxx, with IEAARR DYNSTORAGE=NOTAVAIL you would specify PARAM=yyyyyy.

IEAARR DYNSTORAGE=AVAIL is the compatible default. It may be specified explicitly but is not required.

#### 18.5.6 Time of error locks for ESTAE-type recovery in the SDWA

The SDWA contains information about serialization held at the time of error. This information was previously available only to FRR recovery exits. Even though it is not “current”, this information may be valuable to someone examining an ESTAE-type recovery SDWA.

The following fields are now valid in an ESTAE-type recovery SDWA:

- ▶ SDWAHLHI – a copy of PSAHLHI
- ▶ SDWACLSE – a copy of PSACLHSE
- ▶ SDWASUPR – a copy of PSASUPER
- ▶ SDWASPN – a copy of LCCASPIN

The descriptions for these fields have been updated to explain that they contain the serialization at the time of error, minus serialization that was released by any FRRs that have previously received control. This has not changed, it just was not documented correctly.

## 18.5.7 SDWANMFS - not my fault summary

A common recovery question is “Should I capture documentation about this situation?” One good way to answer that is to categorize abends into “my fault” and “not my fault” errors:

- ▶ “My fault” errors are typically unexpected failures such as ABEND0C4. It is generally useful to capture documentation for these failures.
- ▶ “Not my fault” errors are typically *expected* system situations such as Cancel or Detach. For most programs, if an ABTERM was issued against your program it would be a “not my fault” situation. It generally is not useful to capture documentation for these failures.

The new SDWANMFS (Not My Fault Summary) bit is an indicator that summarizes the most common “not my fault” situations. When SDWANMFS is on, your recovery probably should not collect documentation about the problem.

“Not my fault” situations included in SDWANMFS are:

- ▶ SDWAABTM – ABTERM was issued against your program.
- ▶ SDWASRBM – an SRB abended and its recovery percolated to your program.
- ▶ SDWAIRB – an IRB has interrupted your program and abended.
- ▶ SDWAMCHK – a Machine Check has interrupted your program.
- ▶ SDWARKEY – a Restart has interrupted your program.
- ▶ SDWACTS – your job step is terminating because another task requested Convert to Step.

Cancel and Detach are included in SDWANMFS because they are issued via CALLRTM TYPE=ABTERM, which means that SDWAABTM is always on for these abends.

**Note:** The final decision about whether to collect documentation or retry depends upon the intended design of your program and its recovery.

## 18.5.8 New TCB flag -TCBENDNG

The existing TCBDYING bit was defined to help programs determine that a task is going to terminate. However, a recovery routine might encounter the following situations, where the task will eventually terminate but TCBDYING has not been turned on yet:

- ▶ When an EUT FRR has received control for a non-retriable abend
- ▶ When an ESTAE-type recovery exit has received control for an ABTERM issued with RETRY=NO or for ABEND 33E (detach with STAE)

We cannot change the meaning of TCBDYING because some programs are known to be dependent upon it being off in the above situations.

The new TCBENDNG bit is a superset of TCBDYING that also includes the above situations. TCBENDNG support was also rolled down to z/OS V1R8 via APAR OA29419.

From IKJTCB:

```
TCBENDNG EQU  X'08' - If on, indicates that this TCB will be @OCA
*               terminating (normally or abnormally) and its
```

```

*          mainline processing will not be allowed to run
*          again. The key difference between TCBENDNG and
*          TCBDYING is that TCBENDNG is set before all
*          types of recovery routine if they will not be
*          allowed to retry. TCBDYING is not set before
*          FRRs and is set before Estae-type recovery
*          routines only for Cancel and Detach abends.
*          TCBENDNG is set when any of the following occur:
*          - The TCB is terminating normally
*          - before recovery routines (including FRRs)
*            receive control for all non-retriable abends
*            including Cancel, Detach, Detach with STAE,
*            and RETRY=NO abterms
*          - after all recovery routines have percolated
*            for retriable abends
*          Ownership - RTM.

```

### 18.5.9 New TCB flag - TCBEndingAbnormally

TCBENDNG indicates that a task is terminating. However, it may be useful to also know that the task is terminating abnormally and that its recovery will not be allowed to retry to mainline. Some programs currently use TCBFA to detect an in-progress abnormal termination but TCBFA is not an intended interface.

z/OS V1R12 uses new TCB flag TCBEndingAbnormally to indicate an in-progress abnormal termination. This is the intended interface equivalent of TCBFA.

TCBEndingAbnormally is turned on:

- ▶ Before ESTAE-type recovery (not FRRs) receives control for all non-retriable abends including Cancel, Detach, Detach with STAE, and RETRY=NO abterm.
- ▶ After all recovery routines have percolated for a retriable abend.
- ▶ In all subtasks before TERM=YES ESTAE-type recovery is invoked for a Cancel or Detach of the current task.

### 18.5.10 New STCB field - STCBCMPC

The TCB field TCBCMPC contains the system or user task completion code. However, there is no interface to determine whether TCBCMPC represents a return code or an abend code after a task terminates. In addition, TCBCMPC may be changed by the initiator to reflect the final state of a job step.

z/OS V1R12 uses the new field STCBCMPC to provide a task-specific completion code:

- ▶ After the task terminates; if TCBEndingAbnormally is on, STCBCMPC contains the abend code for which this task ended abnormally.
- ▶ After the task terminates; if TCBEndingAbnormally is off, STCBCMPC contains the contents of GPR15 when the last program returned normally to the system.

**Note:** STCBCMPC is only valid after a TCB terminates.

When combined with TCBEndingAbnormally, STCBCMPC can be thought of as the “RTM view” of a terminated subtask. The contents of STCBCMPC may differ from the contents of TCBCMPC because TCBCMPC may be changed to reflect job step status.

Like TCBCMPC, when STCBCMPC contains an abend code, the first 12 bits are the system code and the last 12 bits are the user code.

The containing field for STCBCMPC, STCBCMP, also contains a flag field called STCBCMPF. This field does not contain the same flags as TCBCMPF and is reserved for future use.

**Note:** TCBEndingAbnormally together with TCBCMPC cannot be used instead of STCBCMPC. TCBCMPC is subject to change. It is possible that TCBEndingAbnormally might be on but TCBCMPC has been changed to a different value such as zero.

### 18.5.11 TCB address in formatted LOGREC record

In a formatted software LOGREC record, it is often desirable to know what the related TCB address was. This information has always been available in SDWATCHB but it was not clear that this was the failing TCB address.

With new function provided by z/OS V1R12, formatted software LOGREC records now contain either:

- ▶ THIS SRB'S PURGEDQ ASID/TCB: yyyy/xxxxxxx
- ▶ THIS TASK'S ASID/TCB: yyyy/xxxxxxx

**Note:** A value of 0000/00000000 is possible for an SRB's PURGEDQ ASID/TCB because SRBPASID and SRBPTCB may legitimately be zero.

A new time-of-error message is added when appropriate:

```
THIS WAS AN SRB TO TASK PERCOLATION.
```

## 18.6 Miscellaneous updates

This section covers the following supervisor code updates made in z/OS V1R12:

- ▶ D SYMBOLS
- ▶ LCCA/PCCA health checks

### 18.6.1 D SYMBOLS

With z/OS V1R12, a new SUMMARY keyword of the DISPLAY SYMBOLS command is designed to provide summary information about symbols used on the system, including how many are in use. This can help you determine how many additional symbols can be defined.

The DISPLAY SYMBOLS command can be used to display the symbols defined via the IEASYMxx parmlib member. In z/OS V1R12 two new parameters have been added to the DISPLAY SYMBOLS command:

```
D SYMBOLS [{,DETAIL | ,SUMMARY}]
```

The default is `DETAIL`, which results in the pre z/OS V1R12 display. The new parameter, `SUMMARY`, returns the number of symbols defined, the current symbol table size, and the maximum table size.

Figure 18-6 shows the output from the `D SYMBOLS,DETAIL` and `D SYMBOLS,SUMMARY` commands.

```
D SYMBOLS,DETAIL
IEA007I STATIC SYSTEM SYMBOL VALUES
      &SYSALVL. = "2"
      &SYSCLONE. = "7F"
      &SYSNAME. = "SP7F"
      &SYSPLEX. = "LOCAL"
      &SYSR1. = "SD112"
      &SYSR2. = "SD112"
      &SYSR3. = "SD112"
      &SYSR8. = "SD112"

D SYMBOLS,SUMMARY
IEA994I STATIC SYSTEM SYMBOL INFO
SYMBOLS DEFINED: 8
CURRENT TABLE SIZE: 232 BYTES
MAX TABLE SIZE: 32512 BYTES
```

Figure 18-6 `D SYMBOLS` display

## 18.6.2 LCCA/PCCA health checks

The `CBLOC` keyword in the `DIAGxx` parmlib member can be used to specify whether the LCCA and/or PCCA control blocks reside above or below the 16 MB line.

To specify that both LCCA and PCCA control blocks reside above the 16 MB line, the following `CBLOC` statement would be added to the `DIAGxx` parmlib member:

```
CBLOC VIRTUAL31(IHALCCA,IHAPCCA)
```

In z/OS V1R12, the default for the LCCA and PCCA control blocks was changed to `RMODE 31`.

New health checks are provided to check the location of the LCCA and PCCA control blocks. There are also z/OS V1R12 migration health checks for the LCCA and PCCA control blocks.

### **RCF\_LCCA\_ABOVE\_16M**

Checks to see whether the residency mode (`RMODE`), specified for the LCCA control block in the `CBLOC` parameter of the `DIAGxx` parmlib member, is the expected value. The default `RMODE` for the LCCA control block is `RMODE 31`. The check will look for `RMODE 31` for the LCCA control block unless you specify an `RMODE` of 24 in the `RMODE` parameter for the check.

The new `RCF_LCCA_ABOVE_16M` is shown in Figure 18-7 on page 371.



```

UPDATE,
CHECK(IBMRCF,RCF_LCCA_ABOVE_16M),
INTERVAL(ONETIME),
SEVERITY(LOW),
PARM('CBLOC(31)'),
DATE('20100124')
Reason('Your reason for making the update.')
```

Figure 18-7 RCF\_LCCA\_ABOVE\_16M Health check syntax

Parameters accepted:

- ▶ CBLOC(31), which is the default, specifies that you want the check to generate an exception if it finds that IHALCCA had been specified in the CBLOC VIRTUAL24 parameter of the DIAGxx parmlib member.
- ▶ CBLOC(24) specifies that you want the check to generate an exception if it finds that IHALCCA either:
  - Had been specified in the CBLOC VIRTUAL31 parameter of the DIAGxx parmlib member.
  - Had not been specified in the CBLOC VIRTUAL24 parameter of the DIAGxx parmlib member because CBLOC VIRTUAL31 is the default for the LCCA.

RCF\_LCCA\_ABOVE\_16M messages:

- ▶ IEAVEH090I - Expected matches actual.
- ▶ IEAVEH091E - Expected does not match actual.

### RCF\_PCCA\_ABOVE\_16M

Checks to see whether the residency mode (RMODE), specified for the PCCA control block in the CBLOC parameter of the DIAGxx parmlib member, is the expected value. The default RMODE for the PCCA control block is RMODE 31. The check looks for RMODE 31 for the PCCA control block unless you specify an RMODE of 24 in the RMODE parameter for the check.

The new RCF\_PCCA\_ABOVE\_16M is shown in Figure 18-8.

```

UPDATE,
CHECK(IBMRCF,RCF_PCCA_ABOVE_16M),
INTERVAL(ONETIME),
SEVERITY(LOW),
PARM('CBLOC(31)'),
DATE('20100124')
Reason('Your reason for making the update.')
```

Figure 18-8 RCF\_PCCA\_ABOVE\_16M Health check syntax

Parameters accepted:

- ▶ CBLOC(31), which is the default, specifies that you want the check to generate an exception if it finds that IHAPCCA had been specified in the CBLOC VIRTUAL24 parameter of the DIAGxx parmlib member.

- ▶ CBLOC(24) specifies that you want the check to generate an exception if it finds that IHAPCCA either:
  - Had been specified in the CBLOC VIRTUAL31 parameter of the DIAGxx parmlib member.
  - Had not been specified in the CBLOC VIRTUAL24 parameter of the DIAGxx parmlib member because CBLOC VIRTUAL31 is the default for the LCCA.

RCF\_PCCA\_ABOVE\_16M messages:

- ▶ IEAVEH100I - Expected matches actual.
- ▶ IEAVEH101E - Expected does not match actual.

### ZOSMIGV1R12\_SUP\_LCCA\_ABOVE\_16M

Checks to see whether the residency mode (RMODE), specified for the LCCA control block in the CBLOC parameter of the DIAGxx parmlib member, is the expected value. The default RMODE for the LCCA control block on z/OS systems at the pre-z/OS R12 level is RMODE 24. The check looks for RMODE 24 for the LCCA control block unless you specify an RMODE of 31 in the RMODE parameter for the check.

The ZOSMIGV1R12\_SUP\_LCCA\_ABOVE\_16M was shipped disabled in z/OS V1R10 and z/OS V1R11; it is shown in Figure 18-9.

```
CHECK(IBMSUP,ZOSMIGV1R12_SUP_LCCA_ABOVE_16M),
INTERVAL(ONETIME),
SEVERITY(LOW),
PARM('CBLOC(31)'),
DATE('20100124')
Reason('Your reason for making the update.')
```

*Figure 18-9 ZOSMIGV1R12\_SUP\_LCCA\_ABOVE\_16M Health check syntax*

Parameters accepted:

- ▶ CBLOC(31), which is the default, specifies that you want the check to generate an exception if it finds that IHALCCA had been specified in the CBLOC VIRTUAL24 parameter of the DIAGxx parmlib member.
- ▶ CBLOC(24) specifies that you want the check to generate an exception if it finds that IHALCCA either:
  - Had been specified in the CBLOC VIRTUAL31 parameter of the DIAGxx parmlib member.
  - Had not been specified in the CBLOC VIRTUAL24 parameter of the DIAGxx parmlib member because CBLOC VIRTUAL24 is the default for the LCCA.

ZOSMIGV1R12\_SUP\_LCCA\_ABOVE\_16M messages:

- ▶ IEAVEH091E - Expected does not match actual.

### ZOSMIGV1R12\_SUP\_PCCA\_ABOVE\_16M

Checks to see whether the residency mode (RMODE), specified for the PCCA control block in the CBLOC parameter of the DIAGxx parmlib member, is the expected value. The default RMODE for the PCCA control block on z/OS systems at the pre-z/OS R12 level is RMODE 24. The check looks for RMODE 24 for the LCCA control block unless you specify an RMODE of 31 in the RMODE parameter for the check.

The ZOSMIGV1R12\_SUP\_PCCA\_ABOVE\_16M was shipped disabled in z/OS V1R10 and z/OS V1R11; it is shown in Figure 18-9.

```
CHECK(IBMSUP,ZOSMIGV1R12_SUP_PCCA_ABOVE_16M),  
INTERVAL(ONETIME),  
SEVERITY(LOW),  
PARM('CBLOC(31)'),  
DATE('20100124')  
Reason('Your reason for making the update.')
```

*Figure 18-10 ZOSMIGV1R12\_SUP\_PCCA\_ABOVE\_16M Health check syntax*

Parameters accepted:

- ▶ CBLOC(31), which is the default, specifies that you want the check to generate an exception if it finds that IHAPCCA had been specified in the CBLOC VIRTUAL24 parameter of the DIAGxx parmlib member.
- ▶ CBLOC(24) specifies that you want the check to generate an exception if it finds that IHAPCCA either:
  - Had been specified in the CBLOC VIRTUAL31 parameter of the DIAGxx parmlib member.
  - Had not been specified in the CBLOC VIRTUAL24 parameter of the DIAGxx parmlib member because CBLOC VIRTUAL24 is the default for the LCCA.

ZOSMIGV1R12\_SUP\_PCCA\_ABOVE\_16M messages:

- ▶ IEAVEH101E - Expected does not match actual.





## BCP contents supervisor updates

This chapter discusses BCP contents supervisor enhancements for z/OS V1R12. There were a number of minor changes to enhance usability of contents supervisor commands and functions:

- ▶ LLA function enhancements
- ▶ Dynamic exits enhancements
- ▶ Dynamic LINKLIST enhancements
- ▶ Dynamic LPA enhancements
- ▶ Defining PROGxx defaults
- ▶ PROGxx SYSLIB enhancement
- ▶ SVCR 0 trace entries for ATTACH/LINK/SYNCH/XCTL
- ▶ RTLS withdrawal

## 19.1 LLA function enhancements

There are enhancements to three LLA functions:

- ▶ Dynamic exits CSVLLIX1 and CSVLLIX2
- ▶ Avoiding LLA busy response
- ▶ LLA automatic restart when SUB=MSTR is omitted

### 19.1.1 Dynamic exits CSVLLIX1 and CSVLLIX2

You can improve the performance of module fetching on your system by allowing library lookaside (LLA) to manage your production load libraries. LLA reduces the amount of I/O needed to locate and fetch modules from DASD storage.

LLA determines which modules, if staged, would provide the most benefit to module fetch performance. LLA evaluates modules as candidates for staging based on statistics it collects about the members of the libraries it manages (such as module size, frequency of fetches per module (fetch count), and the time required to fetch a particular module).

If necessary, you can directly influence LLA staging decisions through the installation exit routines CSVLLIX1 and CSVLLIX2. For information about coding these exit routines, see *z/OS MVS Installation Exits, SA22-7593*.

Prior to z/OS V1R12 the LLA fetch exit CSVLLIX1 and the LLA staging exit CSVLLIX2 were defined via the EXIT1 and EXIT2 statements in the CSVLLAxx parmlib member.

#### **z/OS V1R12 enhancements**

New function in z/OS V1R12 allows CSVLLIX1 and CSVLLIX2 to be added via the dynamic exits facility. This means that they can be managed as other dynamic exits through PROGxx parmlib members.

When EXIT1(ON) is specified, or if EXIT1 is not specified, in the CSVLLAxx parmlib member used to start LLA, the system attempts to add the CSVLLIX1 exit routine to the CSVLLIX1 dynamic exit when no exit routines are already added by PROGxx or SETPROG processing. The same is true for EXIT2. This preserves the pre z/OS V1R12 behavior for clients who do not use the dynamic exit facility via the PROGxx parmlib member or the SETPROG command.

If exit routines have been associated via PROGxx parmlib member and/or the SETPROG command, then the default processing is not done. In that case, all manipulations are to be done via the dynamic exits facility.

Once the exit routines are managed with PROGxx, the use of EXIT statements in CSVLLAxx is ignored. This means that you do not disable the EXIT by changing the EXIT1 or EXIT2 statement in CSVLLAxx to indicate OFF, but use the dynamic exits facility. This could be via the EXIT statement in PROGxx with the DISABLE option or the SETPROG EXIT command.

### 19.1.2 Avoiding an LLA busy response

LLA is manipulated by a modify command:

```
MOFDIFY LLA,command  
F LLA,command
```

Prior to z/OS V1R12, if one modify is in-process when another is received, the second gets a busy response.

New function in z/OS V1R12 allows up to 255 modifies to be started without getting a busy response. The 256th command would get a busy response. Requests received while an earlier one is in-process get queued and processed in turn and so the requestor does not get a busy response. This satisfies FITS request MR1111084222.

### 19.1.3 LLA automatic restart when SUB=MSTR is omitted

LLA should always be started by specifying SUB=MSTR to allow the address space to start independent of JES. If SUB=MSTR is incorrectly omitted when LLA is started, for example via the following command:

```
START LLA,nn=XY which asks to use parmlib member CSVLLAXY
```

LLA detects the omission of the recommended SUB=MSTR, terminates this START, and begins a new one adding SUB=MSTR.

Prior to z/OS V1R12 the specification of nn=XY was not propagated. The restart used the default parmlib member or the CSVLLAxx parmlib member used by an earlier successful start of LLA rather than the CSVLLAxx member specified via nn=XY of the failed START command. This would likely result in the wrong CSVLLAxx parmlib member being used for the restart of LLA.

New function in z/OS V1R12 ensures that the restart propagates the nn=XY specified in the failed START LLA command.

## 19.2 Dynamic exits enhancements

z/OS V1R12 introduces new function for dynamic exits. There are four enhancements, discussed in this section:

- ▶ Dynamic exits - replace
- ▶ Dynamic exits - param
- ▶ Dynamic exits type
- ▶ Dynamic exits foundbuterror

### 19.2.1 Dynamic exits replace

Prior to z/OS V1R12 it was not possible to replace a dynamic exit in such a way that the exit was always active. It could be done by using DELETE then ADD, or MODIFY to INACTIVE followed by MODIFY back to ACTIVE. The problem with both of these methods is that there is some time before the re-ADD when the exit routine does not exist at all. This can be critical for a security-related exit routine where it could be vital that the exit routine always be active.

z/OS V1R12 introduces the REPLACE function for dynamic exits addresses. The REPLACE function is available via:

- ▶ The EXIT REPLACE statement in PROGxx
- ▶ The SETPROG EXIT,REPLACE command
- ▶ The CSVDYNEX REQUEST=REPLACE macro

When REPLACEing an active exit routine, there is always one (and only one) copy of an exit routine that gets control at the exit point. There is never a point in time where there are zero or two copies.

## 19.2.2 Dynamic exits parameter

An exit routine might be able to take advantage of having a constant parameter passed to it. In z/OZ V1R12 a parameter can be defined on the EXIT ADD statement of PROGxx or the SETPROG EXIT,ADD command, or by the CSVDYNEX REQUEST=ADD macro.

The 8-byte parameter is placed into access registers 0/1 on entry to the exit routine. This is a little bit strange but it is the only place available for all situations. The z/OS command is:

```
DISPLAY PROG,EXIT,EXITNAME=xx,DIAG
```

This displays the data; it assumes the data is printable, so you will only see the data if it is printable. It is recommended that exploiters use printable data.

## 19.2.3 Dynamic exits type

There are two types of dynamic exits:

- ▶ Installation exits - type is "installation"
- ▶ Exits intended for use by programs - type is "program"

z/OS V1R12 provides functionality by which the owner of an exit can identify which of these types their exit is. Many exits are not yet identified and so an exit will have a type of installation, program, or not defined.

The DISPLAY PROG command can be filtered by type to limit the display.

```
DISPLAY PROG,EXIT[INSTALLATION, PROGRAM, NOTPROGRAM]
```

- ▶ INSTALLATION displays all exits with type "installation"
- ▶ PROGRAM displays all exits with type "program"
- ▶ NOTPROGRAM displays all exits with type "installation" or no type defined

## 19.2.4 Dynamic exits option foundbuterror

When using the CSVDYNEX macro to ADD or REPLACE a dynamic exit, you can request that the system issue a message (or not) when there is an error.

In z/OZ V1R12 a new option is added:

```
Message=FoundButError.
```

This indicates to write a message in any of the cases covered by Message=Error except for the case of "Exit routine not found".

This was used in the implementation of the LLA dynamic exits, but could be useful to other dynamic exits exploiters.



## 19.3 Dynamic LINKLIST enhancements

It is well known that using LNKLIST UPDATE either via PROGxx parmlib member or a SETPROG command is unpredictably dangerous. Part of the problem is that in-flight processing running against the old LNKLIST can be impacted because there is a window where processing occurs for the old LNKLIST data set after it is closed and freed.

z/OS V1R12 introduces an update delay during LNKLIST UPDATE processing. With the DELAY operand value (numbers of seconds), you can ask that the completion of the LNKLIST UPDATE command is to wait the specified DELAY time before closing and hence freeing the old link list data sets. This gives in-flight requests additional time to complete.

The command syntax is:

```
SETPROG LNKLIST,UPDATE,JOB=jobname,DELAY=nn
SETPROG LNKLIST,UPDATE,ASID=asid#,DELAY=nn
```

**Note:** The command itself will not complete until this delay has been taken into account.

## 19.4 Dynamic LPA enhancements

z/OS V1R12 introduces new function for dynamic LPA. There are five enhancements that are discussed in this section:

- ▶ Dynamic LPA AddAlias
- ▶ Dynamic LPA SVC Number
- ▶ Dynamic LPA Add by fully qualified HFS pathname
- ▶ Dynamic LPA Deferred LPA wait
- ▶ Dynamic LPA Query Only

### 19.4.1 Dynamic LPA AddAlias

When adding a module to dynamic LPA that has aliases prior to z/OS V1R12, the aliases had to be provided by name.

z/OS V1R12 allows aliases to be included without having to specify the alias names via the AddAlias keyword. This can be done via PROGxx, SETPROG, or the CSVDYLPA macro:

- ▶ PROGxx parmlib member, LPA ADD MOD(...) ADDALIAS
- ▶ SETPROG LPA ADD MOD(...) ADDALIAS
- ▶ CSVDYLPA REQUEST=ADD,ADDALIAS=YES

When the AddAlias keyword is found, the processing is similar to MemberMask. All directory entries are read because that is the only way to determine what aliases exist.

### 19.4.2 Dynamic LPA SVC number

When replacing an SVC routine on behalf of some authorized application it can be difficult even if you can get the module into storage. In addition to getting the new module into storage, the SVC table must be updated via the SVCUPDTE programming interface to reflect the new module's address.

z/OS V1R12 allows the replacement of SVC routines to be simplified in some cases. You can specify on a dynamic LPA operation both to fetch the module and also to update the SVC table using the SVC number information that you provide.

This can be done via PROGxx, SETPROG, or the CSVDYLPA macro:

- ▶ PROGxx parmlib member, LPA REPLACE MOD(...) SVCNUMDEC=svcnum
- ▶ SETPROG LPA REPLACE MOD(...) SVCNUMDEC=svcnum
- ▶ CSVDYLPA REQUEST=REPLACE,SVCNUMDEC=svcnum

As implied by the name, SVCNUMDEC is the decimal SVC number to be replaced.

**Note:** SVCNUMDEC can also be specified for ADD for a new SVC but is not as useful because this function updates only the module address. SVCs usually have other attributes such as the “type” that are not supported by this mechanism.

### 19.4.3 Dynamic LPA Add by fully qualified HFS pathname

Prior to z/OS V1R12, dynamic LPA modules could come from a PDS or a PDSE but they could not come from an HFS.

In z/OS V1R12 the CSVDYLPA macro is enhanced to allow the module to be fetched via the fully-qualified HFS pathname:

```
CSVDYLPA REQUEST=ADD, . . . , PATHNAME=pathname, PATHNAMELEN=pn1
```

**Note:** This functionality is not provided for use in PROGxx or by the SETPROG command. It is available only by the CSVDYLPA macro.

### 19.4.4 Dynamic LPA deferred LPA wait

LPA built during IPL (PLPA, MLPA, FLPA) cannot use PDSEs. Prior to z/OS 1.12, PDSE program objects could be added to LPA at the tail end of the IPL via the COMMNDxx parmlib member. The problem with this is that an application could not know that the LPA modules loaded from PDSEs via dynamic LPA were complete. This left it up to you to sequence these events.

**Note:** Prior to z/OS V1R12, LPA ADD statements in PROGxx were not processed at IPL time. In order to have the LPA ADD statements processed, a SET PROG=xx command would need to be added to COMMDxx.

With z/OS 1.12, LPA ADD statements can be included in the IPL-time PROGxx specification. They will be processed at a deferred point late in the IPL. In addition, new function is provided for an application to ask that it wait until all of the deferred ADD processing is complete.

This means that the LPA ADD statements will be processed via PROGxx at IPL time and the application will WAIT. The application will come out of the WAIT when the LPA ADD processing is complete and the dynamic LPA modules are ready for use. If the ADD processing is already done when the WAIT request is received, the WAIT is a no-op.

The new function can be exploited either by changing the application code or by modifying the JCL that runs the application.

## Changing the program code

The CVSDYLPA macro can be used to query whether the dynamic LPA ADD is complete. The query request does not require authorization. The following is sample code that uses the CVSDYLPA macro to issue a query request.

```
CVSDYLPA REQUEST=QUERYDEFLPA,DEFLPASTATE=the_state
CLI  the_state,Csvdy1paDefLpaComplete
JE   deferred_LPA_is_complete
```

If the dynamic LPA ADD is not complete, then the program must WAIT until deferred LPA processing completes.

- ▶ An authorized program could issue:

```
CVSDYLPA REQUEST=DEFLPAWAIT
```

- ▶ An unauthorized program could issue:

```
LINK EP=CSVDLPAW
```

## Changing the JCL

If a step is dependent on dynamic LPA ADD, then a prior step can be added to the JCL:

```
EXEC PGM=CSVDLPAW
```

This step completes when deferred LPA processing is complete.

## 19.4.5 Dynamic LPA Query Only

When adding an entire library to dynamic LPA or a subset using a mask, it may be important to know how much common storage ((E)CSA, (E)SQA) was going to be used. For example, you may want to avoid causing an extra megabyte of CSA to be used that would reduce the private area size.

In z/OS V1R12 the CVSDYLPA macro for REQUEST=ADD with MODINFOTYPE=MEMBERMASK supports a QueryOnly=YES option.

It does much of the processing for a normal REQUEST=ADD but stops short of doing the actual ADD. Instead, it just keeps track of how much storage would have been used if the ADD were done.

The output is mapped by the new DSECT LPMEAQ in macro CSVLPRET.

The idea is to use this prior to a re-IPL so that you can adjust your (E)CSA and (E)SQA amounts accordingly.

## 19.5 Defining PROGxx defaults

In many cases there are options that you know you always want when issuing particular commands but they are not required and you may forget to include them.

A typical example is when creating a new LNKLIST you almost always want to copy from what you currently have by including COPYFROM(CURRENT) in the SETPROG LNKLIST DEFINE NAME(XXX) command. If you omit COPYFROM(CURRENT) you will get a linklist with the five default data sets, which is most likely not what you want.

z/OS V1R12 allows you to define defaults for SETPROG commands issued via:

- ▶ PROGxx LPA statement
- ▶ SETPROG LNKLST command
- ▶ SETPROG LPA command
- ▶ DISPLAY PROG,EXIT command

These defaults are specified via the DEFAULTS statement in PROGxx. The syntax of the DEFAULTS statement is:

```
DEFAULTS LNKLST [REQCOPYFROM | NOREQCOPYFROM] [COPYFROMCUR | NOCOPYFROMCUR]
DEFAULTS LPA [ADDALIAS | NOADDALIAS]
DEFAULTS EXIT [ EXITTYPE({ALL | INSTALLATION | NOTPROGRAM}) ]
```

These defaults can be overridden by specifying the desired keyword. For example, the default is NOCOPYFROMCUR. This means that without specifying a DEFAULTS statement for LNKLST, a SETPROG LNKLST DEFINE NAME(xxx) command would not do a copy from the current linklist.

If you have specified

```
DEFAULTS LNKLST COPYFROMCUR
```

then a copy from the current linklist would be done by default.

To override this, you would issue:

```
SETPROG LNKLST DEFINE NAME(xxx) NOCOPYFROMCUR
```

**Note:** The defaults do not apply to the programming interfaces (CSVDYNEX, CSVDYNL, and CSVDYLPA).

## 19.6 PROGxx SYSLIB enhancement

The SYSLIB statement can be used to change the default system data sets that are placed at the beginning of the LNKLST concatenation or LPALST concatenation.

z/OS V1R12 allows a volume to be specified in the SYSLIB statement. This allows data sets that are cataloged in a user catalog rather than the master catalog to be specified.

The syntax is:

```
SYSLIB LINKLIB(name) [ {VOLUME|VOLSER|VOL} (volser) ]
SYSLIB MIGLIB(name) [ {VOLUME|VOLSER|VOL} (volser) ]
SYSLIB CSSLIB(name) [ {VOLUME|VOLSER|VOL} (volser) ]
SYSLIB LINKLIBE(name) [ {VOLUME|VOLSER|VOL} (volser) ]
SYSLIB MIGLIBE(name) [ {VOLUME|VOLSER|VOL} (volser) ]
SYSLIB LPALIB(name) [ {VOLUME|VOLSER|VOL} (volser) ]
```

**Note:** It is important that data sets placed into the LNKLST be cataloged in the catalog used by LLA after IPL. This means that even if you use VOLUME, you must catalog the LNKLST-related data sets in a user catalog. All the SYSLIB data sets other than the LPALIB data set are LNKLST-related.

## 19.7 SVCR 0 trace entries for ATTACH/LINK/SYNCH/XCTL

You may have seen unexpected SVCR 0 system trace entries for which there is no corresponding SVC 0 trace entry. These are not related to SVC 0, ECXCP, but rather to ATTACH, LINK, SYNCH, and XCTL.

When ATTACH, LINK, SYNCH, and XCTL are issued, the system creates an SVRB that is used to build the PRB for the target. Upon doing that, the SVRB is ordered “newer” than the PRB and then the SVRB terminates. The processing is slightly different for XCTL.

SVRB termination results in an SVCR trace entry. This is created by looking at the SVC's interrupt code that is captured in the RB pointed to by the SVRB. However, in this case there was no SVC that produced the SVRB because the SVRB was just “created”. The SVRB does point to another RB and the information in that RB is used. As there was no SVC and that PRB was initialized to 0s, the SVC interrupt code field is 0s, and so the SVCR trace entry is for SVC 0. The PSW in the trace entry shows the address of the loaded program, which is a useful piece of information.

New function in z/OS V1R12 causes an SVCR FF00 system trace entry rather than SVCR 0 when exiting from ATTACH, LINK, SYNCH, and XCTL.

Figure 19-1 shows a system trace extract for a LINK on z/OS V1R12.

00	0009	005F4320	SVC	6	070C0000	80BEC684	0849FEC4	0849EE70	00AD1E00	Link
00	0009	005F4320	SSRV	78		81127626	1000FF52	00000088	005F69F0	Getmain
							00090000			
00	0009	005F4320	SVCR	FF00	070C0000	85BC7000	85BC7000	0849EE70	00AD1E00	

Figure 19-1 System trace entries for LINK

## 19.8 RTLS withdrawal

In z/OS V1R11, APAR OA29995 provided support for the z/OS tracking facility to track the use of the following:

- ▶ SET RTLS command
- ▶ DISPLAY RTLS command
- ▶ CSVRTLS macro

Run-Time Library Services is withdrawn in z/OS V1R12 and the z/OS tracking facility support is provided via APAR OA29019. The RTLS system commands have been removed in z/OS V1R12.





## Extended Address Volume

This chapter describes the Extended Address Volume (EAV) enhancements provided in z/OS V1R12. This includes supporting additional data set types that can be allocated in the Extended Addressing Space (EAS) on an EAV. This is the third release in which EAV function has been provided. z/OS addressable disk storage is significantly increased with EAV and provides constraint relief for environments that are approaching the 4-digit device number limit in z/OS.

For z/OS V1R12, this chapter describes the following:

- ▶ Support additional non-VSAM data set types
- ▶ Binder support of data sets in the extended addressing space (EAS)
- ▶ JES2 EAV support for spool and checkpoint data
- ▶ JES3 EAV support for spool and checkpoint data sets
- ▶ Standalone dump support for EAV2
- ▶ Superzap support for EAV3

## 20.1 Introduction

This chapter describes a further enhancement to Extended Address Volumes introduced with z/OS V1R12. z/OS V1R10 and V1R11 already introduced the extended address volume (EAV), which allowed DASD storage volumes to be larger than 65,520 cylinders. The space above the first 65,535 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.

Extended Address Volumes (EAVs) are 3390 Model A volumes with an increased number of cylinders above what was previously supported as the maximum. This is accomplished with a new track address format, referred to as 28-bit cylinder track address. First introduced in z/OS V1R10, changes included the following:

- ▶ z/OS V1R10 allowed VSAM data sets to be defined in EAS, V1R11 added extended format sequential data sets, and V1R12 allows almost all additional data set types to reside in EAS.
- ▶ With z/OS V1R10 and higher releases, z/OS has added support for DASD volumes having more than 65,520 cylinders.
- ▶ The name in z/OS publications for these larger volumes is Extended Address Volumes, or EAVs.
- ▶ Since z/OS V1R10, the new DASD volume maximum size is 262,668 cylinders.

This chapter describes the following EAV functions and enhancements provided by z/OS V1R12 support in DFSMS:

- ▶ Support additional non-VSAM data set types
- ▶ Binder support of data sets in the extended addressing space (EAS)
- ▶ JES2 EAV support for spool and checkpoint data
- ▶ JES3 EAV support for spool data sets
- ▶ Stand-alone dump support for EAV2
- ▶ Superzap support for EAV3

## 20.2 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 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, which 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.



## 20.2.1 3390 Model A

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. Figure 20-1 on page 388 illustrates the 3390 device types.

**Important:**

- ▶ 3390 Model A support is provided in DS8000 3.1 versions.
- ▶ The EAV support is provided in the DS8000 4.0 version.

**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 should 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.

## 20.2.2 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 65,520 cylinders defined. With current z/OS releases, this maximum size is not supported.

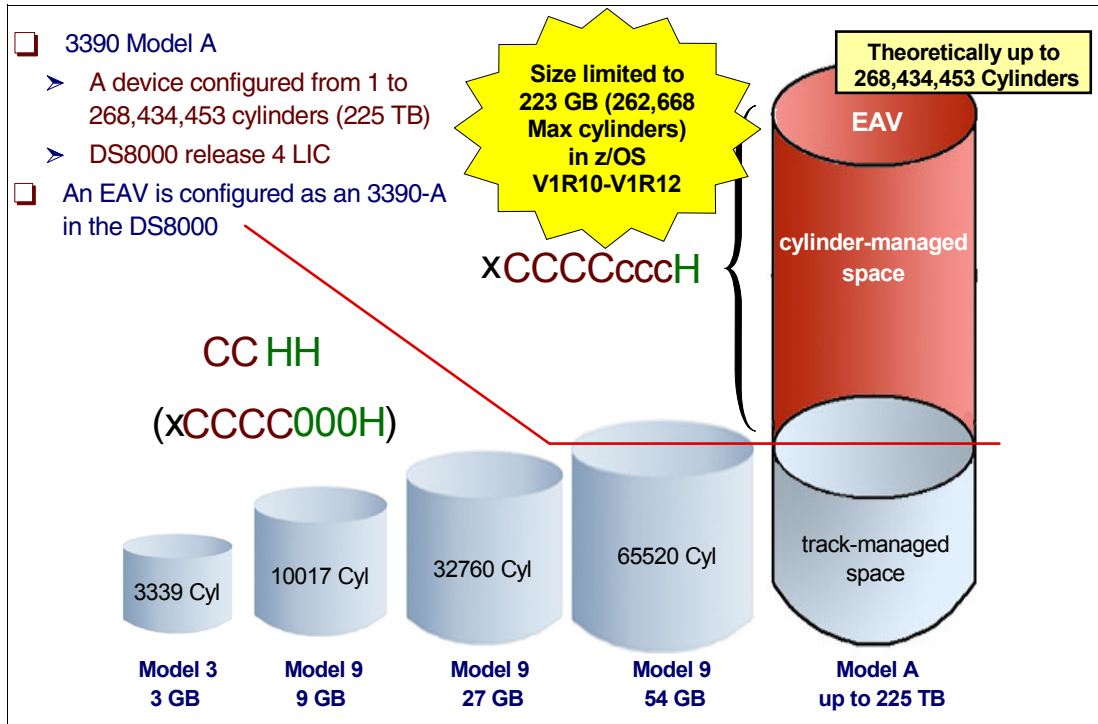


Figure 20-1 DS8000 support for device type 3390

## 20.3 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 refers to a volume with more than 65,520 cylinders. Only 3390 Model A devices can be an EAV.

### track address

This 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 refers to cylinders with addresses that are equal to or greater than 65,536. These cylinder addresses are represented by 28-bit cylinder numbers.

### base addressing space

On an extended address volume, this refers to cylinders with addresses below 65,536. These cylinder addresses are represented by 16-bit cylinder numbers or by 28-bit cylinder numbers whose high order 12 bits are zero (0).

### multicylinder unit

This 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 refers to the space on the volume that is managed only in multicylinder units. Cylinder-managed space begins at cylinder address 65,520. Each data set occupies an integral multiple of multicylinder units. Space requests targeted for the cylinder-managed space are rounded up to the next multicylinder unit. The cylinder-managed space exists only on EAVs.

**track-managed space** This refers to the space on a volume that is managed in tracks and cylinders. Track-managed space ends at cylinder address 65,519. 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 uses the track-managed space, or 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 uses the cylinder-managed space, or both areas.

## 20.4 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 is 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 20-2 on page 390.

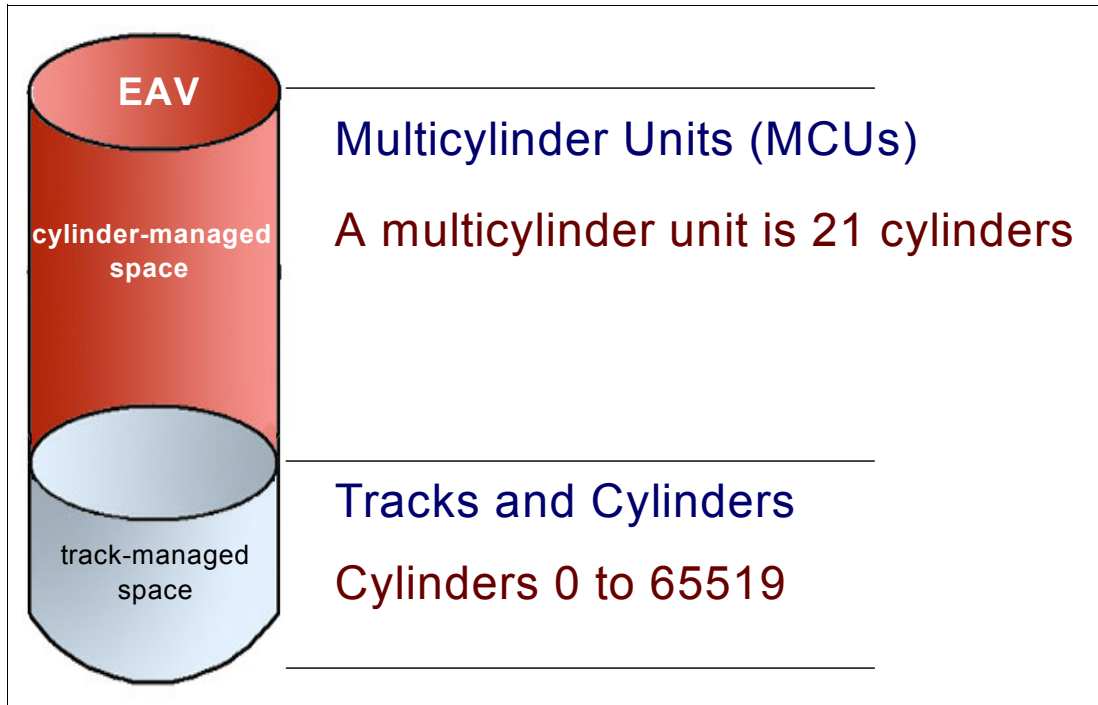


Figure 20-2 EAV and multicylinder units

### 20.4.1 Track-managed space

The track-managed space on a volume is managed in track and cylinder increments. All volumes today have track-managed space, which ends at cylinder address 65,519. 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.

### 20.4.2 Cylinder-managed space

The cylinder-managed space on a volume is managed only in multicylinder units (MCUs). Cylinder-managed space begins at cylinder address 65,520. Each data set occupies an integral multiple of multicylinder units. Space requests targeted for the cylinder-managed space are 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 block (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.

### 20.4.3 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 20-2.

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 1,113 cylinders.

These segments are allocated in multiples of 1,113 cylinders starting at cylinder 65,536.

#### 20.4.4 DASD track address format

As explained, an EAV is defined to be a volume with more than 65,520 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 388 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 20-3 on page 392. 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), shown in Figure 20-3 on page 392, is the area on an 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 volumes” on page 393.

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 a method to prevent 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 20.4.8, “New format DSCBs for EAVs” on page 397.

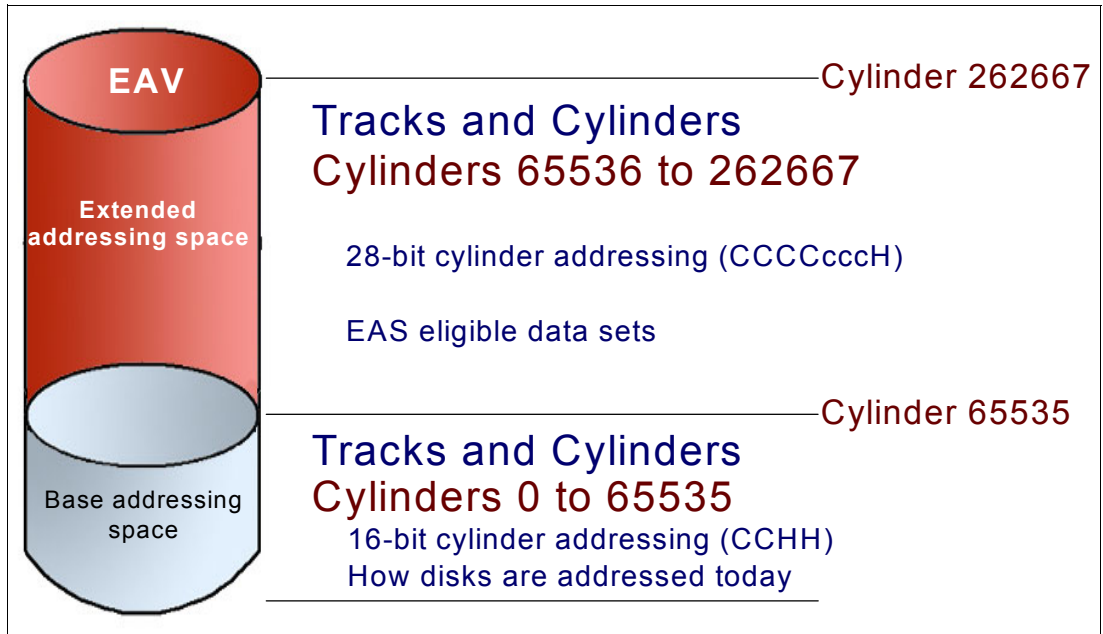


Figure 20-3 New track addressing for EAV volumes

### Old track address

The base addressing space is the area on an EAV located within the first 65,536 cylinders. As shown in Figure 20-4, 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 in 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.

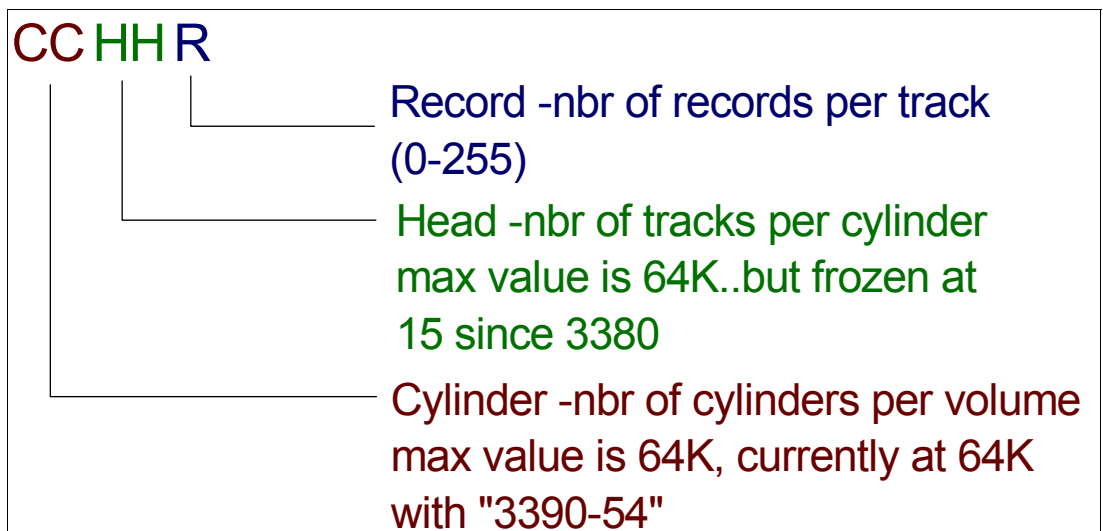


Figure 20-4 Old track address

## New track address for extended address volumes

The extended addressing space (EAS) is the area on an EAV located above the first 65,536 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.

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 are 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 20-6 on page 394.

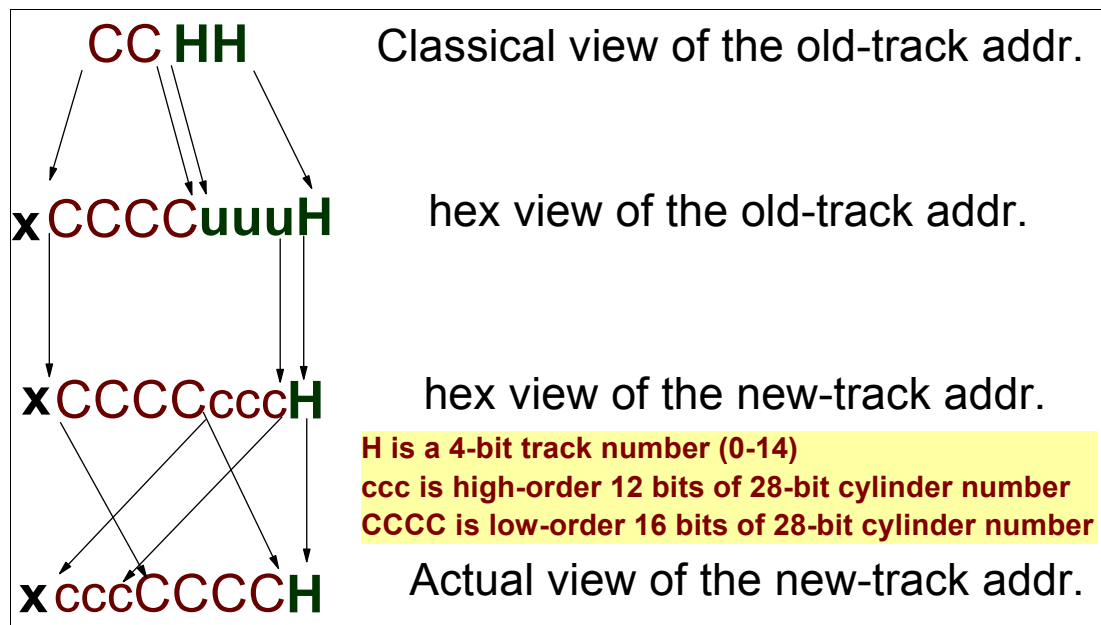


Figure 20-5 From old track address to new track address

### 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 20-6 on page 394. This format preserves the 3390 track geometry. Track addresses in existing channel programs and extent descriptors in DSCBs and elsewhere are in the form 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: CCCCcchH.

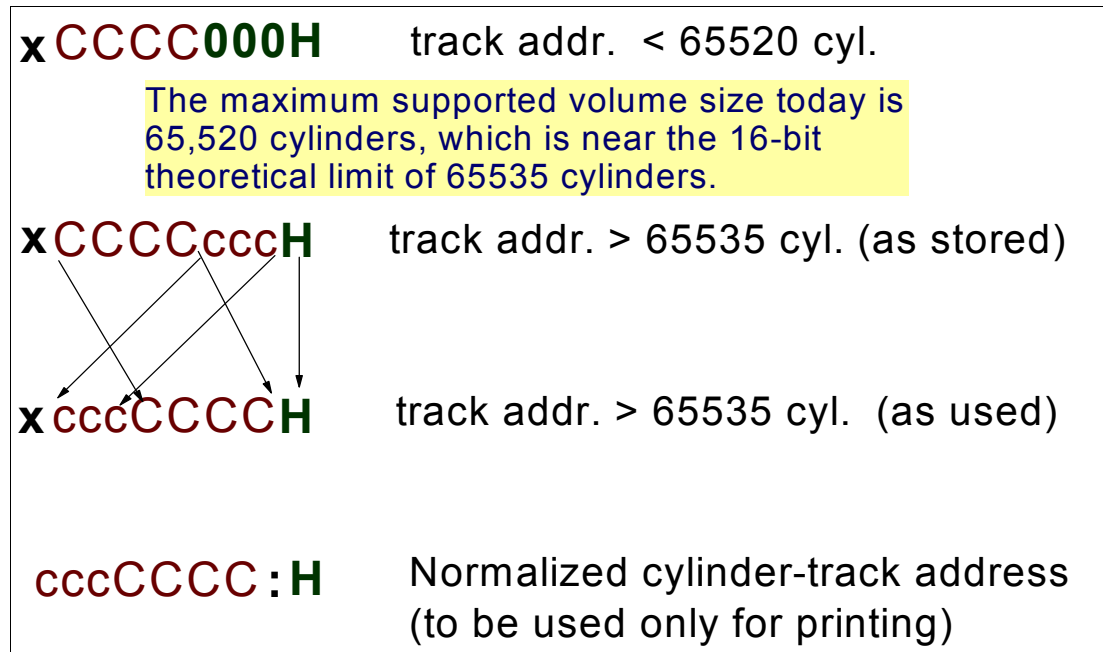


Figure 20-6 EAV: the two types of track address

### 20.4.5 Extended address volume attributes

In z/OS V1R10, each extended address volume, as shown in Figure 20-7 on page 395, 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's format-4 DSCB and the UCB's device class extension. Each of these attributes is to be tested independently. An exception is that a volume with more than 65,520 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 three attributes:

- ▶ The volume supports an extended addressing space.
- ▶ The volume supports cylinder-managed space.
- ▶ The volume 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.



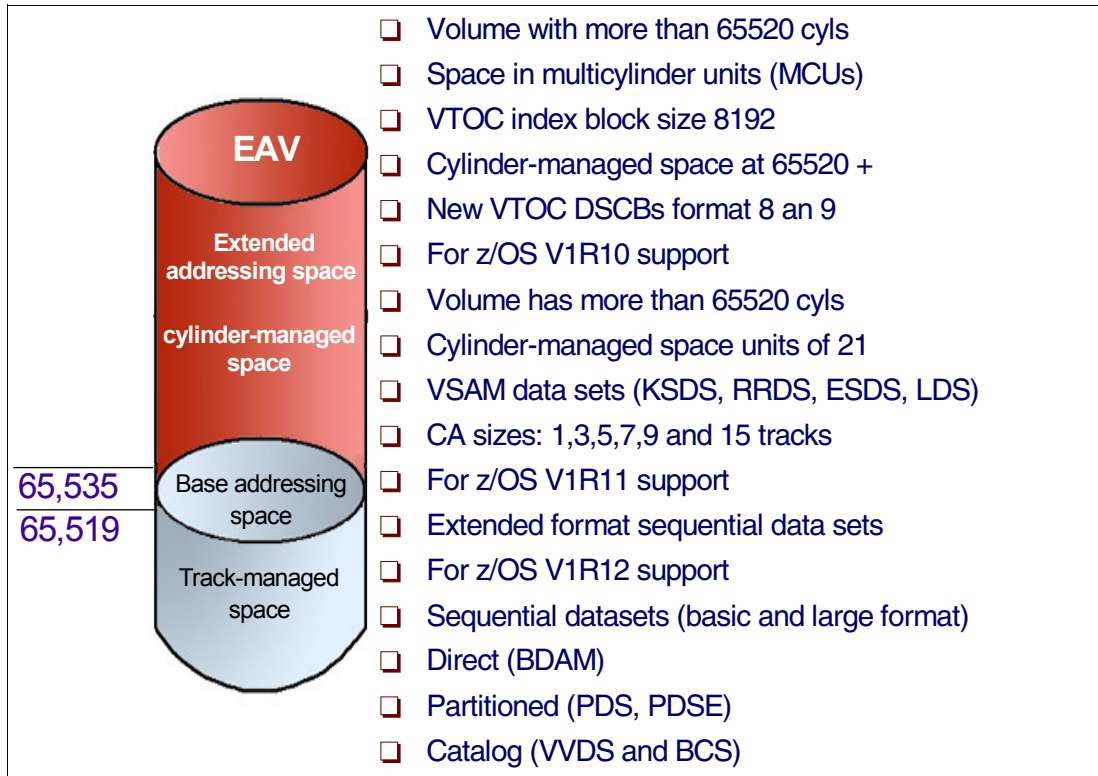


Figure 20-7 EAV attribute summary

## 20.4.6 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 65,536 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 or 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
- ▶ In z/OS V1R11: Extended-format sequential data sets that are SMS-managed can be allocated in the extended addressing space of an EAV.
- ▶ In 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 (CA) for non-striped VSAM, and minimum allocation units (MAU) 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 and MAU sizes 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.

## 20.4.7 EAS non-eligible data sets

An EAS-ineligible data set may exist on an EAV, but 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

**Note:** In a future release, several of these data sets might become EAS-eligible. All data set types, even those listed here, can be allocated in the track-managed space on a device with cylinder-managed space on an EAV.

Eligible EAS data sets can be created and extended anywhere on an EAV. Data sets that are not eligible for EAS processing can only be created or extended in the track-managed portions of the volume.

## 20.4.8 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 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 20-8 on page 399.

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 and 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 that point directly to up to ten 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.

### **Format-8 DSCB**

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.

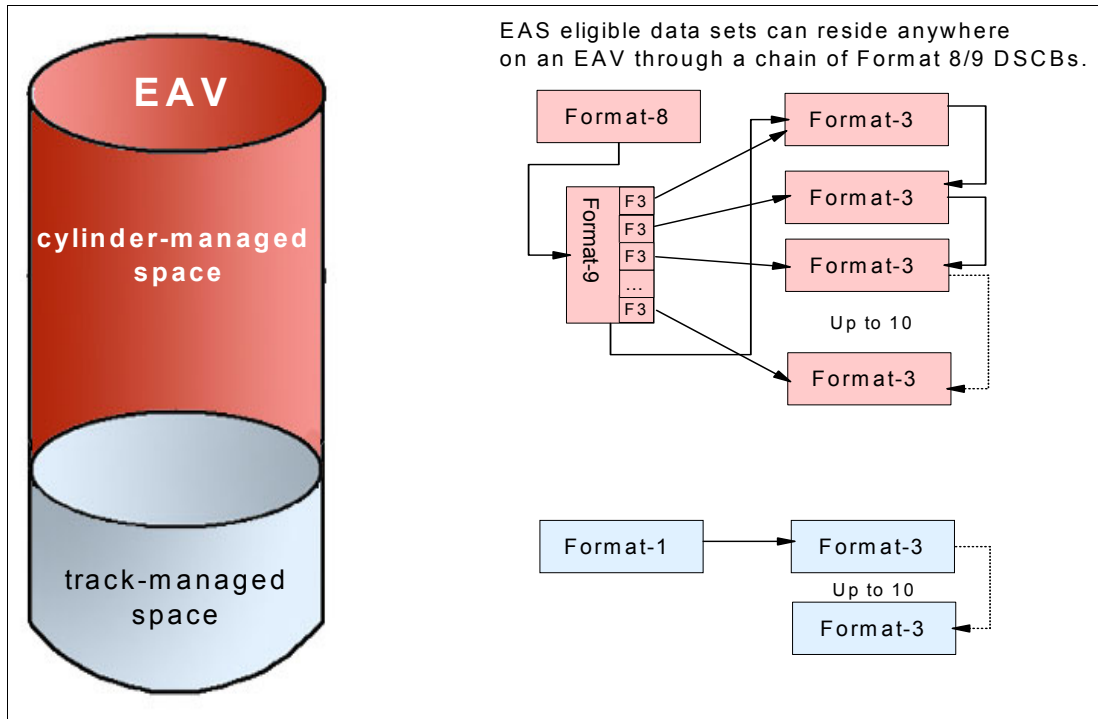


Figure 20-8 Access to EAS-eligible data sets on EAV

### The 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.

The 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.

**Important:** 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

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 that 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.

## 20.4.9 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 you to control whether a data set can have extended attribute DSCBs and thus control whether it can be allocated in EAS.

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 on an extended address volume (EAV). Valid values for the EATTR are NO and OPT.

EATTR of 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 of 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.

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.

The EATTR value is listed by IEHLIST ISPF, ISMF, LISTCAT, Catalog Search Interface (CSI), and DCOLLECT.

Volume selection uses the EATTR values for SMS and HSM non-SMS volume processing. EATTR is determined as follows, in this order:

- ▶ In the JCL interface by merging EATTR from the JCL
- ▶ LIKE= parameter
- ▶ 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 20-9.

```

DS1FLAG1
.... ..00EATTR not specified.
.... ..01DS1EATTR_NOEATTR=NO.
.... ..10 DS1EATTR_OPTEATTR=OPT.
    
```

Figure 20-9 Reading EATTR value

## 20.5 z/OS V1R12 enhancements for EAVs

Figure 20-10 shows enhancements provided in previous releases.

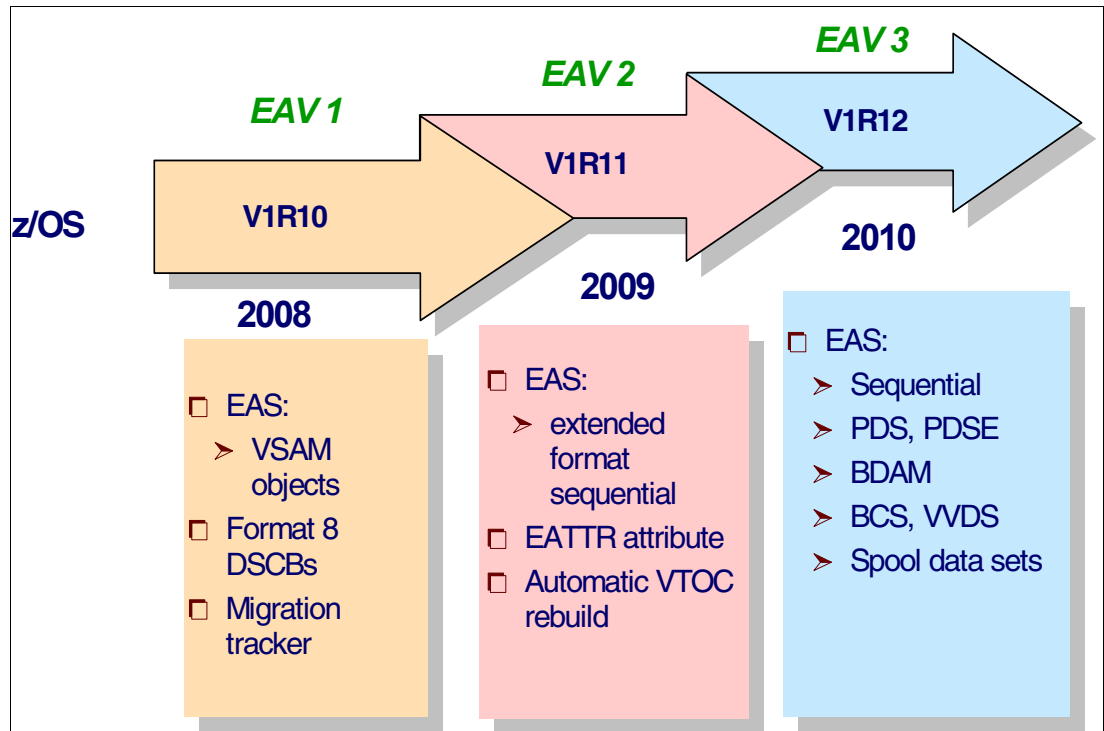


Figure 20-10 EAV evolution

The following enhancements to EAV are provided in z/OS V1R12:

- ▶ Basic catalog structure
- ▶ VSAM volume data sets
- ▶ DFSMSHsm support

- ▶ DFSORT
- ▶ Catalog data sets in EAS
- ▶ Some other minor enhancements

## 20.5.1 Basic catalog structure (BCS)

Every catalog consists of one BCS and one or more VVDSs. A BCS does not “own” a VVDS—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 kinds of data sets, as follows:

- ▶ 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, 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 20-11 on page 403 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.



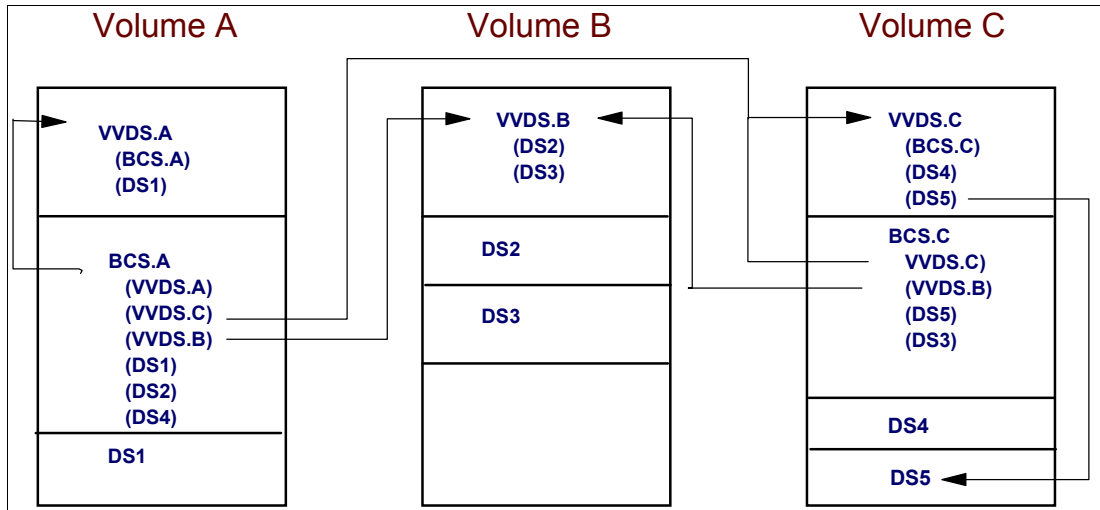


Figure 20-11 Relationship of the BCS and the VVDS

### 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, while 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 the SMS data class DSNTYPE=EXT parameter and subparameters R (required) or P (preferred) on the ISMF DATA CLASS DEFINE/ALTER panel. Use R to ensure that 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.

## 20.5.2 VSAM volume data sets

This support is provided in z/OS V1R12 only. It is for DEFINE CLUSTER for VVDS objects. AMS is changed to support EATTR as a keyword on DEFINE CLUSTER where the object is a VVDS data set.

To specify the EATTR value on IDCAMS DEFINE CLUSTER for a VVDS, you would use one of the following:

- EATTR(NO)** An EATTR value of NO indicates that the VVDS object cannot have extended attributes (format 8 and 9 DSCBs) or optionally reside in EAS.
- EATTR(OPT)** An EATTR value of OPT indicates that the VVDS object can have extended attributes and optionally reside in EAS. The actual use of extended attribute DSCBs depends on the target volume being an EAV.

The specified value for EATTR of NO or OPT is recorded in the VVDS in the DSCBs that get created in the VTOC. A DEFINE without the EATTR keyword will result in the VVDS object

restricted to track-managed space. This is the action taken by the pre-V1R12 system for VVDS defines.

The releases prior to z/OS V1R12 did not support the EATTR external on the DEFINE for a VVDS data set. Data Class/Model does not apply to DEFINE of a VVDS. The only way is with the EATTR keyword.

### **LISTCAT support**

**CI/CA field**      The number of control intervals per control area, if you wish to learn the number of tracks in each CA.

**EATTR field**      Indicates whether the VSAM data set can reside in the cylinder-managed space of an EAV.

- ▶ NULL - Indicates that the attribute was not specified or that the data set type is non-VSAM.
- ▶ NO - Indicates that the VSAM data set can only reside in track-managed space.
- ▶ YES - Indicates that the VSAM data set can reside in track-managed and cylinder-managed space.

## **20.5.3 DFSMSHsm support**

In z/OS V1R12, DFSMSHsm is enhanced to support the additional non-VSAM data set types in the EAS.

### **ML1, ML2, and backup EAV support**

DFSMSHsm enables all the EAV volume space to be used when they are defined as HSM-owned disk volumes (ML1, ML2, and backup volumes). Migration copies, backup versions, VTOC and catalog copies can be allocated in the EAS. A special flag is added to the MCC and MCD records, respectively, to indicate that the migration copy or backup version is located in the EAS.

In z/OS V1R12, DFSMSHsm does allow use of the cylinder-managed space on EAVs for migration copies and/or backup versions. A new SETSYS value is introduced to manage allocation of ML1, ML2, or backup data sets. The new command is:

```
SETSYS USECYLINDERMANAGEDSPACE(Y|N)
```

The two possible values are:

- (Y)**      Migration copies and backup versions can reside in EAS. For allocation, DFSMSHsm uses EATTR=OPT for its migration copy or backup version.
- (N)**      Migration copies and backup versions cannot reside in EAS. For allocation, DFSMSHsm does not specify an EATTR value.

The default value is N. The command can be abbreviated as USECMS.

The parameter has the same meaning when applied to SMS-managed or non-SMS-managed DASD volumes or data sets.

**Notes:**

1. For the FREEVOL command, if the SETSYS USECYLINDERMANAGEDSPACE(Y) command is issued and migration copies, backup versions and/or VTOC copies are going to be moved, then EATTR = OPT will be used. Otherwise, migration copies, backup versions, and/or VTOC copies being moved by the FREEVOL command will not specify an EATTR value.
2. For the ARECOVER command, the value of the USECYLINDERMANAGEDSPACE parameter (YIN) manages allocation of migrated data sets specified in the INCLUDE list.

The DFSMSHsm Journal (basic or large format SAM) can now be allocated in the EAS. Additionally, all of the backup data sets for the HSM CDSs and journal can now be allocated in the EAS. All other DFSMSHsm sequential data sets (PDA, LOG) can now be allocated in the EAS.

DFSMSHsm supports the data set level attribute EATTR for all EAV-eligible data sets and process SMS and non-SMS recalls or recovers accordingly.

## 20.5.4 DFSORT support

In z/OS V1R12, DFSORT data sets that can be allocated in EAS:

- ▶ SORTIN, SORTOUT, OUTFIL support
  - Maximum size of large format sequential is supported.
- ▶ SORTWK support
  - If basic format sequential, a data set is limited to 65,534 tracks.
  - If large format sequential
    - A data set is limited to 1,048,574 tracks (blocks).
    - Larger data sets can be allocated but excess space will not be used.

## 20.5.5 Catalog data sets in EAS

In z/OS V1R12, DFSMS volume selection is enhanced to support catalog data sets in EAS. During allocation of an SMS-managed catalog data set, SMS checks the EATTR value to determine its EAS eligibility. If it is EAS eligible (EATTR= OPT is specified), SMS prefers EAV volumes over non-EAV volumes when the requested space is equal to or greater than the BPV, and treats EAV and non-EAV volumes equally when the requested space is less than the BPV. If it is not EAV-eligible (EATTR=NO or not specified), SMS treats both EAV and non-EAV volumes equally regardless of the requested space amount.

**Note:** Although catalogs are VSAM data sets, the default behavior taken for EATTR by the system for catalogs is not the same as the default behavior taken for other VSAM. For catalogs, the system processes EATTR when not specified as if NO had been specified.

### LIKE= processing for PDS

When the data set referenced by LIKE= is a PDS, SMS does determine the number of directory blocks in the referenced data set by reading the first extent from the beginning until the EOD marker.

## Other changes

Support for additional data set types that are now EAS-eligible is provided for DFSMSdss, EREP, and IBM utilities such as IEBCOPY.

## 20.6 Job entry subsystems support

Detailed descriptions of the way Job Entry Subsystems support EAV can be found for JES2 in 30.1, “EAV support for spool and checkpoint data sets” on page 620 and for JES3 in 31.1, “EAV support for spool, checkpoint and JCT data sets” on page 632.

## 20.7 Service aids support

Two tools have been updated in z/OS V1R12 in order to provide support for EAVs, namely:

- ▶ Standalone dump
- ▶ Superzap

9.1, “SADMP support for EAV volumes” on page 184 and 9.2, “Superzap support for EAV3” on page 188 in the chapter on Service Aids describe these new supports.

## 20.8 Migration considerations

z/OS V1R12 supports the following types of data sets in the extended addressing space (EAS) on EAV volumes:

- ▶ Additional non-VSAM data set types I
- ▶ Sequential (basic, large)
- ▶ Partitioned (PDS/PDSE)
- ▶ Catalogs
- ▶ BDAM data sets

This new support has the following consequences for previous EAV-supported releases V1R10 and V1R11:

- ▶ Prior to z/OS V1R12, any attempt to open one of these types of non-VSAM data sets, which have been allocated with a format 8 DSCB (from a z/OS V1R12 system), will fail with ABEND IEC144I 313-0C.
- ▶ Prior to z/OS V1R12, if the next data set in the concatenation is one of these types of non-VSAM data sets which have been allocated with a format 8 DSCB, EOVS will fail with new ABEND IEC023I 237-24.
- ▶ In z/OS V1R12, any attempt to open one of these types of non-VSAM data sets which have been allocated with a format 8 DSCB, will succeed with standard BSAM, BPAM and QSAM access:
  - DCBE with EADSCB=OK is not required.
  - Cannot rely on open ABEND.
  - Data set extents in the DEB may contain 28-bit cylinder numbers.
    - Programs need to be changed to support 28-bit cylinder addressing.

- The system has no way to detect whether an application program can tolerate 28-bit cylinder numbers.
- EXCP access still requires EADSCB=OK and will continue to get the open ABEND if EADSCB=OK is not specified on the DCBE macro.

### 20.8.1 EATTR attribute

z/OS V1R12 can share EAVs with pre-z/OS V1R12 systems until explicit action is taken to begin allowing non-VSAM data sets to reside in the EAS of an EAV.

This applies to basic and large sequential, partitioned, and direct data sets, because EATTR will default to NO for non-VSAM data sets. A non-VSAM data set allocated with extended attribute DSCBs on z/OS V1R12 cannot be opened on pre-z/OS V1R12 systems.

EATTR is specifiable for non-EAS-eligible data sets:

- ▶ EATTR=OPT could have been specified on pre-z/OS V1R12 systems and ignored.
- ▶ While in z/OS V1R12, the EATTR=OPT setting will take effect.
- ▶ Be certain that applications can handle extended attribute DSCBs and 28-bit cylinder numbers because the data sets may become EAS-eligible in z/OS V1R12.

### 20.8.2 EOVS like concatenation

Like concatenation, DASD to DASD, with a non-SMS-managed EAS data set (a data set with format 8 or 9 DSCBs) is now possible with the support in z/OS V1R12. For example, a non-SMS-managed non-EAS non-VSAM data set could be first in the concatenation followed by a non-SMS-managed EAS non-VSAM data set.

Supposing that the DCBEEADSCBOK in the DCBE is not set on, open routines now allow the first data set to be opened even though DCBEEADSCBOK is not on because it is not EAS. However, when EOVS attempts to switch to the EAS data set that is not SMS-managed, EOVS is going to check to see whether DCBEEADSCBOK is specified. If it is not specified, a new ABEND 237-28 with message IEC023I is issued. This ABEND is described as shown in Figure 20-12.

**237-28** During EOVS concatenation, a format 8 DSCB was read for a data set that was EAS eligible but EADSCB=OK was not specified.

*Figure 20-12 Abend 237-28*

### 20.8.3 OPEN considerations

Since BDAM is eligible for EAS in z/OS V1R12, an OPEN with OPTCD=A of a BDAM data set described with extended attribute DSCBs (format 8 or 9) is failed with existing ABEND 113-44 unless DCBEEADSCBOK is set.

In other words, when using a DCB opened for BDAM access with an OPTCD=A (which indicates use of actual device addresses, MBBCCHHR), the EADSCB=OK keyword on the DCBE macro is required to open it when the data set is described with extended attribute DSCBs (format 8 or 9). If OPTCD=R is specified or no OPTCD is specified, then EADSCB=OK is not required.

## 20.8.4 XRC considerations

XRC control, master, and cluster data sets can be allocated on an EAV, but should not be allocated in EAS. Such data sets, if allocated with extended attribute DSCBs, will be inaccessible.

Journal data sets can be allocated in EAS at z/OS V1R11 and higher.

State data sets may be allocated in EAS at z/OS V1R12 and higher.

An error will result, and the data set will be inaccessible, if the XRC control, master or cluster data set is allocated in EAS on any z/OS release, or if the journal or state data set is allocated in EAS and accessed from a down-level system. It is therefore recommended that SDM journal data sets be allocated in EAS only when all active and recovery SDM systems are at z/OS V1R11 or higher. Because of their small size, little benefit is gained by allocating an SDM data set other than the Journal in EAS.

## 20.8.5 Format 8 DSCBs

To help in migrating, a program can test the bits in DFAFEAT8 in the DFA, mapped by IHADFA, to learn which data set types can be opened with format 8 DSCBs on a given release, as shown in Figure 20-13.

```
1... ..DFAVSAMFOREASVSAM
.1.. ..DFASEQFOREASBasic & large format sequential
..1. ..DFAPDSEFOREASPDSE
...1 ...DFAPDSFOREASPDS
.... 1...DFADIRFOREASDirect (BDAM)
.... .1..DFAEFSEQFOREASExtended format sequential
.... ..1.DFAUNDEFFOREASNo DSORG
```

Figure 20-13 Data set types that can be opened with format 8 DSCBs

- ▶ In V1R10, only the VSAM bit will be on (with PTF on V1R10).
- ▶ In V1R11, the VSAM and extended format sequential bits will be on.
- ▶ In V1R12, all the bits will be on.

## 20.8.6 Tracking facility

The objective of the EAV migration assistance tracker is three-fold:

- ▶ First, to identify executions of select system services by job and program name where the invoking programs may require analysis for changes to use new services provided by the system service. These program calls and the reported output are not considered in error because valid information is returned. They will be considered as informational instances.
- ▶ Second, to identify the possible improper use of returned information, like parsing 28-bit cylinder numbers in output as 16-bit cylinder numbers. These will be considered as warning instances.
- ▶ Third, to identify instances that will either be failed or identified with an informational message if they are run on an extended address volume. These will be considered as error instances. These errors apply to the following functions when the target volume of the operation is non-EAV and the function invoked did not specify the EADSCB=OK keyword:

- ▶ DADSM OBTAIN
- ▶ CVAFDIR
- ▶ CVAFSEQ
- ▶ CVAFFILT
- ▶ EXCP OPEN of an EAS eligible data set (except extended format sequential and PDSE data sets because EXCP OPEN is not allowed for these data set types)

In z/OS V1R12, access to all EAS-eligible data sets is tracked. In z/OS V1R10 only access to VSAM data sets is tracked. In z/OS V1R11 only access to VSAM and extended-format data sets is tracked. An instance will be recorded for any EAS-eligible data set when the EADSCB=OK keyword is not specified. This test applies to non-EAVs.

The other instances are volume oriented and are not affected by additional data set types becoming eligible for EAS. They include:

- ▶ CVAFVSM
- ▶ CVAFDSM
- ▶ LISTCAT
- ▶ DCOLLECT
- ▶ IEHLIST
- ▶ DEVTYPE
- ▶ LSPACE
- ▶ LISTDATA PINNED
- ▶ EXCP OPEN VTOC

## 20.9 Coexistence considerations

Different coexistence scenarios have to be considered for having pre-V1R10, V1R10, V1R11 and V1R12 systems coexist.

### Sharing EAVs across systems

z/OS V1R12 systems can share EAVs with systems at the z/OS V1R10 and z/OS V1R11 level, until the installation takes explicit action to allow non-VSAM data sets to reside in the EAS of an EAV.

For VSAM data sets allocated on EAVs between z/OS V1R12, z/OS V1R11, and z/OS V1R10, sharing is implied because all levels support VSAM files as being EAS-eligible. By specifying an EATTR value of NO for VSAM files the allocation is done without extended attribute DSCBs.

The explicit action of defining a data set with an EATTR value of OPT to allow a data set to be allocated with extended attribute DSCBs can be done from either a z/OS V1R12 or V1R11 system.

### Open, close, EOVS

In a mixed environment with z/OS V1R10 or V1R11, the APAR OA28651 should be considered for proper open, close, and EOVS handling, as follows:

- ▶ UA52685 for z/OS V1R10
- ▶ UA52686 for z/OS V1R11

## EOV like concatenation

When running on a z/OS V1R11 system with fix for OA28651, a new ABEND will be issued because the EAS data set allocated in z/OS V1R12 is not eligible for EAS when accessed from a z/OS V1R11 system. In this case a new ABEND, 237-24, message IEC023I, is issued.

This ABEND is described as shown in Figure 20-14.

**237-24** - During EOV concatenation, a Format 8 DSCB was read for a data set that was not EAS eligible

*Figure 20-14 ABEND 237-24*

## LIKE= processing

Fixes for OA29877 in z/OS V1R10 and V1R11 provide toleration support for additional non-VSAM data sets on these systems. This toleration allows SMS to use LIKE= to non-VSAM data sets that reside in the EAS or have format 8 DSCBs. This is the only way that a V1R10 or V1R11 system can use a non-EAS-eligible data set.

This coexistence code for z/OS V1R10 and z/OS V1R11 is required to handle the following:

- ▶ LIKE= to PDS data sets that reside in extended address space
- ▶ LIKE= to non-VSAM data sets that have format 8 DSCBs
- ▶ ACS source and listing data sets that have format 8 DSCBs

## DFSMSHsm considerations

In z/OS V1R12, DFSMSHsm supports the use of EAVs for L0, migration, and backup volumes. However, since they cannot be successfully accessed on a release prior to z/OS V1R12, their use should be restricted until all systems in a mixed sysplex environment with z/OS V1R12 are all at V1R12.

Existing z/OS V1R10 and V1R11 toleration support for DFSMSHsm will detect when EAVs are used for migration and backup volumes and fail the request if a data set being recalled or recovered is EAS-eligible. Some of the HSM-owned data sets (Journal, LOGx-y, CDS backup copies, and PDA) could be allocated in EAS on z/OS V1R12 and not be accessible on z/OS V1R11 or V1R10. Existing z/OS V1R10 and V1R11 DFSMSHsm toleration support will detect this situation and fail with a message stating that EAVs should not be used until all systems are at a z/OS V1R12 level, which supports all data set types in the EAS.

EAVs should not be used as HSM-owned volumes until all systems in the HSMplex are at z/OS V1R12 or higher.

Fixes for OA30606 on z/OS V1R10 and V1R11 provide support for z/OS V1R11 and V1R10 levels of DFSMSHsm to coexist with V1R12 levels of DFSMSHsm when using DFSMSHsm functions to manage all EAS-eligible data sets on Extended Address Volumes. With this support added, one can expect the following functions to recognize non-VSAM data sets (except non-VSAM SAM EF in V1R11) that have format 8 DSCBs but fail when processing them. These are the DFSMSHsm functions:

Migration, Backup, Abackup, Recall, Recover, FRRecover, Arecover, Audit Mediacontrols, ARCBJRLN and ARCIMPRT utilities, FREEVOL, Primary Space Management, Interval Migration, Secondary Space Management, and processing of DFSMSHsm-owned data sets that have format 8 DSCBs.



## **BCS and VVDS catalogs in EAS**

Coexistence code for z/OS V1R10 and z/OS V1R11 (provided with fixes for OA29932, OA31449, OA29933, and OA30161) allows master catalogs on those environments to be accessed if they were created in EAS on a z/OS V1R12 system. A PTF is used to provide that support in these two releases. Catalogs and VVDSs will continue to not be EAS-eligible from these two releases. In other words, you will still not be able to DEFINE catalogs and/or VVDSs in EAS from z/OS V1R10 and z/OS V1R11.

Coexistence code for z/OS V1R10 and z/OS V1R11 allows the EATTR data set attribute between releases to be preserved for the BCS, just as it does for VSAM files as done in z/OS V1R10 and z/OS V1R11. This support is necessary to allow the AMS functions of EXPORT and IMPORT to preserve the EATTR data set attribute of a BCS between releases. However, this support does not allow a user to DEFINE catalogs or the VVDS in EAS with the EATTR keyword from z/OS V1R10 and z/OS VR11 because the EATTR is not a valid external.

## **IDCAMS**

IDCAMS allows the EATTR for a UCAT in z/OS V1R12 to be EXPORT to the portable data sets. A subsequent import for a UCAT in z/OS V1R12 will preserve the original EATTR value. The prior releases z/OS V1R10 and z/OS V1R11 will be changed to perform the same function, thus allowing the EATTR value to be preserved during export and import processing between system levels. This is provided by APAR OA29933.

## **DADSM**

During the creation of the data sets, the system honors the user-specified EATTR value for data sets that were previously restricted to track-managed space. This is APAR OA30161 (UA52742 for z/OS V1R11).

## **DFSMSdss support**

Fixes for OA29471 on z/OS V1R10 and V1R11 allow these systems to recognize VVDS, BCS, basic and large sequential, partitioned and direct data sets with extended attribute DSCBs in order to prevent processing them by a DFSMSdss COPY or DUMP operation from a pre-z/OS V1R12 system.

For z/OS V1R10 and V1R11 DFSMSdss support, all VVDS, BCS, and non-VSAM data sets that have format-8 DSCBs will not be supported for COPY and DUMP operations. In order to COPY or DUMP these types of data sets, z/OS V1R12 is required. Extended-format sequential data sets are an exception on V1R11, because they are supported with format-8 DSCBs at this release level.

## **DFSORT V1R10**

z/OS V1R10 or V1R11 systems need the coexistence support applied in order for DFSORT message ICE289A to be issued if an unsupported data set in the EAS on an EAV is opened.

This is provided with fix UK45047 for DFSORT V1R10, which runs on z/OS V1R10 and z/OS V1R11.





# HyperSwap

Basic HyperSwap on z/OS is enabled by the IBM TotalStorage Productivity Center for Replication for System z and is designed to provide a low-cost, single-site, high-availability disk solution that allows the configuration of disk replication services using an intuitive GUI from z/OS.

Unplanned Basic HyperSwap masks primary disk subsystem failures by transparently switching to use secondary disks. Planned Basic HyperSwap provides the ability to perform disk maintenance without requiring applications to be quiesced.

This chapter describes the enhancements for basic HyperSwap in the following areas:

- ▶ Basic HyperSwap
- ▶ Metro Mirror sessions with HyperSwap capabilities
- ▶ Metro Global Mirror with HyperSwap
- ▶ Basic HyperSwap and data migration coexistence
- ▶ Couple data set avoidance in z/OS V1R11
- ▶ Enhanced device discovery
- ▶ Basic HyperSwap resiliency as provided in z/OS V1R12

## 21.1 Basic HyperSwap overview

Basic HyperSwap is an enhancement designed to improve z/OS availability by allowing z/OS to recover from the failure of one or more volumes (even entire storage controllers) without causing system or application outage.

Basic HyperSwap is enabled by IBM Tivoli Storage Productivity Center for Replication for System z (TPC-R). IBM Tivoli Storage Productivity Center for Replication Basic Edition for System z (TPC-R Basic Edition) is offered as a no charge license to z/OS clients so they can enable Basic HyperSwap capability. When enabled with this version of TPC-R, Basic HyperSwap provides a high-availability solution. Today, when enabled by the appropriate licensed version of TPC-R, Basic HyperSwap can provide two-site and three-site disaster recovery capabilities.

Basic HyperSwap utilizes the IBM synchronous storage replication feature (Metro Mirror) in order to maintain an identical copy of data in a second storage controller. In the event of a storage failure, the Basic HyperSwap code in z/OS will detect the error, and if the affected volumes are HyperSwap-managed, seamlessly swap the UCBs over to the copies, directing subsequent I/Os to the new primary devices. This is depicted in Figure 21-1 on page 415 where HyperSwap is enabled, and Figure 21-2 on page 416 after the HyperSwap has occurred and replication has been restarted in the opposite direction.

A planned failover can also be initiated to secondary storage controllers before performing hardware maintenance on primary storage controllers; or simply to periodically test the function.

TPC-R provides the graphical user interface for Basic HyperSwap and enables it through three session types:

- ▶ HyperSwap
- ▶ Metro Mirror with HyperSwap
- ▶ Metro Global Mirror with HyperSwap

This chapter describes Basic HyperSwap and recent enhancements to TPC-R and Basic HyperSwap.

## 21.2 Basic HyperSwap

Basic HyperSwap is a component of z/OS that actually performs the HyperSwap. It is comprised of two address spaces:

- ▶ The HyperSwap management address space

The HyperSwap management address space communicates over XCF to its peers on the other members of the sysplex, and manages the HyperSwap.

- ▶ The HyperSwap API address space

The HyperSwap API address space performs the actual functions of the HyperSwap. These functions are performed on a system basis, under the direction of the HyperSwap management address space, or in a GDPS environment, under the direction of GDPS.

Functions performed by the HyperSwap API address space include, for example, saving the configuration passed to it (LOAD), periodically verifying connectivity to the secondary devices, and performing the HyperSwap.

These address spaces can be started by adding simple procedures to SYS1.PROCLIB and then by issuing a START procname command manually, or by including the command in the COMMNDxx parmlib member of your SYS1.PARMLIB.

In order to enable Basic HyperSwap, you need a version of IBM Tivoli Storage Productivity Center for Replication for System z. TPC-R provides the graphical user interface for Basic HyperSwap, maintains the HyperSwap configuration repository, and provides additional management and monitoring functions for the HyperSwap session.

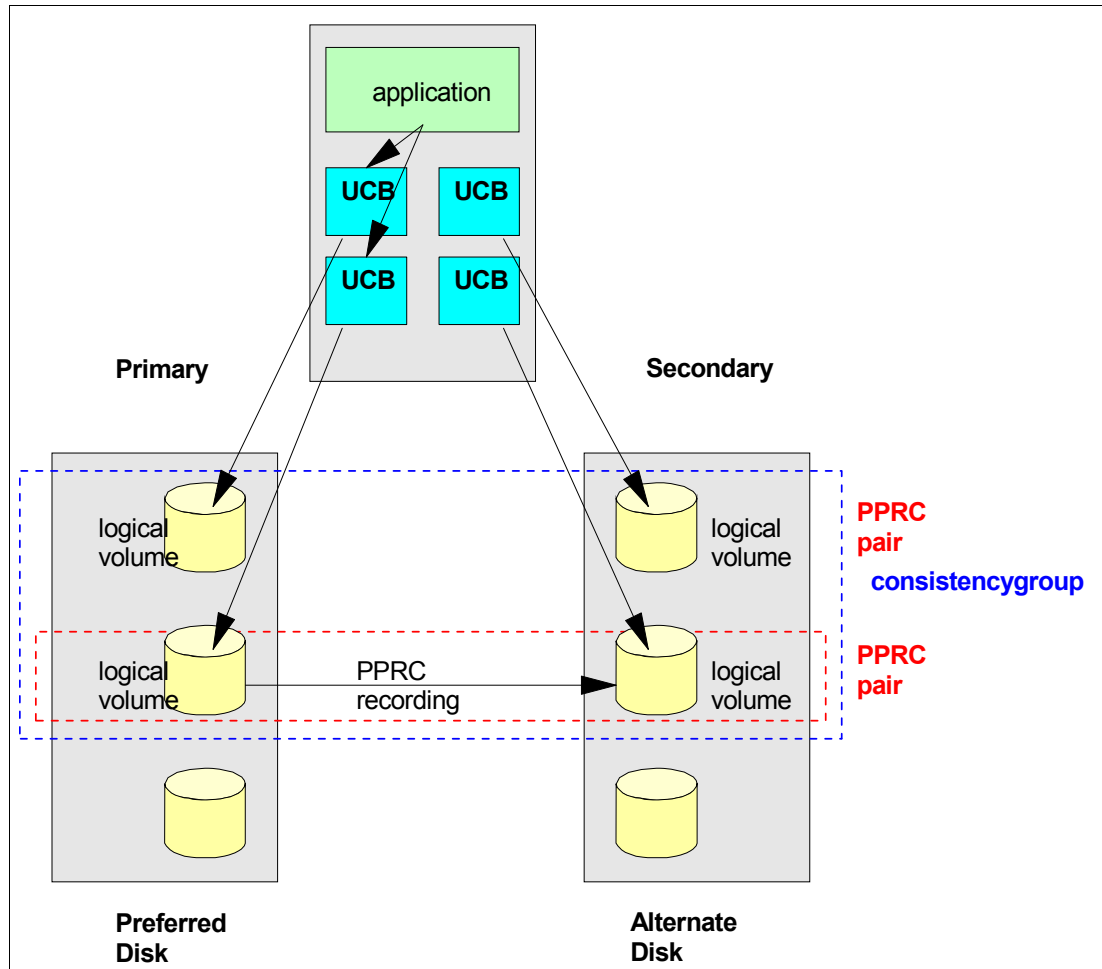


Figure 21-1 HyperSwap in place

## 21.2.1 Relationship of GDPS HyperSwap and Basic HyperSwap

HyperSwap was originally a function introduced by GDPS. Similar to Basic HyperSwap, this function allowed I/O requests to Metro Mirror (PPRC) primary devices to be swapped to secondary devices.

- ▶ Transparently to applications
- ▶ Without manual intervention

The HyperSwap API address space is the same address space currently used by GDPS HyperSwap. So except for the management code that coordinates the HyperSwap across members of the sysplex, the code that performs the actual HyperSwap is identical, allowing both GDPS and Basic HyperSwap to benefit from enhancements to this code.

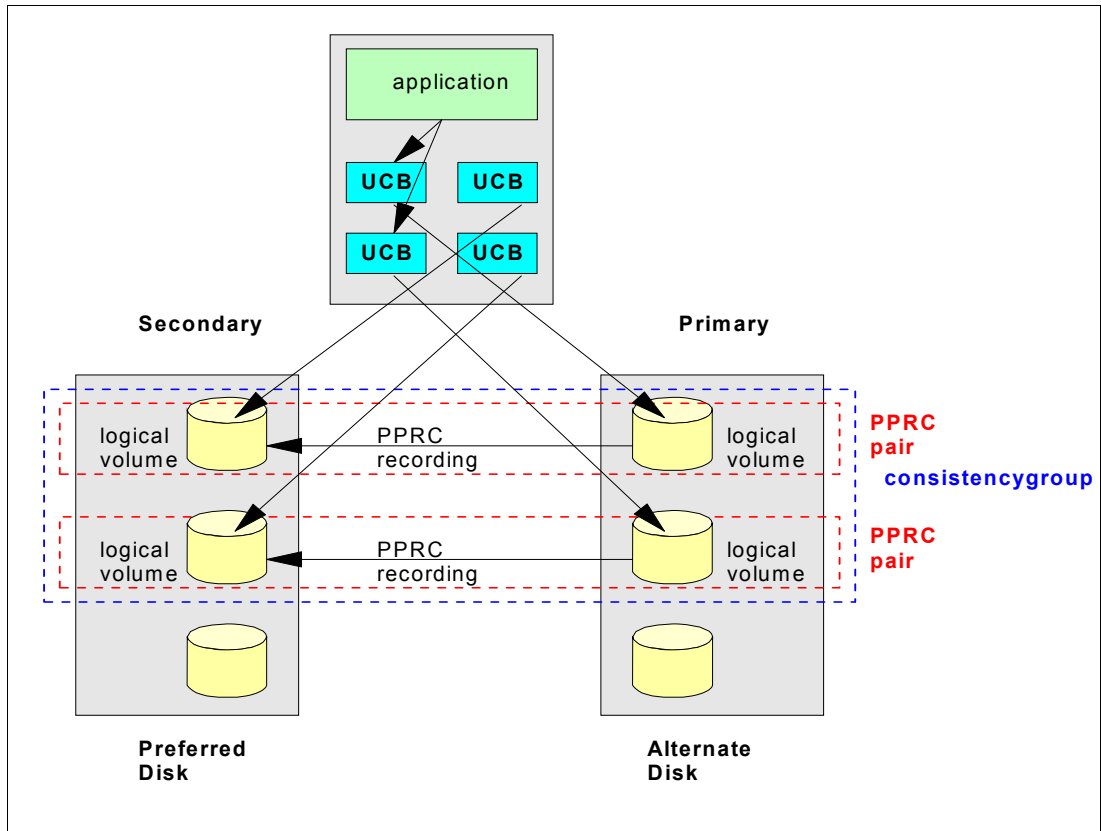


Figure 21-2 HyperSwap after a swap

## 21.2.2 Basic HyperSwap

Basic HyperSwap is a system function, as shown in Figure 21-3 on page 417, that was introduced in z/OS V1R9 with APAR OA20658. In order to exploit it, TPC for Replication v3.4 or higher must also be installed. TPC-R provides the following functions:

- ▶ A graphical user interface for configuration and commands
- ▶ A repository for the configuration
- ▶ Device discovery process
- ▶ A wizard to allow volume matching
- ▶ Creation of Peer-to-Peer Remote Copy (PPRC) paths
- ▶ Setting up of Peer-to-Peer Remote Copy (PPRC) pairs
- ▶ Monitoring of all the environment
- ▶ Setting up of required automation
- ▶ Error checking on the configuration
  - The mandatory actions to freeze the hardware replication to secondary copy
  - Optionally stops all the I/O to the primary site and swaps to secondary copy

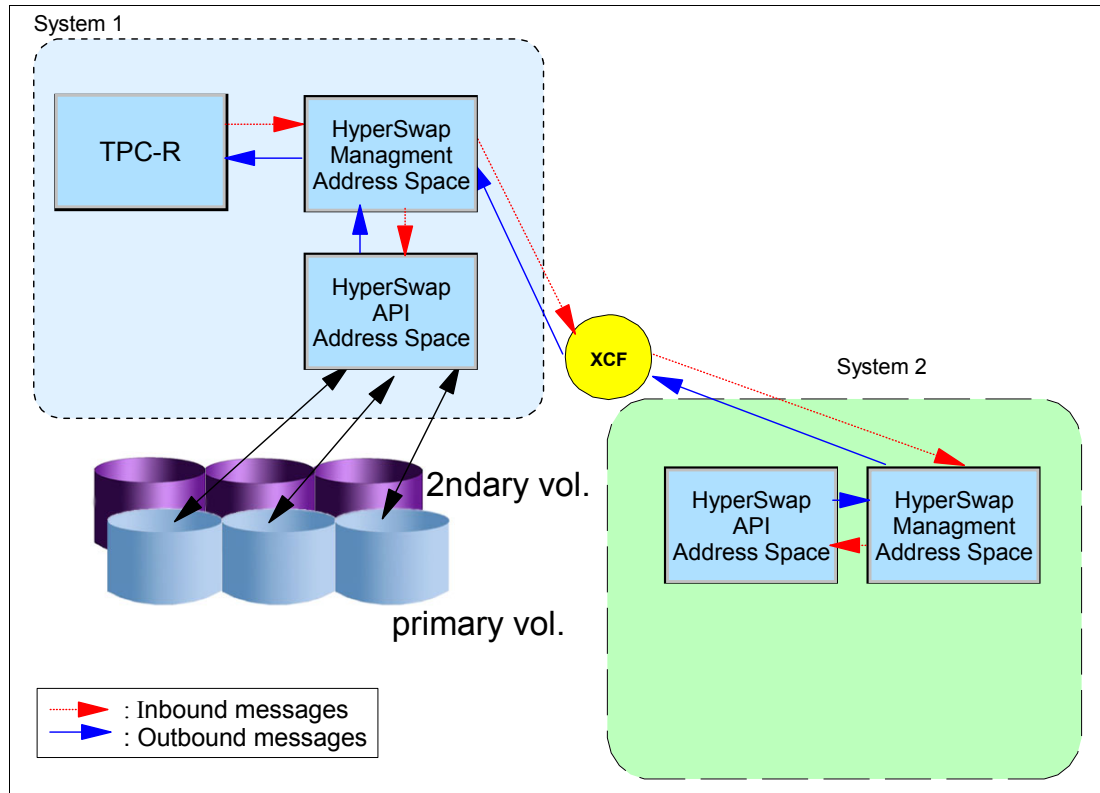


Figure 21-3 Basic HyperSwap and TPC-R

## TPC-R support

TPC-R controls all activities during the definition and management of a HyperSwap session. These activities include defining the HyperSwap session itself, populating the session with the appropriate Metro Mirror volume pairs (in TPC-R terminology these are called copy sets), establishing the PPRC paths, and starting the PPRC pairs to begin mirroring.

TPC-R then passes the configuration definition to Basic HyperSwap, a subcomponent of the I/O Supervisor (IOS). Once the session has reached full duplex state (that is, all target volumes are now identical copies of their corresponding source volumes), Basic HyperSwap becomes enabled, and is able to respond to freeze or HyperSwap triggers without further assistance from TPC-R.

## Basic HyperSwap

Basic HyperSwap provides z/OS native support to maintain a device configuration, monitor the devices for HyperSwap triggers and perform the HyperSwap processing across the sysplex when a trigger is detected.

When Basic HyperSwap receives the device configuration from TPC-R, it maintains a copy in page-fixed storage. Basic HyperSwap also provides operator commands such as DISPLAY HS or SETHS.

The DISPLAY HS command allows you to display the status of the HyperSwap session, and information about the configuration. The SETHS command allows you to initiate a planned HyperSwap, or to temporarily disable and later re-enable HyperSwap processing.

## 21.3 Evolution of TPC-R support of Basic HyperSwap

In May 2007, TPC-R for System z V3.3 was the first release of TPC-R to support z/OS. Since then, there have been several significant enhancements to TPC-R for System z. In April 2008, TPC-R V3.4 provided support enabling Basic HyperSwap. This also introduced IBM Tivoli Storage Productivity Center for Replication Basic Edition for System z, which allows clients that only have a requirement to use TPC-R to manage Basic HyperSwap, to get a license at “no charge”.

TPC-R V4.1 was announced in April 2009. This version provided support for three-site mirroring (Metro Global Mirror) with HyperSwap. In this environment, you can have a HyperSwap session using synchronous mirroring between site 1 and site 2, while cascading to a third site using asynchronous mirroring.

Also new in this version was the introduction of WAS OEM Edition. This version of WebSphere is packaged not only in TPC-R Basic Edition, but also the priced versions of TPC-R for System z. This enhancement eliminates the requirement for a separate license for WebSphere, and allows you to migrate more easily from TPC-R Basic Edition to one of the priced versions, without requiring a reinstall of TPC-R.

Most recently, in October 2009, TPC-R V4.1.1 was announced. This provided a function called “Metro Mirror sessions with HyperSwap capabilities.” This enhances Basic HyperSwap to include two-site business continuity capabilities.

### 21.3.1 Metro Global Mirror with HyperSwap support

Since V3.4, TPC-R has supported Metro Global Mirror (MGM) for the DS8000. MGM provides three-site mirroring, but providing synchronous mirroring between sites 1 and 2, and then cascading to a third site using asynchronous mirroring.

MGM provides failover and failback support, fast re-establishment of three-site mirroring, quick resynchronization of mirrored sites using incremental changes only, and data currency at the remote site.

In v4.1, TPC-R was enhanced to allow the Metro Mirror leg of an MGM session to be HyperSwap-managed, providing the continuous-availability benefits of HyperSwap, while also allowing an asynchronous copy of the data to be maintained at a remote location, for disaster recovery.

### 21.3.2 Metro Mirror with HyperSwap support

This provides the ability to define Metro Mirror (with failover or failback) sessions that have HyperSwap capabilities. Enabling HyperSwap with TPC-R Metro Mirror combines an availability solution with a business continuity solution.

Using the Metro Mirror with HyperSwap sessions means that, if an error occurs on a primary device, z/OS will redirect I/O requests to the secondary devices (that is, HyperSwap), thereby masking the application and the system from the I/O error.

If an error occurs on a secondary device, or on a replication link between primary and secondary devices, all PPRC paths in the HyperSwap session are suspended before I/O is allowed to continue to any of the primary devices affected by the failure, ensuring that the target site remains data consistent. This allows you to recover at the secondary site if the link failure was actually the start of a primary site failure.



### 21.3.3 Metro Mirror suspend policy

When defining “Metro Global Mirror with HyperSwap Support” and “Metro Mirror Sessions with HyperSwap Support” session types, you must choose what policy you wish TPC-R and Basic HyperSwap to follow when a PPRC link suspend is detected.

When PPRC links are suspended, I/O requests to devices on the affected primary LSS are automatically held (frozen) by the storage system. This gives the replication management software time to suspend (or freeze) all of the other PPRC links in order to preserve data consistency at the alternate site.

The issue is that PPRC links may become suspended for different reasons. In one case, the last link between a pair of LSSs may have failed, or the target storage subsystem may have failed, but nothing more. In this case, once all links have been suspended, TPC-R HyperSwap can issue a “run” to all of the affected LSSs, allowing I/O to continue. Once the cause of the link failure has been corrected, the replication session may be restarted.

Another possibility is that this link failure is one of the first failures detected of what will ultimately result in a primary site failure, as might occur during a power failure. In this case, if all links have been suspended, and a “run” issued to all of the affected LSSs, then it is possible that transactions would have completed after the “run,” but before the power failure took the primary site down. In this case, not all of the changes during this period would have been mirrored to the target site.

So you must decide what your policy is for a link failure. If you have a requirement for zero data loss, then TPC-R provides a Metro Mirror Suspend Policy option of “Hold I/O after Suspend”. In this case a run is not issued after the links have been suspended, and the system will wait until either the Release I/O command is issued from the TPC-R GUI, the SETHS RESUMEIO command is issued, or the hardware time-out value on the storage system expires.

If you have a requirement for continuous availability, then TPC-R provides a policy option of “Release I/O after Suspend.” In this case the run is issued after all links have been suspended. However, in the case of a site failure, data may be lost.

These options are only available with the priced versions of TPC-R. TPC-R Basic Edition is an availability-only solution, meaning that it is designed to protect clients from storage failures only. With TPC-R Basic Edition, when Basic HyperSwap detects a suspend trigger, it immediately disables HyperSwap and issues a “run” to the affected LSS(s).

#### **New TPC-R options with HyperSwap support**

With the HyperSwap support, TPC-R provides new options:

- ▶ Hold I/O after Suspend
  - Issue Freeze to all affected LSSs in the configuration.
  - Run is not issued.
- ▶ Release I/O after Suspend
  - Issue Freeze to all affected LSSs in the configuration.
  - Issue Run to all affected LSSs in the configuration.

#### **Basic HyperSwap operation**

To further illustrate this point, Figure 21-4 on page 420 shows the result of processing when a PPRC suspension event is detected in a Basic HyperSwap session.

All primary devices affected by the suspend event (8000-80003) are placed in “extended long busy” state. This prevents further I/O to the primary devices that can no longer copy data to their secondaries by queueing the I/O for the device in the host. Basic HyperSwap detects the suspension due to message IEA4911, or via a signal from TPC-R, and issues a Run command (or what is sometimes called unfreeze) to the affected LSS pair (LSS1:LSS2). This clears the “extended long busy” allowing application I/Os to continue to the affected primary devices.

Due to this behavior, if the link failure was the first failure detected in what ultimately resulted in a primary site failure, then you could not recover using the secondary storage, because data consistency cannot be assured.

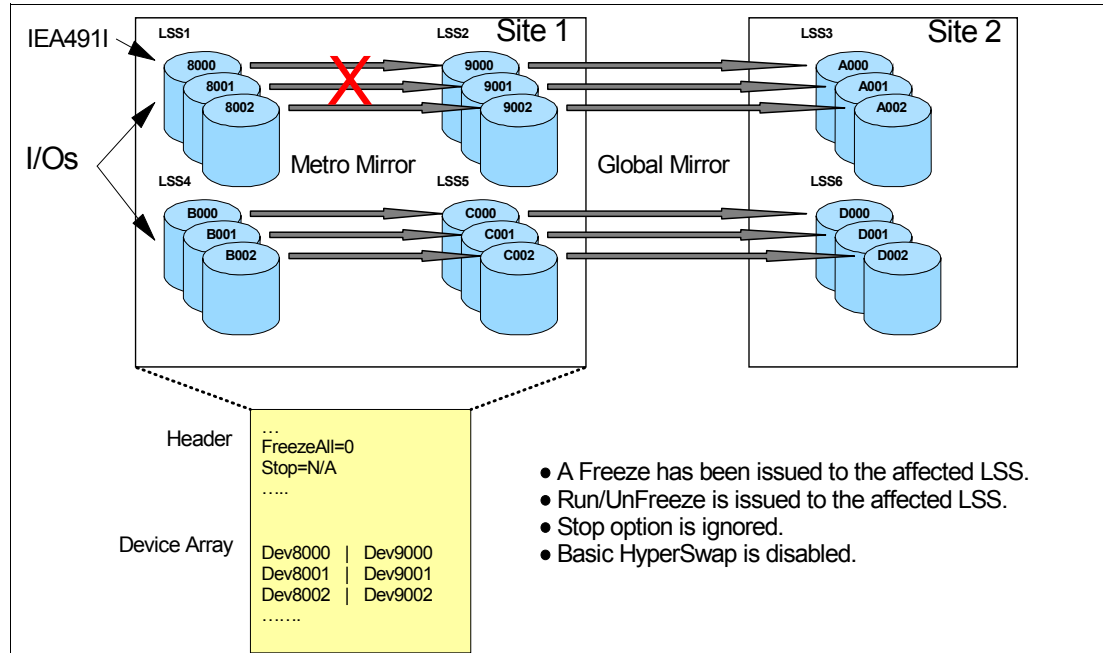


Figure 21-4 PPRC link suspension event with Freeze All=NO

### FreezeAll=Yes and Stop=No

Figure 21-5 on page 421 shows the result of processing when a PPRC suspension event is detected in a Metro Global Mirror with HyperSwap session with the “Release I/O after Suspend” policy in effect. Processing for a Metro Mirror with HyperSwap session is similar.

When a suspend occurs, all primary devices affected by the suspend event (8000-8003) are placed in “extended long busy” state. Basic HyperSwap detects the suspension due to message IEA4911, or via a signal from TPC-R, and issues a FREEZE command to all other LSS pairs (LSS4:LSS5) that are part of the session; see Figure 21-5 on page 421. This places all HyperSwap primary devices in “extended long busy” state, and suspends all further mirroring.

Basic HyperSwap is also disabled for swapping, since mirroring to the secondary devices has stopped. The configuration passed to Basic HyperSwap by TPC-R is also purged at this time.

Once this is complete, the secondary devices are data consistent, and HyperSwap issues the Run command (or what is sometimes called unfreeze) to all affected LSS pairs (LSS1:LSS2 and LSS4:LSS5). This clears the “extended long busy,” allowing application I/Os to continue to the affected primary devices.

If the failure is only a link or secondary storage controller, then there is only a small impact to the applications (the pause while the above processing was being performed), and once the problem is corrected, replication can be restarted.

If the failure was the beginning of what ultimately resulted in a primary site failure, then you can recover from the secondary storage, although some data loss may have occurred.

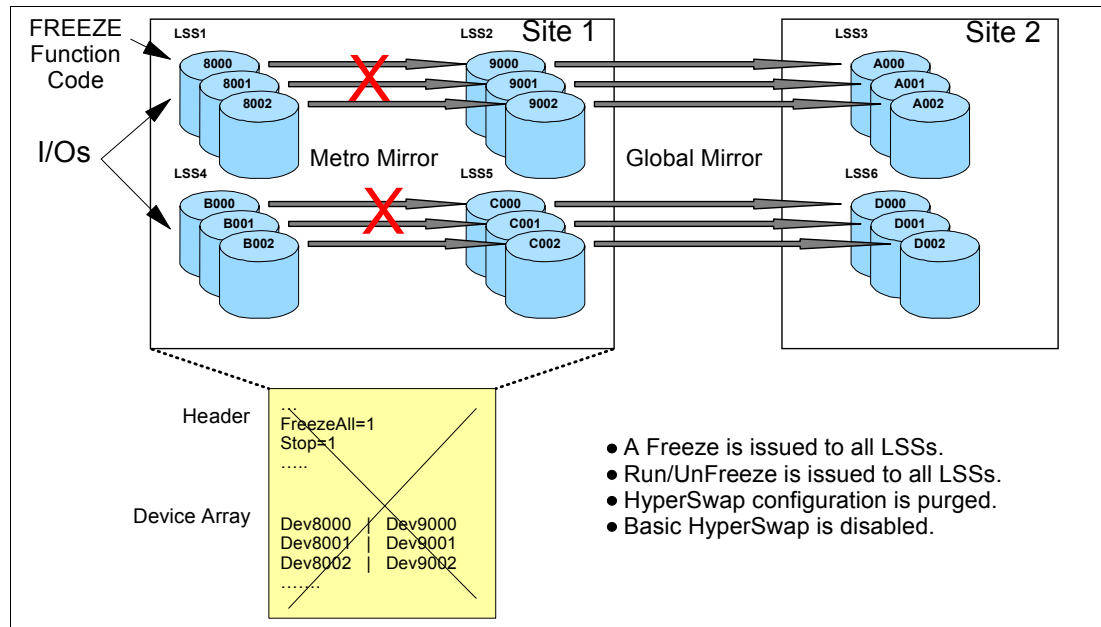


Figure 21-5 Freeze function code with FreezeAll=Yes, Stop=No

### FreezeAll=Yes and Stop=Yes

Figure 21-6 on page 422 shows the result of processing when a PPRC suspension event is detected in Metro Global Mirror with HyperSwap Session with the “Hold I/O after Suspend” policy in effect. Processing for a Metro Mirror with HyperSwap session is similar.

When a suspend occurs, all primary devices affected by the suspend event (8000-8003) are placed in “extended long busy” state. Basic HyperSwap detects the suspension due to message IEA491I, or via a signal from TPC-R, and issues a FREEZE command to all other LSS pairs that are part of the session (LSS4:LSS5). This places all HyperSwap primary devices in “extended long busy” state, and suspends all further mirroring.

Basic HyperSwap is also disabled for swapping, since mirroring to the secondary devices has stopped. The configuration passed to Basic HyperSwap by TPC-R is also purged at this time.

Once this is complete, the secondary devices are data consistent. However HyperSwap does not issue the RUN command to affected LSS pairs, since Stop=Yes was specified.

To resume I/O operations, a manual intervention is required using the MVS system command `SETHS RESUMEIO`, the TPC-R session command `RELEASE I/O`, or wait until the hardware time-out expires on the storage subsystem.

**Note:** In case paging packs have been frozen, it is possible that you will not be able to release the I/O using either the TPC-R session command, or the MVS system command. In this case, you can either issue the TPC-R session command from a standby server that is not impacted by the failure, or wait until the hardware time-out expires.

If the failure was the beginning of what ultimately resulted in a primary site failure, then you can recover from the secondary storage, and no data loss would have occurred. If the failure was only a link or secondary storage controller failure, then there would have been an impact to the applications, since the pause while the operator decided whether or not to resume I/O would likely have taken minutes. Once the problem is corrected, replication can again be restarted.

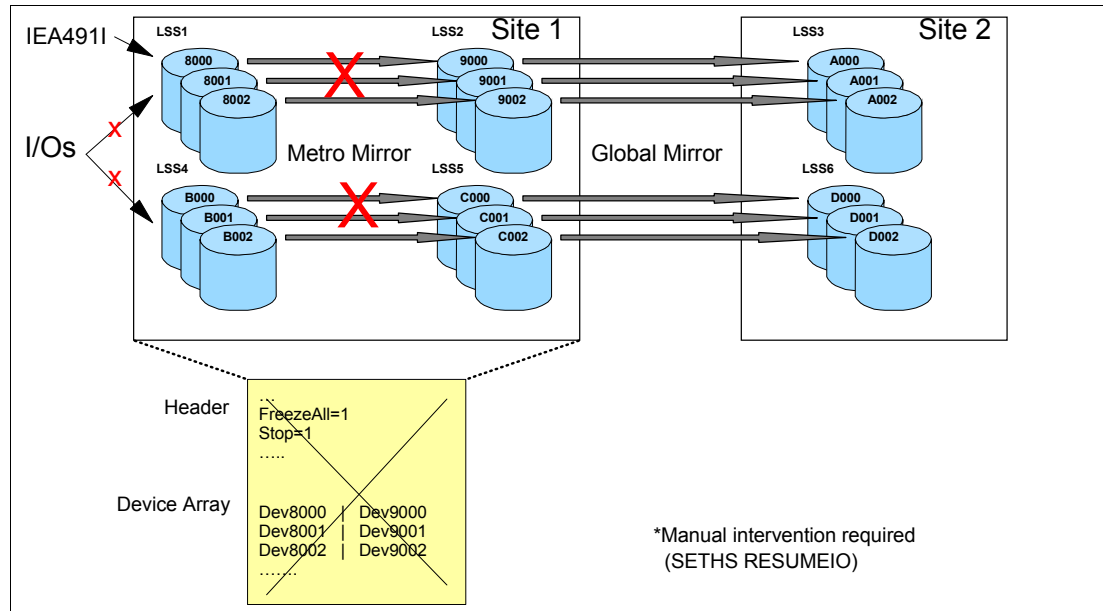


Figure 21-6 Freeze function code with FreezeAll=Yes, Stop=Yes

## 21.4 Basic HyperSwap in z/OS V1R11

Some enhancements to Basic HyperSwap have been added in z/OS V1R11 in the following areas:

- ▶ A programming interface has been provided to allow data migration products to temporarily disable HyperSwap, without stopping mirroring.
- ▶ HyperSwap now enforces the restriction that couple data sets not be on HyperSwap-managed volumes.

## 21.5 Basic HyperSwap resiliency in z/OS V1R12

HyperSwap is a key resiliency item for the System z platform.

It allows for rapid, nondestructive failover from a primary to a secondary set of DASD volumes.

One of the benefits of Basic HyperSwap is that it supports system paging volumes as part of the HyperSwap session. Basic HyperSwap does this by page fixing code in both the HyperSwap management address space and the HyperSwap API address space, along with control blocks that it will need to access during a HyperSwap. By doing this, HyperSwap is able to avoid accessing paging volumes, which may be “frozen” during a HyperSwap.

In z/OS V1R12, in order to further reduce the likelihood of a page fault in other system components interfering with HyperSwap processing, z/OS now supports the concept of critical address spaces. This support has also been rolled back to z/OS V1R10.

When enabled in z/OS V1R12, RSM “hardens” critical address spaces against page faults, meaning that all pages associated with the address space will be treated as if they are fixed—they will not be paged out. This includes all data spaces associated with the critical address spaces as well as common storage in general (PLPA, CSA, etc).

In addition, RSM resolves first-reference page faults in critical address spaces during HyperSwap failovers by treating them the same as a system-critical request; XCF provides the means to enable this new critical paging behavior and PPT provides the means to define additional address spaces deemed critical to HyperSwap.

The benefit of this z/OS V1R12 change is to improve HyperSwap resiliency by reducing the likelihood that another system component will interfere with a HyperSwap completing due to a page fault.

Before enabling CriticalPaging, you should assess your real storage assessment requirements to determine whether additional real storage is required.

Here is the list of the address spaces characterized as critical to HyperSwap:

- ▶ CONSOLE
- ▶ GDPS
- ▶ GRS
- ▶ HSIB
- ▶ HSIBAPI
- ▶ RASP
- ▶ XCF

**Disclaimer:**

- ▶ This enhancement just reduces the chance of a deadlock (it does not eliminate it).
- ▶ Protecting address spaces deemed critical to HyperSwap from page faults should not be done at the expense of the health of the system.

## 21.6 Basic HyperSwap and data migration coexistence support

Clients often use DASD migration products to help them migrate from old to new DASD hardware, while still allowing their applications to run and thereby avoiding an outage.

While suspended for Data Migration reasons, if a swap trigger is detected, a HyperSwap will not occur. However, instead of this window being minutes or even hours, the window in most configurations should be under a second.

### 21.6.1 HyperSwap and data migration products

To allow better coexistence between HyperSwap and the data migration products, the following functions are created:

- ▶ A programmatic interface to temporarily disable the Basic HyperSwap activities
  - Allows Data Migration to temporarily suspend Basic HyperSwap activities.

It coordinates the tasks between Data Migration and Basic HyperSwap. Data Migration performs its swap only while Basic HyperSwap is inactive. Basic HyperSwap performs swaps only if Data Migration is prevented from swapping.

- Prevents Data Migration and Basic HyperSwap from swapping the same set of devices at the same time.
- Minimize data protection exposure window by not requiring HyperSwap configuration to be purged and reloaded.
- ▶ A programmatic interface to re-enable the Basic HyperSwap activities
  - Automatic maintenance of device configuration.
  - z/OS Basic HyperSwap recognizes data migration activities and adjusts device configuration information accordingly.

## 21.6.2 HyperSwap programming enhancement

Basic HyperSwap allows only one block to be in effect in the sysplex. In the rare case that there are multiple data migration products running in the installation, subsequent requests will be rejected with a reason code and return code.

If a HyperSwap master takeover occurs, the new Basic HyperSwap master maintains current block status for the sysplex. When a new Basic HyperSwap member joins the sysplex and it has the correct level of support, the current block status is passed to the new member. This is so if a master takeover occurs and the new member happens to become the new master. It will be able to maintain the block status going forward.

### After the BLOCK request is processed

The HyperSwap configuration cannot be replaced or purged. HyperSwap disablement triggers will not be detected:

- ▶ Loss connectivity to Metro Mirror Secondary devices
- ▶ Suspended Metro Mirror connection between monitored devices

HyperSwap swap triggers will not be processed:

- ▶ Permanent I/O errors on monitored devices
- ▶ Operator SETHS SWAP command

Processing requests will be queued until an UNBLOCK request is processed.

### After the UNBLOCK request is processed

If no other Basic HyperSwap requests were queued during a blocked period, Basic HyperSwap resumes monitoring devices in the original configuration. If other Basic HyperSwap requests were queued during the blocked period, the following actions can occur:

- ▶ Queued requests will be processed.
- ▶ They may not be in the same order as when they are received by Basic HyperSwap.
- ▶ A queued swap request will be processed ahead of others.

### Operational considerations

All members of the sysplex must have the same supporting PTFs installed, or a BLOCK/UNBLOCK request will be rejected with a return code and reason code.

If clients are using data migration products to migrate from, for example, device 0400 to device 0A00, then IBM recommends that you have both devices in your HyperSwap session, so that, for example, device 0400 is being mirrored to 2400, and device 0A00 is being mirrored to 2A00. By doing this, if a swap trigger is detected on one of the volumes being used by the data migration product, and that product has not yet reached the point where it is going to initiate the swap, a HyperSwap can be performed without impact to the migration.

Once the migration is complete, the “old” device pairs may be removed from the HyperSwap session and the devices retired.

### **Command considerations**

Using the SETHS UNBLOCK command provides a way for an installation to reset outstanding BLOCK requests when the application holding the BLOCK request terminates or is unable to issue an UNBLOCK successfully.

Using the command requires RACF UPDATE or master console authority, or message IOSHM0315I is issued.

The DISPLAY HS,STATUS command produces message IOSHM0303I; it displays whether or not a BLOCK is currently in effect and if so, the EBCDIC name of the product currently holding the block is specified.

### **New messages**

The following new messages are part of this new support:

- ▶ IOSHM0310I  
Result of an UNBLOCK request
- ▶ IOSHM0311I  
Result of a BLOCK request
- ▶ IOSHM0313I  
Non-supporting system is detected in sysplex when a BLOCK request is received.
- ▶ IOSHM0314I  
New non-supporting system is prevented from becoming a member of the Basic HyperSwap sysplex group.
- ▶ IOSHM0315I  
Result of the SETHS UNBLOCK command.

## **21.7 Couple data set avoidance**

In general, Basic HyperSwap relies on XCF messaging to communicate between members of the sysplex. However, it also uses XCF services to maintain state information about the other HyperSwap members in the CDS.

Since I/Os to all devices are inhibited during a HyperSwap, if couple data sets are included in the HyperSwap configuration, I/Os to them will get hung during a swap, causing the HyperSwap to fail. Furthermore, XCF provides duplication for the data on the couple data sets, so there is no requirement to include them in the HyperSwap configuration.

## In z/OS V1 R11

In order to enforce this restriction, in z/OS Release 11:

- ▶ Basic HyperSwap rejects a Load configuration request when a CDS is included as a primary device in the HyperSwap configuration.
- ▶ Basic HyperSwap is disabled when a CDS is activated on one of the current primary devices in the HyperSwap configuration.
- ▶ Basic HyperSwap is re-enabled after the CDS is deactivated or removed from the device.

## 21.8 Enhanced device discovery

TPC-R v4.1.1 has been enhanced so that it can now “discover” IBM storage systems through a z/OS connection, in addition to the traditional TCP/IP connection.

Discovery of devices is limited to devices that are defined to and have paths to the z/OS system that is running TPC-R.

This support was released as service on top of z/OS V1R11, and rolled back to z/OS V1R9. Following is a list of the required APARs:

- ▶ IOS OA28415
- ▶ AOM OA29175
- ▶ SDM OA23766
- ▶ TPC-R PK96652

New and enhanced TPC-R CLI commands are available to manage z/OS connections:

<b>addstorsys</b>	Adds a specific storage system and its volumes through a z/OS connection.
<b>chlocation</b>	Changes the location associated with specific storage systems.
<b>lsdevice</b>	This command (which lists attributes of storage subsystems) has been enhanced to include z/OS as a connection type (in addition to direct connect, and hardware management console (HMC)).
<b>lsstorcandidate</b>	Lists storage systems that can be discovered through the z/OS connection.
<b>rmstorsys</b>	Removes a specific storage system and its volumes, which are attached through a z/OS connection.
<b>showdevice</b>	This command (which lists attributes of a single storage subsystem) has been enhanced to include z/OS as a connection type (in addition to direct connect, and hardware management console (HMC)).

## 21.9 HyperSwap resiliency in z/OS V1R12

For the new function enhancing HyperSwap resiliency in z/OS V1R12, one must, first of all, make a storage assessment and then actually activate the function. This is described in the following paragraphs.



## 21.9.1 Capacity planning

In order to enhance HyperSwap with the enhanced resiliency provided in z/OS V1R12, and for this new critical paging behavior to become active, installations must tell XCF to enable it as follows:

- ▶ The CRITICALPAGING parameter must be set in the FUNCTIONS statement of the COUPLExx parmlib member.
- ▶ Before enabling this new z/OS V1R12 function, you may need to configure additional memory to your system to ensure that keeping critical address spaces in memory does not cause real storage constraints and performance problems.

Multiple RMF STORF reports should be examined in order to determine the peak auxiliary slot count for each critical address space. Figure 21-7 is an example of an RMF STORF report.

RMF V1R12 Storage Frames												
Line 1 of 103												
Scroll ==> CSR												
Command ==>												
Samples: 9 System: SYSF Date: 11/28/09 Time: 08.44.00 Range: 30 Sec												
Jobname	Service	C	Class	Cr	Frame Occup.	Active Frames	AUX	PGIN				
					TOTAL	ACTV	IDLE	WSET	FIXED			
								DIV	SLOTS			
									RATE			
INIT	S		SYSSTC		197K	0	197K	0	197K	0	53	0
I0SAS	S		SYSTEM		133K	133K	0	133K	755	0	49	0
THR64GMD	B		JESL0W5		115K	115K	0	115K	115K	0	43	0
STGTHR32	B		JESL0W5		77390	77390	0	77390	367	0	62388	0
STGTHR33	B		JESL0W5		77390	77390	0	77390	367	0	59546	0
STGTHR31	B		JESL0W5		77390	77390	0	77390	367	0	60125	0
STGTHR34	B		JESL0W5		77389	77389	0	77389	367	0	64878	0
INIT	S		SYSSTC		66778	0	66778	0	66627	0	24	0
THR64FMC	B		JESL0W5		65927	65927	0	65927	65720	0	60	0
THR64AMB	B		JESL0W5		65925	65925	0	65925	65720	0	62	0
THR64EMC	B		JESL0W5		65925	65925	0	65925	65720	0	62	0
THR64DMC	B		JESL0W5		65925	65925	0	65925	65720	0	62	0
THR64BMC	B		JESL0W5		65847	65847	0	65847	65612	0	18	0
THR64CMC	B		JESL0W5		65846	65846	0	65846	65612	0	18	0

Figure 21-7 RMF STORF report

### RMF STORF report analysis

After reviewing a number of RMF STORF reports, collect the peak auxiliary slot count for each critical address space. Ideally, the best RMF STORF reports to sample are for time periods when the system was busy.

The following illustrates how you can obtain the numbers needed to plug into the following formula (used to calculate the additional memory needed):

$$\text{Additional memory needed} = ((\text{The sum of the peak auxiliary slot count for each critical address space}) + (\text{The size of the PLPA and CSA area})) * 4096$$

### RMF Virtual Storage Activity report

Examine the RMF VSTOR report (Common Storage Summary section) to determine the size of your PLPA and CSA area (as shown in Figure 21-8 on page 428 in a truncated sample report).

The common storage summary section enables you to measure the use of virtual storage with minimal overhead. It contains the information you need to understand your current use of virtual storage. If you archive the data, you can use differences over time to predict a problem or constraint before it becomes critical. It also helps you to verify the size values set for CSA and SQA at IPL time and determine if you are using common storage effectively. Because

RMF does not sample virtual storage data at every cycle. The value reported for NUMBER OF SAMPLES is less than the number of cycles.

Where:

- Static storage map**                    The major storage areas above and below the 16-megabyte line. It includes the name of each area, the address of its lower boundary, and its size, reported in bytes.
- Allocated CSA/SQA**                    The MIN, MAX, and AVG values for allocated CSA and SQA, both below and above the 16-megabyte line. RMF calculates each size by adding the number of bytes assigned to each SQA or CSA subpool. The report also breaks down allocated CSA by key.
- SQA expansion into CSA**                The MIN, MAX, and AVG size of any expansion of SQA into CSA.
- PLPA intermodule space**                The amount of unused space between the modules in both the PLPA and the EPLPA (the expanded PLPA). If your installation uses a pack list (in the IEAPAK00 Parmlib member), the values reported can help you determine the cost of your packing algorithm in relation to its benefit, a reduction in LPA paging rates, as shown in the paging report.
- PLPA space with MLPA/FLPA**            The amount of space for PLPA occupied by modules that also exist in (E)MLPA and/or (E)FLPA. For EPLPA, reports the amount of space occupied by modules that also exist in (E)MLPA or (E)FLPA.
- Free pages (bytes)**                      The MIN, MAX, and AVG values, in bytes, for the amount of free storage.

VIRTUAL STORAGE ACTIVITY									
z/OS V1R12		SYSTEM ID TRK1		DATE 11/20/2000		INTERVAL 00.50.000		PAGE 1	
		RPT VERSION V1R12 RMF		TIME 10.13.00		CYCLE 1.000 SECONDS			
COMMON STORAGE SUMMARY									
NUMBER OF SAMPLES 6									
STATIC STORAGE MAP		ALLOCATED CSA/SQA							
AREA	ADDRESS	SIZE	----- BELOW 16M -----		----- EXTENDED (ABOVE 16M) -----				
EPVT	21200000	1510M	MIN	MAX	AVG	MIN	MAX	AVG	AVG
ECSA	E51E000	301M	SQA 280K 10.12.50	280K 10.12.50	280K	35.6M 10.12.50	35.6M 10.12.50	35.6M	35.6M
EMLPA	0	0K	CSA 304K 10.12.50	304K 10.12.50	304K	37.5M 10.13.50	37.6M 10.13.30	37.5M	
EPLPA	E51B000	12K							
EPLPA	A4FC000	64.1M	ALLOCATED CSA BY KEY						
ESQA	10E5000	130M	0	148K 10.12.50	148K 10.12.50	148K	10.2M 10.12.50	10.2M 10.12.50	10.2M
ENUC	1000000	0.00M	1	112K 10.12.50	112K 10.12.50	112K	952K 10.12.50	952K 10.12.50	952K
----- 16 MEG BOUNDARY -----			2	36K 10.12.50	36K 10.12.50	36K	4K 10.12.50	4K 10.12.50	4K
NUCLEUS	FD4000	176K	3	0K 10.12.50	0K	0K	0K 10.12.50	0K	0K
SQA	E53000	1540K	4	0K 10.12.50	0K	0K	4K 10.12.50	4K 10.12.50	4K
PLPA	C04000	1700K	5	4K 10.12.50	4K 10.12.50	4K	5012K 10.12.50	5012K 10.12.50	5012K
FLPA	C03000	4K	6	84K 10.12.50	84K 10.12.50	84K	13.4M 10.13.50	13.5M 10.13.30	13.4M
MLPA	0	0K	7	0K 10.12.50	0K	0K	52K 10.12.50	52K 10.12.50	52K
CSA	000000	3660K	8-F	0K 10.12.50	0K	0K	0K 10.12.50	0K	0K
PRIVATE	2000	0200K	SQA EXPANSION INTO CSA						
PSA	0	0K		0K 10.12.50	0K	0K	0K 10.12.50	0K	0K
PLPA INTERMODULE SPACE - 5K IN PLPA AND 205K IN EPLPA									
PLPA SPACE REDUNDANT WITH MLPA/FLPA - 0K IN PLPA AND 11K IN EPLPA									
----- BELOW 16M -----									
			MIN	MAX	AVG	----- ABOVE 16M -----			
CSA			MIN	MAX	AVG	MIN	MAX	AVG	
FREE PAGES (BYTES)			3276K 10.12.50	3276K 10.12.50	3276K	263M 10.13.30	263M 10.13.50	263M	
LARGEST FREE BLOCK			3276K 10.12.50	3276K 10.12.50	3276K	263M 10.13.30	263M 10.13.50	263M	
ALLOCATED AREA SIZE			304K 10.12.50	304K 10.12.50	304K	37.5M 10.13.50	37.6M 10.13.30	37.5M	
SQA									
FREE PAGES (BYTES)			1252K 10.12.50	1252K 10.12.50	1252K	104M 10.12.50	104M 10.12.50	104M	
LARGEST FREE BLOCK			806K 10.12.50	806K 10.12.50	806K	103M 10.12.50	103M 10.12.50	103M	
ALLOCATED AREA SIZE			644K 10.12.50	644K 10.12.50	644K	130M 10.12.50	130M 10.12.50	130M	
MAXIMUM POSSIBLE USER REGION -			0020K BELOW AND 1500M ABOVE						

Figure 21-8 RMF Common Storage Summary report

## 21.9.2 Usage considerations

In order for this new critical paging behavior to become active, installations must tell XCF to enable it via:

The CRITICALPAGING parameter in the FUNCTIONS statement of the COUPLExx parmlib member

- ▶ CRITICALPAGING can only be enabled at IPL via COUPLExx.
- ▶ The function cannot be enabled after IPL.
- ▶ The function can be turned off dynamically after IPL.

To disable this new critical paging behavior, specify the CRITICALPAGING parameter in the SETXCF FUNCTIONS system command:

```
SETXCF FUNCTIONS,DISABLE=CRITICALPAGING
```

Though the function is RSM-related, the XCF command environment has been picked up for two reasons:

- ▶ RSM has no command interface.
- ▶ XCF has a sysplex scope, which is the very scope of HyperSwap for which this new z/OS V1R12 function has been developed.

## 21.9.3 Adding critical address spaces

Additional critical address spaces can be defined by specifying:

- ▶ The CRITICALPAGING parameter in the PPT statement of the SCHEDxx parmlib member.
- ▶ They can be undefined by *not* specifying CRITICALPAGING or specifying NOCRITICALPAGING (which is the default).

Any of the address spaces deemed critical by IBM (CONSOLE, GDPS, GRS, HSIB, HSIBAPI, RASP, and XCF) cannot be undefined using the SCHEDxx parmlib member.

- ▶ Using SET SCH=xx to modify critical address spaces is effective immediately.  
If the address space is already running when you issue the SET SCH, that address space is marked critical immediately.

**Note:** Harden is only possible for a currently paged-in page. RSM will *not* page in data that is currently paged out.

## 21.9.4 New messages and reason codes

Following are the new messages and the reason codes associated with them:

- ▶ IOSHM0201I - Displays new reason code x'64' when a Load configuration is rejected for CDS inclusion.
- ▶ IOSHM0303I - Displays new reason text for Basic HyperSwap disablement.

## Message IEF731I

With the z/OS V1R12 resiliency enhancement, system message IEF731I (Figure 21-9) has changed with the following information:

- ▶ kw: 22 CRITICALPAGING; 23 NOCRITICALPAGING
- ▶ rc: 48 The keyword cannot be specified with this program name.  
CRITICALPAGING/NOCRITICALPAGING cannot be specified with the IEFIIC program or any program IBM has deemed critical to HyperSwap.
- ▶ rc: 52 The CRITICALPAGING keyword specified for a program whose associated storage already has pages in auxiliary storage.  
PPT entry will be honored but address space is at risk. Restart the address space or re-IPL as soon as possible.

```
IEF731I SCHEDxx LINE num: PPT STMT [FOR PGMNAME name] [IGNORED. | ACCEPTED.]  
REASON=kwrc.
```

Figure 21-9 System message IEF731I

## 21.10 Software dependencies

Couple data set avoidance is available with z/OS V1R11 only. HyperSwap resiliency enhancement is available with z/OS V1R12 only. TPC-R 4.1 is required for Global Mirror support.

**Note:** Even though TPC-R is a priced product from Tivoli, z/OS clients can get a version of TPC-R that only allows a client to define Basic HyperSwap sessions. This is a “no charge” product called “IBM Tivoli Storage Productivity Center for Replication Basic Edition for System z”.

The following products are being enhanced by Basic HyperSwap APAR OA26509 (with corresponding PTFs for V1R9-V1R11) for data migration coexistence:

- ▶ TDMF® 5.1
- ▶ Innovation’s FDRPAS

Basic HyperSwap along with either of the following required GDPS and HSIB are both dependent and exploiters of the new z/OS V1R12 resiliency enhancement.

## 21.11 Migration and coexistence

Required APARs for z/OS V1R9-V1R10:

- ▶ HyperSwap Global Mirror support: OA25299
- ▶ Data Migration Coexistence: OA26509

Diagnostic abend 2E0 is issued if one of the systems in the sysplex does not have the proper code level for Global Mirror support.

A Block/Unblock request will be rejected if one of the systems in the sysplex does not have the proper code level for Data Migration Coexistence support.

## 21.11.1 z/OS V1R12 resiliency enhancement

The resiliency enhancement provided in z/OS V1R12 has some requirements in terms of migration and coexistence, which are described here.

### Migration

Paging behavior should remain the same until you enable this function.

When this function is enabled, paging behavior is changed as follows:

- ▶ Address spaces deemed critical to HyperSwap will always have their storage in real (no pages going to auxiliary storage).
- ▶ As a result, pages from non-critical address spaces are more likely to be paged out to auxiliary storage.
- ▶ Real storage assessment is needed to eliminate this change in paging behavior and (more importantly) to ensure that address spaces deemed critical to HyperSwap do not have any pages go to auxiliary storage.
- ▶ See “Capacity planning” on page 427 for performing the necessary real storage assessment.

### Coexistence prerequisites

For z/OS V1R10 and z/OS V1R11 releases, install the PTFs for the following APARs:

- ▶ OA31691 (XCF Support for HyperSwap Resiliency)
- ▶ OA31707 (RSM Support for HyperSwap Resiliency)
- ▶ OA31331 (IOS Support for HyperSwap Resiliency)

PTF availability for all APARs is before the end of the 3rd quarter of 2010.





## BCPii

IBM provides support in 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) as well as 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) and 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 a System z hardware communication from any other network traffic within the intranet and Internet.

This chapter describes the enhancements provided in z/OS V1R12 for BCPii.

- ▶ CTRACE enhancements
- ▶ Improved storage utilization and serviceability of BCPii transport code
- ▶ Additional CPC/Image attributes and commands.

It also provides (in conjunction with code provided in Appendix D, “BCPii Metal C-example” on page 823) an example in C in order to program the BCP internal interface. Such C-programs can hence be wrapped up to become new MVS commands available from any automation

script. This opens up the possibility to integrate HMC (and zManager for z196 systems) actions into existing automation tools, and not keep them limited to manual directives from the HMC. A paragraph details this potential and how to implement it in your shop.



## 22.1 BCPii overview

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.

As illustrated in Figure 22-1, 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. The address space provides a new API that allows authorized programs to communicate with the HMC.

It is totally unlike existing interfaces such as:

- ▶ Performing a topology function used by HiperDispatch to communicate with PR/SM.
- ▶ Diagnose (HVC or x'83') to communicate with the hardware or the Hypervisor.
- ▶ HWMxxxx service calls to communicate with the STP network.

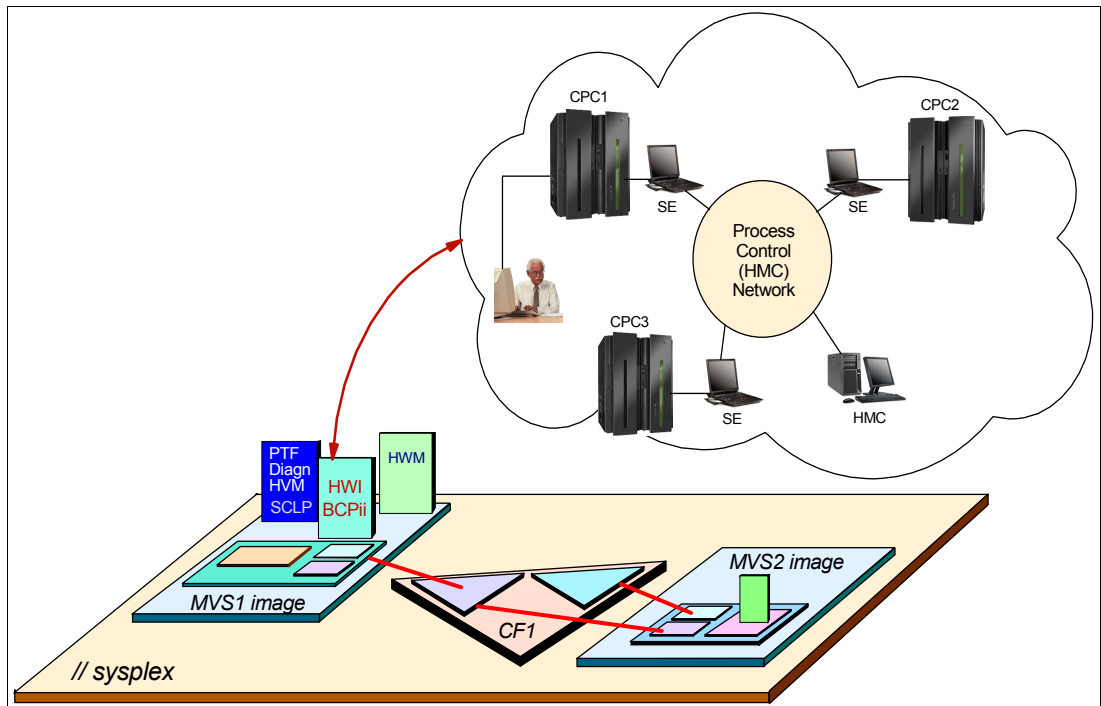


Figure 22-1 BCP internal interface

### 22.1.1 Using BCPii

As mentioned, BCPii provides an interface to authorized z/OS applications through the process control (HMC) network to implement the following:

- ▶ Monitor status or capacity changes.
- ▶ Obtain configuration data related to CPC (not necessarily zSeries system) or image.
- ▶ Re-IPL an image.
- ▶ Allow authorized z/OS applications to have HMC-like control over systems in the process control (HMC) network shown in Figure 22-2 on page 436.

- ▶ Complete communication isolation of existing networks (intranet/Internet) from the process control (HMC) network.  
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.
- ▶ A communications transport between a z/OS application and the local SE.  
It also provides a 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.  
A z/OS programming 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.

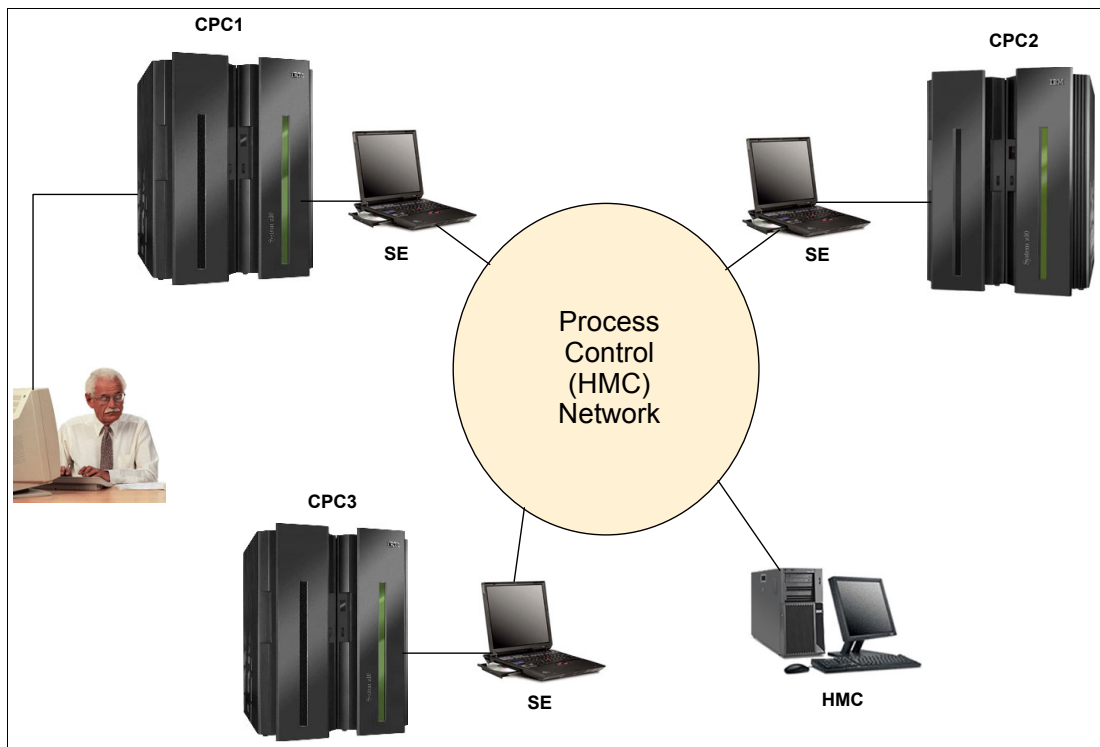


Figure 22-2 HMC network

### 22.1.2 BCPii address space

The BCPii address space, shown in Figure 22-3 on page 437, starts automatically at IPL. It requires initial setup before becoming active. It can be stopped if necessary and restarted by using the 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.

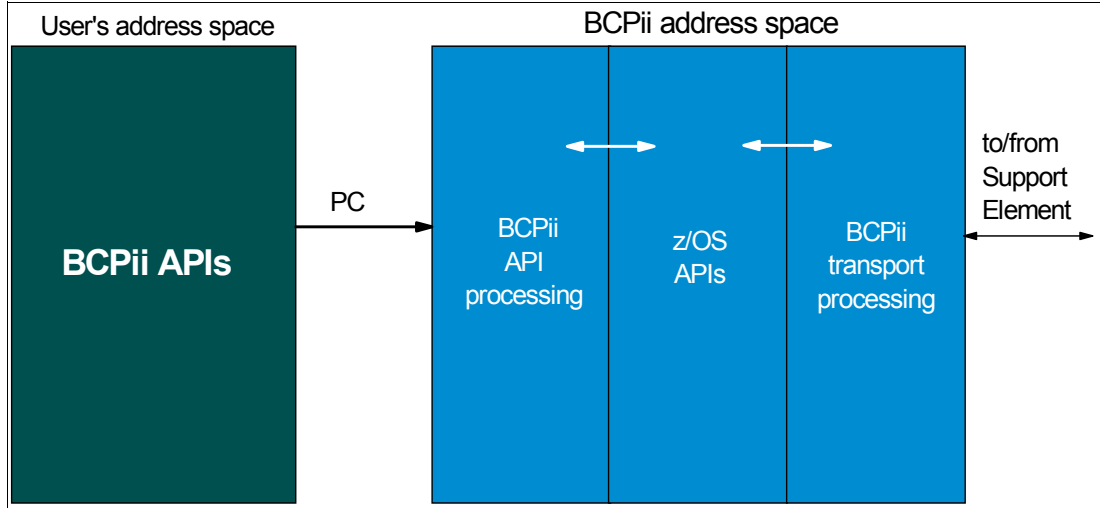


Figure 22-3 BCPii address space

### Users of BCPii

BCPii is used by the following:

- ▶ Vendor applications
  - Control center, systems management applications
- ▶ In-house applications for installations to manipulate System z resources, as follows:
  - Using a more automated, programmatic approach
  - From a z/OS environment rather than a hands-on HMC environment
  - Avoiding the need to have an intranet, Internet, or internal network connect to the process control (HMC) network
- ▶ z/OS operating system components such as XCF

### 22.1.3 BCPii services

BCPii services are available in any address space, from a program that 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:

- ▶ Obtain the System z topology of the current interconnected CPCs, images (LPARs) and capacity records
- ▶ Query various CPC, image (LPAR) and capacity record information
- ▶ Issue commands against both the CPC and image to perform hardware and software-related functions

- ▶ Listen for various hardware and software events that can take place on various CPCs and images.

## BCPii service types

BCPii offers various types of services:

**HWICMD** Issue a BCPii HMC console command against an HMC-managed object that is associated with CPCs, CPC images (LPARs), and capacity records.

There can be CPC commands such as:

- ▶ Activate, deactivate an entire CPC
- ▶ CBU request (activate or undo)
- ▶ On/off capacity on demand request (activate or undo)

There can be image commands such as:

- ▶ Sysreset, load, start, stop, add, or remove temporary capacity, issue operating system commands

**HWICONN** Establish a BCPii logical connection between the application and a CPC, a CPC image (LPAR), a capacity record, or other types of activation profiles (available in V1R11 via APAR OA29638). 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, 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.
- ▶ 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 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 via APAR OA29638) to list the currently defined activation profiles contained on a individual CPC.

**HWIQUERY** BCPii retrieval of SE/HMC-managed objects managed by the Support Element (SE) or hardware management console (HMC) related with CPCs, CPC images (LPARs), capacity records, or other types of activation profiles (available in V1R11 via APAR OA29638).

**HWISET** BCPii change or set SE/HMC-managed objects data for HMC-managed objects associated with CPCs, CPC images (LPARs), or activation profiles (available in V1R11 via APAR OA29638), and for a C application running in the z/OS UNIX System Services environment—according to the principle that anything that can be queried, can be set.

## 22.1.4 BCPii services

You can use base control program internal interface (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 to allow that application to potentially modify these resources.

To use base control program internal interface (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 as follows:

**HWIBeginEventDelivery** Begin delivery of BCPii event notifications to allow a C application running in the UNIX System Services environment to begin delivery of event notifications. This service must be issued before the HWIManageEvents service.

<b>HWIEndEventDelivery</b>	End delivery of BCPii event notifications to allow a C application running in the UNIX System Services environment to end delivery of event notifications. This service unregisters the registration made by the HWIBeginEventDelivery service.
<b>HWIManageEvents</b>	Manage the list of BCPii events to allow a C application running in the 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.
<b>HWIGetEvent</b>	Retrieve outstanding BCPii event notifications to allow a C application running in the UNIX System Services environment to retrieve outstanding BCPii event notifications.

### 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.
- ▶ UNIX System Services callers can receive event notifications through UNIX System Services-only services using the Common Event Adapter (CEA).

**Important:** To give proper authority to a program in an address space to request BCPii services opens up the possibility for this program, for instance, to shut off the entire CEC, even though the given program could only be authorized within one minor test LPAR of the given CEC.

As with z/OS V1R12 and current LIC levels, there is no known record after the potential incident, to figure out which program shut off the CEC.

A tentative solution is proposed in “Potential advantages of a new command set” on page 460.

## 22.1.5 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 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) as well as 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) or 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 and 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 in 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\_CRYPTACTIVITY\_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)

## 22.1.6 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. The mechanism can be used to reduce the energy consumption of the system in periods of low utilization and to partially power off systems designed mainly for disaster recovery—CBU-systems. The mechanism is under full customer control. There is no autonomous function to perform

changes under the covers. The client controls are implemented in the HMC, in the SE, and in the Active Energy Manager with one power saving mode. The expectation is that the frequency reduction is 17%, the voltage reduction 9%, and the total power savings are from 10 to 20% depending on the configuration. Figure 22-4 shows the Set Power Saving panel.

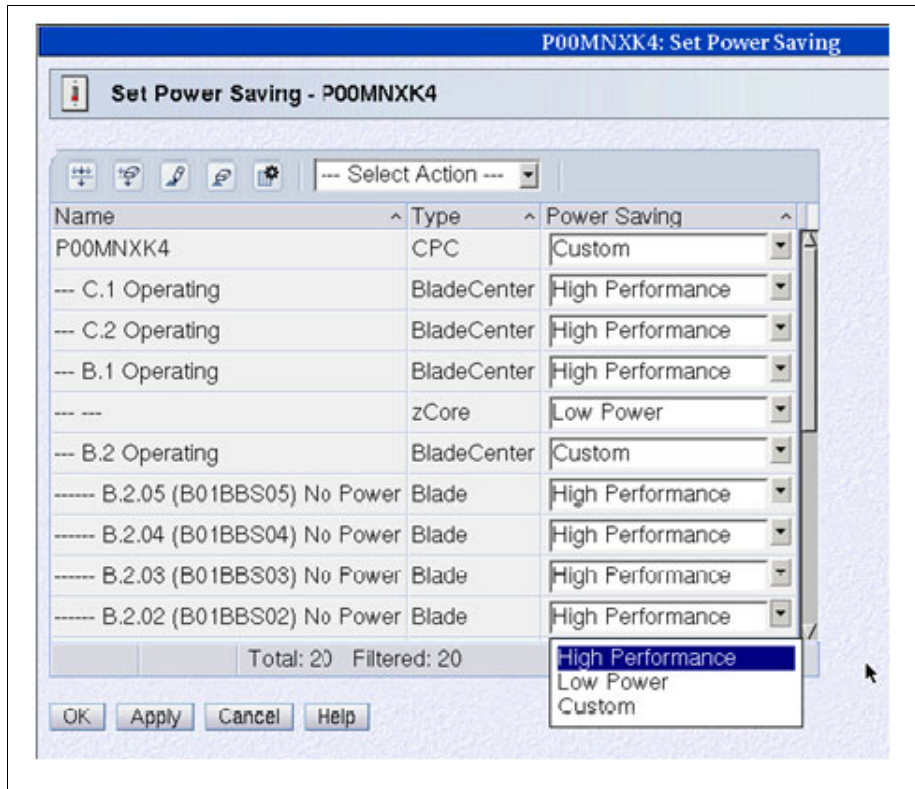


Figure 22-4 Power saving panel

When the server provides a new static power saving mode, it is:

- ▶ Another step in the green initiative for System z
- ▶ Allowing you to reduce performance for significant energy savings
- ▶ Allowing you to reduce power during periods of lower utilization
- ▶ Anticipating CPM users as first exploiters

### BCPii support

BCPii in z/OS V1R12 exploits this new functionality by providing support in its APIs to allow users to:

- ▶ Query the power mode currently in effect (HWIQUERY), new attributes to query being:
  - HWI\_CURRPOWERMODE
  - HWI\_SUPPPOWERMODE
- ▶ Query the power modes available (HWIQUERY)
- ▶ Issue a command to retrieve the Current Processor Power Savings Mode active on the targeted CPC (HWICMS):
  - HWI\_CMD\_POWER\_CONTROL
- ▶ Register to receive notification of changes made to the power saving mode (HWIEVENT). New event to register is:



- HWI\_Event\_Power\_Change bit needs to be turned on for the HWIEVENT Add command.
- Data structure returned is in the ENF exit if listening for HWI\_Event\_Power\_Change.

## 22.2 BCPii installation

BCPii installation steps consist of the following:

- ▶ Configuring the local SE 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)

### 22.2.1 Configure 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 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 by your installation.

**Note:** Configuring the local SE should be unique for both z/OS BCPii and TSA BCPii. It should be defined only once.

You must enable cross-partition authority on the Support Element to allow the SE 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. See Figure 22-5 on page 444, where it is possible to check the partition A01 that is circled. At a minimum, the image (LPAR) where the BCPii address space is running must have this authority activated.

Logical Partition	Active	Performance Data Control	Input/Output Configuration Control	Cross Partition Authority	Partition Isolation	Basic Counter	Problem State Counter	Crypto Activity Counter	Extended Counter	Group Counter	Basic Sampling
A0A	Yes	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
A0B	Yes	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
A0C	Yes	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
A0D	Yes	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
A0E	Yes	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
A0F	Yes	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
A01	Yes	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
A02	Yes	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
A03	Yes	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
A04	Yes	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
A05	Yes	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
A06	No	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
A07	Yes	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
A08	No	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
A09	No	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
A1A	No	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
A1B	No	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
A1C	Yes	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
A1D	Yes	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
A1E	Yes	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Figure 22-5 Change Logical Partition Security panel

- When **Save** and **Change** is clicked, it might take time to update the various profiles. During this time the panel shown in Figure 22-6 is displayed.

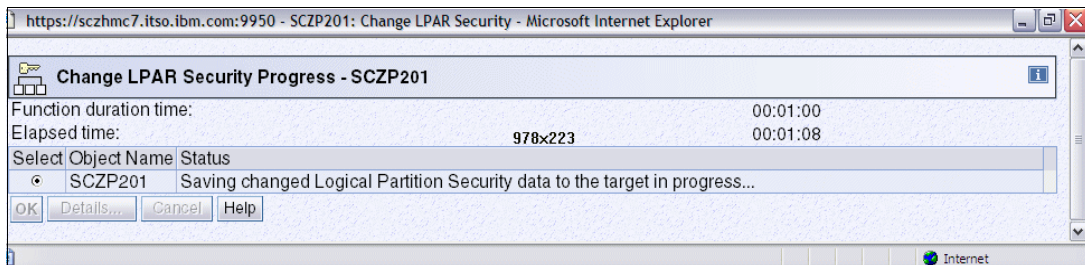


Figure 22-6 Change LPAR Security - save in progress

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 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 just to the CPC from which the BCPii application is going to run on.

## Define 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**.

**Note:** The Customize API Settings button appears in the Support Element Settings panel only if you are logged on the ACSADMIN HMC user ID.

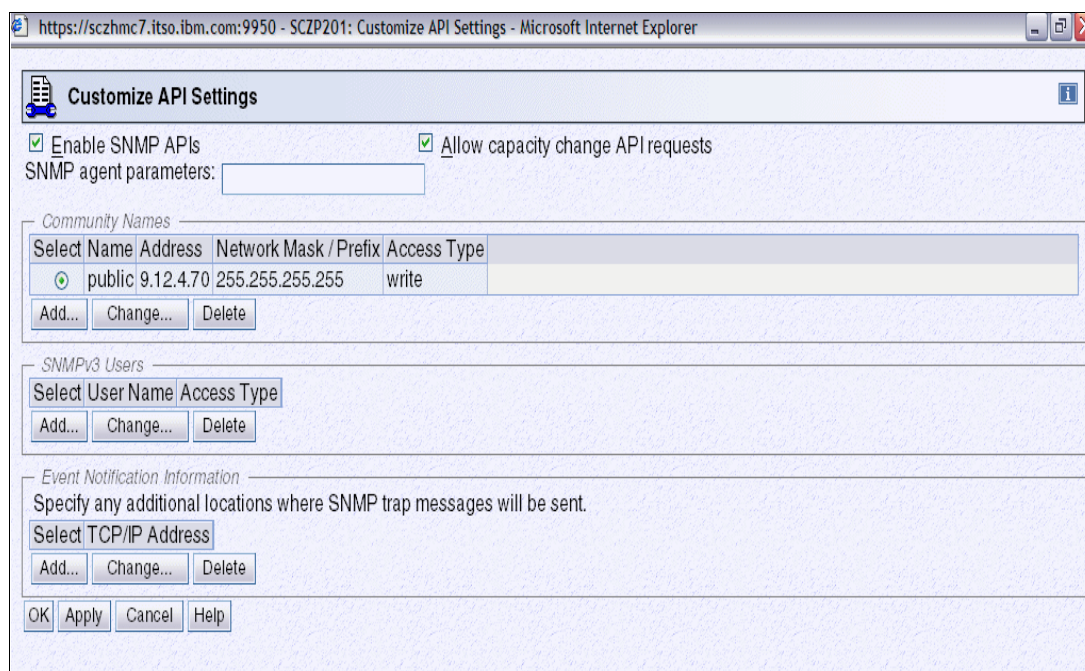


Figure 22-7 Customize API settings

6. Check the **Enable SNMP APIs** check box as shown in Figure 22-7.
7. Consider checking the **Allow capacity change API requests** check box on a z10 or higher system (as shown in Figure 22-7) if the installation is to allow a BCPii application to perform temporary capacity upgrades.

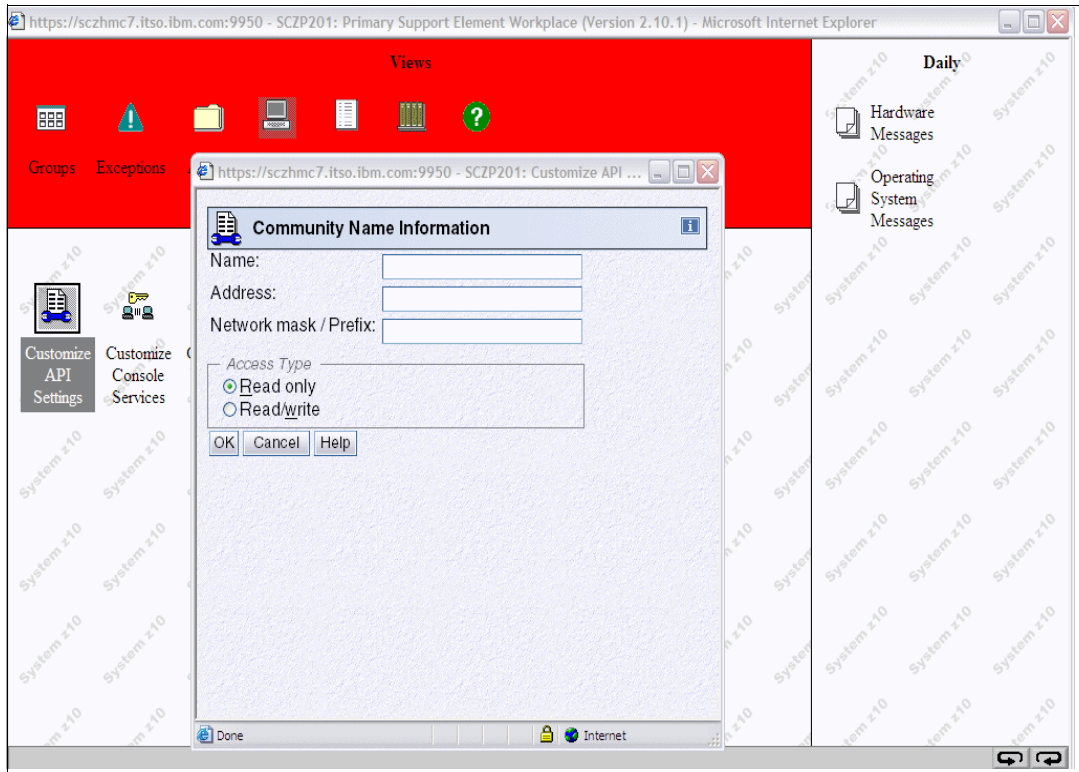


Figure 22-8 Community name information

On a Customize API Settings panel, you can add a Community Name as shown in Figure 22-8.

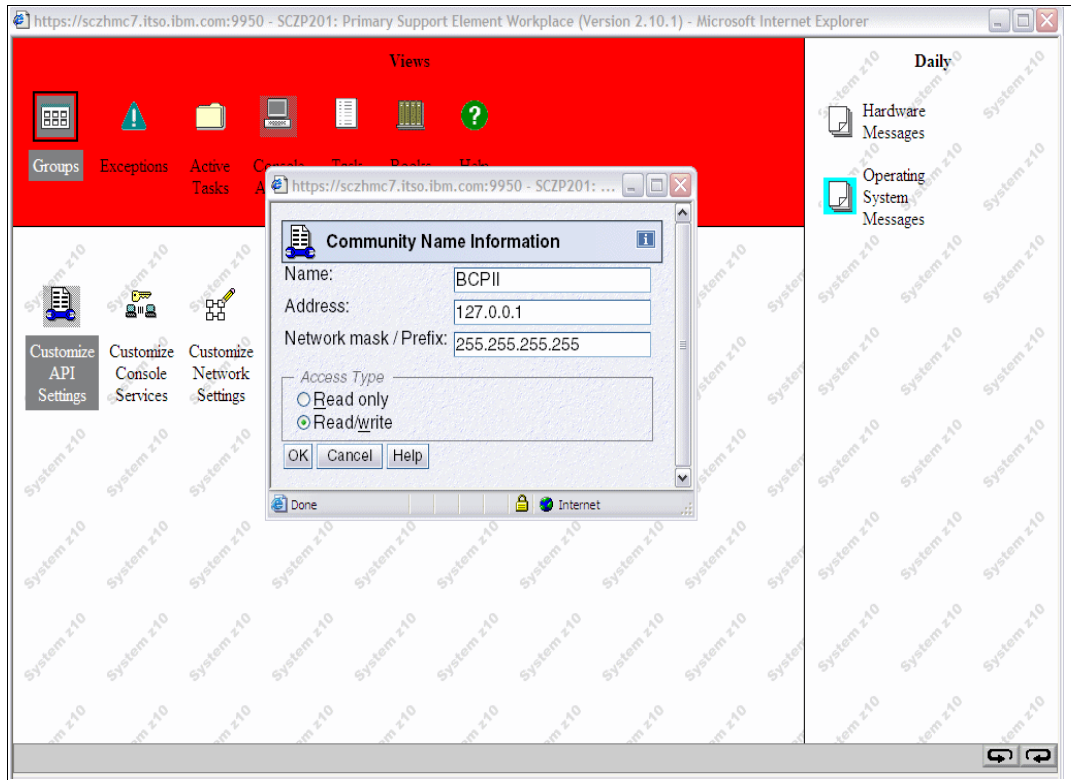


Figure 22-9 Community name in Customize API Settings

8. Make sure that the SNMP agent parameters are blank.
9. Add a BCPii community name, as shown in Figure 22-9. 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 upper-case on the local SE.

**Address:** For BCPii, the address, which is referred to as a loop-back address, must be 127.0.0.1.

**Network mask/Prefix:** This must be 255.255.255.255.

**Note:** Assuming the community name entry is intended to be only used for BCPii purposes, then the address should 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 recommended approach because it opens up potential security concerns.



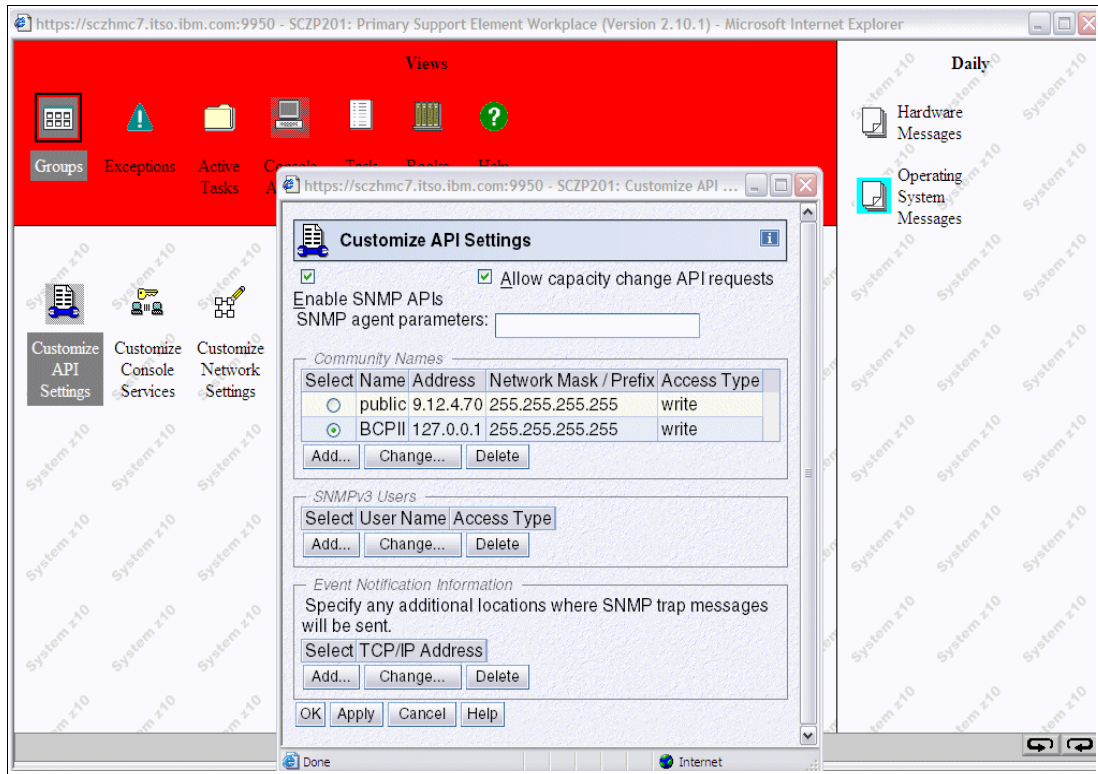


Figure 22-10 Custom API settings after apply

When you select **Apply**, the new Community Name is reflected in the Customize API Settings as shown in Figure 22-10. 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.

## 22.2.2 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 be RACLIST-specified.

Figure 22-11 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 22-11 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 22-1 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 comprised of the Netid.name defined at the SE. The Netid is found in the panel Customize Network Settings, as shown in Figure 22-12 on page 450.

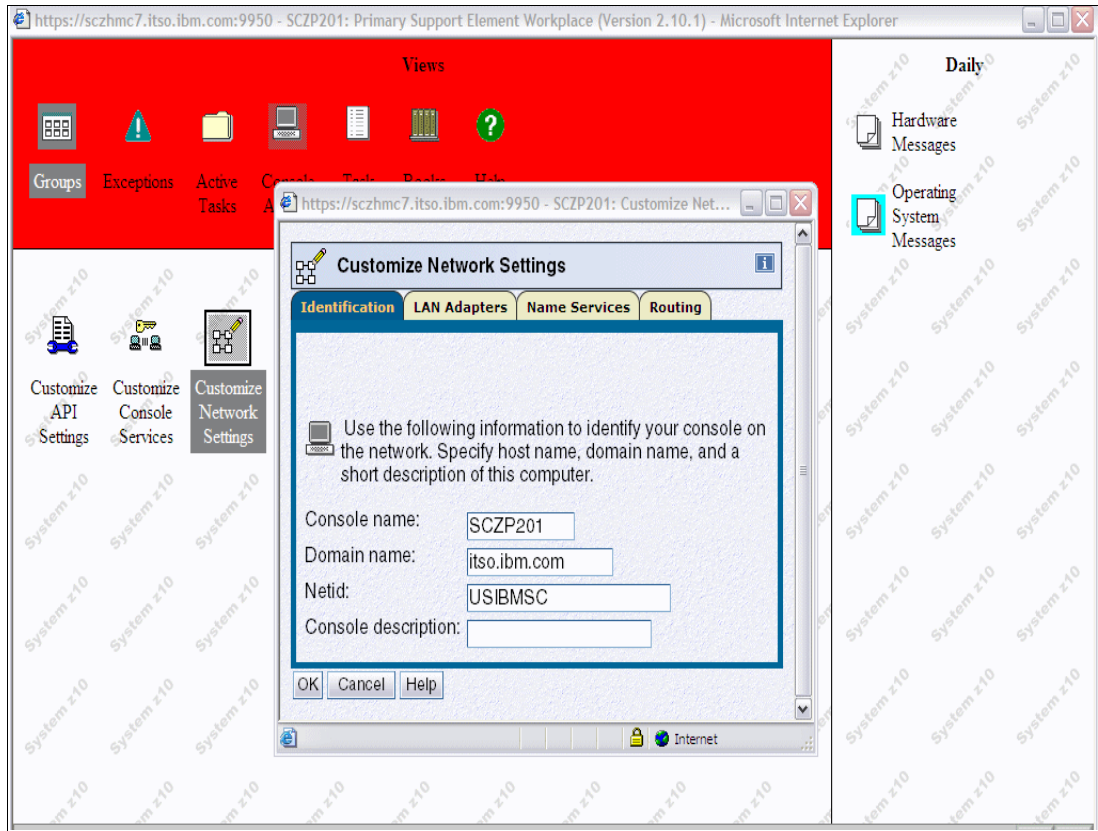


Figure 22-12 Netid on SE panel

The value of 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 22-13 on page 451.



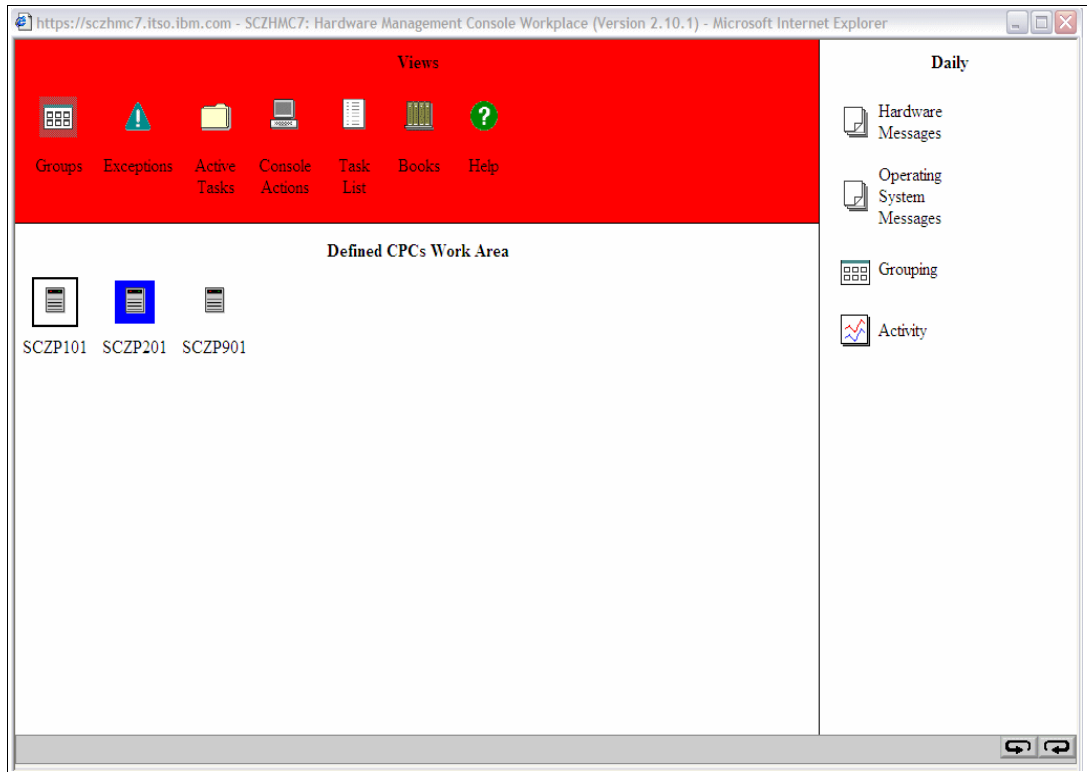


Figure 22-13 CPC name

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 22-14 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 22-14 Authorizing access to CPC resources

Figure 22-15 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 22-15 Authorizing access to LPAR resources

### 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 Support Element. 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 22-16 shows a RACF example assigning a BCPii community name of BCP11 to an existing CPC definition for a SNA name of CPC USIBMSC.SCZP201.

```
RALTER FACILITY HWI.TARGET.USIBMSC.SCZP201 APPLDATA('BCP11')
SETROPTS RACLIST(FACILITY) REFRESH
```

Figure 22-16 Assigning a BCPii community name

### 22.2.3 Configure 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-level qualifier.SCEERUN data sets to be in the link list concatenation. IBM specifies these data sets in the default link list members (PROGxx) 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.

### 22.2.4 Set up the event notification mechanism for z/OS UNIX callers

Applications running in a started procedure, batch, TSO or other non-UNIX System Services environment can use the HWIEVENT service and provide their own ENF exit, which receives control when the application-requested events occur on the target CPC or image.

Applications running in a 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 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 UNIX System Services callers. See the documentation for the UNIX System Services-only services of BCPii in 22.1.3, “BCPii services” on page 437.

## CEA address space setup

The use of the CEA address space by BCPii requires minor CEA setup steps before UNIX System Services-only services of BCPii can work properly. The Common Event Adapter (CEA) address space must be active to allow the UNIX System Services-only services of BCPii to operate. CEA has two modes of operation, namely minimum mode and full-function mode. If the 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 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 a user is allowed to listen to.

Figure 22-17 shows a RACF example giving generic authority to the user ID associated with a 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 22-17 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 in the ENFREQ documentation 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, '02'x =image) and yy denotes the particular event.

Figure 22-18 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 22-18 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.

## BCPii address space does not start up at IPL

If the HWIBCPii address space is not active after an IPL, look for HWI\* messages in the system log. Most of the time, 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 Support Element that controls the CPC that contains the image of z/OS on which BCPii is being started has the improper configuration. Make sure all the steps have been followed in “Configure the local Support Element to support BCPii” on page 443.
- ▶ The community name of this local CPC is either not defined in the security product or contains an incorrect value. This is accompanied by message HWI014I. See “Community name defined in the security product for each CPC” on page 451.

After these problems have been corrected, restart the BCPii address space, as shown in Figure 22-19.

```
000280 S HWISTART
000080 IRR812I PROFILE ** (G) IN THE STARTED CLASS WAS USED 001
000080 TO START HWISTART WITH JOBNAME HWISTART.
000080 $HASP100 HWISTART ON STCINRDR
000280 IEF695I START HWISTART WITH JOBNAME HWISTART IS ASSIGNED TO USER
        IBMUSER , GROUP SYS1
000080 $HASP373 HWISTART STARTED
000080 $HASP395 HWISTART ENDED
000080 IRR812I PROFILE ** (G) IN THE STARTED CLASS WAS USED 006
000080 TO START HWIBCPPII WITH JOBNAME HWIBCPPII.
```

Figure 22-19 Start HWISTART

After all the security-related definitions have been corrected, the BCPii becomes active, as shown in Figure 22-20.

```
000280 IEE252I MEMBER CTIHWI00 FOUND IN SYS1.IBM.PARMLIB
000280 IEF196I IEF285I SYS1.PARMLIB KEPT
000280 IEF196I IEF285I VOL SER NOS= BH8CAT.
000280 IEF196I IEF285I CPAC.PARMLIB KEPT
000280 IEF196I IEF285I VOL SER NOS= Z19CAT.
000280 IEF196I IEF285I SYS1.IBM.PARMLIB KEPT
000280 IEF196I IEF285I VOL SER NOS= Z1BRC1.
000080 HWI016I THE BCPII COMMUNICATION RECOVERY ENVIRONMENT IS 031
000080 NOW ESTABLISHED.
000280 HWI007I BCPII IS ATTEMPTING COMMUNICATION WITH THE LOCAL CENTRAL 032
        PROCESSOR COMPLEX (CPC).
000080 HWI001I BCPII IS ACTIVE.
000080 IXC104I SYSTEM STATUS DETECTION PARTITIONING PROTOCOL ELIGIBILITY: 034
        SYSTEM CANNOT TARGET OTHER SYSTEMS.
000080 REASON: SYSPLEX COUPLE DATA SET NOT FORMATTED FOR THE PROTOCOL
000080 SYSTEM IS NOT ELIGIBLE TO BE TARGETED BY OTHER SYSTEMS.
000080 REASON: SYSPLEX COUPLE DATA SET NOT FORMATTED FOR THE PROTOCOL
```

Figure 22-20 BCPii is active

## Ending the HWIBCPii address space

The application of certain kinds of code maintenance or other unusual circumstances might require that the BCPii address space be stopped. To stop the BCPii address space, issue the STOP command for the BCPii address space P HWIBCPPII. If that does not work, you can use 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 HWIBCPPII address space. When message HWI001I appears, confirming that BCPii is active, 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.

## 22.3 New BCPii diagnostics in z/OS V1R12

Provided in z/OS V1R12 are BCPii diagnostics tools such as:

- ▶ New API return codes
- ▶ CTRACE
  - BCPii cuts CTRACE records using the SYSBCPII CTRACE component
  - Default CTRACE CTIHWI00 parmlib member shipped
  - Two CTRACE options:
    - Min
    - All
  - Dump is taken whenever CTRACE is turned off.
- ▶ Symptom records
  - Limited first failure data capture for select problems
- ▶ Support Element tracing

## 22.4 BCPii dependencies

If installed on a z/OS V1R10 system, BCPii is packaged as a separate deliverable and is enabled via 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 and 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 and Image attributes and commands

After installation of PTF (V1R10) or after V1R11-V1R12 installation:

- ▶ BCPii will attempt to auto-start during IPL of z/OS.
- ▶ If the proper configuration has not been made for BCPii, it will gracefully terminate.
- ▶ BCPii can be re-started via the START command later once the BCPii has been properly configured.

z/OS BCPii (contrary to TSA BCPii) has no dependency on any other software components or products.

z/OS BCPii, though, has dependencies on the hardware.

### 22.4.1 z/OS BCPii vs. 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)

Tivoli System Automation (TSA) BCPii implementation is similar to z/OS BCPii. It does, though, require TSA, NetView®, and Communications Server.

BCPii transport in TSA is for TSA usage only.

On the other hand, z/OS BCPii can run in *any* address space and has no other product dependencies.

### 22.4.2 Hardware dependencies

To get the full functionality of BCPii on a z10, the recommended microcode levels are:

Driver 79

- MCL034 in the N24415 (HMC-SYSTEM) EC stream - Bundle #22
- MCL088 in the N24409 (SE-SYSTEM) EC stream - Bundle #22

On a z9 system, even with the recommended microcode levels

Some 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, reduced attributes) is made available.

**Note:** BCPii does not run in a z/VM virtual machine.

## 22.5 Exploiters of BCPii

Which functions would use BCPii, as they are three-fold:

- ▶ Vendor applications
  - Control center, systems management applications
- ▶ In-house applications
  - Installations that have the need to manipulate System z resources
    - In a more automated, algorithmic approach
    - From a z/OS environment rather than a hands-on HMC environment
    - Without to worry about an intranet or internet network being connected to the process control (HMC) network.
- ▶ z/OS operating system components.

## 22.5.1 Current z/OS system BCPii exploiters

Current BCPii exploiters are:

- ▶ Capacity Provisioning Manager (CPM)
  - Used to have the capability of adding or removing temporary capacity based on goals defined in the tool.
  - Available in z/OS V1R10 via APAR OA24945.
- ▶ Sysplex Failure Manager (SFM)
  - Used to avoid the “Down error message” on the console when a missing XCF heartbeat occurs.
  - Available starting in z/OS V1R11.

## 22.6 Migration and coexistence

BCPii is sysplex ignorant. It can run anywhere starting with z/OS V1R10 and has no migration or coexistence considerations.

## 22.7 BCP internal interface programming

BCPii is a programming interface that offers the possibility to do actions on resources outside the mere LPAR in which it is invoked.

Appendix D, “BCPii Metal C-example” on page 823 contains an example, written in C, of a simple query unto BCPii in order to obtain the number of other CPCs as shown in Example 22-1. From there, you can modify the code and continue to invoke other BCPii functions that are desired to manage the system.

*Example 22-1 BCPii example program running in batch*

---

```
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
```

---

### 22.7.1 A Metal C-example

Note that the C code (such as provided in Appendix D, “BCPii Metal C-example” on page 823) could be compiled with the normal XL C/C++ compiler; see Figure 22-21 on page 458. This would indeed work and implies that the generated module has to run in a Unix System Services environment called a process.

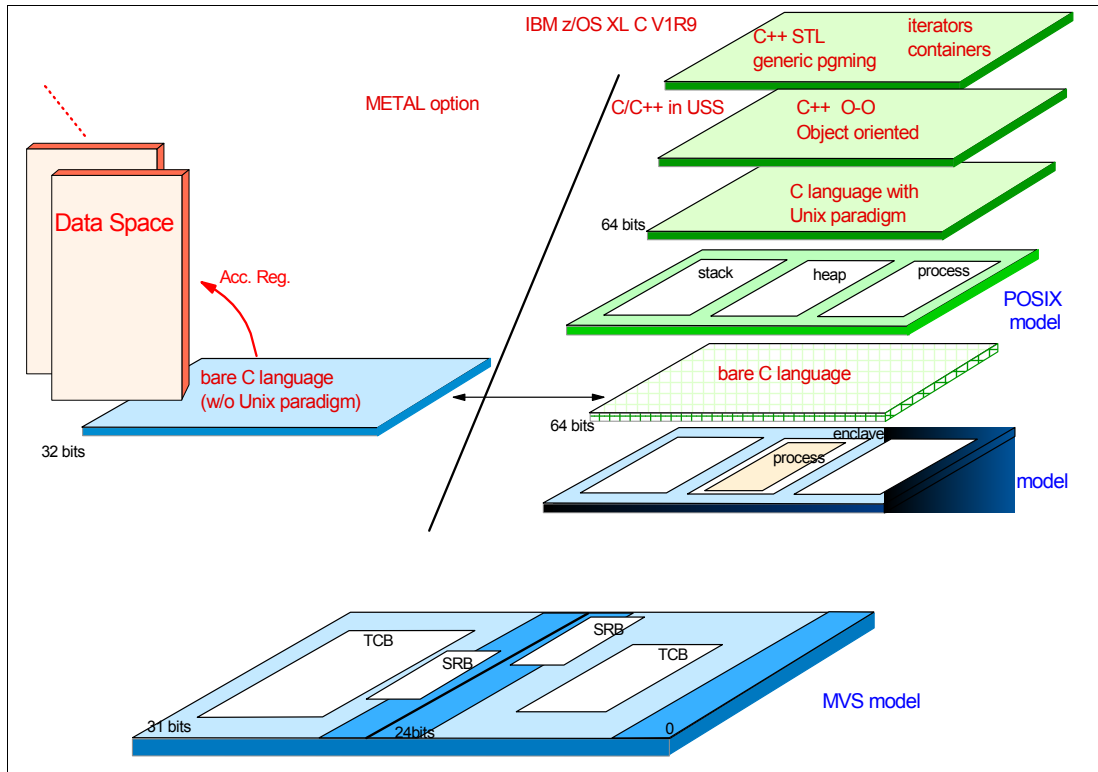


Figure 22-21 Metal C vs XL C/C++

As shown in Figure 22-21, a C program compiled with Metal C is just the expression of an algorithm, just translated into assembler, without any reference to Posix such as process, pipe, heap, and so on. Such is the case with our example in Appendix D, “BCPii Metal C-example” on page 823, which allows us to run the produced module anywhere in z/OS and more specifically, it can be called as a command in a System REXX procedure.

### 22.7.2 BCPii as a set of new z/OS commands

When inserted in a System REXX procedure (as exemplified in D.3, “BCPii example as a System REXX” on page 844), the call to the same program can become a z/OS command (shown in Example 22-2), directly suited for insertion in any automation tool.

Example 22-2 BCPii program called as a System REXX procedure

```

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 869
                        869 EXECNAME=Q
REQTOKEN=0000400000000000C705BA95383ED8A6
                        869 Query zHybrid

```





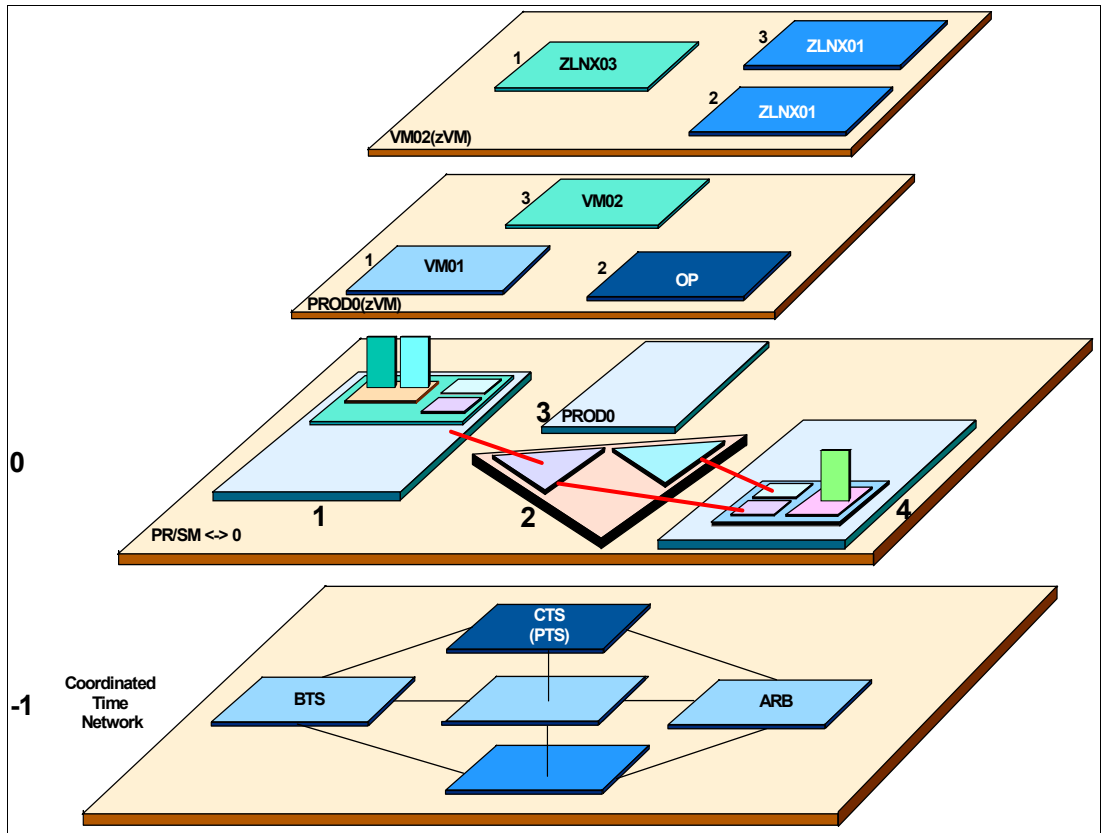


Figure 22-22 Sample context of the proposed BCPii commands

### 22.7.3 Potential advantages of a new command set

A new command set, as proposed in the preceding paragraphs, could help to resolve various BCPii aspects:

- ▶ It could help implement proper RACF security checking within the respective REXX procedures as expressed in “BCPii services” on page 439.
- ▶ It could also be the base for tracing what command is issued by which user on which system, hence building up a log of the actions issued on the BCPii interface.



## About (de)ciphering

Elliptic curve cryptography (ECC) is an approach to public key cryptography based on the algebraic structure of elliptic curves over finite fields. The theory of elliptic curves is one of the most advanced areas in mathematics.

This chapter describes algorithmic and structural enhancements in z/OS V1R12 to ciphering schemes, as follows:

- ▶ New algorithmic techniques to help enhance both performance and strength of the available cryptographic functions on the zServers.
- ▶ New structures for consequently reducing the appearance of clear keys in storage.

## 23.1 Elliptic curves cryptography

In an asymmetric cryptographic process one key is used to encipher the data, and a different but corresponding key is used to decipher the data. A system that uses this type of process is known as a public key system. The key that is used to encipher the data is widely known, but the corresponding key for deciphering the data is a secret. For example, many people can use your public key to send enciphered data to you with confidence, knowing that only you should possess the secret key for deciphering the data.

Public key cryptographic algorithms are used in processes that simplify the distribution of secret keys, assuring data integrity and providing nonrepudiation through the use of digital signatures.

The widely known and tested public key algorithms use a relatively large key. The resulting computer processing time makes them less than ideal for data encryption that requires a high transaction rate. Public key systems, therefore, are often restricted to situations in which the characteristics of the public key algorithms have special value, such as digital signatures or key distribution. PKA calculation rates are fast enough to enable the common use of digital signatures.

ICSF supports these public key algorithms:

- ▶ Rivest-Shamir-Adelman (RSA)
- ▶ Digital Signature Standard (DSS)

A new kind of asymmetric cryptography has been introduced in z/OS V1R12, following FIPS recommendations.

### 23.1.1 From symmetric to asymmetric cryptography

Cryptography has been an old approach trying to make information meaningless for someone else. It is a science that has been used for thousands of years. It concerns encryption as well as decryption of data such that valuable information can remain safe from unauthorized users.

In the 70s, as pictured in Figure 23-1 on page 463, a standard was defined, called Data Encryption Standard (DES).

**Note:** In computer security, the National Institute of Standards and Technology (NIST) Data Encryption Standard, was adopted by the U.S. government as Federal Information Processing Standard (FIPS) Publication 46, which allows only hardware implementations of the data encryption algorithm.

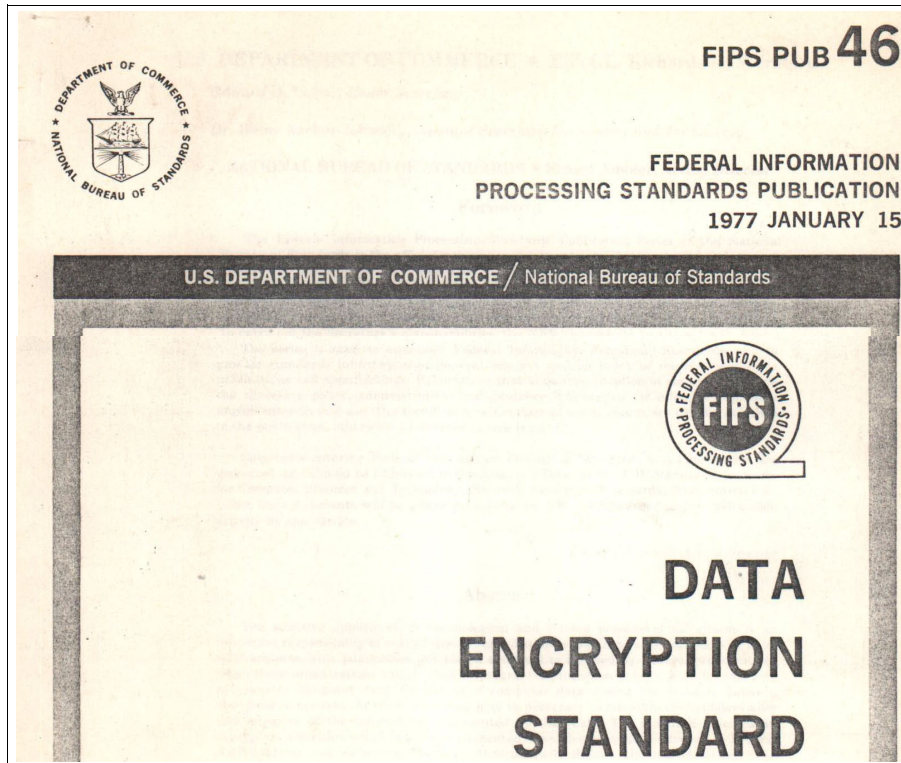


Figure 23-1 DES document definition

Since then, variations such as TDES have been developed. AES, another symmetric cryptographic algorithm, has been adopted, following an open competition among universities to increase DES strength, beyond merely increasing the length of the keys.

In 1978, Rivest, Shamir, and Adleman (RSA), then from MIT (Figure 23-2), proposed an asymmetric cryptographic algorithm, a new class of cryptography.

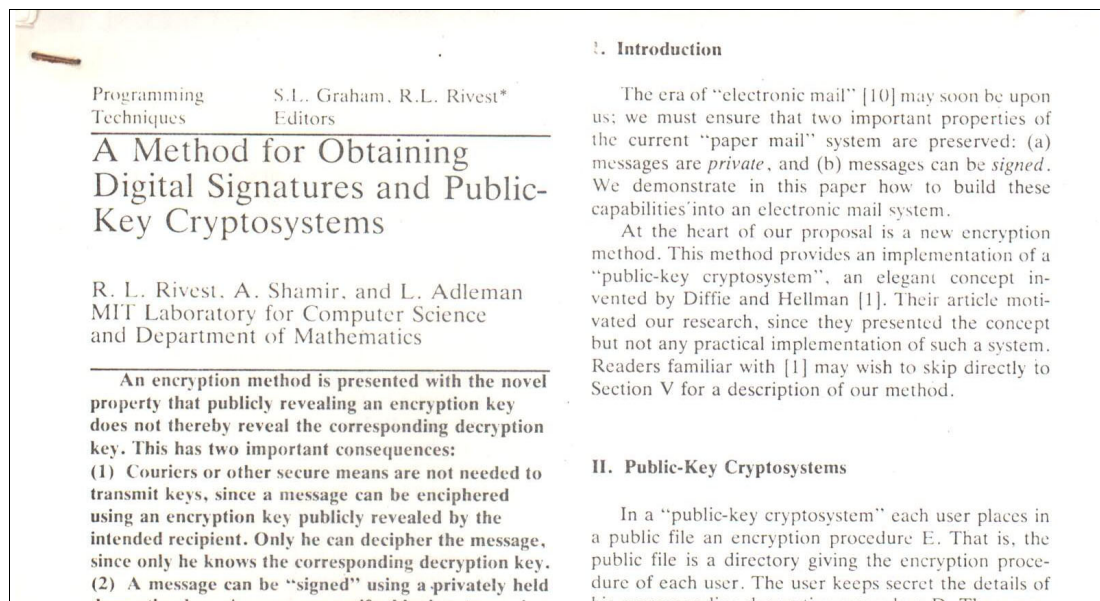


Figure 23-2 RSA document definition

## Symmetric versus asymmetric encryption

Indeed, instead of having the same key to encrypt and decrypt (see the upper part of Figure 23-2 on page 463), RSA proposed to use a different key for encrypting and decrypting (see the lower part of Figure 23-2), hence the qualifier of asymmetric. The strength of an asymmetric method lies in the difficulty of deducing the decrypt key from the encrypt key.

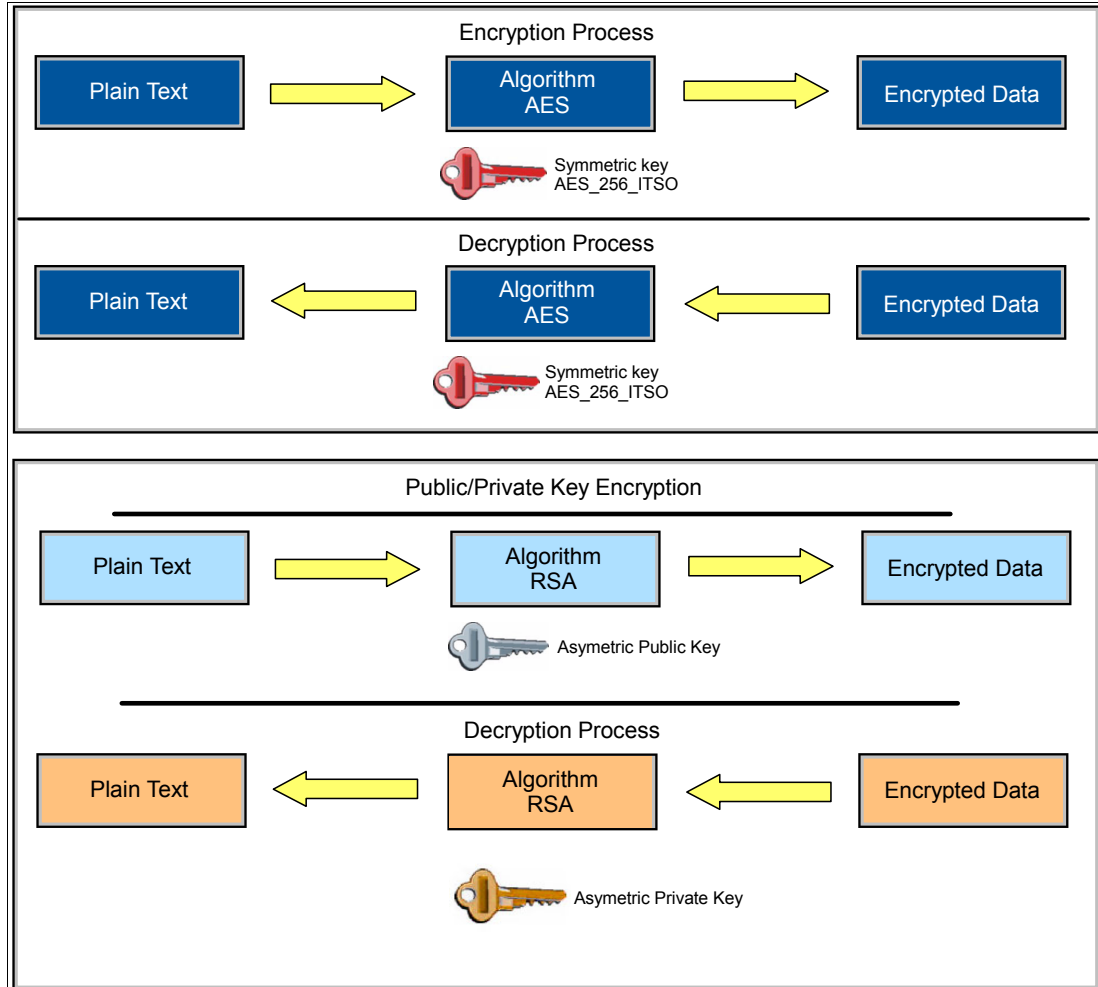


Figure 23-3 Symmetric versus asymmetric encryption

Overall, the longer the key the stronger is the encryption process, whether symmetric or asymmetric.

### 23.1.2 Elliptic curves

More and more specialists in cryptography are using elliptic curves, independently proposed by Victor Miller and Neal Koblitz in 1985. The theory of elliptic curves is a new and rich domain of math which, among other things, allowed Andrew Wiles to prove Fermat's last theorem.

An elliptic curve is a simple object with surprising properties. Algebraic speaking, it is generally of the form shown in Figure 23-4 on page 465.

$$y^2 + a_1 xy + a_3 y = x^3 + a_2 x^2 + a_4 x + a_6$$

Figure 23-4 General elliptic curve

For cryptography, it is simplified with  $a_1$ ,  $a_2$ , and  $a_3$  being equal to 0; generally, cryptography designers rename  $a_4 = a$  and  $a_6 = b$ , resulting in the formula shown in Figure 23-5.

$$y^2 = x^3 + a x + b$$

Figure 23-5 Cryptographic elliptic curve

Figure 23-7 is an example of an elliptic curve, whose equation is shown in Figure 23-6.

$$y^2 = x^3 - 5x + 3$$

Figure 23-6 Equation of an elliptic curve

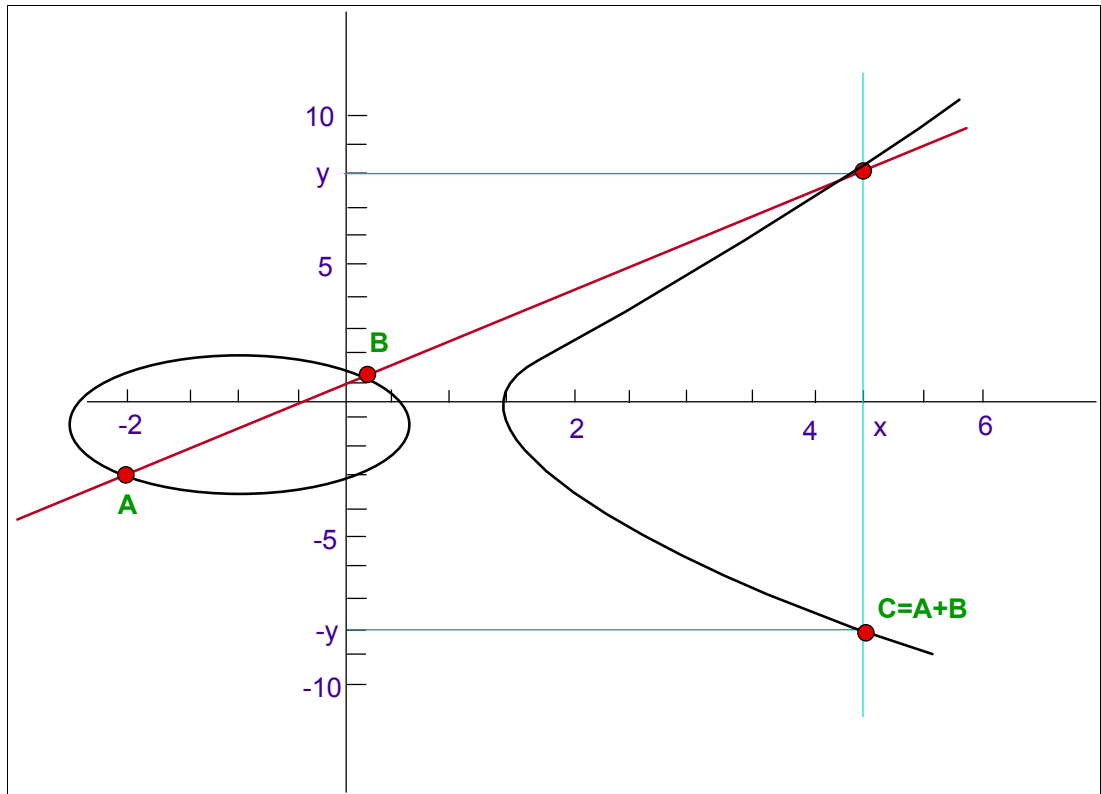


Figure 23-7 Example of an elliptic curve

### Addition of two points

Let us look at points A and B on the curve in Figure 23-7. The line going through A and B crosses the curve in a third point with coordinates  $(x,y)$ . Its symmetric point  $(x,-y)$  is itself on the curve, designated as  $A+B$  to stress that it is built from A and B. The surprising thing is that this '+' operation contains all the properties of the classical addition of numbers. It means that all calculations involving addition, subtraction, and division with an integer (classical on the line of real numbers) can be done on an elliptic curve.

It is possible, as proved by Miller and Noblitz, to encrypt based on this strange '+' operation, as would be done with the classical '+' operation. This makes possible more elaborate calculations, that is, encryption with elliptic curves with a 192-bit (or 4096-bits) long key will be as strong as an RSA encryption with a 1024-bit (160-224-bits) long key, as depicted in Figure 23-8. This explains why elliptic curve cryptography (ECC) will be used more and more for asymmetric encryption.

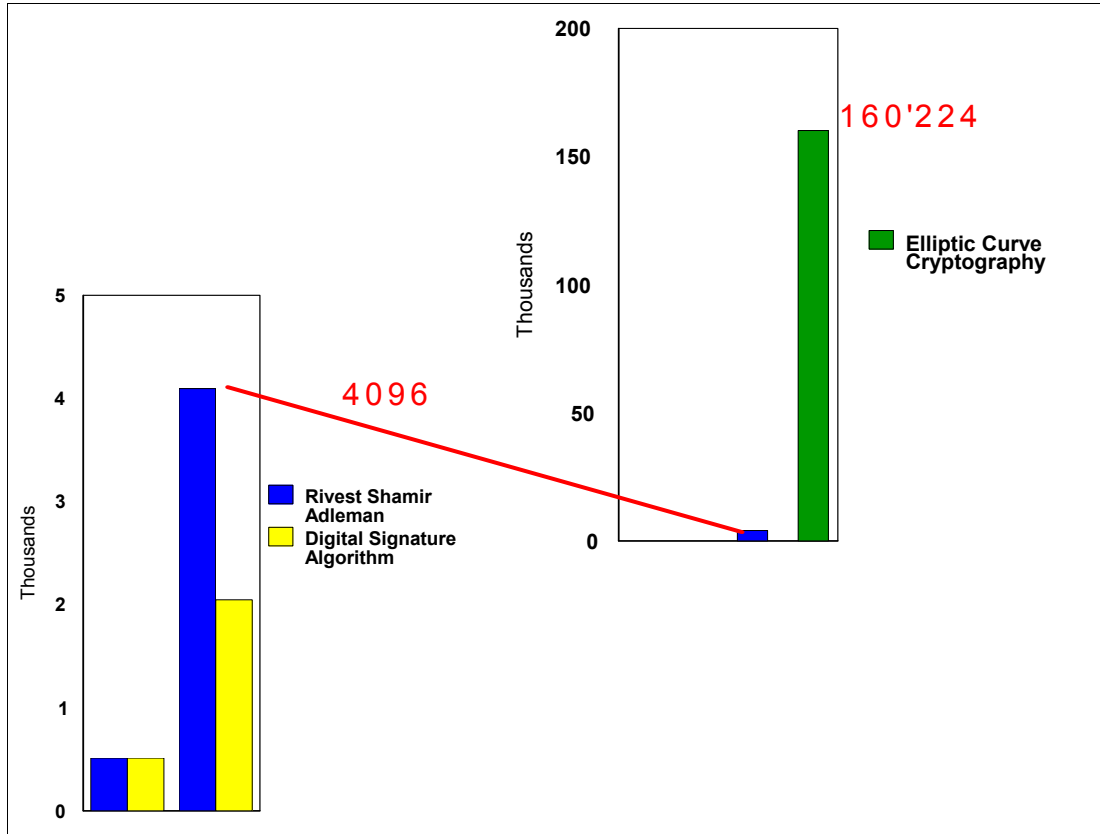


Figure 23-8 Asymmetric key length

### 23.1.3 Elliptic curve cryptography

Elliptic curve cryptography requires the specification of an elliptic curve using EC parameters and public and private keys that relate to points on that curve. To ensure elliptic curves in use are secure, several curves have been predefined using accepted methods and standards, and recommended for use by various security authorities. These are known as *named curves* and can be identified by OIDs rather than specified EC parameters. ICSF provides ECC support for the named curves, listed in Figure 23-9 on page 467, which are defined over prime finite fields only (characteristic Type 2 fields are not supported).

**Note:** In cryptography, Type 2 products are unclassified cryptographic equipment, assemblies, or components, endorsed by the National Security Agency (NSA), for use in telecommunications and automated information systems for the protection of national security information.



NIST Recommended Curves	Brainpool Defined Curves
secp192r1 – {1.2.840.10045.3.1.1}	brainpoolP160r1 – {1.3.36.3.3.2.8.1.1.1}
secp224r1 – {1.3.132.0.33}	brainpoolP192r1 – {1.3.36.3.3.2.8.1.1.3}
secp256r1 – {1.2.840.10045.3.1.7}	brainpoolP224r1 – {1.3.36.3.3.2.8.1.1.5}
secp384r1 – {1.3.132.0.34}	brainpoolP256r1 – {1.3.36.3.3.2.8.1.1.7}
secp521r1 – {1.3.132.0.35}	brainpoolP320r1 – {1.3.36.3.3.2.8.1.1.9}
	brainpoolP384r1 – {1.3.36.3.3.2.8.1.1.11}
	brainpoolP512r1 – {1.3.36.3.3.2.8.1.1.13}

Figure 23-9 Named curves

Supported curves implement key sizes related to the curve name (160 bits, 192 bits, and so on).

### **ECC: the american view**

NIST-recommended curves were defined by Certicom Research in SEC 2: Recommended Elliptic Curve Domain Parameters, and recommended for use by the US Federal Government by NIST in FIPS PUB 186-2 Digital Signature Standard (DSS).

### **ECC: the european view**

Brainpool-defined curves were defined in RFC 5639, Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation, in response to perceived shortcomings of the NIST recommended curves.

## **23.1.4 System SSL elliptic curve cryptography**

The U.S. National Security Agency (NSA) announced Suite B Cryptography in 2005, and implementation guidelines in 2009. One newly listed technology was elliptic curve cryptography (ECC), which is regarded as a faster algorithm that requires a smaller key than RSA cryptography. This type of cryptography is expected to be attractive for use with small devices such as smart cards, which have limited computing power. In z/OS V1R12, PKI Services allows you to create and sign certificates with ECC keys in addition to RSA keys.

Specified EC Parameters define all necessary attributes of an elliptical curve. Only specified EC Parameters that match a supported named curve can be used by System SSL.

Implicit EC Parameters contain a NULL EC Parameters field and inherit their parameters from the ECC key of the parent certificate. This form of EC Parameters is not supported.

Attempting to use an unsupported curve with System SSL will result in a CMSERR\_ECURVE\_NOT\_SUPPORTED (0x0335307E) error.

In V1R12, System SSL provides support for ECC-related data structures, signing data, and verifying signed data using elliptic curve digital signature algorithm (ECDSA). This is intended to allow exploiters of z/OS System SSL to import ECC-style certificates and private keys into key database files or PKCS#11 tokens and use ECDSA certificates to sign and verify operations.

### **ICSF growth (PKCS#11)**

With the focus on data security, cryptographic services must be continuously available. New ICSF designs that use integration between hardware and software components are intended

to help improve the availability of z/OS host applications. In z/OS V1R12, a software cryptographic engine function embedded in ICSF is designed to be used so that no optional cryptographic coprocessors will be required for PKCS11 processing. Additional algorithms to be supported are DSA, DH, EC, AES GCM, BLOWFISH, and RC4. Furthermore, this support provides cryptography services and packaging that is designed to meet NIST FIPS 140-2 level 1 criteria, and support to comply with RFC4869 Suite B Cryptographic Suites for Ipsec and the following new clear key algorithms:

- ▶ Galois-Counter Mode encryption for AES (GCM)
- ▶ Elliptic curve Diffie-Hellman key derivation (ECDH)
- ▶ Elliptic curve digital signature algorithm (ECDSA)
- ▶ HMAC

In addition, support has been added for several new cryptographic algorithms:

HMACSHA-256-128; HMAC-SHA-384-192; HMAC-SHA-512-256; PRF-HMAC-SHA-256;  
PRF-HMAC-SHA-384; PRF-HMAC-SHA-512; AES-128-GCM; AES-256-GCM

**Note:** This function is also available with the z/OS Cryptographic Support for z/OS V1R9-V1R11 web deliverable.

Up until z/OS V1R12, System SSL contained software implementations for all utilized algorithms. Either through direct calls to CPACF or ICSF, System SSL is able to exploit cryptographic support provided through either the CPACF or crypto express cards. With the introduction of elliptic curve cryptography, System SSL will not be having its own software implementation. It will be relying on ICSF's PKCS#11 support for ECC.

ICSF HCR7770 is shipped in the base z/OS V1R12 and provides ECC support. In order for clear key EC digital signature generation and verification to be performed, READ access to CSF1PKS and CSF1PKV in the CSFSERV resource class is needed.

Similar to System SSL, ICSF has been designed to meet FIPS 140-2 Level 1 certification. When System SSL is running in FIPS mode, it calls ICSF for EC digital signature generation, or verification will also request FIPS mode. If ICSF is not executing in FIPS mode, ICSF returns with a failure. This failure results in System SSL failing the cryptographic processing.

**Note:** Because it implies code signature, FIPS 140-2 is an option only available starting with z/OS V1R11.

FIPS mode restricts the allowed name curves to the name curves recommended by NIST. System SSL calls ICSF for ECC unconditionally. If ECC support is not available when required, System SSL returns CMSERR\_ICSF\_NOT\_AVAILABLE (0x03353083).

## Dependencies

ECC and PKCS#11 require ICSF Web Deliverable #9 (HCR7770) to be installed and operating on the system.

ECC and PKCS#11 SAF key ring support requires RACF (HRF7770) to be installed and operating on the system.

Any current exploiters of the System SSL certificate validation process that may encounter ECC-based certificates, such as Communication Server AT-TLS, IPsec and PKI Services, can use ECC.

## **z196 support**

Beyond ECC support as a PKCS#11 service (ICSF HCR7770), ECC is also supported (ICSF HCR7780) on z196 systems as a native CCA service.

## **RACDCERT ECC certificate support**

The U.S. National Security Agency (NSA) announced Suite B Cryptography in 2005, and implementation guidelines in 2009. One newly listed technology was elliptic curve cryptography (ECC). This type of cryptography is expected to be attractive for use with small devices such as smart cards, which have limited computing power.

As higher levels of security are needed, one can either move to larger key sizes or move from RSA or DSA to elliptic curves. ECC is regarded as a faster algorithm that requires a smaller key than RSA cryptography. For example, a RSA 2048-bit key has equivalent strength to a 224-bit ECC key.

With the introduction of ECC as a public key cryptographic algorithm and with the direction to move to ECC as RSA key sizes increase, System SSL needs to provide fundamental certificate and digital signature support for exploiting applications.

In z/OS V1R12, System SSL has added support to understand and process ECC-style certificates and functionality to sign and verify data using an ECC-based public/private key pair.

In z/OS V1R12, the RACF RACDCERT command supports certificates with the ECC keys, in addition to RSA and DSA keys.

Clients are thus able to process ECC-based certificates:

- ▶ ECC offers equivalent security with smaller key sizes than RSA.
- ▶ ECC provides the ability to have a higher level of security.
- ▶ Computational and memory utilization are less than for RSA.

## **23.1.5 Length is strength**

Since the beginning of cryptography, length of the key used to cipher has been in correlation with the strength of the used algorithm. Otherwise, stated ECC provides more “security per bit” than other public-key algorithms. As pointed out in “Addition of two points” on page 465, this is even more so when comparing the RSA class of cryptography with ECC.

Table 23-1 gives NIST-recommended key sizes for the three classes of cryptographic algorithms.

<b>Symmetric Key Size (bits)</b>	<b>RSA and DH Key Size (bits)</b>	<b>EC Key Size (bits)</b>
80	1024	160
112	2048	224
128	3072	256
192	7680	384
256	15360	521

*Table 23-1 NIST-recommended key sizes*

RACF has restrictions for the size of the private key for certificates that have associated private keys.

The maximum key size depends on key type as specified in Table 23-2. Shorter keys of the ECC class, which are generated when NISTECC or BPECC is specified, achieve comparable key strengths when compared with longer RSA keys (thanks to the strange '+' operation described in "Addition of two points" on page 465).

Table 23-2 Maximum key sizes

Private key type	Max key size
BPECC key	512 bits
NISTECC key	521 bits
ICSF RSA key	1024 bits
non-ICSF DSA key	2048 bits
non-ICSF RSA key	4096 bits
PCI-class crypto-coprocessor RSA key	4096 bits

RSA, NISTECC, and BPECC keys of the following sizes are comparable in strength, as specified in Table 23-3.

Table 23-3 Strength comparisons

RSA key size	NISTECC key size	BPECC key size
1024 bits	192 bits	160 or 192 bits
2048 bits	224 bits	224 bits
3072 bits	256 bits	256 or 320 bits
7680 bits	384 bits	384 bits
15360 bits	521 bits	512 bits

## Hash algorithms

RACF signs certificates using a set of secure hash algorithms based on the SHA-1 and SHA-2 hash functions.

When the signing key is a DSA type, the SHA-1 algorithm is used for keys of all sizes.

For NISTECC keys, valid key sizes are 192, 224, 256, 384, and 521 bits. For BPECC keys, valid key sizes are 160, 192, 224, 256, 320, 384, and 512 bits.

For RSA and DSA keys, the minimum key size is 512. The maximum key size is determined by United States export regulations and is controlled by RACF and non-RACF code in z/OS. Depending on the installation, non-RACF code may enforce a lower maximum size.

When the signing key is an RSA, NISTECC, or BPECC type, the size of the signing key determines the hashing algorithm used for signing, as shown in Table 23-4 on page 471.

Table 23-4 Size of the signing key

Hashing algorithm		signing key size	
used for signing	RSA	NISTECC	BPECC
SHA-1	Less than 2048 bits		
SHA-256	2048 bits or longer	192, 224, or 256 bits	160, 192, 224, 256, or 320 bits
SHA-384		384 bits	384 bits
SHA-512		521 bits	512 bits

### 23.1.6 z196 specifics

Only clean and internal ECC keys are supported at the GA1 level of z196 systems. In such systems, ECC keys are protected using a new ECC master key (actually a 256-bit AES key). Hardware can be used as appropriate to accelerate ECC performance.

**Note:** From the TKE, administrators can generate key material, load or clear the new ECC master key register, or clear the old ECC master key register. The ECC key material can be stored on the TKE or on a smart card.

## 23.2 CPACF-protected key

CPACF-protected key is an enhancement (also known as Message-Security-Assist Extension 3 in Principles of Operation) to “Central Processor Assist to Cryptographic Function” (CPACF) since IBM System z10 GA3 servers. The CEX3C feature is designed to help facilitate the continued privacy of cryptographic key material when used by the CPACF for high performance data encryption. Leveraging the unique z/Architecture, protected key CPACF is designed to help ensure that key material is not visible to applications or operating systems when used for encryption operations. Protected key CPACF is designed to provide significant throughput improvements for large volumes of data and low latency for small blocks of data.

### 23.2.1 Principles

In z/OS V1R12, ICSF exploits the enhancements made to the CPACF in support of separate key wrapping keys for DES, TDES, and AES. This is designed to provide the same functions available using the PCI card, but with the advantage of CPACF performance.

With the z10 GA3, the System z hardware adds support for protected keys. Protected keys blend the security of the Crypto Express3 (CEX3) and the performance characteristics of the CPACF. While a secure key is encrypted under a master key, a protected key is encrypted under a wrapping key that is uniquely created for each LPAR. The wrapping key is created each time an LPAR is activated or reset, and there are two variations of the wrapping key. One is for wrapping DES or TDES operational keys and a second for wrapping AES keys. The wrapping key is considered as the contents of a register, implemented in the Hardware System Area (HSA) for the z10 and z196 systems. The wrapping key is only accessible by firmware and cannot be read by the operating system or applications (even if running authorized).

If a CEX3 coprocessor is available, a protected key can begin life as a secure key. That is, ICSF will retrieve the secure key from the CKDS and the clear value will be recovered (decrypted from under the master key) and then that clear key is re-encrypted (or wrapped) under the appropriate wrapping key for that LPAR. The rewrapping is managed by System z firmware in conjunction with the CEX3 coprocessor (CEX3C); the clear key value is never visible in operating system or application storage.

The use of a protected key is driven by the application when it invokes a CPACF API and specifies the label of a secure key. The secure key is retrieved from the CKDS and brought into the CEX3 coprocessor where it is decrypted from under the master key and re-encrypted under the appropriate wrapping key. That wrapped key is passed back to ICSF, which uses it within the CPACF to perform encryption and decryption operations. The underlying clear value of the key never exists in any address space, but only inside the inaccessible part of the system.

Alternatively, if no CEX3C is available, an application could use a clear key as the source for a protected key without using ICSF. In this case, the application is responsible for creating or loading a clear key value and then using the new PCKMO instruction (new on z10 GA3 - Nov. 2009 or later microcode) to wrap the key under the appropriate wrapping key.

Since the wrapping key is unique to each LPAR, a protected key cannot be shared with another LPAR. Either the original secure key or the clear key value, not the protected key, would be shared or stored. The underlying secure key or clear key would need to be retrieved and wrapped with the wrapping key in each LPAR.

**Note:** There are thus two sets (DES-TDES and AES) of wrapped keys kept in registers for each LPAR. This protocol is also valid for each virtual machine running under z/VM whatever the level of SIE in which it is actually running.

## 23.2.2 Protection of cryptographic keys

Globally, protection of cryptographic keys is done in the following ways:

- ▶ The source key is a CCA MK wrapped key residing in the CKDS.
- ▶ The key label of the source key can be specified on existing symmetric ICSF APIs that call CPACF.
- ▶ ICSF is responsible for managing the CPACF wrapping, caching, and validation source key.
- ▶ After initial “CPACF Wrapping” of a key, encryption overhead is essentially the same as for clear key with the added benefit that the key is never in operating system addressable host memory.

It is assumed that the source key must reside in the CKDS and can be a DES, TDES, or AES encrypted key. The source key may be wrapped with either the symmetric master key, DES master key, or AES master key. The following key lengths are supported:

- ▶ AES – 128, 192, and 256 bits
- ▶ DES/TDES – 64, 128, and 192 bits

The first time an encrypted key label is specified as input to one of the symmetric key decipher or encipher services:

- ▶ ICSF initiates the protocol for rewrapping the encrypted key.
- ▶ The encrypted key token is passed to the card, unwrapped, and returned.

- ▶ The System z firmware intercepts the output before it is visible in operating system storage. The key is then wrapped with the appropriate (either DES-TDES or AES) CPACF Wrapping Key and this result is returned to ICSF (operating system) as the result.
- ▶ ICSF will then cache the CPACF-wrapped key for subsequent API calls.

Figure 23-10 illustrates the flow of the CPACF wrapping protocol.

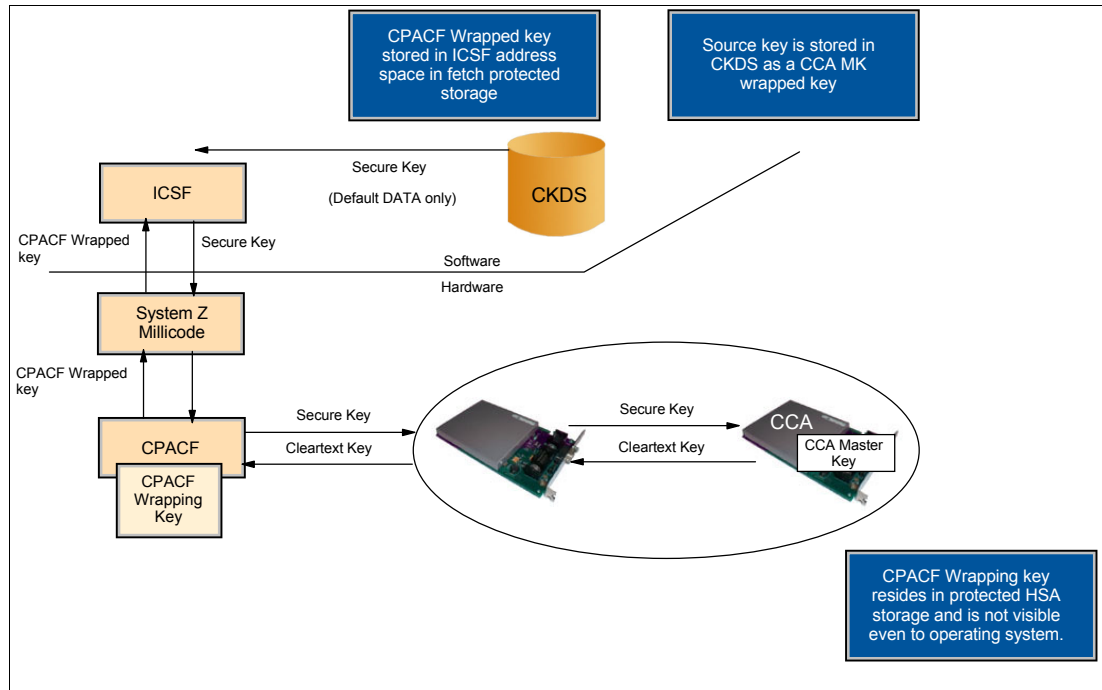


Figure 23-10 CPACF protected key: functional principles

ICSF will then use the CPACF wrapped instance of the key as input to the “encrypted” flavor of the requested CPACF function. If the CPACF operation completes with an indication that the wrapped key is not valid, ICSF will re-initiate the wrapping protocol, cache the rewrapped key, and redrive the operation.

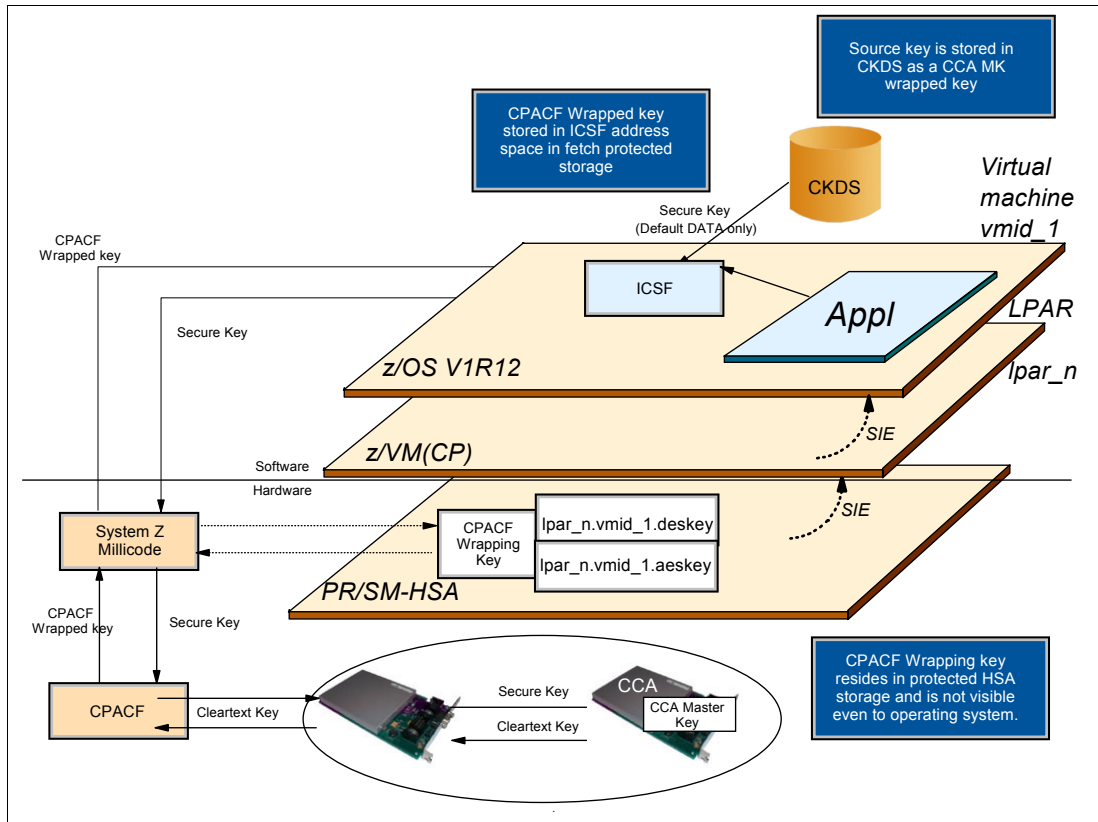


Figure 23-11 CPACF protected key: structural view

The wrapping keyset is created each time an LPAR is activated or reset, and there are two variations of the wrapping key set. One (lpar\_n.deskey) is for wrapping DES or TDES operational keys, and a second (lpar\_n.aeskey) for wrapping AES keys. The wrapping key is stored in the appropriate registers (implemented in Hardware System Area - HSA - for z10 and z196 systems). The wrapping key is only accessible by firmware and cannot be read by the operating system or applications (even if running authorized).

Each set of PCKMO-Encrypt-DEA-Key functions and PCKMO-Encrypt-AES-Key functions can be independently disabled by external means. When a set is disabled, functions in the set appear as if they were not installed.

### 23.2.3 Architectural description

When the message-security-assist-extension-3 facility is installed, two wrapping-key registers and two wrapping-key-verification-pattern registers are provided for each configuration, as depicted in Figure 23-12 on page 475. The two wrapping-key registers consist of a 192-bit DES wrapping-key ( $WK_d$ ) register and a 256-bit AES wrapping-key ( $WK_a$ ) register. The two wrapping-key-verification-pattern registers consist of a 192-bit DES wrapping-key-verification-pattern ( $WK_dVP$ ) register and a 256-bit AES wrapping-key-verification-pattern ( $WK_aVP$ ) register.



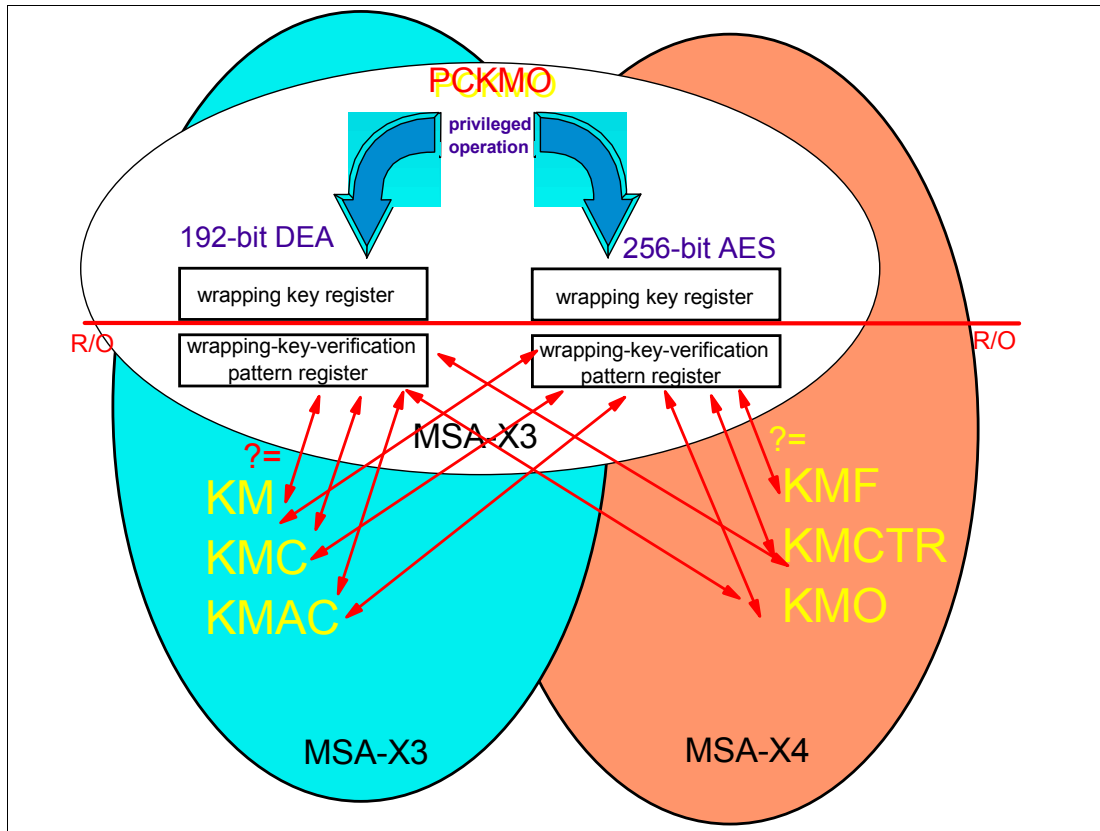


Figure 23-12 New set of registers: one per SIE environment

### DES or TDES keys

AES keys may be clear keys or secure keys protected by encryption under a AES master key. The AES master key always remains within the secure boundary of the cryptographic coprocessor on the server. There is only one AES master key and it is used only to encrypt and decrypt other AES keys. All coprocessors must have the same AES master key for the coprocessors to be active.

**Note:** Secure AES keys are supported on a z9 EC, z9 BC, z10 EC, and z10 BC with a Crypto Express2 Coprocessor with the November 2008 or later licensed internal code (LIC).

### AES key forms

A key that is protected under the AES-MK is in operational form, which means that ICSF can use it in cryptographic functions on the system. When you store a key with a file or send it to another system, the key can be protected using an RSA key pair.

### Types of AES keys

While the DES keys are grouped into different types, that is not the case for AES keys. AES keys are limited to:

- ▶ AES Master key

A 256-bit AES key that is used only to encrypt and decrypt AES operational keys. The ICSF administrator installs and changes the AES master key using the ICSF panels or the optional TKE workstation. The AES master key always remains within the secure boundaries of the cryptographic coprocessors.

- ▶ Data-encrypting keys

All AES operational keys are data-encrypting keys. Data-encrypting (DATA) keys are 128, 192, and 256-bit keys. DATA keys are used to encipher and decipher data.

### **Message-security-assist extension 3**

The message-security-assist extension 3 (MSA-X3) facility provides a means to protect user cryptographic keys by encrypting them under machine-generated wrapping keys. As depicted in Figure 23-12, when this extension is installed, two wrapping keys are provided for each configuration: one for protecting user DEA keys and another for protecting user AES keys. The wrapping keys reside in the machine so that, with an appropriate setting of controls, no clear value of user cryptographic keys is observed anywhere in the system by any program.

Those registers (one per SIE environment) are used in MSA-X3 by KM, KMC, and KMAC instructions.

### **Message-security-assist-extension-4**

The message-security-assist extension 4 (MSA-X4) provides the support for the CFB (cipher feedback) mode, the OFB (output feedback) mode, and the CTR (counter) mode. In addition, a number of primitives are provided to facilitate the support for the CMAC (cipher-based message authentication code) mode, the CCM (counter with cipher block chaining message authentication code) mode, the GCM (Galois/counter mode), and the XTS mode.

As depicted in Figure 23-12 on page 475, the message-security-assist extension 4 requires the message-security-assist extension 3 as a prerequisite, because it uses the two sets of registers by MSA-X3.

## **23.2.4 RACF field support for CPACF-protected keys**

There are new capabilities in RACF and the CSFKEYS profiles to restrict which secure keys can be used as protected keys. By default, all secure keys are considered SYMCPACFWRAP(NO), which means they are not eligible to be used as protected keys. The security administrator must define a CSFKEYS profile for the secure key and specify SYMCPACFWRAP(YES) in the ICSF segment before it can be used as a protected key.

This support provides a new attribute in the ICSF (SYMCPASSISTWRAP(YES/NO)) segment to allow/disallow CPACF wrapping of secure keys stored in the CKDS. This is an update to the RACF ICSF segment in support of CPACF protected key enhancement, CPACF supports in z/OS V1R12.

The ICSF HPCK aims to do the following:

- ▶ Provide a “secure key” high performance bulk encryption solution, where “clear key” material is not present in z/OS host-addressable memory or the ICSF CKDS.
- ▶ Provide a solution that combines the high performance characteristic of clear key and close to the high security characteristic of secure key.

The RACF support for this includes adding a new field to the ICSF segment to indicate whether or not a secure key (that is, a key wrapped under the Symmetric, DES or AES master key) stored in the CKDS can be used as input to the CSNBSYE, CSNBSYD, CSNBSMG, or CSNBSMV APIs and be rewrapped with the CPACF wrapping key.

The ICSF segment applies to profiles in the CSFKEYS/GCSFKEYS and XCSFKEY/GXCSFKEY classes, which protect symmetric and asymmetric keys stored in the ICSF CKDS and PKDS, respectively.

## CPACF-protected key other externals

A new ICSF SMF record is created when an encrypted key is wrapped under the CPACF wrapping key.

### 23.2.5 Summary of CPACF protected key

CPACF protected key is an enhancement that provides an “encrypted key” high performance bulk encryption solution, where “clear key” material is not present in z/OS host-addressable memory or the ICSF CKDS.

Before CPACF-protected key enhancement and its support in z/OS V1R12, the two main encryption solutions provided by z/OS are either high performance or high security.

Clients must choose the solution that best fits their needs. As industry standards and regulations continue to dictate more rigorous security controls, this choice has become increasingly difficult. Table 23-5 summarizes the attributes of the two main symmetric encryption solutions offered on z/OS today and along with z/OS V1R12 on a z10 GA3 System z.

The right-most column of Table 23-5 shows that, with CPACF protected key enhancement, CPACF provides a solution that combines the high performance characteristic of clear key and close to the high security characteristic of encrypted key.

The solution consists of an architecture utilizing the CPACF, the crypto coprocessor (CEX3C) and ICSF to implement a protocol to serve CKDS CCA MK wrapped key material to CPACF instructions without the key material appearing in the clear in z/OS memory.

Table 23-5 Symmetric encryption solutions

Attribute	Clear key	Encrypted key	High perform. encrypted key CPACF
Host storage key wrapping	<b>None</b> - key is visible in the clear in the OS and appl. storage.	<b>CCA Master key</b> - key is never visible in the clear outside the tamper-resistant hdw boundary.	<b>CCA-Master key</b> - key is never visible in the clear outside the tamper resistant hdw boundary
Key store key wrapping	<b>None</b> - key is visible in the clear in the key store.	<b>CCA Master key</b> - key is never visible in the clear outside the tamper-resistant hdw boundary.	<b>CCA Master key</b> - key is never visible in the clear outside the tamper resistant hdw boundary
Key store	CKDS or appl. key file	CKDS or appl. key file	CKDS
Encryption engine	CPACF or software	Crypto express coprocessor	CPACF and CEX3C
Encryption algorithms	DES, TDES, AES	DES, TDES, and AES	DES, TDES, and AES
Benefits	High performance	High security	High performance High security





## 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 whether 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 next two checks became available with z/OS V1R11. The chapter discusses the highlights of PFA and explains these remote checks in more detail. The following topics are covered:

- ▶ Predictive Failure Analysis overview
- ▶ Support in z/OS V1R12
- ▶ PFA infrastructure
- ▶ PFA installation

## 24.1 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.

### 24.1.1 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 generate a system failure that cannot be isolated to a failing component or subcomponent.

There are three general categories of software-detected system failures:

- |                       |   |
|-----------------------|---|
| <b>Masked failure</b> | This is a system failure that is detected by the software and corrected by the software.  |
| <b>Hard failure</b>   | This failure occurs when the software fails completely, quickly, and cleanly, for example when an operating system kills a process.   |
| <b>Soft failure</b>   | This failure is caused by abnormal behavior. A system failure caused by abnormal behavior is defined as unexpected, unusual, or abnormal behavior that 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:

- |                               |   |
|-------------------------------|---|
| <b>Damaged systems</b>        | These are systems suffering recurring or recursive errors caused by software defects.   |
| <b>Serialization problems</b> | These problems can be caused by incorrect priority assignments, by classic deadlocks, or by the resource owner terminating.                             |
| <b>Resource exhaustion</b>    | 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. |
| <b>Indeterminate states</b>   | 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.

## 24.2 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:

- ▶ Some metrics require data for the entire system to be tracked:
  - Exhaustion of common storage for the entire system
  - LOGREC arrivals for the entire system grouped by key
- ▶ Some metrics call for tracking only persistent address spaces:
  - Persistent address spaces: those that start within the first hour after IPL.
  - For example, track frames and slot usage to detect potential virtual storage leaks in persistent address spaces.
- ▶ Some 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)

## 24.3 Using steps to get the most out of PFA

Following are the steps recommended to be taken in order to get the most out of PFA:

- ▶ Reduce the number of false positives.

With z/OS V1R12, 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 that 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 jobs or address spaces from analysis. Supervised learning applies to the following checks:

- “PFA\_LOGREC\_ARRIVAL\_RATE” on page 91
- “PFA\_FRAMES\_AND\_SLOTS\_USAGE” on page 98
- “PFA\_MESSAGE\_ARRIVAL\_RATE” on page 103
- “PFA\_SMF\_ARRIVAL\_RATE” on page 111.

See “Reduce the number of false positives” on page 482 and “Supervised learning support” on page 490.

- ▶ Eliminate jobs causing false positives.

With z/OS V1R12, 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 during processing.

See “Eliminate jobs causing false positives” on page 483.

- ▶ Automate the PFA IBM Health Checker for z/OS exceptions.

Beginning with z/OS V1R12, PFA can use the single ini file for all checks in the /etc/PFA directory, which means you only have to update and maintain one ini file. If you prefer that a specific check uses a different level of Java than what is specified in the /etc/PFA/ini directory, 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 a different level of Java for the PFA\_MESSAGE\_ARRIVAL\_RATE check.

- ▶ Use the check report in SDSF to aid the investigation.

PFA creates a report output using the IBM Health Checker for z/OS and displays it with the z/OS System Display and Search Facility (SDSF) and the message buffer.

- ▶ 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. See “Runtime Diagnostics with z/OS V1R12” on page 485.

### 24.3.1 Reduce the number of false positives

In order to reduce the number of false positives:

- ▶ Use the default configuration parameters that were 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 would cause a damaged or hung system likely show very erratic behavior such that the problem may 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 different 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.

### 24.3.2 Eliminate jobs causing false positives

False positives have to be quickly eliminated during the investigation; they are:

- ▶ 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.
- ▶ With z/OS V1R12, new support is 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 the check.
    - It supports wildcards.
    - Job name and system must match. The rest of the 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.
```

### 24.3.3 Automate the PFA IBM Health Checker for z/OS exceptions

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. 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 (future) behavior of the z/OS images, compares the actual behavior with the expected behavior, and if the behavior is abnormal, 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. PFA 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:

- ▶ The simplest way is to add exception messages to an existing message automation product.
- ▶ A more complex way is to use exception messages and other information to tailor alerts.

#### 24.3.4 PFA and IBM Health Checker for z/OS

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 - This check is new in z/OS V1R12.

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.

##### **PFA\_COMMON\_STORAGE\_USAGE health check**

If PFA detects that there is a potential for the exhaustion of common storage, PFA 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 is shown in Figure 24-1 on page 485. Reports are generated for all PFA checks when everything is normal as well as when an exception occurs.

The PFA\_COMMON\_STORAGE\_USAGE check looks to see whether there is a potential for storage to be exhausted in the upcoming predictive failure analysis (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

```

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

Storage Location  Current Usage in Kilobytes  Prediction in Kilobytes  Capacity When Predicted in Kilobytes  Percentage of Current 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 Name          Storage Location  Current Usage in Kilobytes  Predicted Usage 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 24-1 Common storage usage prediction report

### 24.3.5 Runtime Diagnostics with z/OS V1R12

In z/OS V1R12, Runtime Diagnostics is 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 process 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 usage
- ▶ Looking for an address space that might be in a loop
- ▶ Evaluating local lock conditions

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.

## 24.4 PFA with z/OS V1R10 and V1R11

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.

Indeed, 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 PFS 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 as well as ESQA and ECSA for above-the-line common storage.

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

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.

The message arrival rate check detects damaged address spaces or a damaged LPAR by tracking WTOs and WTORs normalized by processor across time ranges:

- ▶ Chatty, persistent address spaces
- ▶ Non-chatty, persistent address spaces
- ▶ Non-persistent address spaces
- ▶ Total system rate (“chatty” + other persistent + non-persistent)

In summary, two checks were provided to detect recurring failures; they became available with z/OS V1R10:

- ▶ Failures - LOGREC arrival rate (the dump arrival rate)
- ▶ Common storage

Two new checks, designed to detect exhaustion of storage or virtual storage leaks that generate abnormal excessive usage, are added with z/OS V1R11:

- ▶ Storage used by persistent address space
- ▶ Messages - message arrival count per processor

Predictive Failure Analysis shipped with z/OS V1R11 first gave z/OS the capability to detect a damaged z/OS image or address space due to the exhaustion of common resources or recurring errors.

## 24.5 PFA with z/OS V1R12 enhancements

In z/OS V1R12, PFA provides three enhancements:

- ▶ A new PFA check called the SMF arrival rate check. This check is an addition to the other checks that detect a damaged system. It has some similarities to PFA message arrival rate check provided in z/OS V1R11.
- ▶ A supervised learning support is provided as a direct request by many installations. In z/OS V1R11, all of the checks have had “unsupervised” learning. However, it was discovered that there are certain cases where certain address spaces are erratic, such as during development and testing, and these address spaces should be excluded from the checks’ results. This supervised learning support allows you to specify certain jobs to be excluded.
- ▶ Improvements of serviceability and performance for PFA. Some of these changes improve performance while others improve functionality or allow additional configuration.

### **PFA detection of damaged systems**

Figure 24-2 on page 488 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.

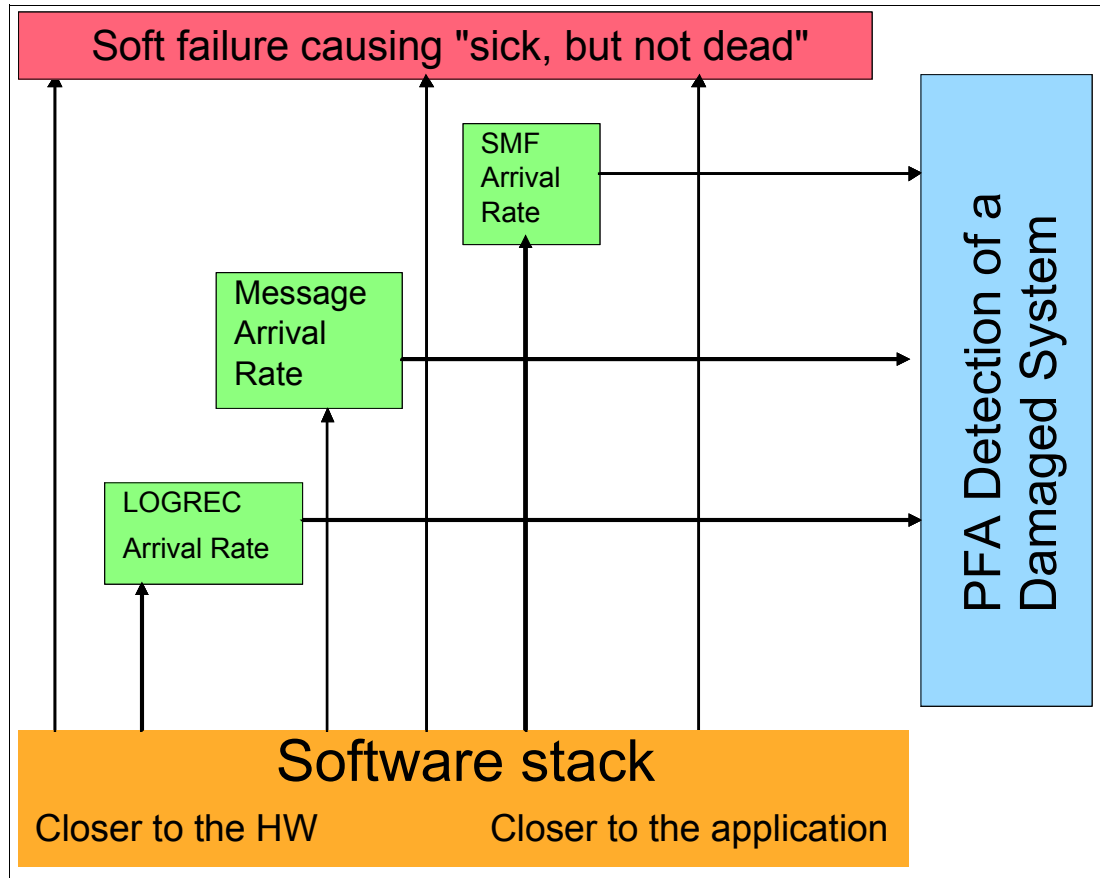


Figure 24-2 How PFA detects soft failures: a layered approach

Some of the metrics that PFA uses in its analysis are closer to the hardware and some are closer to the application. It is easier for PFA to detect an error the closer the metric is to the hardware. On the other hand, the closer the metric is to the application, the harder it is to detect an error.

For instance, as depicted in Figure 24-2:

- ▶ 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 detects 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 so alerted by PFA prior to the soft failure occurring.

### 24.5.1 SMF arrival rate check

In z/OS V1R12, PFA adds one additional check (the SMF arrival rate check), to detect a damaged system so that we now have two checks that detect resource exhaustion and three checks that detect a damaged system. These checks detect errors at different layers of the software stack and complement each other such 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 greater 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. This check accumulates the SMF arrivals in a collection interval and normalizes it by the processor 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 different 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 detects this change and automatically deletes the previously collected data and re-enters the warm-up phase to detect which jobs to track.

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 the jobs that had the highest arrivals after a 6-hour warm-up phase that begins an hour after IPL or when PFA starts, whichever is later. If PFA had not previously been running or 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. And, 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 should help in determining 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:

- ▶ A STDDEV parameter
- ▶ An EXCEPTIONMIN parameter

If exceptions are being produced for very 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.

## 24.5.2 Supervised learning support

PFA provides both the ability to do supervised and unsupervised learning to determine what is abnormal behavior. In general, PFA is designed to require the minimum customer configuration, and therefore has always used unsupervised learning. However, there are some address spaces whose behavior is not predictable for certain checks and there are scenarios such as combined test and production systems where PFA is unable to accurately determine abnormal behavior. In those 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 you 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.

Starting in z/OS V1R12, a new configuration option is available for all checks except the PFA\_COMMON\_STORAGE\_USAGE check. PFA now allows an EXCLUDED\_JOBS file that must exist in the /config subdirectory of the check's directory. All address spaces 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 24-3.

```
/u/pfauser/PFA_MESSAGE_ARRIVAL_RATE/config/EXCLUDED_JOBS
```

Figure 24-3 EXCLUDED\_JOBS file placement

### Using the PFA operator commands

The EXCLUDED\_JOBS file is always read and processed when PFA starts. In order 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 wildcard in both the job name and system name
- ▶ If PFA is running, issue the PFA **modify** 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:

```
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 PFA **modify** update command in order for the file to be reread.

The display option of the PFA **modify** command has also been enhanced to display the list of jobs being excluded for each check

### 24.5.3 Miscellaneous improvements

The common storage usage check has been enhanced to detect common storage exhaustion on a more granular basis. Whereas in z/OS V1R10 and z/OS V1R11 only the two storage locations “above the line” and “below the line” were tracked:

- ▶ Below the line: SQA + CSA
- ▶ Above the line: eSQA + eCSA

Starting in z/OS V1R12, the following storage locations have been separated, and the granularity has been increased:

- ▶ SQA
- ▶ CSA
- ▶ eSQA
- ▶ eCSA

In addition, it was reported that in z/OS V1R10 and z/OS V1R11, the CSA check could have high processor 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 processor utilization do not occur during collection or modeling. The only time that the extra processing is done that caused the high processor utilization is when an exception occurs or at “run check” time when the debug parameter is on and the check has recently modeled.

Some of the enhancements made to the checks to improve the modeling and comparison algorithms are as follows:

- ▶ 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 in order for an exception to occur. The default value is 25. Therefore, installations who have very 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 check 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 and so that 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 so as not to skew the predictions for LOGRECs generated during shutdown and IPL.

- ▶ 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.
- ▶ Improved performance by reducing the number of model requests when the system is stable.

## 24.6 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 24-4 shows a PFA check summary.

The syntax of the PFA `modify` command is very 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 24-4 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 you 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 is displayed in the following five figures; Figure 24-5, Figure 24-5, Figure 24-7 on page 494, Figure 24-8 on page 494, and Figure 24-9 on page 495.

The figures show a single command response, with each figure displaying one of the five checks available for PFA.

```
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 24-5 PFA\_COMMON\_STORAGE\_USAGE 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 24-6 PFA\_LOGREC\_ARRIVAL\_RATE 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 24-7 PFA\_FRAMES\_AND\_SLOTS\_USAGE 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 24-8 PFA\_MESSAGE\_ARRIVAL\_RATE check output

```

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 24-9 PFS\_SMF\_ARRIVAL\_RATE check detail

### 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 is the number of PFA checks that are ACTIVE(ENABLED) in Health Checker as shown in Figure 24-10.

```

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 24-10 PFA checks status

### 24.6.1 SDSF support for PFA

After PFA has been started, the SDSF Health Checker panel (CK) displays the PFA checks, as shown in Figure 24-11 on page 496.

```

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_OPERCMD5_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 24-11 SDSF CK panel showing the PFA checks

**Browsing check output**

To browse the output of a check, Enter an S, as shown in Figure 24-11, to display the common storage usage check. The check output is displayed in Figure 24-12 on page 497.

```

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(IBM PFA,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 24-12 PFA Common storage usage check output

**Note:** The best practice is 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, the next two steps should be taken to analyze the problems:

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 in 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). The files are text files in comma-separated value (.csv) format and 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 24-13 on page 498.

```

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 24-13 Common Storage Usage Prediction report

## 24.7 PFA infrastructure

The PFA infrastructure manages the PFA address space, connects to IBM Health Checker for z/OS (referred to as Health Checker from now on), 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; automate the monitoring. There are no APIs to exploit. The software dependencies for PFA are shown in Figure 24-14 on page 499. They are IBM Health Checker for z/OS, z/OS UNIX file system, and Java 1.4 or later.



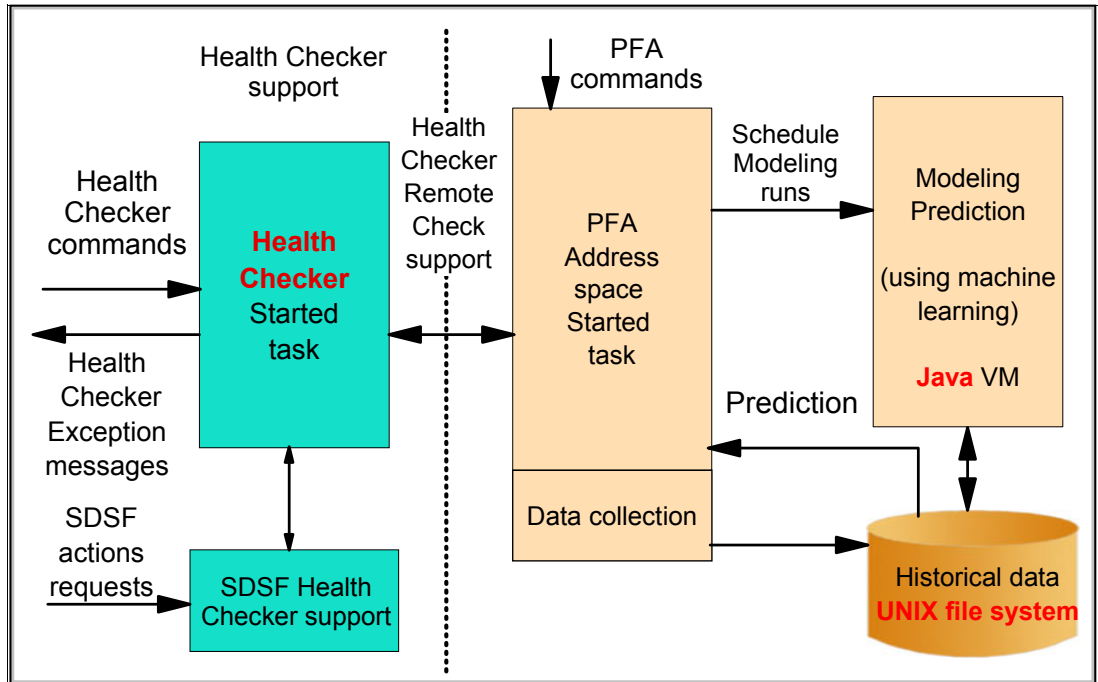


Figure 24-14 PFA infrastructure

### 24.7.1 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 an 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 you. 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 issue 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 done 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:

- ▶ The COLLECTINACTIVE parameter is set to yes by default and 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 you to allow for greater flexibility on a per-system basis.

## **24.7.2 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 very useful to need to input all parameters just 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, the `modify PFA display` command must be used.

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 you or COLLECTINACTIVE is turned on.

Figure 24-15 illustrates the overall PFA processes.

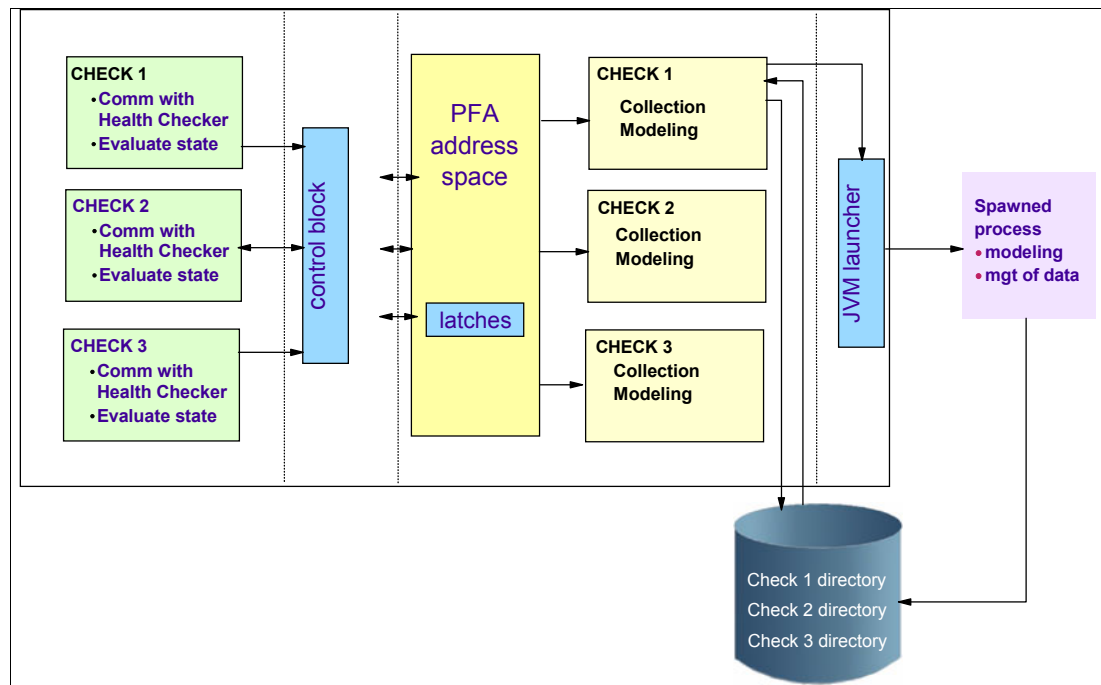


Figure 24-15 Overall PFA process

## 24.8 PFA installation

PFA is shipped with z/OS, starting in V1R11. It contains everything available in the SPE as well as 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 required PTFs 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.

## Create 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.

## Define the PFA started task

Define the PFA started task by creating a RACF profile for 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)`)

## Copy the sample PFA procedure

Copy the sample PFA procedure, AIRPROC, from SYS1.SAMPLIB to the PFA member of 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.

## 24.8.1 Migration considerations for PFA with z/OS V1R12

There is a new parameter when running the PFA install script AIRSHREP.sh, called migrate. It is used to install only new checks that are added by z/OS V1R12; see Figure 24-16 on page 503.

When your installation is migrating from z/OS V1R10 or V1R11 to z/OS V1R12, provide one of the following parameters when running the install script AIRSHREP.sh, from the home directory PFA is using or when using the sample JCL for batch provided in SYS1.SAMPLIB.

If you do not append the migrate parameter or 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 migrate option).

Beginning with z/OS V1R12, PFA can use the single ini file for all checks in the `/etc/PFA` directory, which means you only have to update and maintain one ini file. If you prefer that a specific check use a different level of Java than what is specified in the `/etc/PFA/ini` directory, 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 a different 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 with z/OS V1R12

**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.

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.

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 24-16 shows the new migrate option; the display of the PFA directories is shown in Figure 24-17.

```
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 24-16 PFA installation for z/OS V1R12

```
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 OMVSRP    1120 Jul 26  2006 ITS0-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 24-17 PFA directory structure

## Directory structure with z/OS V1R12

Under UNIX System Services (for example, OMVS as shown in Figure 24-18), you can see the files needed for PFA in directory /usr/lpp/bcp.

```
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 24-18 Directory structure with z/OS V1R12

## Installing PFA in a z/OS UNIX shared file system environment

If your context is 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.

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 24-19, using the UID you assigned to pfauser.

```
cd
ln -s \${SYSNAME}/pfa pfa
```

Figure 24-19 Define symlink

This results in the home directory /systemname/pfa.

3. Create the PFA directory in each of the system directories by entering the following command 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; see Figure 24-20 on page 505.

```
SYS1.PARMLIB(BPXPRM00)
MOUNT FILESYSTEM('OMVSSPT.&SYSNAME..PFA.ZFS') TYPE(ZFS)
MODE(RDWR) MOUNTPOINT('/&SYSNAME./pfa') UNMOUNT
```

Figure 24-20 BPXPRMxx add-ons

## 24.8.2 PFA in z/OS V1R12 supports one or multiple ini files

In z/OS V1R10, it did not seem unreasonable to have one ini file for each check when PFA was first introduced with only two checks, but after z/OS V1R11 when there were four checks, it was obvious that having one ini file per check was becoming cumbersome. Therefore, in z/OS V1R12, we have enhanced PFA to support having one ini file for all checks or having multiple ini files such that individual checks could have their own while other checks could still use the default.

The only reason to have multiple ini files (one per check) is if you want 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 ini file is used. Otherwise, the ini file in /etc/PFA is used.

The /etc/PFA directory is automatically created when z/OS V1R12 is installed. However, if your installation uses steps that would cause the /etc/PFA directory to no longer exist, the new ini file processing cannot be used. If you do not have an /etc/PFA directory after installing, you can create it manually and assign it the proper authorities, as described in the z/OS Problem Management manual.

## 24.8.3 New IBM health checks with z/OS V1R12

The new IBM health checks for z/OS V1R12 exception messages provide for automation of the SMF arrival rate check:

- ▶ AIRH187E
- ▶ AIRH188E
- ▶ AIRH191E
- ▶ AIRH174E

## 24.9 Serviceability information

Originally, PFA was designed to be a black box; you were supposed to start it and forget it. However, it was later decided through client interaction that more information is needed to ensure that PFA is running correctly. Therefore, a comprehensive “modify PFA” command has been built to display status information for the PFA infrastructure as well as 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 have 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 24-21.

```
f hzsproc,delete,check(ibmpfa,PFA_COMMON_STORAGE_USAGE)
```

Figure 24-21 Deleting a check from Health Checker

## 24.9.1 Serviceability improvements 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 will have several log files that are consistently named. There will be a log file for configuration changes, a log file for collection processing, a log file for modeling, and a log file for when the check is run:

CONFIG.LOG, COLLECT.LOG, MODEL.LOG, and RUN.LOG

There will be other log files for each check depending on what other types of processing the check does.

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 was issued for the PFA\_COMMON\_STORAGE\_USAGE check, the new directory with the data and log files would be created in the /u/pfauser/PFA\_COMMON\_STORAGE\_USAGE directory. The name of the new directory will start with EXC\_ and the timestamp of the exception will be concatenated to it.

Once 30 directories have been created, the oldest directory and its contents are deleted prior to creating a new directory.



## Software dependencies

PFA requires Java version 5 or greater.

## Interactions

PFA is built into the operating system. It is looking for a small number of generic events that could cause a soft failure. It is not looking for events or soft failures in specific address spaces unless they could 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 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 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 and provides the information for the soft failure via IBM Health Checker for z/OS, which issues the exception to the console (if so configured) as well as the exception and the report data to the health check output in SDSF.

From the messages provided by PFA via the Health Checker support to the console, other products can be used to further analyze the situation; see Figure 24-22.

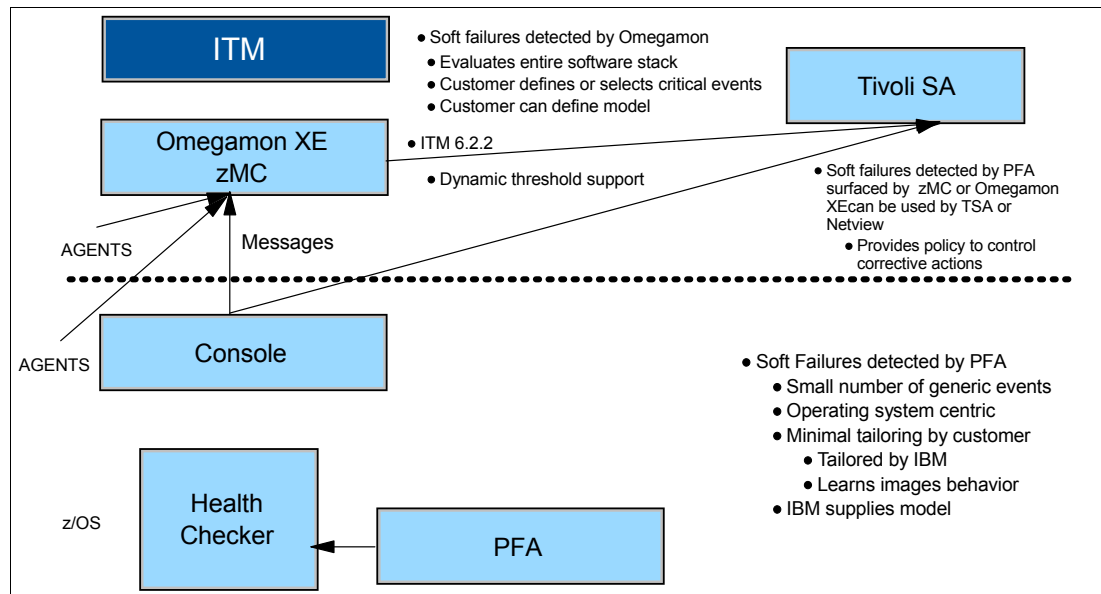


Figure 24-22 How PFA and Tivoli products work together

The OMEGAMON® XE for Management Console will see all health check alerts including the PFA ones. You can build a situation 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.

### 24.9.2 Migration actions to z/OS V1R12

If your installation saves the /etc directory and then restores it after a release upgrade, you must create the /etc/PFA directory manually after the release upgrade since it has now been overlaid. If you omit this step and the /etc/PFA directory does not exist on your system, you cannot take advantage of this new feature.

Consider the following installation and customization items:

- ▶ PFA is also providing 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 permissions are correct.
- ▶ If you are using AIRINJCL to run AIRSHREP.sh, you must update the AIRINJCL file to specify your pfauser'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. In either case, 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, it is recommended that you choose 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 that you should 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.

Once the installation is complete, update the Java configuration in /etc/PFA/ini if it is not correct for your installation. If you still desire multiple ini files, create ini files in the checks' directories and update those as well.



## C language

This chapter describes enhancements in z/OS V1R12 to XL C/C++ and Metal C compiler:

- ▶ XL C/C++
  - Performance
  - C++0x
  - Usability and portability
- ▶ Metal C updates
- ▶ A new memory model is introduced in the z196 that is quite different from previous processors. This new memory model is embodied in the form of new instructions provided in z/OS V1R12 Assembler. This new set of instructions is described in this chapter.

## 25.1 XL C/C++

XL C/C++ in z/OS V1R12 comes with improvements affecting performance, the C++0x standard, as well as usability and portability.

### 25.1.1 Performance via a restrict keyword

Existing source code with function parameters pointing to disjoint sections of memory can be made to execute faster by adding the `restrict` keyword. Modifying a large set of those sources can be expensive, however.

The `restrict` keyword was introduced with the C99 standard, so old standards-compliant code did not use it.

With z/OS V1R12, the compile time option `RESTRICT` can be specified, making all pointer parameters in functions `restrict`-qualified.

With this enhancement, z/OS V1R12 provides the following options:

- ▶ Saving the manual source code modification.
- ▶ Tuning which functions' pointers should be `restrict`-qualified so they can be tailored to specific sets of functions.

Using this new feature in z/OS V1R12, you can:

- ▶ Automatically `restrict`-qualify pointers in functions.
- ▶ Apply to all functions, or to the given list of functions only.

The XL C compiler can use the information given by `restrict` (that the pointers point to objects that are disjoint in memory) to make better optimizations.

Source code can be compiled in an older language level (Ex. `STDC89`) and still have the benefits of `restrict`.

This is invoked by the `RESTRICT` compiler option:

```
xlc -qrestrict myfuncs.c
```

#### IPA object

Inter-procedural analysis (IPA) enables high-level optimization. Isolating potential issues from IPA compile-time optimizations requires a recompile of all objects with `IPA(OBJONLY)`, as follows:

- ▶ 1st compile: `IPA(OBJECT)` and then linking with IPA would get you both compile-time and link-time optimizations.
- ▶ 2nd compile: `IPA(OBJONLY)` and then linking with IPA would get you compile-time optimizations only (to isolate them from IPA-specific link-time optimizations).

The behavior of the `IPA(OBJECT)` option is not consistent with XL compilers on other platforms. Isolating potential issues from IPA compile-time optimizations would require recompiling all objects with `IPA(OBJONLY)`.

As depicted in Figure 25-1 on page 511, this enhancement allows you to merge the previous behavior of `IPA(OBJECT)` and `IPA(OBJONLY)`, and provides:

- ▶ Consistency for users of the XL compilers on other platforms

- ▶ No need for an IPA link to locate the problematic source file due to IPA compile-time optimizations (can simply perform a regular link).

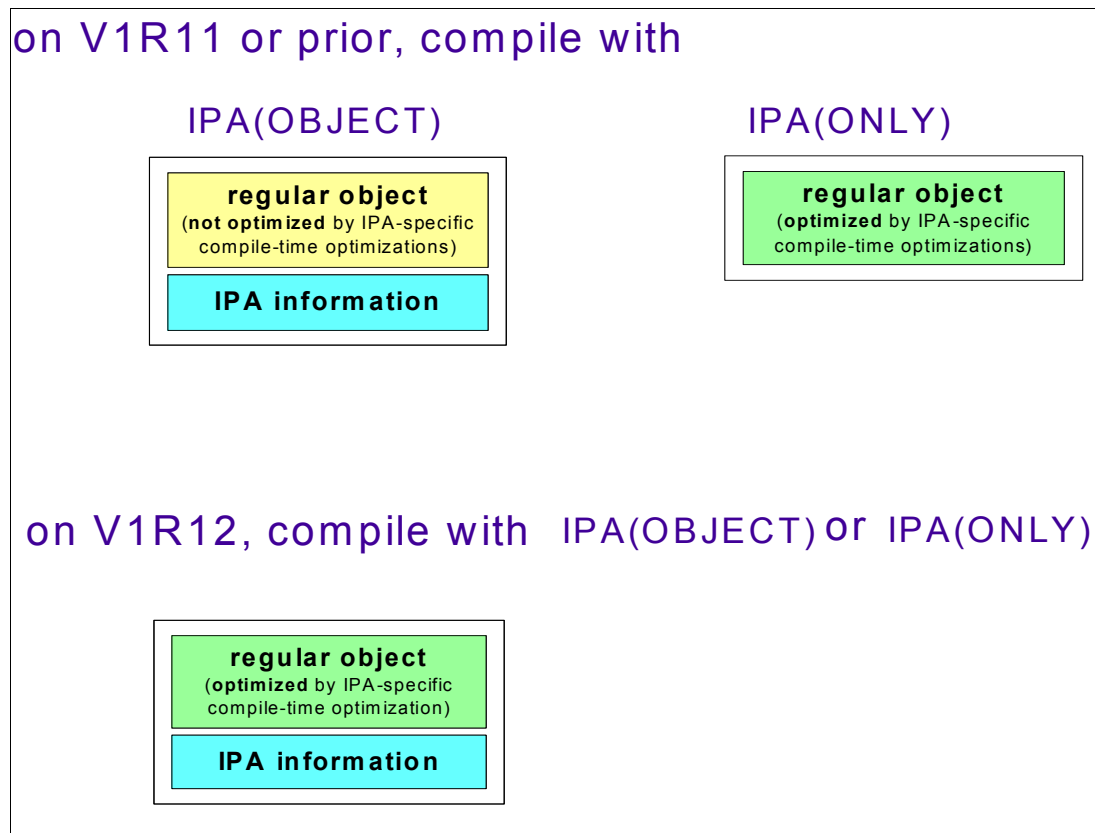


Figure 25-1 IPA object

### Using z/OS V1R11 systems

As shown in Figure 25-2 on page 512, on a z/OS V1R11 system, when doing a compile:

- ▶ With IPA(OBJECT)
  - Regular object does not contain IPA-specific compile-time optimizations.
  - When the object file is fed into the IPA link, IPA link-time optimizations will be performed.
- ▶ With IPA(OBJONLY)
  - Regular object is optimized by IPA-specific compile-time optimizations.
  - When the object file is fed into the IPA link, no IPA link-time optimizations will be performed, because no IPA information was written to the object file.

On V1R12, when compiling with:

- ▶ IPA(OBJECT)
  - Regular object is optimized by IPA-specific compile-time optimizations.
  - When the object file is fed into the IPA link, IPA link-time optimizations will be performed.
- ▶ IPA(OBJONLY)
  - Recognized silently for backward compatibility.
  - Treated as a synonym for IPA(OBJECT).

## Migration and coexistence

To get the previous level of optimizations in the executable program, when compiling in V1R11 with certain options, do in V1R12 as specified in Figure 25-2.

V1R11		IPA specific optimizations on executable program	V1R12 migration	
Compiled with	Linked with		Compiled with	Linked with
IPA(OBJECT)	IPA	Compile-time and link-time	IPA(OBJECT)	IPA
IPA(OBJECT)	Non-IPA	None	Remove IPA	Non-IPA
IPA(OBJONLY)	IPA	Compile-time	IPA(OBJECT)	Non-IPA
IPA(OBJONLY)	Non-IPA	Compile-time	IPA(OBJECT)	Non-IPA

Figure 25-2 Migration and coexistence

## Additional improvements

Additional improvements include:

- ▶ CCNEIPA1 (31-bit) and CCNQIPA2 (64-bit) modules have been merged into a single 64-bit CCNQIPA module. This might make it possible to reduce the amount of memory required in job cards.
- ▶ The new environment variable `_CCN_IPA_WORK_SPACE` can significantly reduce storage demands when linking with IPA.

## Reusable PDF files

Previously, both stages of Profile-Directed Feedback (PDF) had to be done with identical source files and compiler options, or else PDF2 would terminate with the error message shown in Figure 25-3.

```
Profiling data matching that from the pdf1 phase could not be found for
function ...
```

Figure 25-3 PDF2 error message

At PDF2, this enhancement tolerates and warns about source differences and different compiler options from PDF1. It reduces the burden of redoing the PDF instrumentation step when only incremental changes are made to an application. In particular, this will enable clients doing daily PDF builds to instrument their application on a less frequent basis.

## PLO built-in function

The zSeries Perform Locked Operation (PLO) is a powerful hardware facility for atomic read-modify-write operations.

The PERFORM LOCKED OPERATION instruction can be used in a multiprogramming or multiprocessing environment to perform compare, load, compare and swap, and store operations on two or more discontinuous locations that can be words or doublewords.

The operations are performed as an atomic set of operations under the control of a lock that is held only for the duration of the execution of a single PERFORM LOCKED OPERATION instruction, as opposed to across the execution of multiple instructions.

Since lock contention is resolved by the processor and is very brief, the program need not include a method for dealing with the case when the lock to be used is held by a program being executed by another processor. Also, there need be no concern that the program may be interrupted while it holds a lock, since PERFORM LOCKED OPERATION will complete its operation and release its lock before an interruption can occur.

PERFORM LOCKED OPERATION can be thought of as performing concurrent interlocked updates of multiple operands. However, the instruction does not actually perform any interlocked update, and a serially reusable resource cannot be updated predictably through the use of both PERFORM LOCKED OPERATION and conditional swapping instructions (CS and CDS).

However, before z/OS V1R12, it cannot be exploited directly in the compiler generated code.

With z/OS V1R12, the PLO instruction is made available by the XL C/C++ compiler makes as part of the built-in functions that can be used in the C/C++ source code.

The PLO instructions can thus be exploited in C/C++ programs.

There are 24 individual built-in functions with the function name in the form of `__plo_XXXX` where XXXX represents the Function Symbols defined in the z/Architecture Principles of Operation.

For example, `__plo_CLGR` is to generate the PLO instruction for function code 2.

For specifying this function, notice that:

- ▶ For operations that require a parameter list, macros and types refer to `builtins.h`.
- ▶ For operations with 64-bit or 128-bit operands, the LP64 option is required.
- ▶ For 128-bit operands, quad-word alignment is required and it has to be managed by the user.

For example, the built-in function `__plo_CLGR` is a PLO instruction for function code 2 enabled in LP64 mode since it operates on 64-bit operands.

In terms of dependencies, these built-in functions require the following:

- ▶ Minimum ARCHITECTURE level of 5 (see Figure 25-4 on page 514)
  - To enable long long date type.
  - To include the `builtins.h` header file.

## 25.1.2 Architecture compiler suboption

With the z196 model (and more specifically 2817-xxx models) the new suboption ARCH(9) is being made available with XL C/C++ of z/OS V1R12. It is an indication for the compiler to produce code that uses instructions available on the 2817-xxx models in z/Architecture mode, as indicated in Figure 25-4 on page 514.

XLC Arch ( )	Hardware facility	Hardw. arch.	Machine models	XLC min level	XLC option	Note
9	high word, interlocked-access, load/store-on-condition, distinct-operands, population-count	z/Arch	z196 2817-xxx	1.12		
8	general instruction extensions facility	z/Arch	z10	1.10		
	AR-mode	31/64		1.9	METAL	no LE Envrt
	decimal floating point	z/Arch	z9-GA3	1.8	DFP	LANGLVL (EXTENDED)
7	extended-immediate facility	z/Arch	z9			x- xlation 3
6	long/displacement facility		z990 z890			x - xlation 2
5	64-bit mode	ESAME	z900 z800	1.2		default for TARGET(z/OS 7) and above
4	long long operations	ESA/390	z900 z800			
3	IEEE (binary) floating-pt		G5-G6			
2	Branch Relative and Halfword Immediate		(G2-G4)			
1	Logical String Assist		G1 9021			
0	Produced code is executable on all models					

Figure 25-4 Overall enhancements of IBM z/OS XL C/C++

### Message severity modification - C only

Users want to be able to fit the compilation environment to their particular standards:

- ▶ Ex.: Warn about or do not allow any non-prototyped functions.
- ▶ Ex.: Do not show warnings regarding unknown preprocessing directives (may be valid on other platforms or compilers).

XL C in z/OS V1R12 allows for changing the severity of some of the diagnostic messages issued by the compiler. This gives you the possibility of customizing the diagnostic message levels to the coding standards.

Using this feature, you can change the default severity of some diagnostic messages that are issued by the compiler.

Making changes to message severity by reducing an error to a warning is not allowed because this would lead to the expectation of a valid object being produced.

This allows different classification of messages for different lines or stages of development, for example:

- ▶ Development line vs. production line
- ▶ Code drop phase vs. stabilization phase



## Example of its usage

Figure 25-5 shows an example of modifying message severity.

```
LAFITTE @ SC74:/u/lafitte/jll-zfs>more proto.c

int main(void) {
    int * rc = (int*)malloc(sizeof(int));
    foo(rc);
return *rc; }

int foo(int * rc) {
    *rc = 55;
    return 0; }
^proto.c" (EOF)

LAFITTE @ SC74:/u/lafitte/jll-zfs>xlc proto.c -qinfo=pro -qflag=i
INFORMATIONAL CCN3304 ./proto.c:2    No function prototype given for "malloc".
INFORMATIONAL CCN3304 ./proto.c:3    No function prototype given for "foo".
LAFITTE @ SC74:/u/lafitte/jll-zfs>xlc proto.c -qinfo=pro -qflag=i -qseverity=w=C
CN3304
WARNING CCN3304 ./proto.c:2    No function prototype given for "malloc".
WARNING CCN3304 ./proto.c:3    No function prototype given for "foo".
LAFITTE @ SC74:/u/lafitte/jll-zfs>
```

Figure 25-5 Message severity modification

## Improved aliasing diagnostics

ANSI aliasing rule violations are generally hard to find manually. Some higher optimizations indeed depend on strict ANSI aliasing rules being followed.

In order to address this point, XL C/C++ in the z/OS V1R12 compiler finds some of these ANSI aliasing violations for you.

This has the following benefits:

- ▶ Issuing more diagnostics if the compiler detects some ANSI aliasing violations in the source code.
- ▶ Allowing you to fix the problem and compile at higher optimization levels.

As shown in Figure 25-6 on page 516, using this feature you can:

- ▶ Find some existing ANSI aliasing violations in existing or new code.
- ▶ Determine the potential locations of the ANSI aliasing violations and where they may have impact.

Globally speaking, it provides the following:

- ▶ Allows finding potential problem areas in the source code logic.
- ▶ Helps find potential invalid usage of pointers.

```

// t.C
int main(int argc) {
    int i = argc;
    short *sp = (short*)&i;
    *sp = 1; // line 4
    return 55;
}
^aliasdiag.c" (EOF)

LAFITTE @ SC74:/u/lafitte/jll-zfs>xlc aliasdiag.c -qinfo=als -qflag=i
INFORMATIONAL CCN4393 ./aliasdiag.c:5      Dereference may not conform to the cur
rent aliasing rules.
INFORMATIONAL CCN4394 ./aliasdiag.c:5      The dereferenced expression has type "
short". "sp" may point to "i" which has incompatible type "int".
INFORMATIONAL CCN4395 ./aliasdiag.c:5      Check pointer assignment at line 4 col
umn 13 of ./aliasdiag.c.
LAFITTE @ SC74:/u/lafitte/jll-zfs>

```

Figure 25-6 Improved aliasing diagnostics

### 25.1.3 C++0x standards

In order to follow the evolution of the C++ standard, different features of the next standard level, named C++0x, have already started to be incorporated in XL C/C++ of z/OS V1R12.

All C++0x draft papers mentioned in the description of individual features are superseded by Working Draft, Standard for Programming Language C++ scheduled to be ratified in 2012.

Most C++0x features allow the following options:

- ▶ Have the individual `langlvl` suboption and macro described in the speaker notes.
- ▶ There is also a C++ language level suboption, `extended0x`, which enables all of them.
- ▶ Using the `-qwarn0x` compiler option which can assist in migration from the C++98 standard to the C++0x standard.

#### Variadic templates

There is a growing need to have generic containers with any number of parameters.

Variadic templates are the solution in C++0x to address this requirement, as follows:

- ▶ Type and non-type template parameters can now be specified as type and non-type parameter packs.
- ▶ Template parameter packs can be instantiated with 0 or more arguments.
- ▶ In a function template, the template parameter pack can be used to create a function parameter pack.

This is especially beneficial to library developers because generic containers do not need to be designed with a hard-coded number of template parameters. This allows easier

implementation of generic containers such as tuples or templates to represent function objects.

It has been decided to implement this enhancement to XL C/C++ in z/OS V1R12 as one of the first C++0x features because it will likely be used by library developers in order to replace some complex tricks currently done with macros to implement these types of containers. Also, it is hoped that in the future, the XL C/C++ library might be improved by using variadic templates. Newer versions of g++ headers are already using variadic templates.

To enable variadic templates, use the option `-qlanglvl=VariadicTemplates`, which is enabled by default under `-qlanglvl=extended0x`. When the option is enabled, the macro `_IBM_CPP_VARIADIC_TEMPLATES` is defined.

**Note:** Template parameter packs are specified in the C++ standard, but they have not been implemented in this release.

The example in Figure 25-7 shows the use of the type and non-type parameter packs and illustrates that these can be instantiated with 0 or more arguments.

```
// template parameter pack <class... A>
template <class...A> struct container{};

container<> a1;
container<int> a2;
container<int,char,float> a3;

template <bool...B> struct container1{};

container1<> b1;
container1<true> b2;
container1<true,false,true,true> b3;
```

Figure 25-7 *Template class*

Note in Figure 25-8 on page 518 the use of `sizeof`, which will return the size of a function parameter pack or pack expansion.

```

#include <cassert>

template <class A, class B> struct container{};

// arg is a function parameter pack
template <class... C, class... D> int func( container<C,D>... arg ){
    // sizeof... returns the size of the function parameter pack
    assert( sizeof...(arg) == sizeof...(container<C,D>));
    return sizeof...(container<C,D>);
}

struct a1{}; struct a2{};

int main(void){
    container<a1,a2> a;
    assert( func( a,a,a,a,a ) == 5);
    return 0;
}

```

Figure 25-8 Template function

The example in Figure 25-9 shows one way to access the members of a pack. The first version of `func` is matched when there are 0 arguments in the call; the second version is matched when there are 1 or more arguments in the call.

```

#include <iostream>
using namespace std;

template <class X> struct test{
    test(X data){      cout << "ctor test: " << data <<endl;   }
};

template <class... A> void func(A... arg1){  test<int> data(99);}
template <class head, class... tail>
void func(head arg1, tail... arg2){
    test<head> local(arg1);  // arg1 is the first argument
    func(arg2...);         // arg2... is a pack expansion containing
                           // all the remaining args
}

int main(void){
    func();  // match the first version of func
    func(1); // match the second version of func
    func(1,2,3,4); // match the second version of func
    return 0;
}

```

Figure 25-9 Pack expansion and access to pack members

## Delegating constructors

Common initializations in multiple constructors of the same class to be concentrated in one place should provide a more robust, maintainable schema.

Currently, this is partially alleviated in existing C++ programs by using assignment instead of initialization.

In that perspective, C++0x provides delegating constructors, which makes possible to concentrate common initializations in one constructor that can be used as the target constructor of one or more delegating constructors.

This enhances program readability and maintainability, as well as reduces code size.

## Enabling delegating constructors

To enable delegating constructors, use the option `-qianglvl=delegatingctors`, which is enabled by default under `-qianglvl=extended0x`. When the option is enabled, the macro `__IBMCPP_DELEGATING_CTORS` is defined as the preprocessing number 1; otherwise the macro is left undefined. In both cases, the macro is protected.

### DELEGATINGCTORS | NODELEGATINGCTORS

(C++0x only) This option controls whether the delegating constructors feature is enabled. When `LANG(DELEGATINGCTORS)` is specified, you can concentrate common initializations and post initializations in one constructor, which improves the readability and maintainability of the program. The default is `LANG(NODELEGATINGCTORS)`.

**Note:** This feature provides the ability to concentrate common initializations and post-initialization handling to improve program readability and maintainability.

- ▶ When the delegating constructors feature is enabled, the `__IBMCPP_DELEGATING_CTORS` macro is defined as 1; otherwise, the macro is undefined. In both cases, the macro is protected and a compiler warning is displayed if it is undefined or redefined.
- ▶ `LANGLVL(DELEGATINGCTORS)` is implied in the group option of `LANGLVL(EXTENDED0X)`. You can also use this group option to enable the delegating constructors feature.

The example shown in Figure 25-10 has `A(T)` and `A(U)` delegate to `A(T, U)`. This demonstrates a typical use to have common initialization in a single constructor. Note that `const` non-static data member `t` is initialized with an expression involving two parameters and operator `^` in the non-delegating constructor.

```

#include <cstdio>

template <typename T, typename U> struct A {
    const T t;
    const U u;

    static T tdef;
    static U udef;

    A(T t_, U u_) : t(t_ ^ u_), u(u_) { }
    A(T t_) : A(t_, udef) { }
    A(U u_) : A(tdef, u_) { }
};
template <typename T, typename U> T A<T, U>::tdef;
template <typename T, typename U> U A<T, U>::udef;

int main(void) {
    A<unsigned char, unsigned>::tdef = 42u & 0x0F;
    A<unsigned char, unsigned> a(42u & 0xF0);
    std::printf("%d\n", a.t);
    return 0;
}

```

Figure 25-10 Delegating constructors

In the example in Figure 25-10, `tdef` is initialized with `0x0A` in `main`. In delegated constructor `A(U)` parameter `u_` in is initialized parameter `0x20`. A member `t` is initialized in the target constructor with `0x2A` (42), which is the result from the `0x0A ^ 0x20` operation.

## Namespace association

Namespaces offer a mechanism for separating conflicting names from multiple versions of the same library.

Even though members from one namespace can be used in another without qualifiers, they cannot, in all cases, be used as if the adoptive namespace is their original namespace. C++0x introduces namespace association with inline namespace definitions. Members of an inline namespace can then be used as if they were members of another namespace.

The main benefit is for library vendors who can use the same source and object files (including interface headers and library archives) for all implementations.

To enable namespace association with the Inline Namespace Definitions feature, use the option `-qlanglvl=inlinenamespace`, which is enabled by default under `-qlanglvl=extended0x`. When the option is enabled, the macro `__IBMCPP_INLINE_NAMESPACE` will be defined as the preprocessing number 1; otherwise the macro will be left undefined. In both cases, the macro is protected.

This feature supports the following library organization recipe:

- ▶ The separate implementations are each placed into a namespace under a common enclosing namespace (which may be the global namespace). The namespace of an implementation forms the direct source interface to that implementation.
- ▶ Use conditional inclusion in the interface headers to control whether and which one of the implementation namespaces is inline. If any, the implementation namespace that is inline is the one behind the common source interface. You will find the common interface in the

namespace that encloses the implementation namespaces: the members of the namespace behind the common interface will appear to be in the namespace that encloses the implementation namespaces.

- Implement the library using the direct source interface of each implementation. Regardless of which, if any, implementation is behind the common source interface, the binary interface to the separate implementations remains the same

The examples in Figure 25-11 on page 521 and Figure 25-12 on page 522 can be compiled with

```
xIC -qclanglvi=inlinenamespace myFooCaller.C -I./, for which the resulting executable produces 5,
```

or with

```
xIC -qclanglvi=inlinenamespace myFooCaller.C -I./  
-DSOME_LIBRARY_USE_VERSION_2_, for which the resulting executable produces 6.
```

Note that if the call, `foo(intWrap)`, was qualified with one of the inline namespaces, then the you would need to ensure that the explicit specialization is effective.

```
namespace SomeLibrary {  
  
#ifdef SOME_LIBRARY_USE_VERSION_2_  
    inline namespace version_2 { }  
#else  
    inline namespace version_1 { }  
#endif  
  
    namespace version_1 {  
        template <typename T>  
        int foo(T a) { return 1; }  
    }  
    namespace version_2 {  
        template <typename T>  
        int foo(T a) { return 2; }  
    }  
}
```

Figure 25-11 *Foo.h*

```

#include <Foo.h>
#include <iostream>

// Client code
struct MyIntWrapper { int x;};

// specialize SomeLibrary::foo() using the correct version
namespace SomeLibrary {
    template <> int foo(MyIntWrapper a) { return a.x; }
}

int main(void) {
    using namespace SomeLibrary;
    MyIntWrapper intWrap = { 4 };
    std::cout << foo(intWrap) + foo(1.0) << std::endl;
}

```

Figure 25-12 *myFooCaller.C*

## Long long support

The C++0x Standard introduced the concept of a *long long* type for integers that are larger than what can be represented in 4 bytes. IBM compilers had already introduced a proprietary long long type on its own.

The C++0x long long is the only available long long type under the EXTENDED0X language level.

A diagnostic message is issued to alert you when the new C++0x integral promotional rules produce a different result from the previously existing IBM long long type's promotional rules.

The benefit of this is twofold:

- ▶ Conformance to the C++0x Standard
- ▶ The diagnostic informs you when your use of IBM's long long differs from C++0x behavior (so that you can modify your code to become standards conforming).

Since the promotional rules differ between IBM's long long and C++0x's, code shown in Figure 25-13 produces the following informational message:

```

"./t.C", line 3.19: CCN8928 (I) Integral constant "2147483648" has implied type
unsigned long int under the non-C++0x language levels. It has implied type long
long int under C++0x.

```

```

// Compiled with: -Wc,"LANGLVL(EXTENDED0X)" -qflag=I -qinfo=por
int main(void) {
    // LONG_MAX is 2147483647
    long long x = 2147483648;
}

```

Figure 25-13 *Long long example of new promotional rules*

## C99 preprocessor changes

The C99 Standard introduced some preprocessor rules that were not present in C++98. The C++0x Standard "imported" some of these changes to make C++ and C99 more compatible with each other.



To handle that situation, XL C/C++ of z/OS V1R12 uses the `_Pragma` operator:

- ▶ Increased maximum limit for `#line` preprocessor directive
- ▶ New predefined macros
- ▶ Wide and narrow string concatenation

This way, it increases compatibility and easier source code migration between C99 and C++0x.

The `_Pragma` operator is simply for source compatibility and is functionally equivalent to the regular `#pragma` directive.

The statements in Figure 25-14 are 100% functionally equivalent.

```
#pragma comment(copyright, "IBM 2007")
_Pragma("comment(copyright, \"IBM 2007\")")
```

Figure 25-14 Preprocessor changes

The `#line` directive previously could not accept a value above 32,767, but this limit has been increased to 2,147,483,647.

The following macros were added in z/OS V1R12 to provide compatibility with programs written under C99:

- ▶ `__STDC__` - macro defined to 0 for all language levels (that is, it is ALWAYS defined to 0).
- ▶ `__STDC_HOSTED__` - macro is defined to 1 for `LANGlvl=EXTENDED0x`, and undefined otherwise.

Previous to C++0x, two strings could not be concatenated together if they were not both either “wide” or “narrow”.

The resulting strings will default to “wide” but they can be casted back to narrow.

If the program shown in Figure 25-15 is compiled under C++0x, both these string assignments would be in error.

```
int main() {
    char* narrow = (char*) "This string " L"will be narrow";
    wchar_t* wide = "This string " L"will be wide";
}
```

Figure 25-15 Wide and narrow string concatenation

## 25.1.4 Usability and portability

Source and binary portability improvements are provided with XL C/C++ in z/OS V1R12. They are described here.

### The `typeof` keyword

Porting source code from other compilers such as gcc may have compile time errors for keywords that do not exist in XL C compiler.

Such is `__typeof__`, which is harder to type and read than `typeof`.

To help this, XL C in z/OS V1R12 allows `typeof` as a keyword with the same semantics as `__typeof__`.

This is beneficial because it

- ▶ Saves modifying source code manually.
- ▶ Makes porting source code easier.

The support is invoked by the `KEYWORD(typeof)` compiler option. Externally no error message is issued when `typeof` is used in the source code instead of `__typeof__`

This is shown in Figure 25-16.

```
LAFITTE @ SC74:/u/lafitte/jll-zfs>more typede.c

int main(void) {
    int rc = 66;
    typeof(rc) returnValue = 55;
    return returnValue;
}
^typede.c" (EOF)

LAFITTE @ SC74:/u/lafitte/jll-zfs>xlc typede.c
ERROR CCN3277 ./typede.c:3      Syntax error: possible missing ';' or ','?
ERROR CCN3045 ./typede.c:4      Undeclared identifier returnValue.
CCN0793(I) Compilation failed for file ./typede.c.  Object file not created.
LAFITTE @ SC74:/u/lafitte/jll-zfs>xlc -qkeyword=typeof typede.c
LAFITTE @ SC74:/u/lafitte/jll-zfs>
```

Figure 25-16 *typeof* keyword example

## NAMEMANGLING - new suboption

Linkage issues might arise when combining modules built with different compiler versions due to evolution of `NAMEMANGLING(ANSI)`.

XL C in z/OS V1R12 provides new suboption `zOSV1R12_ANSI` to the option `NAMEMANGLING(...)` and to the language construct `#pragma namemangling(...)`.

Thus, the linkage errors caused by any future changes in the mangling scheme can be resolved by recompiling the new modules using the same mangling scheme that was used in the already-compiled modules (for example, `zOSV1R12_ANSI`).

## 25.2 Metal C enhancements

Metal C in z/OS V1R12 provides several new enhancements that are described in the following paragraphs.

### 25.2.1 RENT option

Without the constructed reentrancy support, a Metal C program is forced to be naturally reentrant, that is, external and/or static variables should not be used. This becomes a restriction for adopting Metal C.

z/OS V1R12 enables the C RENT option with the `METAL` option.

As a benefit, modifiable extern and static variables can be used while maintaining the reentrancy of the program. It is now possible to use Metal C to write programs to run in CICS Transaction Server for z/OS, particularly in situations where code would otherwise directly be using Assembler.

The constructed reentrancy consists of a piece of runtime code to dynamically allocate the Writable Static Area (WSA) initialized with the program initialization values before entering function "main".

The WSA storage can be managed by the user-supplied plug-in routines, which are referenced by the prolog or epilog code in function "main". The default is to use the IBM-supplied routines.

New GOFF classes are used to build the WSA storage image and are retrieved by the runtime code for storage allocation and initialization.

The WSA pointer is passed to functions in GPR0. The Metal C RENT support is independent of and different from the Language Environment RENT support. They should not be mixed.

Programs compiled with RENT and NORENT can be mixed as long as the NORENT programs do not call RENT programs.

Figure 25-17 shows an example of how to compile, assemble, and link with RENT.

```
/bin/xlc -qMETAL -qRENT -S a.c  
/bin/as -mgoff a.s  
export _LD_SYSLIB="//'CBC.SCCNOBJ'"  
/bin/ld a.o
```

Figure 25-17 To compile, assemble and link with RENT

Figure 25-18 shows an example of how to supply WSA storage management plug-in routines.

```
GBLC &CCN_WSA_INIT  
GBLC &CCN_WSA_TERM  
&CCN_WSA_INIT SETC 'MYWSAI'  
&CCN_WSA_TERM SETC 'MYWSAT'
```

Figure 25-18 To supply WSA storage management plug-in routines

Additional new global set symbols are made available:

- ▶ &CCN\_MAIN - "1" if function "main"
- ▶ &CCN\_RENT - "1" if compiled with the RENT option

The compiler sets the value of these symbols, just like every other global set symbols set by the compiler, so the user prolog code can reference them to determine the code to be generated in their macro based on the values of these symbols.

For example, if &CCN\_RENT is not true, you need not preserve GPR0 to supply the user WSA initialization routine plug-in, to not include extra stack space for main, and so on.

## 25.3 Memory model

Since the beginning of the 70s and the introduction of the MP support, atomic instructions of the class Compare-and-Swap have been made available. Different sizes of the operand have been provided over the years, and even a kind of Compare-and-Swap on multiple operands called PERFORM LOCKED OPERATION, which can be thought of as performing concurrent interlocked updates of multiple operands.

With the z196 level of hardware, a new set of instructions under the name of Interlocked-Access Facility (bit 45) is provided. It looks to be inheriting the FETCH-and-OP introduced with the IBM RP3 and presents a new memory model:

LOAD AND ADD can be depicted as in Figure 25-19 on page 526.

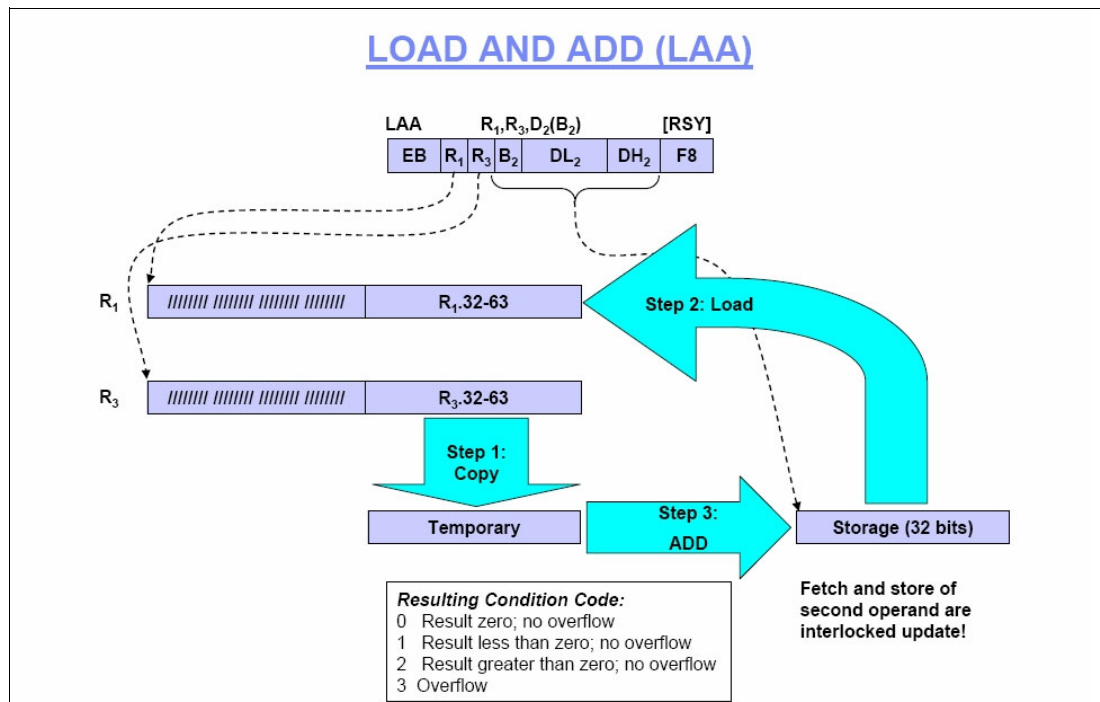


Figure 25-19 LOAD AND ADD

It can, as well, handle 64-bit registers, hence the term G for gigantic, as shown in Figure 25-20.

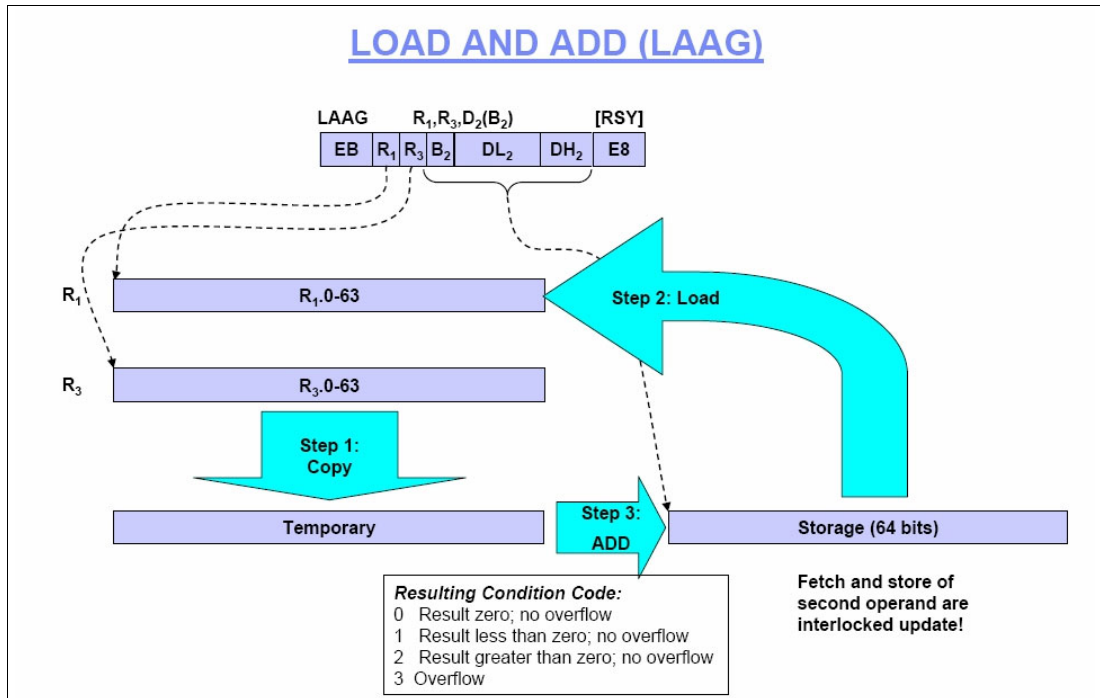


Figure 25-20 LOAD and ADD (gigantic)

LOAD AND ADD LOGICAL (as well as its gigantic equivalent) can be depicted as in Figure 25-21 on page 527.

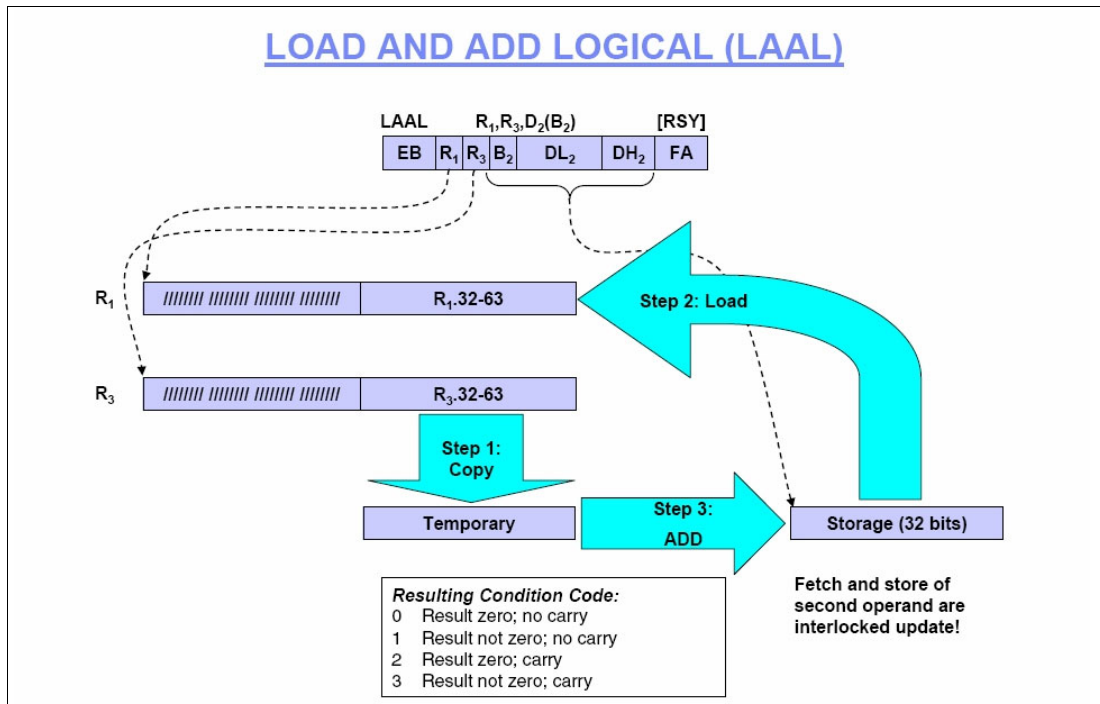


Figure 25-21 LOAD AND ADD LOGICAL

LOAD AND AND (and its gigantic equivalent) can be depicted as in Figure 25-22.

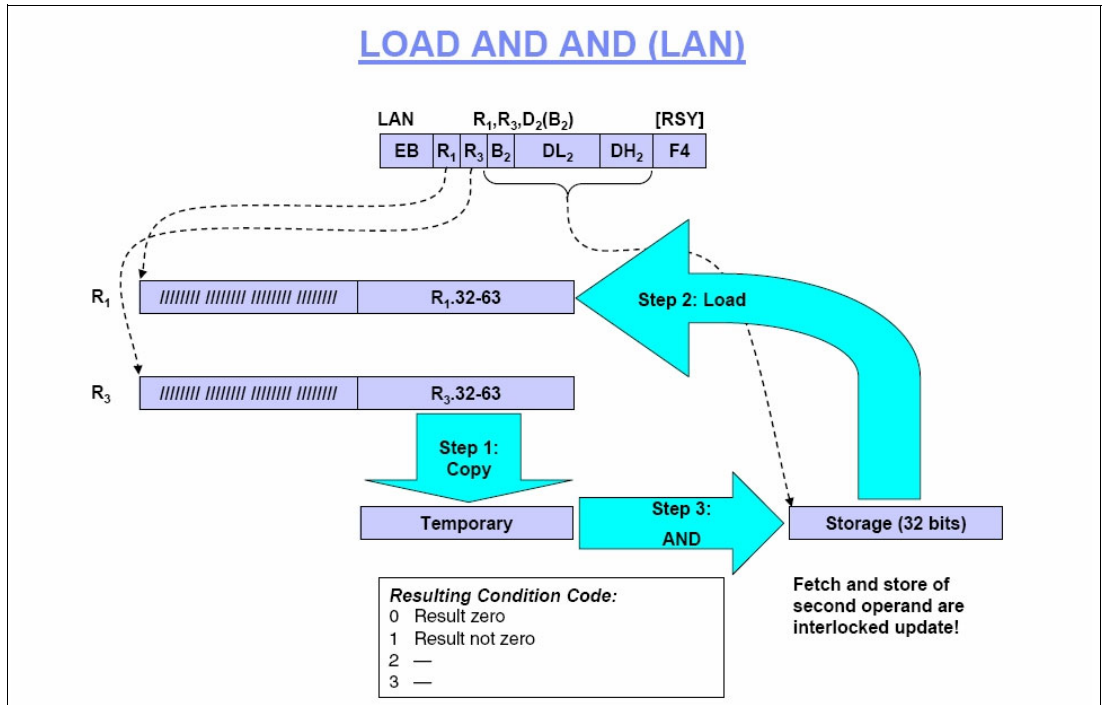


Figure 25-22 LOAD AND AND

LOAD AND OR (and its gigantic equivalent) can be depicted as in Figure 25-23.

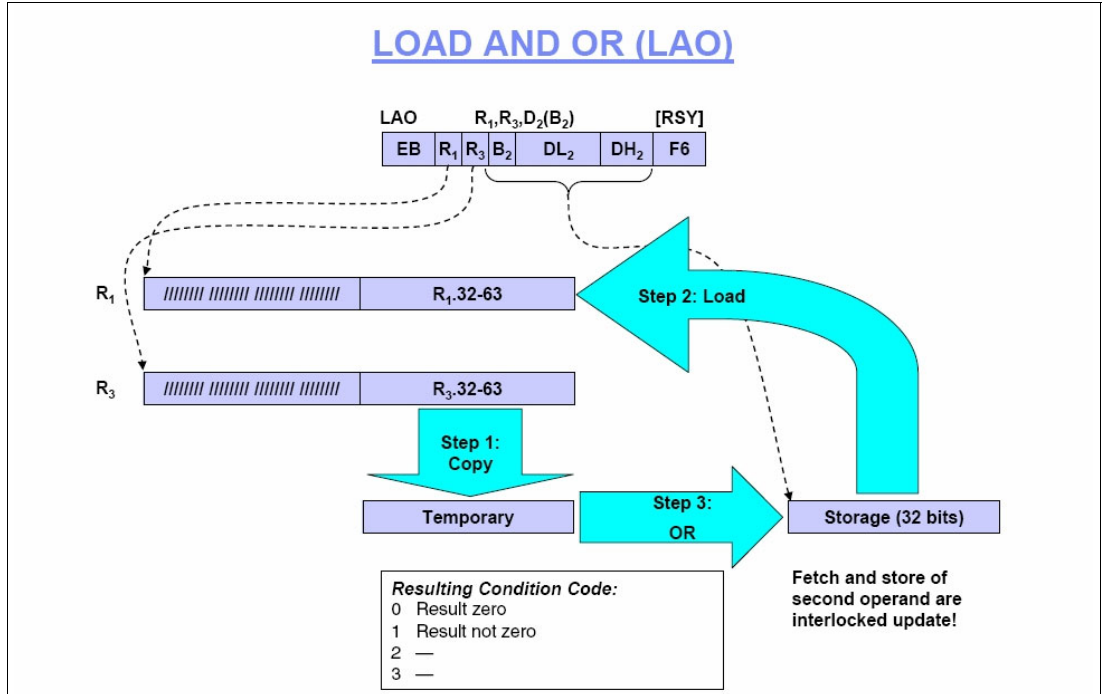


Figure 25-23 LOAD AND OR

LOAD AND EXCLUSIVE OR (and its gigantic equivalent) can be depicted as in Figure 25-24.

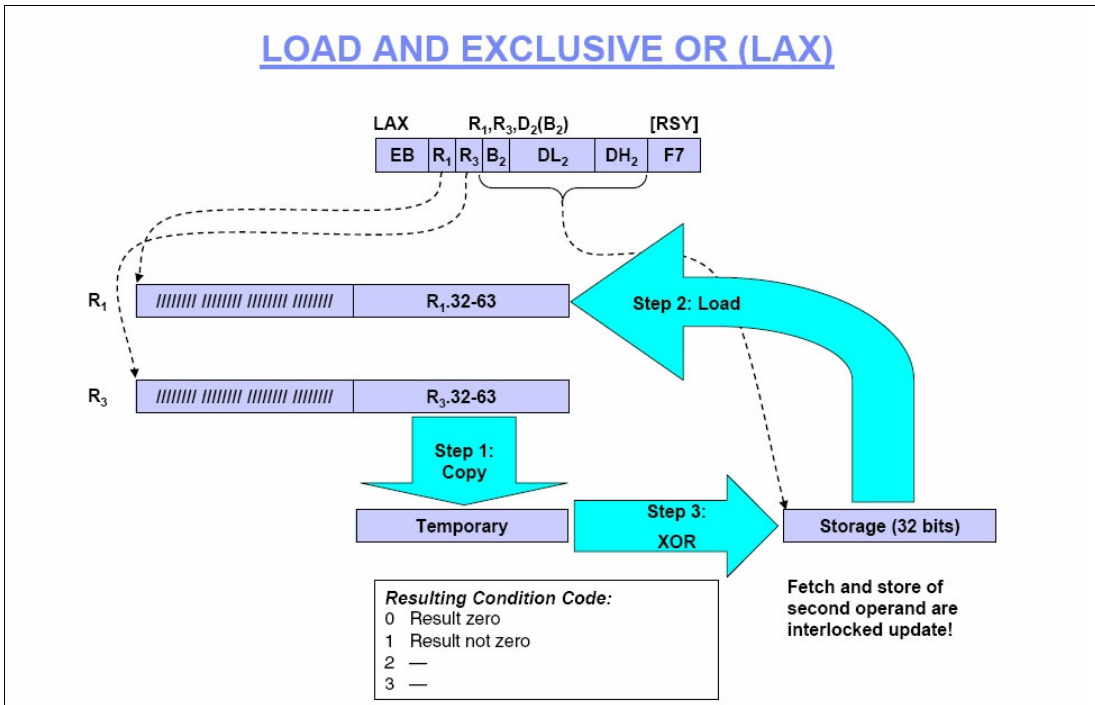


Figure 25-24 LOAD and Exclusive OR

LOAD PAIR DISJOINT (and its equivalent gigantic) can be depicted as in Figure 25-25.

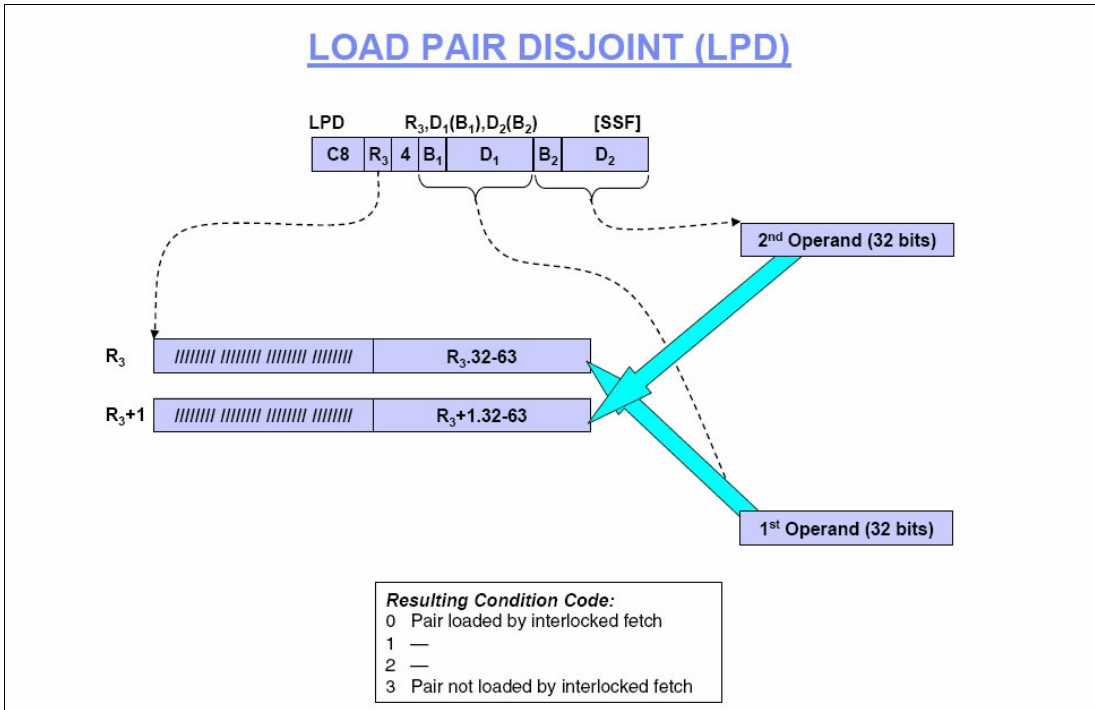


Figure 25-25 Load Pair Disjoint

### 25.3.1 Specific-operand serialization

Certain instructions cause specific-operand serialization to be performed for an operand of the instruction.

As observed by other processors and by the channel subsystem, a specific-operand serialization operation consists of completing all conceptually previous storage accesses by the processor before a subsequent access to the specific storage operand of the instruction may occur. At the completion of an instruction causing specific-operand serialization, the instruction's store is completed as observed by other processors and channel programs.

Specific-operand serialization is performed by the execution of the following instructions:

- ▶ ADD IMMEDIATE (ASI, AGSI) and ADD LOGICAL WITH SIGNED IMMEDIATE, for the first operand, when the interlocked-access facility is installed and the first operand is aligned on a boundary that is integral to the size of the operand.
- ▶ LOAD AND ADD, LOAD AND ADD LOGICAL, LOAD AND AND, LOAD AND EXCLUSIVE OR, LOAD AND OR, for the second operand.

## 25.4 More general purpose registers

High-Word Facility (bit 45) is a suite of instructions to manipulate bits 0-31 of a GPR. For decades, it has been debated whether some 16 more general purpose registers would be helpful for the compilers.

With 64-bit addressing, each of the existing 16 general purpose registers has been enlarged from 32 to 64 bits. Though taking care of the new 2\*\*64 addressing schema, such gigantic (64 bits) registers are only being used when accessing objects or data beyond the 2 GB. For other cases, specifically when used by programs running in 31-bit mode, half of their content is somehow spoiled, while compilers are still in need of more general purpose registers.

To provide register-constraint relief for compilers, starting with the z196, the left part of general purpose registers is somehow being made available as 0-31-bit registers.

For purposes of address-generation interlock (AGI), the leftmost bits (0-31) are treated separately from the rightmost.

For selected 32-bit instructions, the High-Word Facility effectively provides 16 additional 32-bit registers by utilizing bits 0-31 of the 16 64-bit general registers. The facility provides the instructions listed in Table 25-1.

Table 25-1 High-Word Facility

Instruction	Mnemonic	OpCode	1st operand	2nd operand	3rd operand
ADD HIGH	AHHHR	B9C8	R1.0-31	R2.0-31	R3.0-31
ADD HIGH	AHHLR	B9D8	R1.0-31	R2.0-31	R3.32-63
ADD HIGH IMMEDIATE	AIH	CC8	R1.0-31	I2[32bits]	--
ADD LOGICAL HIGH	ALHHHR	B9CA	R1.0-31	R2.0-32	R3.0-31
ADD LOGICAL HIGH	ALHHLR	B9DA	R1.0-31	R2.0-32	R3.32-63



Instruction	Mnemonic	OpCode	1st operand	2nd operand	3rd operand
ADD LOGICAL WITH SIGNED IMMEDIATE HIGH	ALSIH	CCA	R1.0-31	I2[32bits]	--
ADD LOGICAL WITH SIGNED IMMEDIATE HIGH	ALSIHN	CCB	R1.0-31	I2[32bits]	--
BRANCH RELATIVE ON COUNT HIGH	BRCTH	CC6	R1.0-31	RI2[16bits]	--
COMPARE HIGH	CHHR	B9CD	R1.0-31	R2.0-31	--
COMPARE HIGH	CHLR	B9DD	R1.0-31	R2.32-63	--
COMPARE HIGH	CHF	E3CD	R1.0-31	S20[32bits]	--
COMPARE IMMEDIATE HIGH	CIH	CCD	R1.0-31	I2[32bits]	--
COMPARE LOGICAL HIGH	CLHHR	B9CF	R1.0-31	R2.0-31	--
COMPARE LOGICAL HIGH	CLHLR	B9DF	R1.0-31	R2.32-63	--
COMPARE LOGICAL HIGH	CLHF	E3CF	R1.0-31	S20[32bits]	--
COMPARE LOGICAL IMMEDIATE HIGH	CLIH	CCF	R1.0-31	I2[32bits]	--
LOAD BYTE HIGH	LBH	E3C0	R1.24-31	S20[8bits]	--
LOAD HALFWORD HIGH	LHH	E3C4	R1.16-31	S20[16bits]	--
LOAD HIGH	LFH	E3CA	R1.0-31	S20[32bits]	--
LOAD LOGICAL CHARACTER HIGH	LLCH	E3C2	R1.24-31	S20[8bits]	--
LOAD LOGICAL HALFWORD HIGH	LLHH	E3C6	R1.16-31	S20[16bits]	--
ROTATE THEN INSERT SELECTED BITS HIGH	RISBHG	EC5D	R1.I3-I4	R2.0-63	I3,I4,I5

Instruction	Mnemonic	OpCode	1st operand	2nd operand	3rd operand
ROTATE THEN INSERT SELECTED BITS LOW	RISBLG	EC51	R1.32+i3-32 +i4	R2.0-63	I3,I4,I5
STORE CHARACTER HIGH	STCH	E3C3	R1.24-31	S20[8bits]	--
STORE HALFWORD HIGH	STHH	E3C7	R1.16-31	S20[16bits]	--
STORE HIGH	STFH	E3CB	R1.0-31	S20[32bits]	--
SUBSTRACT HIGH	SHHHR	B9C9	R1.0-31	R2.0-63	R3.0-31
SUBSTRACT HIGH	SHHLR	B9D9	R1.0-31	R2.0-63	R3.32-63
SUBSTRACT LOGICAL HIGH	SLHHR	B9CB	R1.0-31	R2.0-63	R3.0-31
SUBSTRACT LOGICAL HIGH	SLHHLR	B9DB	R1.0-31	R2.0-63	R3.32-63

The mechanism provided by this set of instructions can be depicted in Figure 25-26 on page 532 for ADD HIGH.

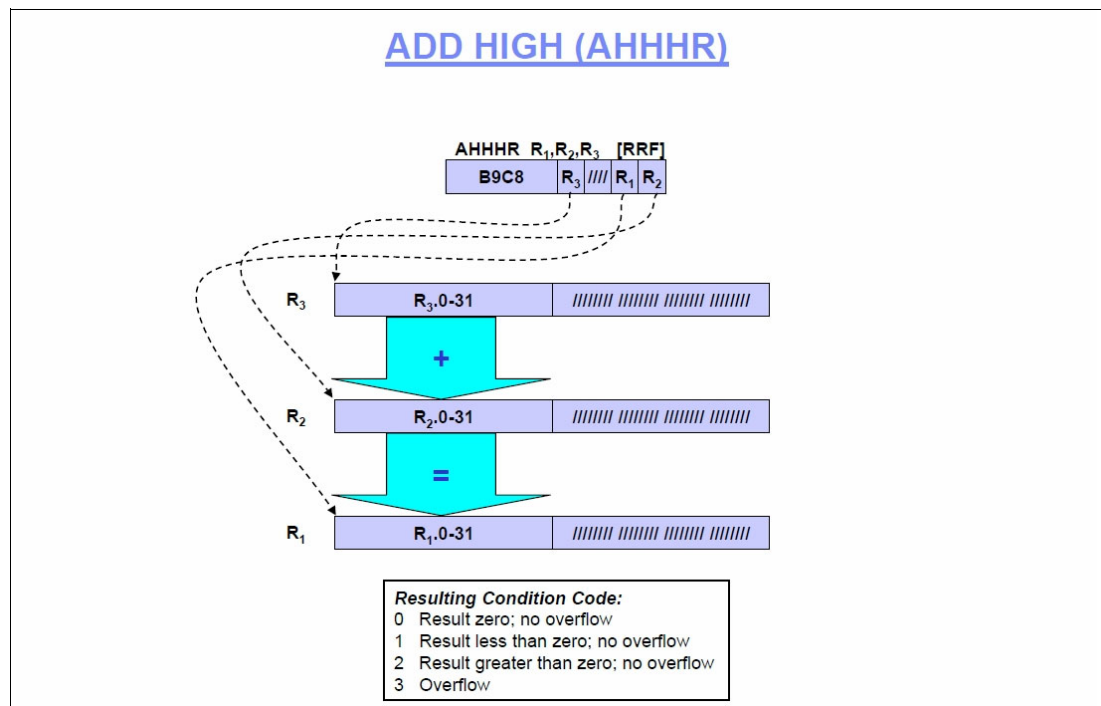


Figure 25-26 Add High

Or by the equivalent depicted in Figure 25-27 for loading with High-Word Facility, the supplemental general purpose registers.

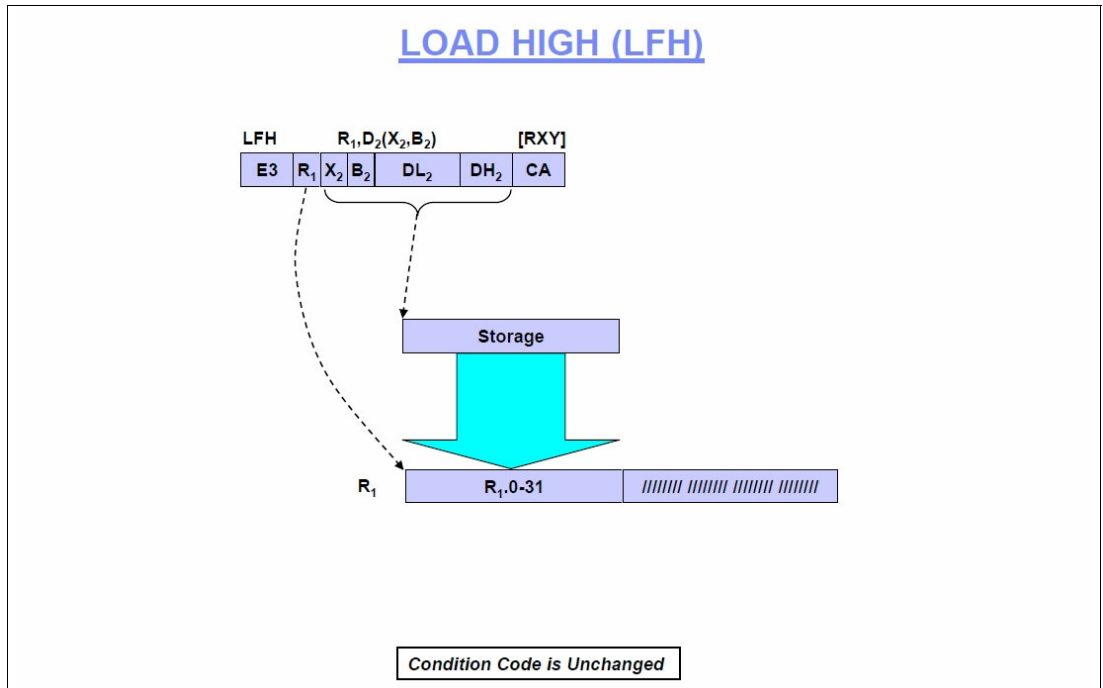


Figure 25-27 Load High

Figure 25-28 depicts a less obvious instruction introduced as part of High-Word Facility.

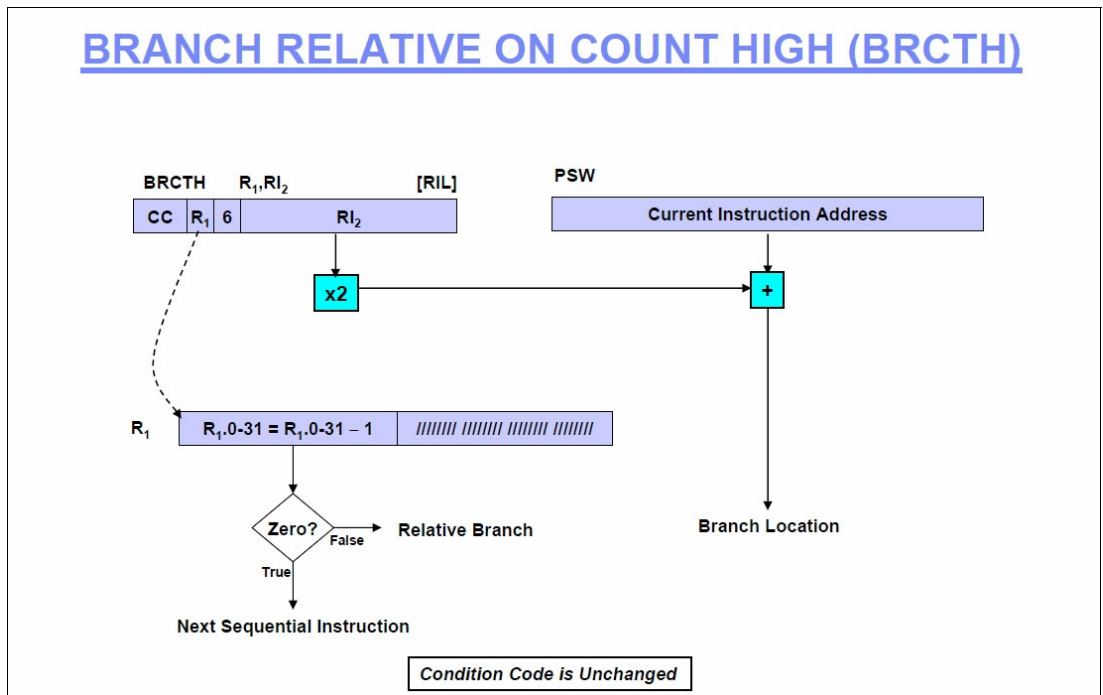


Figure 25-28 BRCTH





## Hardware instrumentation services

The Hardware Instrumentation Services (HIS) function provides a framework for collecting and recording hardware event data for IBM System z10 machines. This function collects hardware event data for processors in SMF records type 113, subtype 2, as well as UNIX System Services output files. Note that you can only use HIS for IBM System z10 or later machines.

z/OS V1R12 provides enhancements in order to ease usage of these services. This chapter describes these changes along with a high-level review of the function. It describes the following:

- ▶ Hardware performance analysis
- ▶ CPU Measurement Counter Facility
- ▶ Hardware Instrumentation Services address space
- ▶ z/OS V1R12 enhancements

## 26.1 CPU Measurement Facility overview

Having discussed several software techniques to improve performance with HiperDispatch in a previous chapter, 3.1, “HiperDispatch overview” on page 72, we focus here on measuring performance and analyzing possible weaknesses. A significant improvement achieved with the System z10 server is hardware support for performance analysis. This support makes understanding performance problems easier and much more precise than the software-based methods used before. The new hardware instrumentation does not disturb the system being observed, although it requires support from z/OS.

These sections provide a brief overview of the CPU Measurement Facility architecture. Subsequent sections explain how to invoke this function in z/OS.

**Note:** Hardware instrumentation services (HIS) is a function that collects hardware event data for processors in SMF records type 113, subtype 2, as well as UNIX System Services output files. You can only use HIS for IBM System z10 or later machines.

Start the HIS address space on each system where you want to collect data. Then configure and activate HIS data collection (hardware event counters and sampling facilities) for each system by issuing the **F hisproc** command.

### 26.1.1 Hardware-based performance analysis

Hardware Instrumentation offers many advantages over the software-based performance tools used in previous systems. The CPU Measurement Facility, available in z10 systems, provides support built into the processor for performance analysis. Exploiting this mechanism allows you to observe the performance behavior of an operating system running client applications, and it does so with little impact to the system being observed.

The CPU Measurement Facility is designed to increase the ability to run hardware instrumentation in the field and to support application performance tuning.

In the past, System z mainframe hardware instrumentation has been used for the following tasks:

- ▶ Debugging hardware performance and enabling performance tuning by the sophisticated compilers of operating systems and internal subsystems
- ▶ Allocating (or deallocating) a sufficient hardware system area for storing measurement data

However, allocation or deallocation of a hardware system area requires a machine power-on. Also, hardware instrumentation runs on CPs at the basic machine-level only; thus, running it requires the entire machine. Because of these constraints, hardware instrumentation has been limited mainly to laboratory use. The CPU Measurement Facility resolves these issues by enabling the control program to allocate real storage in which the machine can store measurement data and to virtualize the hardware instrumentation so that it can run in multiple LPARs concurrently.

The format and meaning of measurement data used to facilitate application performance tuning are defined in the architecture. The remainder of the CPU Measurement Facility, used to establish controls and extract measurement data, is basically the same, independent of the intended use. The discussion from this point focuses on the use of measurement data to tune application performance.

## Installation considerations

This architecture defines the CPU Measurement Facility, which is available in the z/Architecture architectural mode. When the facility is installed on a processor, the same facility is installed on all processors in the configuration.

Measurement data provided by the CPU Measurement Counter Facility or the CPU Measurement Sampling Facility is intended for statistical estimation of performance characteristics. Measurement data is only substantially accurate, and may not be repeatable. For example, it is unpredictable if the instruction count counts an instruction that caused an exception or counts the target of EXECUTE. Also, certain system internal activities may be included in measurement data.

The CPU Measurement Facility consists of two parts:

- ▶ A CPU Measurement Counter Facility  
This facility provides a means to measure activities in the processor and in various shared peripheral processors.
- ▶ A CPU Measurement Sampling Facility  
This facility provides a means to take a snapshot of the processor at a specified sampling interval and requires the SET PROGRAM PARAMETER facility as a prerequisite. Each part consists of a number of instructions to set and extract controls or measurement data, and consists of several events of an external interruption subclass.

**Note:** Disable the CPU Measurement Counter Facility and the CPU Measurement Sampling Facility when they are not being used.

## 26.2 CPU Measurement Counter Facility

The CPU Measurement Counter Facility on each processor consists of a local counter-set-state control register, a number of counters, several external interruption events, a measurement-counter-extraction-authorization control, and a number of instructions.

The following list summarizes the instructions.

```
EXTRACT COPROCESSOR-GROUP ADDRESS
EXTRACT CPU COUNTER
EXTRACT PERIPHERAL COUNTER
QUERY COUNTER INFORMATION
SET CPU COUNTER
SET CPU-COUNTER-SET CONTROLS
SET PERIPHERAL COUNTER
SET PERIPHERAL-COUNTERSET CONTROLS
```

### 26.2.1 Measurement counters

The measurement-counter-extraction-authorization control, which is bit 15 of control register 0, specifies whether selected processor counter contents can be extracted in the problem state. The bit is initialized to zero (0).

The facility provides two kinds of counters: local counters and global counters.

- ▶ Local counters are local to the processor. These counters are used to count logical processor activities and are grouped into several processor counter sets.

- ▶ Global counters are used to count activities of peripheral processors, which are shared among certain processors, and are grouped into different peripheral counter sets.

All counters are 64 bits. A carry-out of bit position 0 of any counter is ignored. When any peripheral counter set is provided, the facility also includes a global counter-set-state control register. The number and type of installed counters are model-dependent. Two 16-bit counter version numbers, called counter first version number and counter second version number, are provided by QUERY COUNTER INFORMATION. Each version number identifies installed counters in certain counter sets.

When there is a change to the meaning of a counter or the number of installed counters in any counter set specified by a counter version number, then another value is indicated by that counter version number. Both counter version numbers start at one and increase consecutively.

### **Basic counter set**

Counters can be grouped as basic, such as:

- ▶ Cycle count
- ▶ Instruction count
- ▶ Level-1 I-cache directory write count
- ▶ Level-1 I-cache penalty cycle count
- ▶ Level-1 D-cache directory write count
- ▶ Level-1 D-cache penalty cycle count

### **Problem state counter set**

Others are grouped as problem state counter:

- ▶ Problem state cycle count
- ▶ Problem state instruction count
- ▶ Problem state level-1 I-cache directory write count
- ▶ Problem state level-1 I-cache penalty cycle count
- ▶ Problem state level-1 D-cache directory write count
- ▶ Problem state level-1 D-cache penalty cycle count

### **Extended counter set**

Some other counters are model-dependant.

## **26.2.2 Cryptographic coprocessor support**

IBM z/Architecture includes a message security assist that supports cryptographic operations. The crypto activity counter set collects information about activities caused by executing cryptographic operations. Because the z10 machine is the first in which two CPs share a cryptographic coprocessor, this counter set provides information about characteristics of the cryptographic workload and cryptographic interference between the two sharing CPs.

A cryptographic coprocessor group consists of a cipher coprocessor for the Data Encryption Standard and the Advanced Encryption Standard and a hash coprocessor for the Secure Hash Algorithm. Both coprocessors can perform cryptographic operations simultaneously. However, when two sharing CPs attempt to use the same coprocessor, one will be blocked until a predetermined time slice has passed.



Additionally, a counter set is provided for each cryptographic coprocessor group to count cryptographic activities contributed by both of the sharing CPs. This counter set provides information about the utilization of each cryptographic coprocessor.

### **Sampling CP snapshots**

Sampling provides a means to take a snapshot of the CP at a program-specified sampling interval. At the end of each sampling interval, the machine stores sample data into measurement blocks allocated by the control program. Each sample data includes an instruction address, the primary address space number, and state information about the CP. This allows tooling programs to map instruction addresses into modules or tasks and facilitates the determination of hotspots. A control program can set up alerts before the measurement blocks are filled up so that it can take appropriate actions to save the sampling data and to free up space in the measurement blocks.

### **Crypto CPACF activity counters**

The following counters are provided for crypto:

- ▶ PRNG function count
- ▶ PRNG cycle count
- ▶ PRNG blocked function count
- ▶ PRNG blocked cycle count
- ▶ SHA function count
- ▶ SHA cycle count
- ▶ SHA blocked function count
- ▶ SHA blocked cycle count
- ▶ DES function count
- ▶ DES cycle count
- ▶ DES blocked function count
- ▶ DES blocked cycle count
- ▶ AES function count
- ▶ AES cycle count
- ▶ AES blocked function count
- ▶ AES blocked cycle count

## **26.2.3 Counter sets**

When both counter version numbers are 1, the facility provides four processor counter sets on each processor. They are: basic counter set, problem-state counter set, crypto-activity counter set, and extended counter set. Counters in the processor counter sets count processor activities on the logical processor basis. The facility also provides one kind of peripheral counter sets, called coprocessor-group counter sets. One coprocessor group counter set is provided for each coprocessor group. Counters in a coprocessor-group counter set count activities of each coprocessor in the group at the basic machine level.

At the basic machine level, a coprocessor group consists of a number of coprocessors and can be attached to one or more processors. When authorized, all processors in the model have access to all coprocessor-group counter sets and the global counter-set-state control register.

When the counter first version number is 1, it identifies installed counters in the basic counter set and problem-state counter set; when the counter second version number is 1, it identifies installed counters in the crypto-activity counter set, the extended counter set, and the coprocessor-group counter sets.

## 26.2.4 CPU Measurement Sampling Facility

The CPU Measurement Sampling Facility on each processor consists of two sampling functions, several sampling control registers, several external interruption events, and two instructions. The instructions are:

- ▶ QUERY SAMPLING INFORMATION
- ▶ SET SAMPLING CONTROLS

The facility provides a means to take a snapshot of the logical processor at a specified sampling interval, which is a processing time interval as seen by the processor. Each snapshot produces a set of sample data, which includes the instruction address of an instruction being executed and state information about the processor. The CPU Measurement Sampling Facility requires the SET PROGRAM PARAMETER facility as a prerequisite.

The facility includes two sampling functions: basic sampling and diagnostic sampling. They both use the same sampling control registers, the same sampling buffer structure, the same external interruption events, and the same instructions. The main difference between these two functions is the sample data. The sample data size and format for each sampling function are model-dependent, and are determined by a 16-bit data entry format code that is stored in each sample data.

The data entry format code starts with 0001 (hex) for the basic-sampling data entry; the code starts with 8001 (hex) for the diagnostic sampling data entry. Both data entry format codes continue consecutively. When the size, the meaning, or the format of a sample data is different from the previous model, then the appropriate data entry format code is incremented.

The sample data provided by the basic sampling function is not included in the sample data provided by the diagnostic sampling function. To get meaningful diagnostic sampling data, both sampling functions must be activated. The state of each sampling function can be individually set by executing SET SAMPLING CONTROLS. Both sampling functions are disabled by initial processor reset, clear reset, or Power-on Reset.

## 26.3 Hardware Instrumentation Services address space

The Hardware Instrumentation Services (HIS) address space must be started on each system where you want to collect data. Then configure and activate HIS data collection (hardware event counters and sampling facilities) for each system by issuing the **f hisproc** command.

**Important:** Assign a sufficiently high dispatch priority to the instrumentation started task hisproc, so that the task can write sampling data to the SMP output files in a timely manner.

### 26.3.1 Setting up hardware event data collection

Hardware Instrumentation Services (HIS) is a function that collects hardware event data for processors at the System z10 Enterprise Class level or later. During a run of hardware data collection, the system writes the data to a UNIX System Services output file as well as to SMF record type 113, subtype 2. The system writes the raw data to SMF record type 113 at 15-minute intervals during the data collection run, and writes the delta data to the UNIX System Services output file at the end of the run. The delta data is the data from between

when the instrumentation run was initiated and the end of the run, showing the delta incremental values of the instrumentation run on the specified processor.

### 26.3.2 Security specifications

Before you start the HIS data collection, you may first need to authorize to the sampling facilities and counter set types you want to use through the support element (SE) console. For information about how to set up the authorization of the sampling facilities and counter sets, see *Support Element Operations Guide for System z10 machine* on the Resource Link™ home page at:

<http://www.ibm.com/servers/resourceLink>

These LPAR security options are new for z10. They allow CPU Measurement Facility to be actively gathering and saving performance data; refer to Figure 26-1.

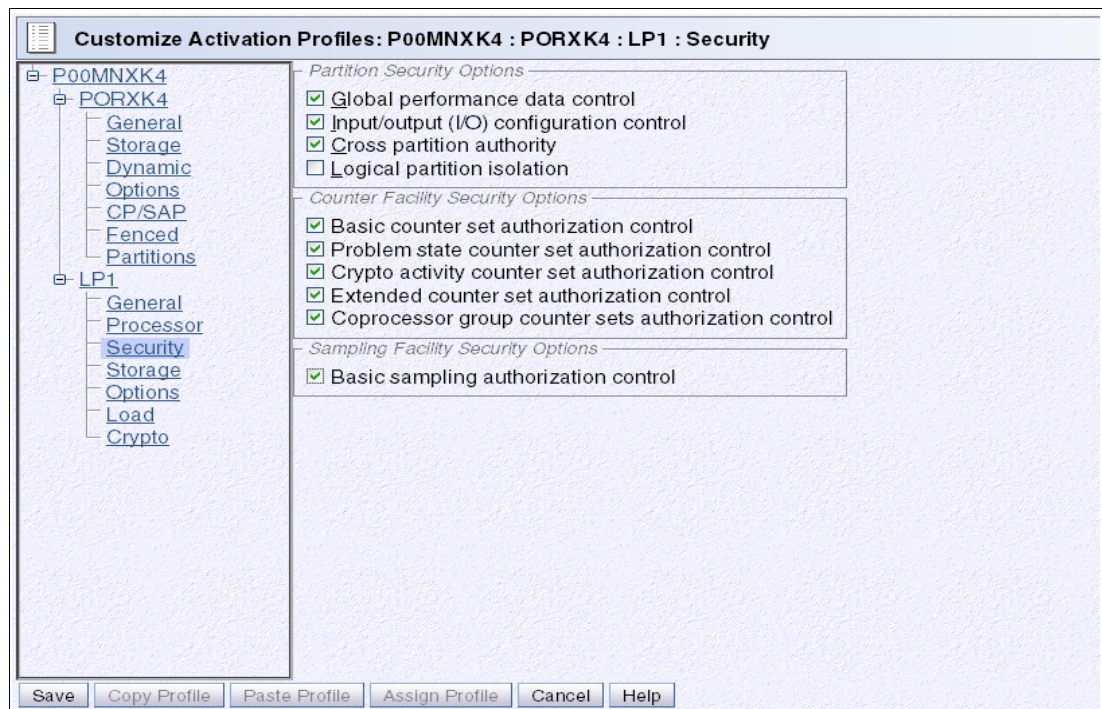


Figure 26-1 CPU Measurement Facility

### 26.3.3 Collecting data with HIS

You can specify in advance the duration of a run of data collection you want by using the DURATION parameter with the **F hisproc,BEGIN** command. If you configure a new processor online in a system after you have already issued the **F hisproc,BEGIN** command to start a data collection run for that system, then HIS might not collect data for that processor. To ensure that data is collected for all the processors on a system, bring the processors online before beginning a hardware data event collection run. The system does not collect data on a processor that is configured offline.

See "Start, configure, and stop hardware event data collection" on page 542, for complete information about the **F hisproc** command.

## Stopping data collection

To explicitly stop the run of data collection on a system, use the following command:

```
F hisproc,END
```

As an alternative, you can use the **DURATION** parameter on the **F hisproc,BEGIN** command to specify when you want a data collection run to end. When the data collection run ends, the system writes all the collected data to the UNIX System Services data set at the path specified and to the SMF data set that was set up by the installation.

You can also use the **p hisproc** command to stop a run of data collection. Note that if you use the **STOP** command, you must restart the address space again with the **START** command before starting the next run of data collection.

## Start, configure, and stop hardware event data collection

Use the **F hisproc** command to manage hardware event data collection for systems at the System z10 Enterprise Class level or higher. Use **F hisproc,BEGIN** to configure and start a run of data collection, and use the **F hisproc,END** command to stop the run. You must explicitly start each run of hardware data collection; you cannot set up data collection to run automatically.

During a run of hardware data collection, the system writes the data to a UNIX System Services output file as well as to SMF record type 113, subtype 2. The system writes the raw data to SMF record type 113 at 15-minute intervals during the data collection run, and writes the delta data to the UNIX System Services output file at the end of the run. The delta data is the data from between when the instrumentation run was initiated and the end of the run, showing the delta incremental values of the instrumentation run on the specified processor.

Before you issue the **F hisproc,BEGIN** command to configure and start hardware event gathering on a system, you must perform several setup tasks; see 26.3.1, “Setting up hardware event data collection” on page 540.

## 26.4 z/OS V1R12 enhancements for HIS

The following enhancements to HIS are provided in z/OS V1R12:

- ▶ Detect changes to the state of the system (state change), and create the concept of collection run intervals which are marked by these state changes.
- ▶ Provide the ability to synchronize SMF Record Type 113 records with other SMF records.
- ▶ Provide the ability to create a load module mapping (.MAP) of the system without having the hardware counters and sampling facilities authorized for use.

### 26.4.1 State change detection

The effective model of the machine and the processor speed can be changed dynamically without a re-IPL. This can occur for clients using Replacement Capacity, On/Off Capacity on Demand, or during cooling problems with the machine. HIS currently relies on the processor speed to remain constant throughout an instrumentation run. If the processor speed were to change from underneath HIS without its knowledge, data collected for that run will either be thrown out or misinterpreted.

The following events can be considered as potential significant hardware events (state change):

- ▶ Replacement Capacity (Customer Initiated Upgrade)
- ▶ On/Off Capacity on demand
- ▶ Cooling problems

A Hardware instrumentation services (HIS) performance run can span across “significant hardware events,” rendering any data collected useless without knowledge of the event. To address this, z/OS V1R12 allows to detect changes to the state of the system (state change), and hence create the concept of collection run intervals that are marked by these state changes.

### **STATECHANGE parameter**

How HIS reacts depends on the STATECHANGE parameter specified at the start of the collection run:

- ▶ STATECHANGE=STOP stops the collection run when the event is detected.
- ▶ STATECHANGE=IGNORE continues the collection run as if the event never happened.
- ▶ STATECHANGE=SAVE which is the default value. It records the previous state of the system (saves all data).
  - Writes and closes the .CNT file.
  - Closes all .SMP files (1 per processor).
  - Cuts SMF Type 113 Records (1 per processor)
- ▶ Continues the collection run with the new state
  - Creates new .SMP files (1 per processor).
  - Cuts SMF Type 113 Records (1 per processor).

### **Output file names**

The output file names have a new format:

- ▶ Before z/OS V1R12, it was: SYSHISyyyymmdd.hhmmss.type.cpu#
- ▶ Starting with z/OS V1R12, it is: SYSHISyyyymmdd.hhmmss.seq.type.cpu#

All files from the same collection run have the same prefix, which is:

SYSHISyyyymmdd.hhmmss

The new sequence number represents a collection run interval. A collection run interval is data during a collection run where the system was in a known consistent state. The contents of the .CNT file are an indicator as to when the state change occurred.

### **Sampling data collection run**

A collection run was started at 09:53:26 on January 2, 2010 on a machine with two processors, collecting both counter and sampling data, with STATECHANGE=SAVE.

During this collection run, CoD increased the capacity of the machine. It then triggers the following output files for the HIS runs:

- ▶ SYSHIS20090102.095326.000.CTR
- ▶ SYSHIS20090102.095326.000.SMP.00
- ▶ SYSHIS20090102.095326.000.SMP.01

- ▶ SYSHIS20090102.095326.001.CTR
- ▶ SYSHIS20090102.095326.001.SMP.00
- ▶ SYSHIS20090102.095326.001.SMP.01

The output file at the end of the general collection:

SYSHIS20090102.095326.001.MAP

The .CNT file adds a new line to describe the state (new version identifier). For instance:

- ▶ When a state change was detected and STATECHANGE=STOP, the contents of the .CNT file is as shown in Figure 26-2.

```
HIS019I  EVENT COUNTERS INFORMATION VERSION 2
FILE NAME:  SYSHISyyyymmdd.hhmmss.000.CNT
COMMAND:  MODIFY HIS,xxxx
STATE CHANGE: YES,STOP
COUNTER VERSION NUMBER 1:  xxxx   COUNTER VERSION NUMBER 2:  xxxx
```

Figure 26-2 STATECHANGE=STOP

- ▶ When a state change was detected and STATECHANGE=IGNORE is specified, Figure 26-3 is displayed.

```
HIS019I  EVENT COUNTERS INFORMATION VERSION 2
FILE NAME:  SYSHISyyyymmdd.hhmmss.000.CNT
COMMAND:  MODIFY HIS,xxxx
STATE CHANGE: YES,IGNORE
COUNTER VERSION NUMBER 1:  xxxx   COUNTER VERSION NUMBER 2:  xxxx
```

Figure 26-3 STATECHANGE=IGNORE

- ▶ When a state change was detected and STATECHANGE=SAVE, is specified, Figure 26-4 is displayed.

```
HIS019I  EVENT COUNTERS INFORMATION VERSION 2
FILE NAME:  SYSHISyyyymmdd.hhmmss.000.CNT
COMMAND:  MODIFY HIS,xxxx
STATE CHANGE: YES,SAVE
COUNTER VERSION NUMBER 1:  xxxx   COUNTER VERSION NUMBER 2:  xxxx
```

Figure 26-4 STATECHANGE=SAVE

- ▶ When no state change was detected, Figure 26-5 is displayed.

```
HIS019I  EVENT COUNTERS INFORMATION VERSION 2
FILE NAME:  SYSHISyyyymmdd.hhmmss.000.CNT
COMMAND:  MODIFY HIS,xxxx
STATE CHANGE: NO
COUNTER VERSION NUMBER 1:  xxxx   COUNTER VERSION NUMBER 2:  xxxx
```

Figure 26-5 no state change

This enhancement and the concept of an HIS run going with it, allows you to analyze data more effectively on a known system state. It achieves a better picture of the system through more synchronized information.

## 26.4.2 SMF Record Type 113 records

Before z/OS V1R12, SMF Type 113 record data was recorded at the start, end, and at 15 minute intervals of a collection run. It was difficult to line up this performance data with other performance-related SMF records. It was also impossible to get data more frequently.

With z/OS V1R12, the new SMFINTVAL parameter provides the ability to modify when SMF Type 113 records are recorded:


- ▶ SMFINTVAL={1-60}: Record every x minutes; the default is 15.
- ▶ SMFINTVAL=SYNC: Synchronize records with the SMF global recording interval.

These new INTVAL and SYNCVAL parameters are to be set in the SMFPRMxx member of the parmlib.

## 26.4.3 Load module mapping

Until now with HIS, mapping load modules required that either the counter facility or the sampling facility be installed and authorized. Support is added in z/OS V1R12 to allow mapping load modules when neither the counter facility nor the sampling facility are authorized.

For that purpose, a new parameter is introduced, shown in Figure 26-6 on page 545.



```
F HIS,B,MAPONLY
```

Figure 26-6 New parameter to start HIS

User authorization is required to map load modules. The hisproc user must have READ authority to the data sets and the z/OS UNIX System Services directories that are used when loading modules, whether into LPA storage or from the LNKLST concatenation, a joblib, or steplib or tasklib concatenation, or a concatenation identified by a program-specified DCB.

The hisproc user must be designated as TRUSTED.

## 26.5 Installation considerations

Starting with z/OS V1R12, the enhancements to HIS are available via APAR OA30486 down to z/OS V1R10.

### 26.5.1 Requirements

The System z10 machine must be at GA2 Driver 76D – Bundle #11 or higher (z10 BC subcapacity models must be at Bundle #20 or higher). HIS is not supported for z/OS running as a z/VM guest.







## FICON dynamic channel-path management

This chapter describes dynamic channel-path management (DCM) for FICON, which allows z/OS to dynamically manage FICON channel paths and control unit ports in response to changing workload demands.

FICON DCM, as it is known, allows you to identify FICON channels and control units that should be managed by the system. This gives you the means to:

- ▶ Define an I/O configuration to maximize availability and performance that otherwise can become quite complex.
- ▶ Avoid to overconfigure for performance peaks.

Overall, FICON DCM:

- ▶ Simplifies I/O configuration definition tasks.
- ▶ Improves overall I/O performance.
- ▶ Provides more efficient use of hardware resources.
- ▶ Dynamically balances I/O resources based on workload demands.
- ▶ Enhances availability by dynamically adding new channel paths for certain error conditions.

Dynamic Channel-path Management (DCM) was initially shipped in z/OS V1R1. At that time, FICON native channels were not supported, and DCM only supported ESCON and FICON bridge channels. This new support in z/OS V1R11 provides support for FICON native channels.

Generally speaking, defining an I/O configuration to maximize availability and performance is very complex and cumbersome. What ends up happening is that many clients tend to overconfigure the I/O configuration to manage performance peaks.

DCM in general (and FICON DCM for that matter) allows z/OS to manage channel paths (FICON and ESCON) dynamically. You need to identify the channels and control units that should be managed.

By assigning channels to the pool of managed channels, the system is able to respond to peaks in a control unit's demands for I/O channel bandwidth. In addition, this reduces the complexity of defining the I/O configuration, since the client is no longer required to define a configuration that will adequately address any variations in workload. This allows to specify a much looser configuration definition.

DCM can provide improved performance by dynamically moving the available channel bandwidth to where it is most needed. Prior to DCM, you had to manually balance your available channels across your I/O devices, trying to provide sufficient paths to handle the average load on every controller. This means that at any one time, some controllers probably have more I/O paths available than they need, while other controllers possibly have too few.

Another advantage of Dynamic Channel-path Management is that you do not have to be as concerned about keeping different rules of thumb about how busy you should run your channels for every channel or control unit type you have installed. Instead, you just need to monitor for the signs of channel over-utilization (high channel utilization combined with high pending times).

Because Dynamic Channel-path Management can dynamically move I/O paths to the LCUs that are experiencing channel delays, you can reduce the CU/channel-level capacity planning and balancing activity that was necessary prior to Dynamic Channel-path Management.

Using DCM, you are only required to define a minimum of one non-managed path and up to seven managed paths to each controller (although a realistic minimum of at least two non-managed paths is recommended), with Dynamic Channel-path Management taking responsibility for adding additional paths as required, and ensuring that the paths meet the objectives of availability and performance.

It is still necessary to understand the performance and availability characteristics of all your installed hardware. DCM can only work within the confines of the physical connectivity that you design and implement. However, to the extent that DCM is self-tuning, it should be possible to spend considerably less time on configuration management and monitoring and capacity planning. This allows you to free up scarce technical resources to work on other valuable tasks.

When attaching a DASD subsystem to a zSeries CPC, there are a number of things you should take into account, if you wish to achieve maximum availability and performance from the configuration. Within the CPC, there are a number of redundant components provided (channel cards, STIs, MBAs, and so on). Ideally, you would attach a control unit to channels that are spread over as many of these components as possible, minimizing the number of single points of failure.

Similarly, in the switch and the DASD controller itself, there are various availability characteristics that you must be aware of when designing the physical connectivity. If you currently go to the trouble of configuring every path to make the best use of the provided availability features, you are probably in the minority! DCM can help you make the best use of the installed availability features, without the time investment that was required prior to DCM. As we stated previously, you still have to take these hardware features into account when planning your physical connectivity; however, DCM can reduce the effort you have to put into deciding which paths should be used for each control unit, and which control units should be placed on the same paths. This reduces your workload and provides you with similar or better reliability and availability.

## 27.1 LPAR clustering

LPAR clustering was announced on October 3, 2000 as part of Intelligent Resource Director (IRD), a set of new capabilities available on the IBM zSeries range of processors and delivered as part of z/OS. Intelligent Resource Director might be viewed as Stage 2 of Parallel Sysplex. Stage 1 provided facilities to let you share your data and workload across multiple system images. As a result, applications that supported data sharing could potentially run on any system in the sysplex, thus allowing you to move your workload to where the processing resources were available.

However, not all applications support data sharing, and there are many applications that have not been migrated to data sharing for various reasons. For these applications, IBM has provided Intelligent Resource Director, which basically gives the ability to move the resource to where the workload is. Intelligent Resource Director uses facilities in z/OS Workload Manager (WLM), Parallel Sysplex, and PR/SM to help you derive greater value from your z/Series investment. Compared to other platforms, z/OS with WLM already provides benefits from the ability to drive a processor at 100% while still providing acceptable response times for your critical applications. Intelligent Resource Director amplifies this advantage by helping you make sure that all those resources are being utilized by the right workloads, even if the workloads exist in different Logical Partitions (LPs).

### IRD principles

Figure 27-1 on page 550 shows a simplified view of what Intelligent Resource Director can offer. It shows a Central Processing Complex (CPC) with two LPs. One LP contains an OLTP workload, defined in WLM as being Importance 1, and a batch workload, defined in WLM as being Importance 3. The other LP contains a Business Intelligence workload that is defined to WLM as being Importance 2. Both the batch and Business Intelligence workloads are capable of using the capacity of the whole CPC if allowed. To provide the OLTP workload with the resources it requires to meet its goals during the prime shift, Intelligent Resource Director sets the LPAR weight of that LP to 75. The weight of the Business Intelligence LP is set at 25.

However, in the evening shift when the OLTP workload has gone away, Intelligent Resource Director will adjust the weights so that the Business Intelligence LP, which is of higher importance than batch, now gets 75% of the CPC, and the LP containing the batch workload now gets 25%. You will also notice that during the prime shift the OLTP DASD have more channels, whereas in the evening, there are more paths to the Business Intelligence DASD. Intelligent Resource Director has also automatically adjusted the channel configuration to provide more paths to the DASD subsystem serving the more important workload.

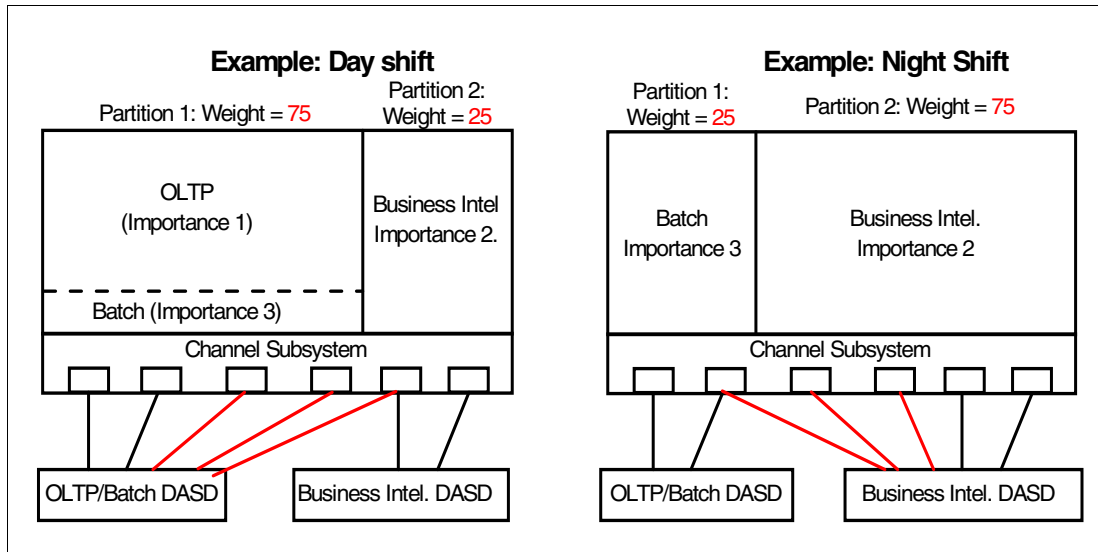


Figure 27-1 IRD principles

Intelligent Resource Director is not actually a product or system component; rather it was announced in 2000 as three separate but mutually supportive functions:

- ▶ WLM LPAR processor management
- ▶ ESCON dynamic channel-path management (ESCON DCM)
- ▶ Channel Subsystem I/O Priority Queueing (CSS IOPQ)

Since then the LPAR processor management has been complemented by:

- ▶ Group capacity limit
- ▶ HiperDispatch (which cannot currently coexist with WLM LPAR processor management) on z10 systems.

With z/OS V1R11 and the GA2 of z10 systems, FICON dynamic channel-path management is introduced and is the main topic of this chapter.

### 27.1.1 LPAR clustering components

The following components are included with LPAR clustering:

- ▶ General purpose processors
  - WLM LPAR processor management (part of IRD, announced in 2000)
  - Group capacity limit (available with GA2 z9 systems and z/OS v1r8)
- ▶ Memory
  - HiperDispatch (available with z10 systems and z/OS V1R9 or higher; cannot coexist with WLM LPAR processor management)
- ▶ I/O connections
  - ESCON DCM (part of IRD announced in 2000)
  - FICON DCM (available with GA2 z10 systems and z/OS V1R11)
  - Channel subsystem I/O priority queueing (CSS IOPQ, part of IRD announced in 2000)

To recap, z10 systems and z/OS V1R11 can henceforth provide LPAR clustering to help with managing, as depicted in Figure 27-2.

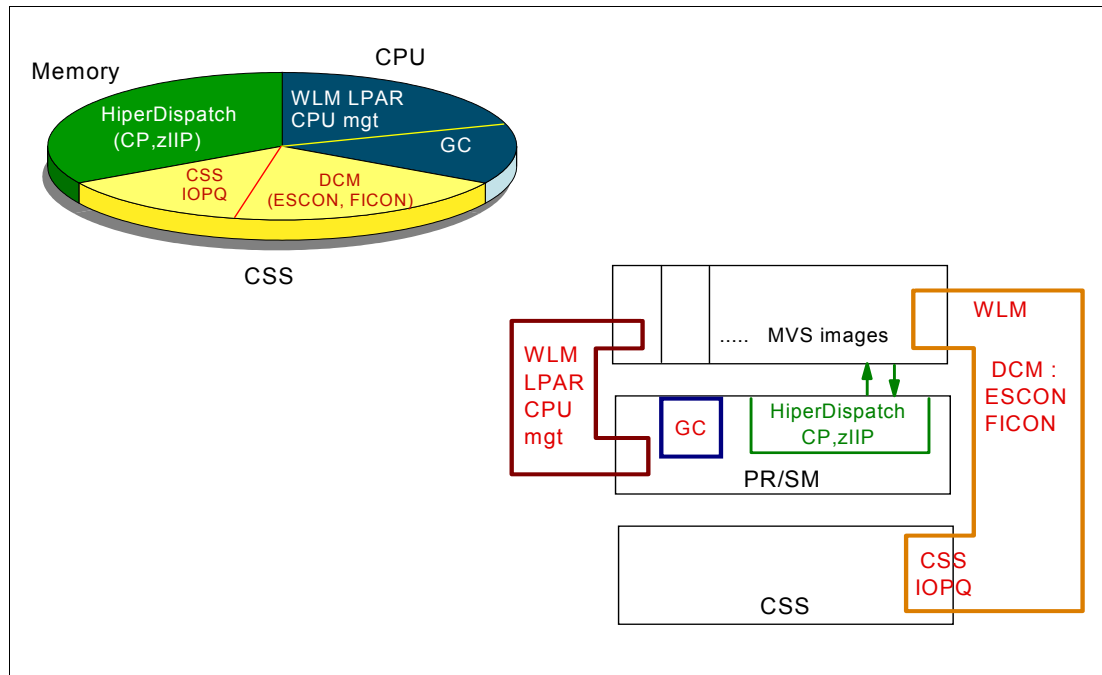


Figure 27-2 LPAR clustering

### Intelligent Resource Director

As the original Intelligent Resource Director, those different LPAR clustering functions are implemented jointly by:

- ▶ z/OS (in z/Architecture mode)
- ▶ Workload manager (WLM)
- ▶ PR/SM part of the IBM zSeries

Also, by using existing function in the following system components:

- ▶ Hardware Configuration Dialog (HCD)
- ▶ Dynamic I/O reconfiguration
- ▶ I/O supervisor (I/O)

## 27.2 Value of dynamic channel-path management

DCM provides value in a number of ways, including performance, availability, and reduced cost of computing. The specific benefits that DCM can provide for your installation depend on the profile of your environment.

The following paragraphs describe the objectives and benefits of DCM in more detail.

## 27.2.1 Improved overall I/O performance

DCM can provide improved performance by dynamically moving the available channel bandwidth to where it is most needed. Prior to DCM, you had to manually balance your available channels across your I/O devices, trying to provide sufficient paths to handle the *average* load on every controller. This means that at any one time, some controllers probably have more I/O paths available than they need, while other controllers possibly have too few. Dynamic Channel-path Management attempts to balance the responsiveness of the available channels by moving channels to the controllers that require additional bandwidth, even when the system is in WLM Compatibility mode.

The chart of Figure 27-3 shows the concept of how DCM operates. It shows the number of channels and their utilization when serving two logical control units (LCU) in two scenarios, one without DCM and another with DCM.

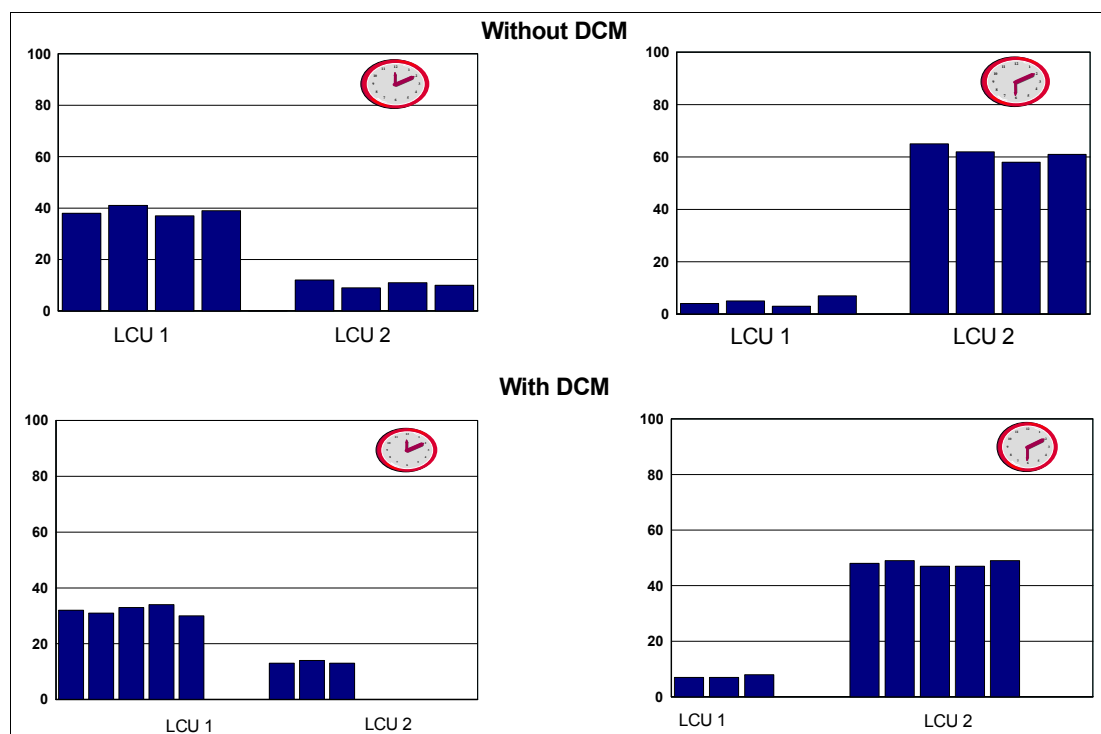


Figure 27-3 Improved system performance with DCM

### DCM and channels

Without DCM, the channels are unbalanced in relation to their utilization due to I/O skew (uneven I/O activity in the LCUs) in the two LCUs. This is the normal mode of operation; it is unusual to see a configuration where all the control units are equally utilized at all times.

With DCM, the load on the LCUs is still uneven because DCM cannot do anything about that, but the channel utilization (and therefore the responsiveness) is more evenly balanced, resulting in decreased Pend times. In the prime shift, when LCU1 is much busier than LCU2, DCM moves two channels from LCU2 to LCU1 and also adds two more channels to LCU1. In the evening, when the load on LCU1 largely disappears, and LCU2 gets much busier, the bulk of the channel bandwidth is moved to LCU2. In practice, DCM may not actually remove the channels from LCU1 in the evening—this depends on the system workload at the time—but the chart is shown like this for illustration purposes.

Another advantage of Dynamic Channel-path Management is that you do not have to be as concerned about keeping different rules of thumb about how busy you should run your channels for every channel or control unit type you have installed. Instead, you just need to monitor for the signs of channel over-utilization (high channel utilization combined with high Pend times). This is made easier by changes in RMF, which now gives a report showing the average aggregate utilization for all managed channels.

## 27.2.2 Simplified configuration definition

Dynamic channel-path management also simplifies the task of defining your configuration. Without dynamic channel-path management, you have to:

- ▶ Identify the bandwidth required for each LCU to provide acceptable performance at peak times.
- ▶ Identify LCUs that are busy at different times, and therefore are good candidates to share a channel if you are forced to have more than one LCU per channel.
- ▶ Allocate your channels among LCUs in a manner that balances the overall channel utilization as much as possible.
- ▶ Select channels that provide the best availability (different channel cards in the CPC, different switches, and so on).
- ▶ Define the desired configuration (up to a maximum of 8 paths per LCU per LP) to HCD.
- ▶ Monitor the configuration on an ongoing basis to ensure the performance and channel utilization remain within acceptable limits.

Because dynamic channel-path management can dynamically move I/O paths to the LCUs that are experiencing channel delays, you can reduce the CU/channel-level capacity planning and balancing activity that was necessary prior to dynamic channel-path management.

Using DCM, you are only required to define a minimum of one non-managed path and up to seven managed paths to each controller (although a realistic minimum of at least two non-managed paths are recommended), with dynamic channel-path management taking responsibility for adding additional paths as required, and ensuring that the paths meet the objectives of availability and performance.

## 27.2.3 Reduced skills requirement

It is still necessary to understand the performance and availability characteristics of all your installed hardware. DCM can only work within the confines of the physical connectivity that you design and implement. However, to the extent that DCM is self-tuning, it should be possible to spend considerably less time on configuration management, monitoring, and capacity planning. This allows you to free up scarce technical resources to work on other tasks.

## 27.2.4 Maximize utilization of installed resources

As one part of the continuing drive to reduce the cost of computing for the zSeries platform, DCM can help you drive more traffic to your DASD subsystems without necessarily having to invest in additional channels. DCM may let you increase the overall average utilization of your channels, without adversely impacting the response time for the connected subsystems.

For example, in the chart on the left of Figure 27-4 on page 554, some channels are heavily utilized (resulting in channel contention for the attached control units), while others are nearly idle.

In the chart on the right, the overall average utilization is, however, more consistent. This is because DCM will help a busy channel by adding an idle channel to the control units that it is attached to. This increases the utilization on the idle channel, and decreases the utilization on the busy one. The net effect is improved overall response times (because the peaks should not be so high), while driving more value from the installed channels (because idle ones will be put to work helping the busy ones). You will also notice that even though the peak utilization has decreased, the number of channels in the DCM scenario has *decreased*, from 14 down to 10, thus allowing you to deliver similar service levels with fewer channels.

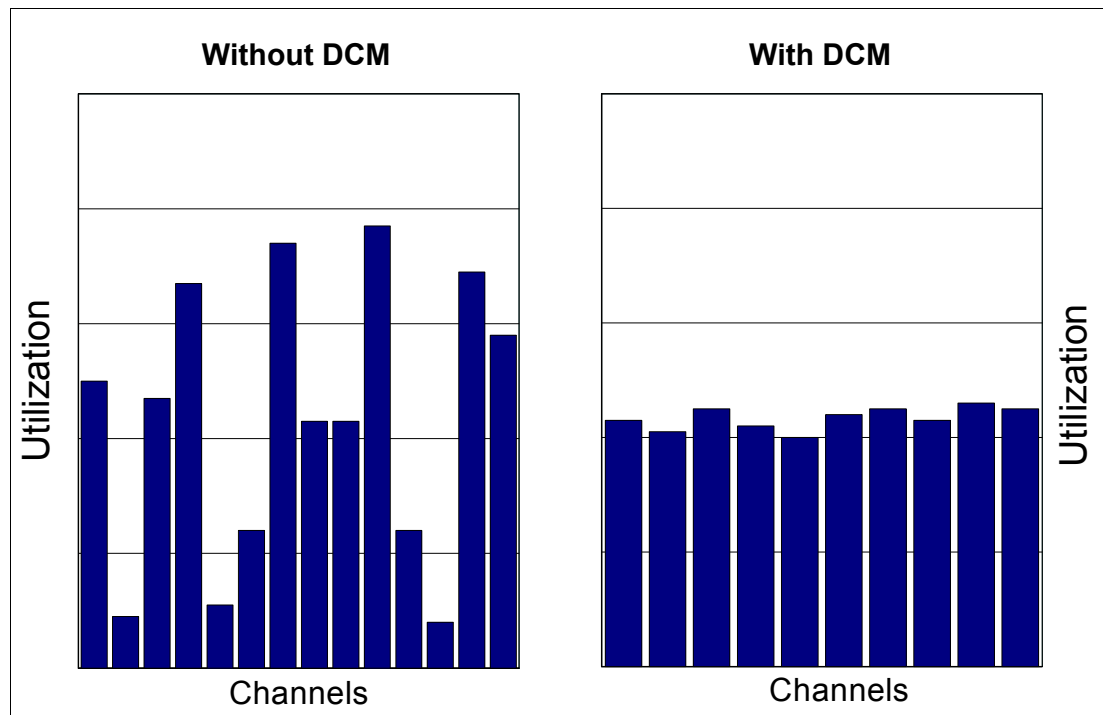


Figure 27-4 Improved channel utilization

### 27.2.5 Enhanced DASD subsystem availability

When attaching a DASD subsystem to a zSeries CPC, there are a number of things you should take into account, if you wish to achieve maximum availability and performance from the configuration. Within the CPC, there are a number of redundant components provided (channel cards, STIs, MBAs, and so on) as depicted in Figure 27-5 on page 555. Ideally, you would attach a control unit to channels that are spread over as many of these components as possible, minimizing the number of single points of failure.



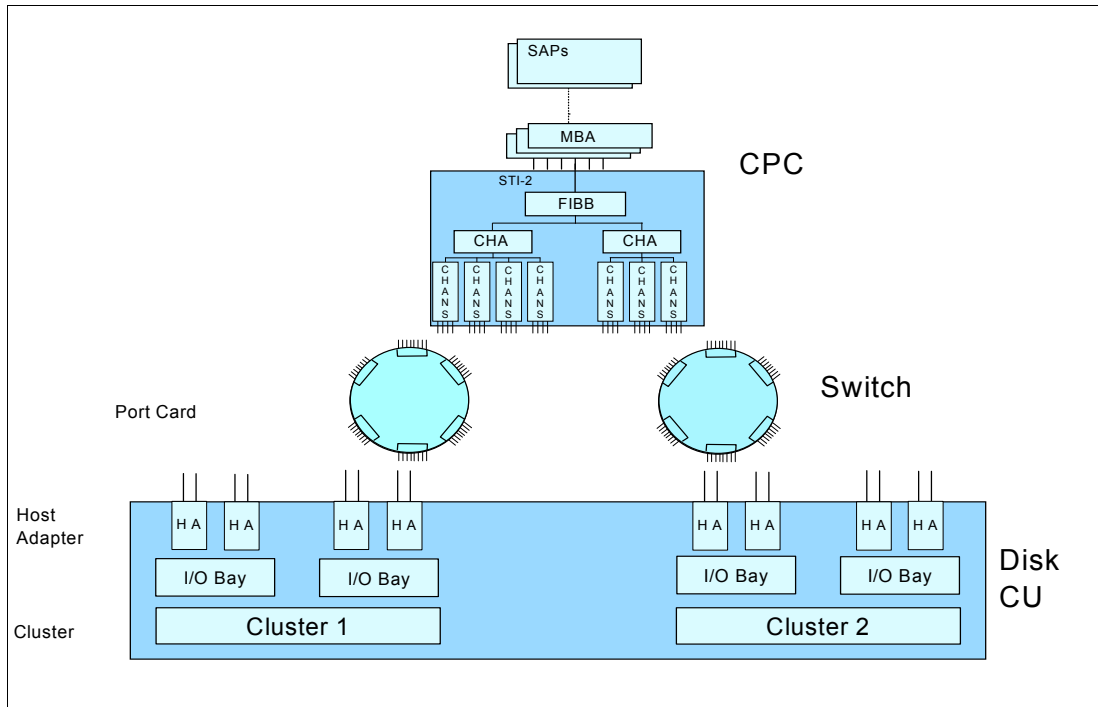


Figure 27-5 Availability considerations

Similarly, in the switch and the DASD controller itself, there are various availability characteristics that you must be aware of when designing the physical connectivity.

If you currently go to the trouble of configuring every path to make the best use of the provided availability features, you are probably in the minority. DCM can help you make the best use of the installed availability features, without the time investment that was required prior to DCM. As we stated previously, you still have to take these hardware features into account when planning your physical connectivity; however, DCM can reduce the effort you have to put into deciding which paths should be used for each control unit, and which control units should be placed on the same paths. This reduces your workload and provides you with similar or better reliability and availability.

Another advantage of DCM is that it works all this out using support provided in the hardware components (CPCs, switches, and control units), so every time you install a new device, you do not have to worry about reconfiguring your paths to match its characteristics. It should be sufficient to follow the connectivity recommendations provided in the installation documents associated with the device when you are designing the physical connectivity. IBM is working with non-IBM hardware vendors to ensure that they also provide information in a format that DCM can use to identify potential single points of failure.

### 27.2.6 Reduced requirement for more than 256 channels

For most installations that require more than 256 channels, that requirement is rarely driven by a need for more than 256 channels worth of bandwidth—current processors do not have sufficient power to drive even 256 channels flat out, so providing more channels would not result in more work being done. Similarly, it is usually not driven by addressability considerations—for most installations, 256 channels provide connectivity for more devices than they will attach to a single CPC. In most cases, the requirement is driven by the complexity of configuring every channel with multiple control units, while also ensuring acceptable performance for every control unit.

DCM can address this last issue—configuring every channel with the maximum number of devices while still delivering acceptable performance across all the control units. Because DCM dynamically adjusts to changing workloads, if you happen to end up with multiple busy control units on a channel (a situation that can easily happen today and that will result in degraded performance), DCM will react to the resulting increased Pending time (more commonly known as *Pend time*) by adding idle or less busy channels to the control units that are suffering high Pend times.

### 27.2.7 WLM role in dynamic channel-path management

The z/OS Workload Manager (WLM) plays an active role in Dynamic Channel-path Management, regardless of whether the system is in WLM Goal mode or WLM Compatibility mode. The diagram in Figure 27-6 shows the role of WLM in both modes.

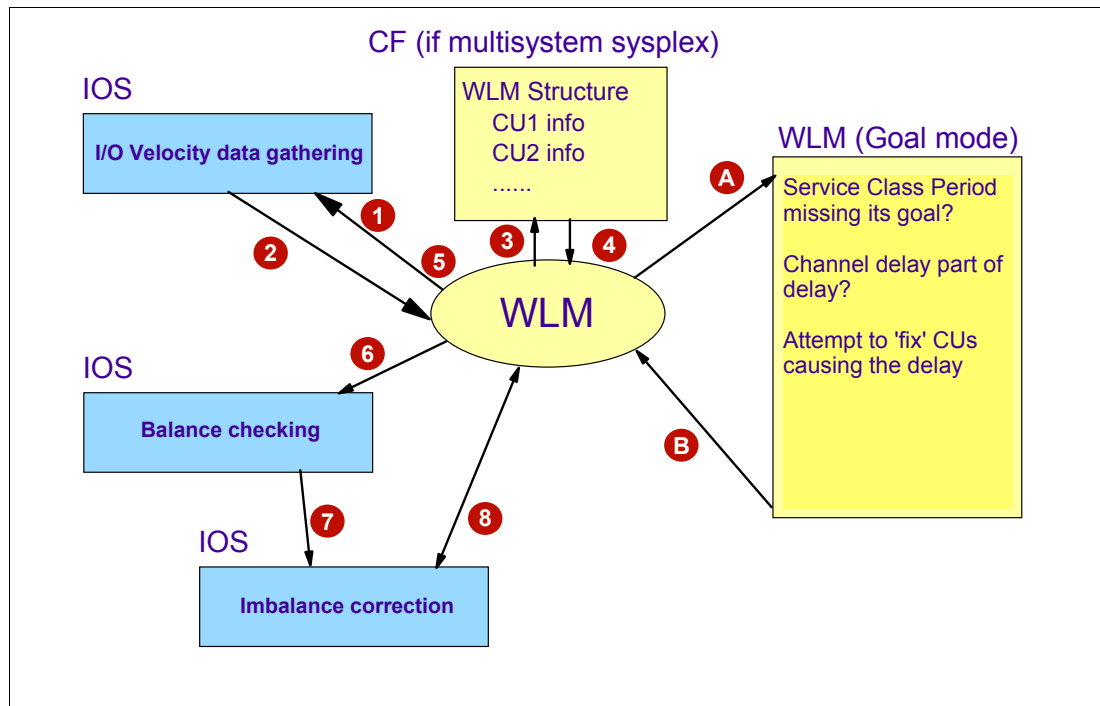


Figure 27-6 WLM role in DCM

WLM is responsible for initiating the process of gathering performance information for all DASD subsystems (Step 1). Together with IOS, WLM calculates a metric, known as the *I/O velocity* (a value that is used to objectively measure the impact of channel contention), for every DASD subsystem (Steps 2 and 5). After this information has been gathered for all DASD subsystems, and optionally kept in a CF structure (Steps 3 and 4), IOS identifies the DASD subsystems whose channel delays fall outside an acceptable range - this process is known as balance checking (Step 6). Balance checking is followed by imbalance correction (Step 7), during which actions are taken to bring the identified DASD subsystems back towards a target I/O velocity. In addition, if DCM is operating in Goal mode, WLM may decide to set explicit velocities for control units that are impacting important workloads. This is shown in Steps A and B.

## I/O velocity concepts

The I/O velocity is conceptually similar to the RMF Monitor III workflow percentage and the WLM Goal mode execution velocity in that it measures the impact of channel contention on the total I/O response time. This metric is the trigger that causes DCM to initiate a configuration change.

If the system is part of a multisystem sysplex and is on a CPC running in LPAR mode, then any decisions relating to configuration changes by DCM must take into account the use of the channels and control units by *all* the systems in the LPAR Cluster (not in the whole sysplex). WLM uses a CF structure as the mechanism by which this information is shared between all those systems. This is shown as steps 3 and 4 in Figure 27-6 on page 556.

Another function available when DCM is operating in DCM Goal mode is that if DCM needs to make a decision about a control unit that may affect another control unit with an explicit velocity, IOS will call WLM to decide which control unit is the more important. This is shown in Step 8 in the diagram.

## 27.3 FICON DCM definition

Some definitions are needed in order to describe FICON DCM functions. The following differentiation is introduced as part of this new support. From now on, there are:

- ▶ Managed channels
  - Any channel that is specified in HCD as being managed by DCM. That is, the channel is defined as being managed and no control units are explicitly defined as being attached to that channel.
  - Managed attribute is assigned in HCD.
  - Defined as shared and not spanned.
  - Associated with a specific LPAR cluster
  - Must be connected to a switch.
- ▶ Non-managed channels
  - This is the term for traditional channels that are not managed by DCM.
  - Non-managed channels are sometimes also referred to as static channels.
  - Each control unit within a managed subsystem must have a least one static channel path defined. IBM recommends two static channels.
  - Static channel paths can be shared by systems outside of the LPAR cluster.
- ▶ Managed Control Unit
  - Any control unit that is specified in HCD as being managed by DCM.
  - That is, some of the CHPIDs that the control unit is attached to are defined in HCD using an asterisk (\*) rather than a specific CHPID number. “\*” indicates that that path is to use a managed channel. That is, some of the CHPIDs that the control unit is attached to are defined in HCD using an asterisk (\*) rather than a specific CHPID number.
  - Managed CUs are identified as control units with DASD devices in common. As with managed channel paths, managed subsystems are defined through HCD as being capable of accepting managed channel paths or having them removed or replaced.

- The control unit definition must specify a number of managed channel paths that can be connected and the managed subsystem must be switch-connected. (Cascaded connectivity is not supported.)
- ▶ Non-managed Control Unit
  - Control units that are defined in HCD as having no managed channel defined to them.
  - Non-managed control units can also be referred to as *static control units*.
- ▶ LPAR cluster or I/O cluster
  - Set of system images on a processor defined within the same sysplex.
  - Image to cluster association is registered at image IPL.
  - Defines the scope of DCM activity.
  - Cluster name is the sysplex name.

Refer to Figure 27-7.

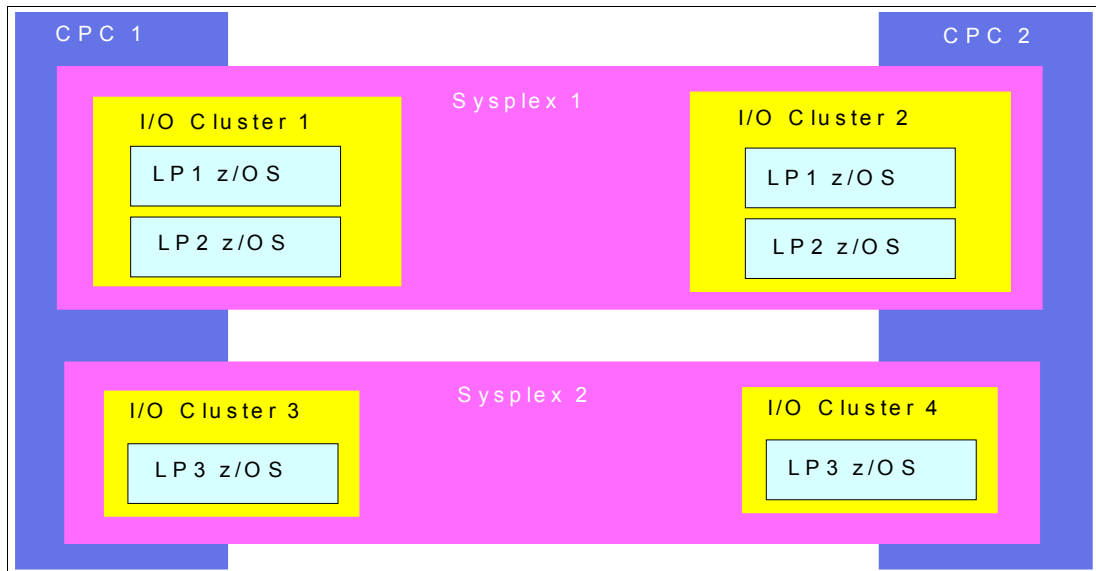


Figure 27-7 LPAR, I/O clusters

- ▶ I/O Cluster
  - An I/O Cluster is the group of z/OS systems, running in z/Architecture mode on a single IBM zSeries CPC that are all in the same sysplex. It is possible to have more than one I/O Cluster per CPC if there are members of more than one sysplex on that CPC. At the time of writing, the term I/O Cluster is synonymous with LPAR Cluster.
- ▶ LPAR Cluster
  - The set of one or more LPs, running on one IBM zSeries 900 or later CPC, that are in the same sysplex and are running z/OS 1.1 or later in z/Architecture mode. The scope of an LPAR Cluster is currently the same as the scope of an I/O Cluster; however, the term LPAR Cluster is also used in relation to WLM LPAR CPU Management and Channel Subsystem I/O Priority Queuing.

## DCM management

Figure 27-8 on page 559 depicts the scope of DCM management. It shows a single processor view with two LPARs or I/O clusters (A and B). LPAR cluster A has three images or LPARs: LPA1, LPA2, LPA3. LPAR cluster B has two images or LPARs: LPB1 and LPB2.

Both LPA1 and LPA2 are part of the logical channel subsystem 0 (LCSS0). LPA3 is identified as part of LCSS1. Since managed channels cannot be spanned between LCSSs, the managed channels used for LCSS0 in LPAR cluster A are different than the managed channels used for LCSS1 in LPAR cluster A. Therefore, even though all three systems belong to the same LPAR cluster, the assignment of managed channels to control units is done on a LCSS level.

The channels are identified as either S or M—S for non-managed or static and M for managed. There are also three (3) switches identified showing how the switches are connected to the host bus adapters (HBA) for the control unit. Managed channels must always be switch-connected and may be used for any managed control unit on the switch.

Four (4) control units are identified as CU1 through CU4. Each has one non-managed or static channel defined. At least one static channel is required for a managed control unit in order to recognize the control unit during z/OS IPL. However, IBM recommends that at least two static channels be defined for a control unit for availability reasons.

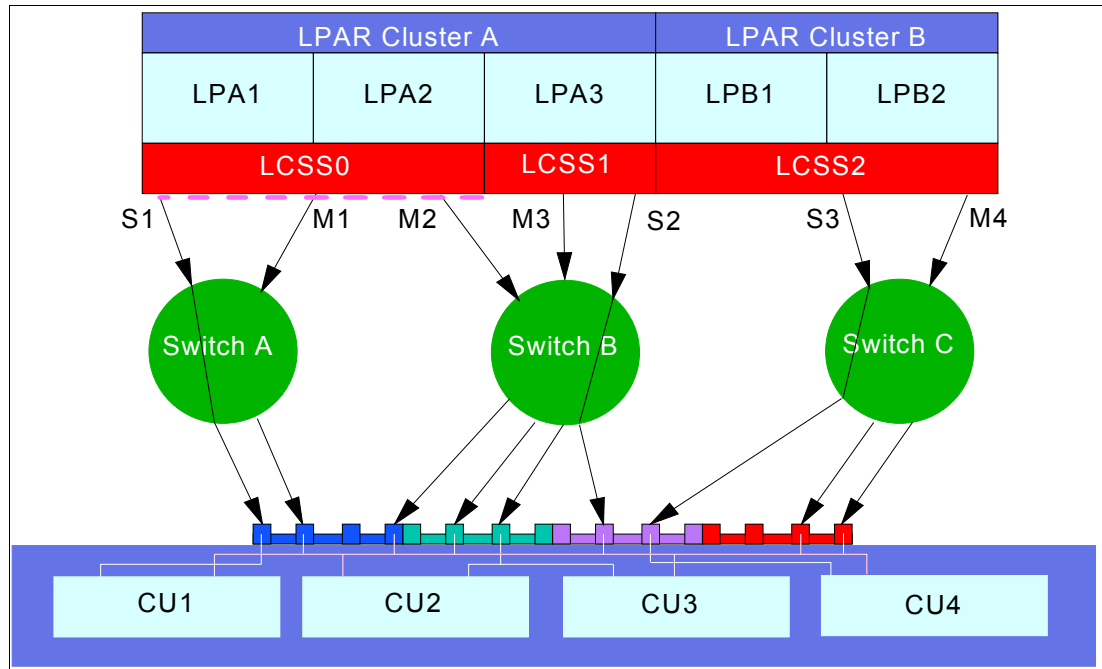


Figure 27-8 FICON DCM topology

### 27.3.1 A high-level overview of the DCM process

As depicted in Figure 27-9 on page 560, DCM per se consists of two functions, data collection and algorithms.

- ▶ Data collection collects performance data and interfaces with WLM to create running averages (over a period of time) of the performance data. Note that WLM gathers data from all systems in the LPAR (I/O) cluster.
- ▶ Data collection then starts the DCM algorithms. The algorithms are responsible for determining the performance issues with all of the defined managed CUs. It pulls topology and other information in about the channels, the switches, the control units, and link port statistics. From this information and the averaged data maintained by WLM, the algorithms determine what changes can be made (if any) to what managed CUs. A list of the potential changes is reviewed and the best change is selected. There are several criteria used to

determine the best change, including availability, single points of failure, and projected utilization.

- In this example we show the old utilizations of the two static channels and the new or projected utilizations of adding the managed channel to control unit 5000.

Once a best change is identified, DCM implements the change to the I/O configuration via dynamic I/O to add, replace, or delete the channel from the control unit (based on what the best change indicates).

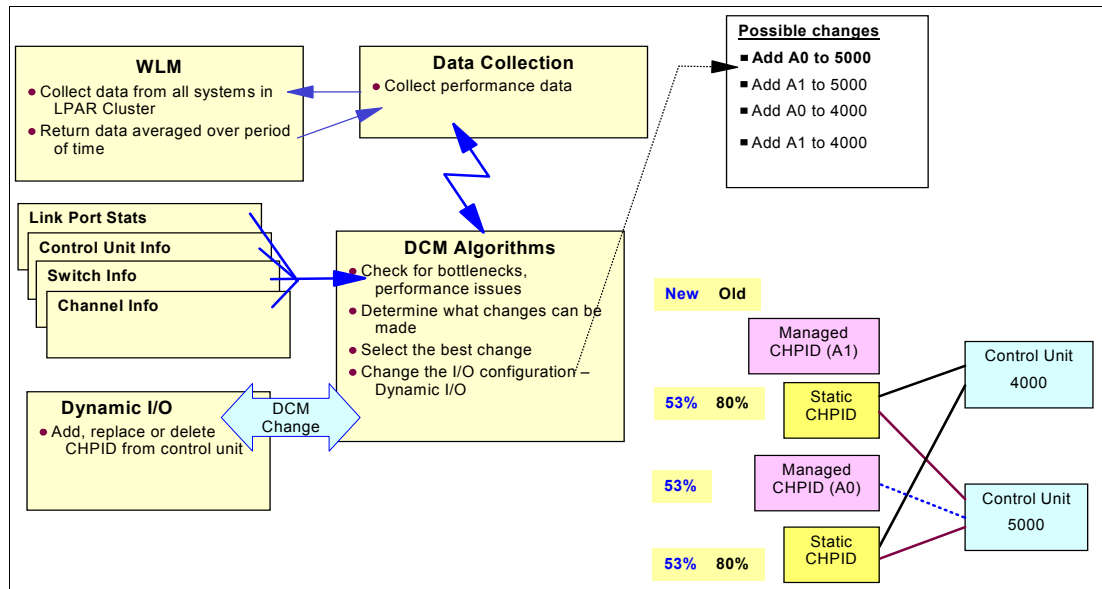


Figure 27-9 FICON DCM process

## 27.4 I/O configuration changes

In HCD, as depicted in Figure 27-10 on page 561, in the Channel Path list the existing column shows which channel paths are managed (Yes) and which channels are non-managed (No). In the same way, when adding a managed channel path, shown in Figure 27-13 on page 562, use the existing "Managed" input to designate whether the channel path will be DCM-managed or not.

In an LPAR environment, such a channel path must be shared (SHR). And since the scope of management for DCM is at the I/O cluster level (that is, LPAR Cluster), the I/O cluster definition must be defined. A managed CHPID must be dynamically switchable on a switch, otherwise an error is issued as shown in Figure 27-11 on page 561.

```

Goto Filter Backup Query Help
-----
                                Channel Path List      Row 1 of 2 More:  >
Command ==> _____ Scroll ==> PAGE

Select one or more channel paths, then press Enter. To add use F11.

Processor ID . . . . : B706STP1
Configuration mode . : LPAR
Channel Subsystem ID : 0

                                DynEntry Entry +
/ CHPID Type+ Mode+ Switch + Sw Port Con Mngd Description
_ 00   CFP  SHR   ___   ___   Y   No  _____
_ 01   CFP  SHR   ___   ___   N   No  _____
***** Bottom of data *****

```

Figure 27-10 HCD definition - Channel Path List

```

- _____ Add Channel Path _____
|
| Specify or revise the following values.
|
| Processor ID . . . . : B706STP1
| Configuration mode . : LPAR
| Channel Subsystem ID : 0
|
| Channel path ID . . . . F0   +          PCHID . . . . ___
| Number of CHPIDs . . . . 1
| Channel path type . . . . FC   +
| Operation mode . . . . DED   +
| Managed . . . . . Yes (Yes or No)  I/O Cluster _____ +
| Description . . . . . _____
|
| Specify the following values only if connected to a switch:
| Dynamic entry switch ID ___ + (00 - FF)
| Entry switch ID . . . . ___ +
| Entry port . . . . . ___ +
| F1=Help   F2=Split   F3=Exit   F4=Prompt   F5=Reset   F9=Swap
| F12=Cancel
|
| _____
| Managed channel path of processor B706STP1 must define a dynamic switch.
|

```

Figure 27-11 Error in HCD definition - managed channel path

As depicted in Figure 27-12 on page 562, any channel path that is not either ESCON or FICON cannot be managed. What appears also, is that the I/O Cluster name is mandatory to be filled—certainly nice to have but nothing more, at least at z/OS V1R11 level of the system.

```

----- Add Channel Path -----

Specify or revise the following values.

Processor ID . . . . . : B706STP1
Configuration mode . . : LPAR
Channel Subsystem ID : 0

Channel path ID . . . . . F0      +          PCHID . . . . . ____
Number of CHPIDs . . . . . 1
Channel path type . . . . . CFP      +
Operation mode . . . . . SHR        +
Managed . . . . . Yes (Yes or No)  I/O Cluster _____ +
Description . . . . . _____

Specify the following values only if connected to a switch:
Dynamic entry switch ID ____ + (00 - FF)
Entry switch ID . . . . . ____ +
Entry port . . . . . ____ +
  F1=Help    F2=Split    F3=Exit    F4=Prompt    F5=Reset    F9=Swap
  F12=Cancel

CFP channel path 0.F0 of processor B706STP1 can not be defined as
managed.

```

Figure 27-12 Error in HCD definition - managed channel path

```

----- Add Channel Path -----

Specify or revise the following values.

Processor ID . . . . . : B706STP1
Configuration mode . . : LPAR
Channel Subsystem ID : 0

Channel path ID . . . . . F0      +          PCHID . . . . . ____
Number of CHPIDs . . . . . 1
Channel path type . . . . . DF      +
Operation mode . . . . . SHR        +
Managed . . . . . YES (Yes or No)  I/O Cluster SVPLEX5_ +
Description . . . . . _____

Specify the following values only if connected to a switch:
Dynamic entry switch ID 5F + (00 - FF)
Entry switch ID . . . . . 31 +
Entry port . . . . . 20 +
  F1=Help    F2=Split    F3=Exit    F4=Prompt    F5=Reset    F9=Swap
  F12=Cancel

```

Figure 27-13 HCD definition - managed channel path

As depicted in Figure 27-14 on page 563, when adding a control unit, only one non-managed (that is, static) channel path per partition has to be defined. This is necessary for IPL and for DCM to be able to identify the CUs and devices that are to be managed. IBM recommends two static channel paths. To specify a managed channel path, simply type in an asterisk as a place holder for the channel path and leave the link address field blank.



```

Goto Filter Backup Query Help
-----
Control Unit List
Command ==> _____ Scroll ==> PAGE
View Control Unit Definition
-----
Command ==> _____ Row 1 of 8 More: >
Scroll ==> PAGE
Control unit number . : 1900          YP CUA 1
Control unit type . . : 2105          Serial number . . . : 0FCA602105
Connected switch.ports: 61.11 62.11 63.15 64.15

ENTER to continue.

-----Channel Path ID . Link Address-----
Proc.CSSID 1----- 2----- 3----- 4----- 5----- 6----- 7----- 8-----
SCZP101.0 98.15 89.11 *----- *
SCZP101.1 98.15 89.11 99.15 8B.11 *      *
SCZP101.2 98.15 89.11 99.15 8B.11
SCZP201.0 53.11 57.11 5B.15 5F.15
F1=Help      F2=Split      F3=Exit      F7=Backward  F8=Forward
F9=Swap      F12=Cancel

```

Figure 27-14 HCD definition - managed Control Unit

## 27.5 WLM changes

There is no “switch” as such in WLM to turn dynamic channel-path management on or off. So if you are running in basic mode, or the system that will be using DCM is in XCFLOCAL or Monoplex mode, there are *no* WLM-related changes required. If the system that will be exploiting DCM will be running in an LP and is in multisystem sysplex mode, you *must* have a WLM structure. If the structure is not defined or is not available, DCM will not function in this environment.

### Defining WLM structures

The structure size recommended in that section is large enough to support both DCM and WLM LPAR CPU Management. Each LPAR Cluster requires a CF cluster structure for FICON and ESCON dynamic channel-path management as well as for WLM LPAR CPU management. Because we have two LPAR Clusters, one for each CPC, we define two CF structures in the CFRM policy.

Because WLM uses System Managed Rebuild to rebuild the structure in the event of planned configuration changes, the CFRM policy must contain support for System Managed Rebuild by specifying ITEM (NAME(SMREBLD) NUM(1)).

Also, due to the use of functions that only exist in CFLEVEL=9 or higher, the WLM structure must exist in a CF running at least that level of CFCC. In order to be able to rebuild, we recommend that you upgrade *all* your CFs to this level before implementing FICON dynamic channel-path management.

```

// POLICY JOB MSGLE VEL=( 1,1)
// EXEC PGM= IXCMIAPU
// SYSPRINT DD SYSOUT =A
// SYSIN DD *
DATATYPE (CFRM) REPORT(YES)
DEFINE POLICY NAME (CTTEST1) REPLACE(YES)
...
STRUCTURE NAME(SYS ZWLM_ 27972064)
MINSIZE(4096)
INITSIZE(6144) /* 1K UNITS */
SIZE(12288) /* 1K UNITS */
PREFLIST(CF01,CF02 )
STRUCTURE NAME(SYSZWLM_ 63422064)
MINSIZE(4096)
INITSIZE(6144) /* 1K UNITS */
SIZE(12288) /* 1K UNITS */
PREFLIST(CF02,CF01 )

```

Figure 27-15 Defining WLM cluster structures

## WLM cluster structures

These WLM cluster structures (see Figure 27-15) require a CF with CFCC LEVEL 9. If PREFLIST is coded, it must specify CFs that have this level. The structure size depends on the number of LPARs to be defined and the area required for dynamic channel-path management.

The CFRM policy is created and updated using the XCF administrative utility IXCMIAPU, which is documented in *z/OS MVS Setting Up a Sysplex*, SA22-7625. The name of the structure must be SYSZWLM\_ sssstttt, where ssss is the last 4 digits of the CPC and the tttt is the type of the CPC. The following are details about the CFRM policy that defines our cluster structures:

- ▶ Our CFRM policy name is CTTEST1.
- ▶ The structure name for cluster structure 1, which is on CPC1, is SYSZWLM\_27972064. The structure name for cluster structure 2, which is on CPC2, is SYSZWLM\_63422064.
- ▶ The minimum size and initial sizes of each structure are 4,096 KB and 6,144 KB, respectively. The maximum size for both structures is 12,288 KB. The structure will initially be allocated with a size of 6,144 KB. If we need a larger structure, we can use the SETXCF ALTER operator command to increase the size of the structure up to the value specified on the SIZE parameter, which in this case is 12,288 KB.
- ▶ In our structure definition, we have placed each cluster structure in a different CF. This is seen in the PREFLIST statement. The preferred CF for cluster structure SYSZWLM\_27972064 is CF01, but it can also reside in CF02, while the preferred CF for cluster structure SYSZWLM\_63422064 is CF02, but it can also reside in CF01.

From an availability point of view, it really does not make any difference whether the structure is in the same CPC as the operating systems in the LPAR Cluster. However, from a performance point of view, it is probably better to balance the load of the two structures across the two CFs rather than placing both in the same CF. Either structure can be moved to the other CF using the SETXCF REBUILD operator command.

Once you have defined the new CFRM policy, it can be activated using the following operator command:

```
SETXCF START,POL,POLNAME=CTTEST1,TYPE=CFRM
```

## CFRM policy considerations

This makes the structure available to WLM. When a new policy is activated, an ENF 35 signal is sent on all systems in the sysplex. WLM in z/OS listens for this ENF. WLM now knows there is a cluster structure available and each system connects to the appropriate cluster structure. The CF structure is actually allocated when the first WLM instance connects to it. WLM is ready to use the cluster structure for WLM LPAR CPU Management once all the remaining software and hardware actions are completed.

Do not forget to give the RACF user ID associated with the WLM address space access to the LPAR Cluster structure. This is protected by the IXLSTR.structure\_name profile in the SAF class CLASS(FACILITY). If you have not defined a user ID for WLM, and you have a RACF profile that covers the LPAR Cluster structure, the RACF violations (ICH408I messages) will not tell you which address space is failing to obtain access.

## 27.6 HMC considerations

There are no changes to the HMC specifically for dynamic channel-path management. However, in order for an LP to use the Dynamic I/O Reconfiguration capability, you must have enabled this function in the CPC Reset Profile, as shown in Figure 27-16 on page 566.

In the CPC Reset Profile, named SCZP201, you must enable two settings. In the Dynamic selection we check the box allowing dynamic changes to the channel subsystem, as shown in Figure 27-16 on page 566.

In addition, all LPs in the LPAR Cluster must be authorized to make I/O reconfiguration changes for the CPC. Otherwise, only a subset of the systems in the LPAR Cluster would be able to initiate DCM configuration changes.

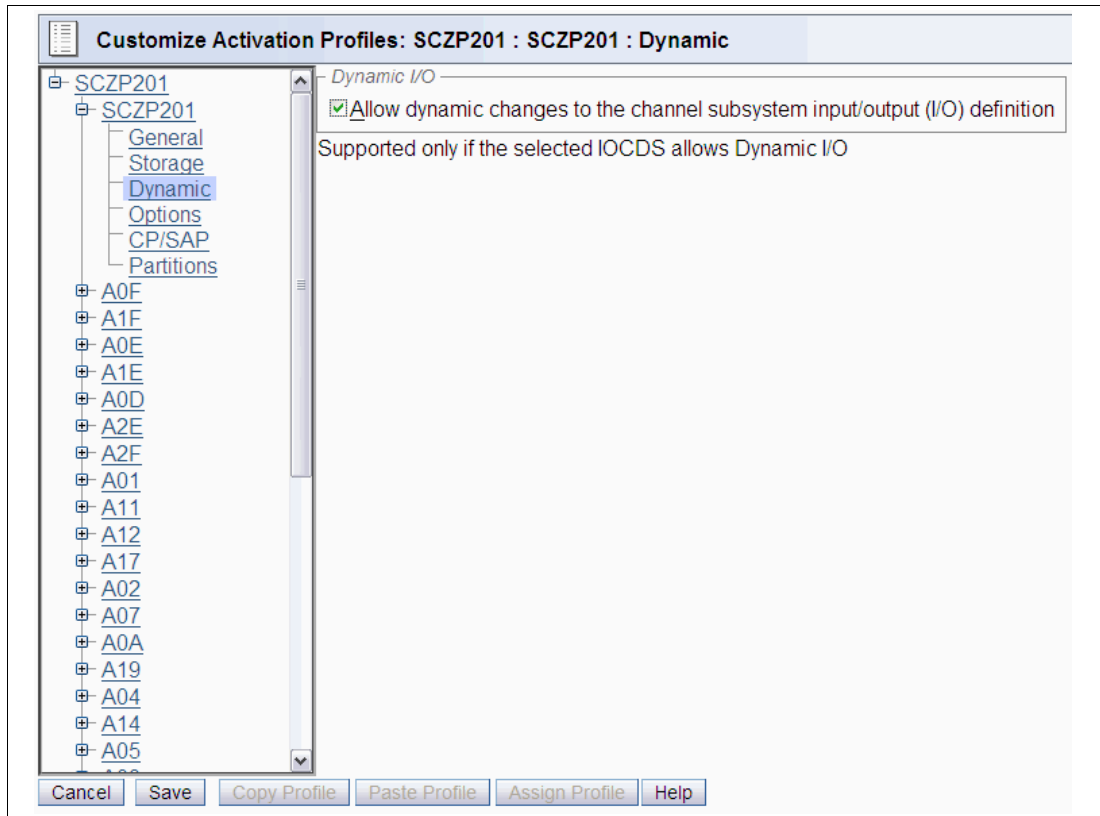


Figure 27-16 HMC changes

### Selecting the Options tab

Also, on the Options tab of the CPC Reset Profile, you must enable the “automatic input/output (I/O) interface reset” option. In the Options selection we check the box allowing automatic input/output interface resets, as shown in Figure 27-17 on page 567.

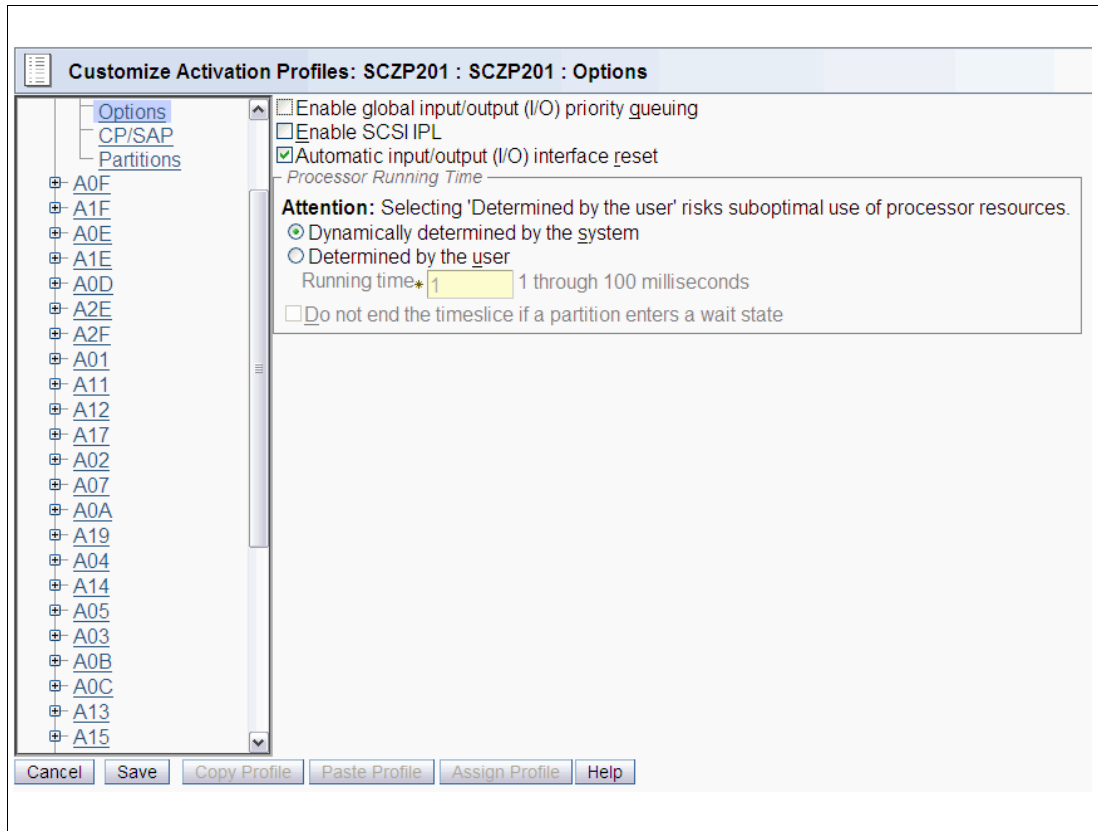


Figure 27-17 The Options tab

In the Image profile for every LP participating in the LPAR cluster you must enable Input/output configuration control. In an Image profile, named A01 in this example, in the Security selection we select the box allowing Input/output configuration control, as shown in Figure 27-18 on page 568.

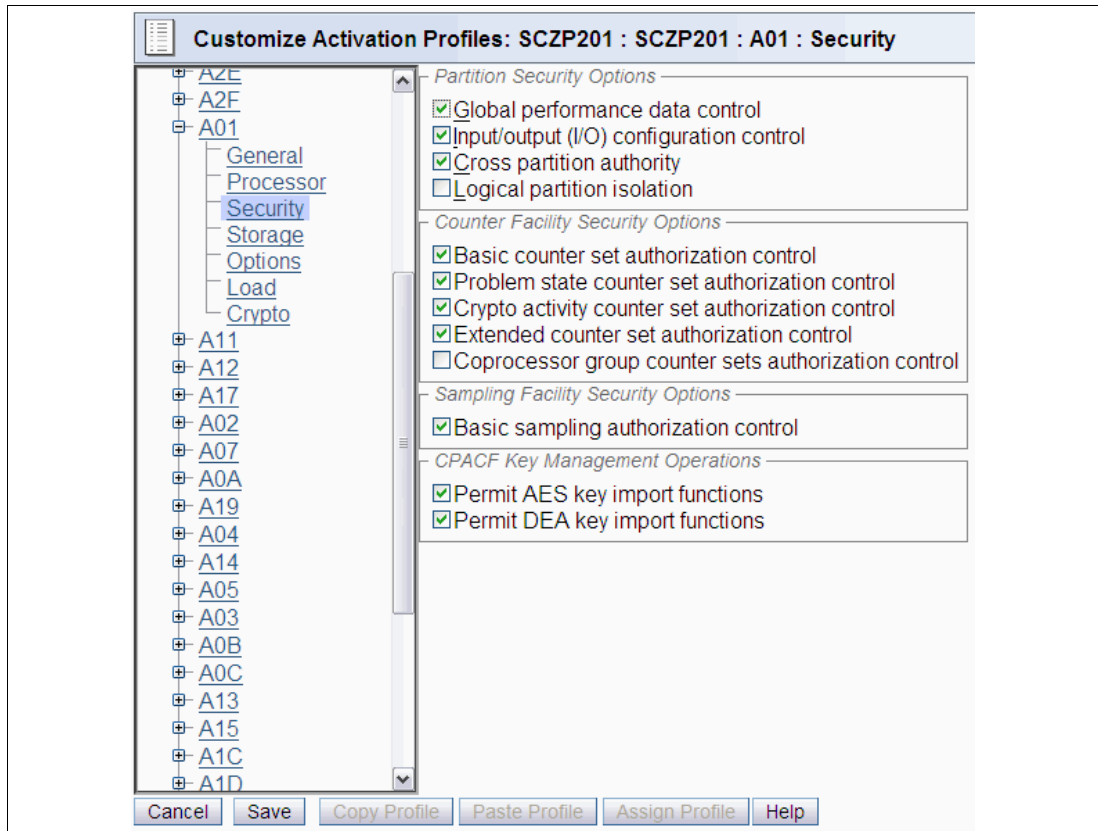


Figure 27-18 Enabling Input/output configuration control

## 27.7 Building the IOSTmmm module

The IOSTmmm load module must reside in a LNKLIST library. It contains information used to determine the points of failure within a given control unit. The mmm in the load module name represents the control unit manufacturer name as provided in the manufacturer field of the Node Descriptor. The IOSTIBM module, the one that maps the IBM control units, is shipped in SYS1.LINKLIB as part of the system, and is updated as part of the changes provided with every new IBM control unit.

For non-IBM control units, you will need to obtain a copy of the IOSTmmm load module from each DASD vendor that you use.

Updates to the IOSTmmm load module can be activated dynamically using the SETIOS DCM,REFRESH command.

## 27.8 Activating the changes

Having completed all these changes, you are now ready to activate them. First, configure offline any channels that you are converting from non-managed to managed channels. This must be done in *every* LP that is currently using the channel.

Once the channel is offline everywhere, do a Software Activate using the new IODF on all but one of the systems on the CPC. Do not forget that there could be systems in other sysplexes that might be using this channel, so *all* affected systems should be updated. When this has completed successfully, do a Hardware Activate on the final system. Obviously, this last system has to be a z/OS system.

When the Hardware Activate has completed, the managed channel will be offline, and must be configured online in every LP in the LPAR Cluster. To ensure that the change was successful, issue a **D M=CHP(xx)** command for every managed channel to ensure that it is in fact now a managed channel, and that it is online and attached to a switch. Also, issue a **D M=DEV(yyyy)** command for a device behind a managed control unit to ensure that the control unit has the expected number of managed paths defined.

The managed channels are now online and available to DCM, and they should now start being assigned to the various managed control units. You can use the operator commands to display information about the managed channels and control units, and the enhanced RMF reports to show performance information for these entities.

## 27.9 Invocations

To enable DCM, the following must be done:

- ▶ Define managed control units in the IODF as described in 27.4, “I/O configuration changes” on page 560.
- ▶ Define managed channels in the IODF that are connected to the switches for the managed control units as described in 27.4, “I/O configuration changes” on page 560.
- ▶ Define switch devices in the IODF and vary them online to z/OS.
- ▶ Ensure that at least one system in the LPAR (I/O) cluster is authorized to make dynamic configuration changes, located in the LPAR image profile on the HMC.

DCM is enabled automatically (at IPL time) when:

1. A managed control unit is defined in the IODF and specifies at least one non-managed (static) channel. IBM recommends that two non-managed channels be defined. Ensure that managed channels can be assigned to the control unit by specifying one or more asterisks for the channels. An asterisk indicates a managed channel.
2. A managed channel is defined in the IODF by specifying YES in the managed attribute when defining the channel. Ensure that an I/O or LPAR Cluster name is specified and remember that managed channels must be switch-attached. (For a cascaded environment, the managed CU must have some ports connected to the entry switch.)
3. Switch devices are defined in the IODF that are connected to managed channels and managed CUs. It is important to remember that a non-managed channel must be defined to access the control unit port or CUP. The CUP is used by DCM to retrieve switch topology information.
4. Finally enable at least one partition in the I/O or LPAR Cluster to make dynamic I/O configuration changes. This is located in the LPAR image profile on the HMC. For z9 this is the input/output configuration control option. Refer to the PR/SM Planning Guide for your processor.

When DCM was initially released for ESCON channels, new commands and updates to existing commands were introduced. These are commands to display information about the configuration and commands to change the configuration. This section is a review of the commands since they will prove useful with FICON DCM. The commands presented here are

not new and were not updated for FICON DCM. They were either introduced when DCM was introduced or updated for DCM in z/OS V1R1.

## 27.9.1 SETIOS DCM=ON/OFF

The SETIOS DCM=ON/OFF command allows a change to the DCM status after IPL:

```
S703 SETIOS DCM=ON
S703 IOS353I 23.09.00 DCM STATUS 342
DYNAMIC CHANNEL PATH MANAGEMENT IS ACTIVE IN GOAL MODE
```

SETIOS DCM=OFF:

```
S703 SETIOS DCM=OFF
S703 IOS358I DYNAMIC CHANNEL PATH MANAGEMENT HAS BEEN TURNED OFF
```

**Important:** There is no parmlib member allowing to specify DCM ON/OFF.

Hence, if you wish to change the status of DCM after the IPL, use the SETIOS command. Do not forget that this command has an LPAR Cluster-wide scope. Issuing the command on any system in the LPAR Cluster will affect all systems in the LPAR Cluster.

## 27.9.2 Display IOS,DCM

The D IOS,DCM command shows the status of DCM on this system, as shown in Figure 27-19), and in the LPAR Cluster. This command is a bit different from most commands you are familiar with, as most commands display information about either a single system, or about the whole sysplex. This command displays information about a subset of systems in the sysplex (the LPAR Cluster).

### DISPLAY IOS,DCM

```
IOS353I 23.10.05 DCM STATUS 462
DYNAMIC CHANNEL PATH MANAGEMENT IS NOT ACTIVE
CF CONNECTIVITY ERROR IN MULTISYSTEM CONFIGURATION

IOS353I 23.10.05 DCM STATUS 462
DYNAMIC CHANNEL PATH MANAGEMENT IS NOT ACTIVE
TURNED OFF BY A COMMAND

IOS353I 23.10.05 DCM STATUS 462
DYNAMIC CHANNEL PATH MANAGEMENT IS NOT ACTIVE
DYNAMIC CHANGES TO THE CHANNEL SUBSYSTEM NOT ALLOWED IN THIS LOGICAL PARTITION

IOS353I 23.10.05 DCM STATUS 462
DYNAMIC CHANNEL PATH MANAGEMENT IS NOT ACTIVE
DYNAMIC CHANGES TO THE CHANNEL SUBSYSTEM NOT ALLOWED

IOS353I 23.10.05 DCM STATUS 462
DYNAMIC CHANNEL PATH MANAGEMENT IS NOT ACTIVE
DYNAMIC CHANNEL PATH MANAGEMENT DECISIONS CAN NOT BE MADE ON SYSTEM = sys_name
```

Figure 27-19 DISPLAY IOS,DCM



## DISPLAY IOS,DCM command considerations

One thing to be aware of when using this command is that if you issue it while a system in the LPAR Cluster is in the middle of IPLing and has not yet joined the IOS XCF group, the other systems in the LPAR Cluster will be aware that a new member is about to join the group and will cease making any configuration changes until the new system successfully joins the group. During the interval, if you issue the D IOS,DCM command, you may get the response indicated in the first message. While this message may appear to indicate an error, if one of the members of the LPAR Cluster is IPLing, it is actually informing you that one of the members of the cluster is not yet connected to the LPAR Cluster CF structure, which is correct at that time.

If the SETIOS DCM=OFF command has been issued, then the message (2nd message in Figure 27-19 on page 570) is displayed. The 3rd and 4th messages are an indication that "Allow dynamic changes to the channel subsystem" has not been enabled in the Support Element (SE) activation profile for the system or the Logical Partition of the system.

The final message of Figure 27-19 on page 570 indicates that even though dynamic channel path management is active, the algorithm portion of DCM cannot run on this system in a multisystem configuration. For example, there are no switches online or there is the possibility also that there is no dynamic I/O capability.

While an example is not shown here, devices that do not have channel path measurement blocks will also be listed. It could be only one device, a range of devices, or more than one device not in a range, that do not have channel path measurement blocks, which contain information unique to each system. This data collection process takes place on every system in the LPAR Cluster. A device or devices with channel path measurement blocks cannot be measured and thus cannot participate in DCM.

## DCM messages

Refer to message IOS353I in *z/OS MVS System Messages, Volume 9, SA22-7639* for all the possible reasons DCM would or would not be active.

For backup purposes, here is a list of those reasons:

### ► DYNAMIC CHANNEL PATH MANAGEMENT IS NOT ACTIVE Reason(s)

Dynamic channel path management is not active or not capable of managing within a specific partition, where Reason(s) can be one or more of the following:

- DYNAMIC CHANNEL PATH MANAGEMENT DECISIONS CAN NOT BE MADE ON SYSTEM = sys\_name

Dynamic channel path management algorithms cannot run on this system in this multisystem configuration.

- FACILITY IS NOT SUPPORTED

Dynamic channel path management facility is not supported.

- NO MANAGED CHANNEL PATHS DEFINED

No managed channel paths are defined.

- NO CONFIGURATION TOKEN OR AN INCOMPATIBLE TOKEN IN HSA

Configuration token is not defined or there is an incompatible token in the hardware system area.

- TURNED OFF BY A COMMAND

Dynamic channel path management was turned off by a command.

- CHANNEL TABLE COULD NOT BE BUILT  
There was an error building the channel table on the system on which the command was issued.
- SWITCH TABLE COULD NOT BE BUILT  
There was an error building the switch table on the system on which the command was issued.
- NO MANAGED SUBSYSTEMS ARE DEFINED OR ACCESSIBLE  
There are no managed subsystems defined or managed subsystems are not accessible.
- NO MANAGED SUBSYSTEMS ARE DEFINED OR ACCESSIBLE IN THIS LOGICAL PARTITION  
There are no managed subsystems defined or managed subsystems are not accessible in this logical partition of the LPAR Cluster.
- FUNCTION IS NOT AUTHORIZED  
Dynamic channel path management function is not authorized on this logical partition of the LPAR Cluster.
- FUNCTION IS NOT AUTHORIZED IN THIS LOGICAL PARTITION  
Dynamic channel path management function is not authorized in this logical partition of the system.
- CF CONNECTIVITY ERROR IN MULTISYSTEM CONFIGURATION  
CF connectivity error exists in the configuration of the multisystem LPAR Cluster. At least one system does not have connectivity to the SYSZWLM\_xxxxxyyy structure.
- DYNAMIC CHANGES TO THE CHANNEL SUBSYSTEM NOT ALLOWED  
"Allow dynamic changes to the channel subsystem" has not been enabled in the Support Element (SE) activation profile.
- DYNAMIC CHANGES TO THE CHANNEL SUBSYSTEM NOT ALLOWED IN THIS LOGICAL PARTITION  
"Allow dynamic changes to the channel subsystem" has not been enabled in the Support Element (SE) activation profile in this logical partition of the system.
- SYSTEM IS NOT A MEMBER OF A DYNAMIC CHANNEL PATH MANAGEMENT GROUP  
The Dynamic Channel Path Management group has not been joined by this system.
- NEIGHBOR NODE TOPOLOGY COULD NOT BE BUILT  
There was an error building the neighbor node descriptor topology on the system on which the command was issued.
- THE FOLLOWING DEVICE(S) DO NOT HAVE MEASUREMENT DATA: Dddd {, Dddd-Dddd}  
Displays the devices which do not have measurement data defined. It could be only one device or more than one device in a range, or more than one device not in a range, where Dddd {, Dddd-Dddd} is the device number.
- **WARNING: DISPLAY IS INCOMPLETE DUE TO SYSTEM ERROR**  
An error occurred while displaying the multiline message. Display of the system command is incomplete.

### 27.9.3 Vary switch command

The VARY SWITCH command is used to make a switch port available or unavailable for DCM to use, as follows:

- ▶ LPAR Cluster in scope.
- ▶ When a port is made unavailable, all managed channels using that port will be removed from their control units.
- ▶ If static paths use the port, a list of devices and paths that need to be varied online or offline is displayed (this is not done automatically by the command).

The VARY SWITCH command is used to tell Dynamic Channel-path Management on all systems in the LPAR Cluster whether it is allowed to use the specified port on the switch or not. It is only necessary to issue the command on one system in the cluster—IOS looks after routing the command to all systems in that cluster.

Specifying DCM=OFFLINE will stop DCM from setting up a managed path using this port. If it is already using this port for an existing managed path, the associated path is varied offline and removed from the control units.

When a VARY SWITCH,OFFLINE command (Figure 27-20) completes and there are static device paths defined and online using that switch port, the command response includes a list of those devices that need to have the paths varied offline.

```
S703 VARY SWITCH(B000,60),DCM=OFFLINE
S703 IEE632I SWITCH B000
IEE633I SWITCH B000, PORT 60, DCM STATUS=OFFLINE
ATTACHED NODE = 003990.OCC.IBM.XG.000000000006
THE FOLLOWING DEVICE PATHS ARE ONLINE THROUGH THIS PORT:
(0220,58)
```

Figure 27-20 VARY SWITCH OFFLINE

When a VARY SWITCH,ONLINE command completes, as shown in Figure 27-21), if there are static device paths defined and offline using that switch port, the command response will include a list of those devices that need to have paths varied online.

```
S703 VARY SWITCH(B000,60),DCM=ONLINE
S703 IEE632I SWITCH B000
IEE633I SWITCH B000, PORT 60, DCM STATUS=ONLINE
ATTACHED NODE = 003990.OCC.IBM.XG.000000000006
THE FOLLOWING DEVICE PATHS ARE OFFLINE THROUGH THIS PORT:
(0220,58)
```

Figure 27-21 VARY SWITCH ONLINE

Because Dynamic Channel-path Management is solely responsible for the managed paths, you cannot use the Vary Path command to either vary on or vary off a managed path. If you try to use this command with a managed path, the command will be rejected, as in the examples of Figure 27-22 on page 574.

```
VARY PATH(980,5E),ONLINE
IEE777I VARY PATH REJECTED, CHPID 5E DEFINED AS MANAGED

VARY PATH(980,5E),OFFLINE
IEE777I VARY PATH REJECTED, CHPID 5E DEFINED AS MANAGED
```

Figure 27-22 VARY PATH REJECTED

CONFIG CHP does allow for the disabling of a managed channel path. You may want to do this for several reasons, including dynamic configuration changes or serviceability needs or processes.

It is important to understand that DCM assumes that the state of any given managed channel is consistent across all of the systems, that is to say, if it is offline, then it is offline on all systems in the LPAR Cluster, and conversely if it is online then it is online for all systems in the LPAR Cluster. Any inconsistency will cause DCM to fail when DCM attempts to make a change affecting that managed channel.

Also note that when a managed channel is configured offline, then DCM removes that managed channel from all of the managed control units that it is connected to.

**Important:** About configuring managed channel paths on and offline: the state of the managed channels is consistent across all systems in the LPAR Cluster (actually the LCSS of the LPAR Cluster). The managed channel is online on all systems or offline on all systems. Failure to do so will cause DCM to fail while making changes.

A CONFIG OFF of a managed CHPID will cause DCM to remove the CHPID from all Control units that it is attached to.

## 27.9.4 Display commands

The DISPLAY M=CHP command has been updated to provide additional information related to Dynamic Channel-path Management, as depicted in Figure 27-23 on page 575. If the D M=CHP command is issued without specifying a CHPID number, a two-part report will be provided. The first part indicates the status of each channel (online, offline, managed, and so on), and the second part indicates the channel type (ESCON, CTC, FICON, and so on). Managed channels are indicated in the first part of the report by an asterisk (\*) if the channel is managed and online, and by a “#” if the channel is managed and offline.

```

D M=CHPD

CHANNEL PATH STATUS
  0 1 2 3 4 5 6 7 8 9 A B C D E F
0  + + + + + + + . . . + + + +
1  . . + . + . + . . . . . . .
...
D  - + . . + . * + + . . . + + + +
E  + + + + . . . + + + + + . .
F  + + + + + + + . . . . . . .
***** SYMBOL EXPLANATIONS *****
+ ONLINE      @ PATH NOT VALIDATED  - OFFLINE      . DOES NOT EXIST
* MANAGED AND ONLINE  # MANAGED AND OFFLINE
CHANNEL PATH TYPE STATUS
  0 1 2 3 4 5 6 7 8 9 A B C D E F
0  00 22 00 00 00 22 22 22 00 00 00 00 00 22 22
22 22 11 00 11 00 11 22 22 00 00 00 00 00 22 22
...

```

Figure 27-23 D M=CHP

If you specify a CHPID number on the display command, as depicted in Figure 27-24 on page 575, the response will include the channel type, whether it is online or not, the status of any devices on the channel, and the device number of the switch it is attached to (if it is attached to a switch). The provision of the switch device number is new, and this information will be used to display information about the switch configuration. If you specify a range of CHPID numbers, you will get the same information, but for every CHPID that you specified.

```

D M=CHP(C5)

IEE174I 10.53.40 DISPLAY M 943
CHPID C5: TYPE=1B, DESC=MANAGED FICON SWITCHED, ONLINE
DEVICE STATUS FOR CHANNEL PATH C5
  0 1 2 3 4 5 6 7 8 9 A B C D E F
0300 + + + + + + + + + + + + + +
0301 HA HA HA HA HA HA HA HA HA HA HA HA HA HA HA HA
0320 + + + + + + + + + + + + + +
0321 HA HA HA HA HA HA HA HA HA HA HA HA HA HA HA HA
SWITCH DEVICE NUMBER = B55F
DEFINED ENTRY SWITCH - LOGICAL SWITCH ID = 5F
ATTACHED ND = 006140.001.MCD.01.0000013A2795
PHYSICAL CHANNEL ID = 03F1
***** SYMBOL EXPLANATIONS *****
+ ONLINE      @ PATH NOT VALIDATED  - OFFLINE      . DOES NOT EXIST
* PHYSICALLY ONLINE  $ PATH NOT OPERATIONAL
BX DEVICE IS BOXED          SN SUBCHANNEL NOT AVAILABLE
DN DEVICE NOT AVAILABLE     PE SUBCHANNEL IN PERMANENT ERROR
HA DEVICE IS A HYPERPAV ALIAS HU HYPERPAV ALIAS UNUSABLE

```

Figure 27-24 D M=CHP(xx)

The DISPLAY M=DEV command was expanded to include an indicator of whether a given path is managed or not, as depicted in Figure 27-25 on page 576. It also lists the maximum number of managed paths that can be used by the control unit the device is attached to.

Additional information provided for each path includes the CUADD of the LCU the device is attached to (in the DEST LOGICAL ADDRESS field) and the port on the switch that the associated control unit host adaptor port is attached to (in the DEST LINK ADDRESS field).

**D M=DEV(3000**

```

IEE174I 11.03.47 DISPLAY M 242
DEVICE 3000 STATUS=ONLINE
CHP          AB  A8  95  C5
ENTRY LINK ADDRESS  73  F1  04  31
DEST LINK ADDRESS  42  B2  5E  5E
PATH ONLINE        Y   Y   Y   Y
CHP PHYSICALLY ONLINE Y   Y   Y   Y
PATH OPERATIONAL   Y   Y   Y   Y
MANAGED            N   N   Y   Y
CU NUMBER          3000 3000 3000 3000
MAXIMUM MANAGED CHPID(S) ALLOWED:  4
DESTINATION CU LOGICAL ADDRESS = 10
SCP CU ND          = 002107.900.IBM.75.000000002471.0143
SCP TOKEN NED      = 002107.900.IBM.75.000000002471.1000
SCP DEVICE NED     = 002107.900.IBM.75.000000002471.1000
HYPERPAV ALIASES CONFIGURED = 16
FUNCTIONS ENABLED = MIDAW, ZHPF

```

Figure 27-25 D M=DEV(xxxx)

The D M=SWITCH command allows you to display information about the selected switch and the components that are attached to it (channels, control units, or other switches), as depicted. If a switch device number is specified without a port address, a matrix is displayed that shows the status of each port. This includes status from the switch itself (the port is blocked or offline) and status set as a result of a VARY SWITCH command (DCM not allowed by operator).

**D M=SWITCH(B55F)**

IEE174I 12.19.10 DISPLAY M 410

SWITCH B55F, PORT STATUS

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	.	.	.	.	c	c	u	u	c	c	u	u	u	c	+	u
1	u	u	u	u	c	u	u	u	c	c	c	u	u	u	c	u
2	u	c	u	u	+	u	u	u	+	u	u	u	u	u	u	u
3	+	c	u	u	c	c	u	c	c	p	u	u	u	u	u	u
4	c	+	u	u	u	u	u	c	u	+	u	c	u	c	c	u
5	u	c	c	u	u	c	u	u	u	c	u	u	u	u	+	u
6	u	u	+	u	c	u	c	u	u	+	c	u	u	c	u	u
7	u	u	u	u	u	u	u	c	u	u	u	c	c	u	+	c
8	u	u	+	c	.	.	.	.	u	u	u	u	u	u	u	u
9	u	u	u	x	.	.	.	.	.	.	.	.	.	.	.	.
A	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
B	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
C	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
D	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
E	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
F	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.

\*\*\*\*\* SYMBOL EXPLANATION \*\*\*\*\*

- + DCM ALLOWED                    - DCM NOT ALLOWED BY OPERATOR
- x NOT DCM ELIGIBLE            p DCM NOT ALLOWED DUE TO PORT STATE
- c CHANNEL ATTACHED          \$ UNABLE TO DETERMINE CURRENT ATTACHMENT
- u NOT ATTACHED                . DOES NOT EXIST

Figure 27-26 D M=SWITCH

If you specify a port address on the display command (D M=SWITCH(ssss,pp)), the response includes the node descriptor of the control unit that is connected to the port, as well as the DCM status of the port.

It is important to understand that the Switch itself has no knowledge of DCM, and is not involved in the decision as to whether DCM can use a specific port. The information about whether DCM can use a specific port or not is held entirely within the operating systems in the LPAR Cluster. It is entirely possible that one LPAR Cluster will not be allowed to use a specific port, but a different LPAR Cluster will be able to use it. The information provided in the response to the D M=SWITCH command only reflects the status of the LPAR Cluster that it was issued in.

This command shows a status of DCM NOT ALLOWED DUE TO PORT STATE for any port that has been blocked, is in a dedicated connection, or was varied offline from the director (switch) console.

A port status of DCM ALLOWED simply indicates that the port is connected to a DASD control unit. It does not mean that the control unit is a managed control unit, or even that it is a control unit type that is capable of being managed by DCM.

## 27.10 Interactions and dependencies

The hardware dependencies are as follows:

- ▶ Processor - All currently supported processors (z900 and up).
- ▶ Channels - All currently supported FICON channels.
- ▶ Coupling Facility required if running multisystem.
- ▶ Managed control units must be switch-connected.
- ▶ Managed channels cannot span. The scope of a managed channel is the LCSS within the LPAR Cluster.
- ▶ Non-managed channels
  - Can be shared or spanned.
  - One non-managed (static) channel must be defined to a managed CU - IBM recommends two non-managed channels.
- ▶ Existing ESCON DCM function is not affected.
- ▶ Switches
  - Must have control unit port (CUP) function.
  - Must be defined in the IODF.
- ▶ Control Unit - No special microcode needed.

The software dependency is that all systems in an LPAR Cluster must be z/OS V1R11 with FICON DCM APAR OA28321 installed.

FICON DCM will be enabled only when all systems in the LPAR Cluster are at the z/OS V1R11 level. If a system in the LPAR cluster falls back to a prior level, DCM will be disabled.

## 27.11 Installation and planning for DCM

When planning for DCM, it is suggested that the installation “start small”, for example:

- ▶ Take a few control units with eight static channels and make two or three of them managed.
- ▶ Try out on a test LPAR that is isolated from the production LPARs (different sysplex or LCSS).
- ▶ Convert more control units and channels to DCM management as you get comfortable.

**Note:** DCM changes will potentially affect:

- ▶ Control units that share non-managed channels with the managed control units
- ▶ All images sharing the control unit

To determine the number of managed channels and control unit interfaces, a workload analysis should be performed and the following questions should be answered:

- ▶ Which subsystems have predictive unusual demands
- ▶ Which and how many of the defined channels are assigned specifically for the peak demands, and conversely how many channels are needed for normal or steady state demands



- ▶ Are the peak demand channels assigned to other control units and do those control units have different peak demands, and how many channels are assigned specifically to handle peak demand requirements where the demands occur at different times of the day

### Installing FICON DCM

The following steps should be taken to install FICON DCM:

- ▶ First, z/OS V1R11 and APAR OA28321 must be installed on all systems in the LPAR Cluster (rolling IPLs can be used).
- ▶ Second, at least one system in the LPAR Cluster is enabled to make dynamic configuration changes - refer to the PR/SM Guides for your installation's processors.
- ▶ Third, ensure that the control unit port (CUP) feature is installed on switches that will be connected to managed channels. A static (non-managed) channel must be defined to access the control unit port (CUP).

Update the I/O configuration as follows:

- ▶ Define managed control units.
- ▶ Define managed channels - some rearrangement of the static channels may need to occur, and remember that the managed channels must be switch-attached.
- ▶ Define switch devices - specify a destination port of FE for the CUP and connected to the operating system configurations used by the LPAR Cluster.

### SYS1.PARMLIB considerations

Once the I/O configuration has been created, activate the new I/O configuration. It is important to bring the switch devices online to z/OS, which can be done through vary commands in the COMMNDxx parmlib member or through automation. Specify OFFLINE NO in the HCD device parameters.

Some additional recommendations include enabling DCM component tracing. To do this, update the IECIOSxx parmlib member to include a CTRACE(CTIIOSxx) parmlib member specification, as shown in Figure 27-27.

```
TRACEOPTS
  ON
  OPTIONS('EXTEND,DCM,DS=xx')
```

Figure 27-27 CTRACE option

- ▶ In the CTIIOSxx parmlib member, include DCM in the OPTIONS keyword.
- ▶ Include a control size for the CTRACE data space (specified with the DS=xx keyword, the default value being 512 M).

Finally, utilize RMF. IBM recommends that the channel, device, IOQ, and ESS reporting functions be “turned on” and that the recording interval should be no more than 15 minutes. For ESP clients it is strongly suggested that the RMF interval be five minutes or less. This will allow for validation of DCM data.

## 27.12 Summary considerations

We discussed the benefits that FICON DCM provides, as follows:

- ▶ Improves overall I/O performance.
- ▶ Makes efficient use of hardware resources.
- ▶ Simplifies the I/O configuration definition task.
- ▶ Dynamically balances I/O resources based on workload demand.

We reviewed the following:

- ▶ How to update, change, and modify the I/O configuration through HCD, that is, define managed control units and managed channels.
- ▶ The existing commands and messages from these commands that will be useful when running FICON DCM or DCM for that matter. These commands and messages are existing commands that were introduced or updated for the initial release of DCM.
- ▶ Migration and coexistence considerations.
- ▶ Suggestions were made for migration and that APAR OA28321 is required to “activate” FICON DCM. Also, that FICON DCM is enabled only when all systems in the I/O or LPAR Cluster are at z/OS V1R11 level, and that if a system in the I/O or LPAR Cluster falls back to a prior z/OS level, DCM is disabled.
- ▶ The steps of “turning on” or activating DCM for FICON channels.



## Workload Manager

MVS workload management provides a solution for managing workload distribution, for workload balancing, and for distributing resources to competing workloads. MVS workload management is the combined cooperation of various subsystems (CICS, IMS/ESA®, JES, APPC, TSO/E, z/OS UNIX System Services, DDF, DB2, SOM, LSFM, and Internet Connection Server) with the MVS workload management (WLM) component.

With workload management, you define performance goals and assign a business importance to each goal. You define the goals for work in business terms, and the system decides how much resource, such as processor and storage, should be given to the work to meet its goal.

z/OS V1R12 delivers truly significant improvements in system availability, workload performance, simplified usability, and cross system integrated connectivity. z/OS V1R12 takes smart systems to a whole new dimension by providing automatic and real-time capabilities for higher performance and less operator intervention, with fewer system disruptions and response time impacts to z/OS and the business applications that rely on z/OS.

## 28.1 IBM zEnterprise System

The IBM zEnterprise System configuration is shown in Figure 28-1 on page 583. It is made of z/OS, z/VM and z/Linux closely related along with other non-z environments. The expectation is to narrow the gap for non-z environments to come closer to z Systems-managed data such as online transactional data.

The zEnterprise System brings about a revolution in the end-to-end management of diverse systems, while offering expanded and evolved traditional System z capabilities. With zEnterprise, a system of systems can be created where the virtualized resources of both the zEnterprise 196 (z196) and selected IBM blade-based servers, housed in the zEnterprise BladeCenter Extension (zBX), are pooled together and jointly managed. End-to-end solutions based on multi-platform workloads can be deployed across the zEnterprise System structure and benefit from System z's traditional qualities of service, including high availability, and simplified and improved management of the virtualized infrastructure.

### 28.1.1 zEnterprise BladeCenter Extension

The zEnterprise BladeCenter Extension Model 002 (zBX), shown in Figure 28-1 on page 583, is available as an option with the z196 servers and consists of the following:

- ▶ Up to four IBM Enterprise racks
- ▶ Up to eight BladeCenter8 chassis with up to 14 blades9 each
- ▶ IBM blades, up to 112, based on POWER7™ technology
- ▶ Two Top of Rack (TOR) 1000BASE-T switches for the intranode management network (INMN). The INMN provides connectivity for management purposes between the z196 support elements and zBX.
- ▶ Two Top of Rack (TOR) 10 GbE switches for the intraensemble data network (IEDN). The IEDN is used for data paths between the z196 and the zBX, and the other ensemble members.
- ▶ 8 Gbps Fiber Channel switch modules for connectivity to an SAN
- ▶ Power Distribution Units (PDUs) and cooling fans

The zBX is configured with redundant components to provide quality of service similar to that of System z, such as firmware management and the capability for concurrent upgrades and repairs.

**Statement of Direction:** In the first half of 2011, IBM intends to offer a System x® blade running Linux and a WebSphere DataPower® Appliance, for the zBX Model 002.



Figure 28-1 The zEnterprise System

## zEnterprise System and WLM

In the zEnterprise System, the logical partitions (LPARs) running in the z196 are called virtual servers. The z/OS LPARs have the workload performance defined, monitored, and adjusted through the z/OS Workload Manager (z/OS WLM), but that is only part of the picture. zEnterprise System performance management functions provide a more comprehensive performance management. It permits setting objectives and tracking the performance of any virtual server in the ensemble:

- ▶ The logical partitions (LPARs) running z/OS
- ▶ The z/VM guests running Linux
- ▶ The virtual servers running AIX on the POWER7 blades

The Workload Manager (WLM) component of z/OS has the information at the task level, but is unaware of physical processors. This disconnect is solved by enhancements on z196 that allow PR/SM and WLM to work more closely together. They can cooperate to create an affinity between task and physical processor rather than between logical partition and physical processor. This is known as HiperDispatch.

## HiperDispatch and the z196

As HiperDispatch was introduced with the z10, it is now enhanced with the z196 to combine two functional enhancements, one in the z/OS dispatcher and one in PR/SM. This is intended to improve efficiency both in the hardware and in z/OS.

In general, the PR/SM dispatcher assigns work to a minimum number of logical processors needed for the priority (weight) of the LPAR. PR/SM also attempts to group the logical processors into the same book and, if possible, the same chip. The end result is to reduce the multiprocessor effects, maximize use of shared cache, and lower the interference among multiple partitions.

**Note:** To effectively exploit HiperDispatch, the WLM goal adjustment might be required. We suggest that you review WLM policies and goals, and update them as necessary. You may want to run with the new policies and HiperDispatch on for a period, turn it off and use the older WLM policies while analyzing the results of using HiperDispatch, re-adjust the new policies and repeat the cycle, as needed. In order to change WLM policies, turning HiperDispatch off then on is not necessary.

### Classical view of the new environment

Figure 28-2 depicts the following: The blue left side is a new classical z environment, the yellow right side is the non-z environment, the brown body in the center is the overall IBM zEnterprise 196 (z196) infrastructure coupling the blue and yellow sides in order to achieve a unique environment providing deep computing capabilities.

Following the color code of Figure 28-2 is the functional configuration of an IBM zEnterprise 196 (z196) machine depicted in Figure 28-1 on page 583.

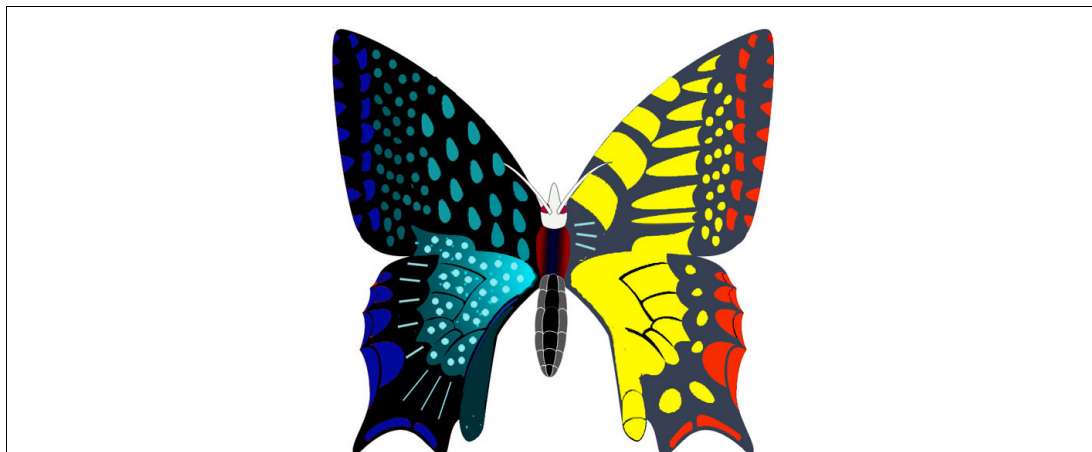


Figure 28-2 Hybrid computing

## 28.2 WLM and the z196

The IBM zEnterprise 196 (z196) environment provides:

- ▶ Platform workload management
- ▶ Platform performance monitoring

The guest platform management provider (GPMP) is the interface between the IBM zEnterprise 196 (z196) intranode management network (INMN) and the z/OS Workload Manager, as shown in Figure 28-3 on page 586.

### WLM service REQLPDAT

The WLM service for requesting LPAR-related data (REQLPDAT) is enhanced to include character-based data about the machine model:

- ▶ A Model-Permanent-Capacity Identifier
- ▶ A Model-Temporary-Capacity Identifier
- ▶ The Model-Capacity Rating
- ▶ The Model-Permanent-Capacity Rating
- ▶ The Model-Temporary-Capacity Rating

This new data is intended to be used for reporting.

## 28.3 Guest Platform Management Provider

The Guest Platform Management Provider (GPMP) is an optional suite of applications that is installed in specific z/OS, Linux, and AIX operating system images to support platform management functions. For example, the GPMP collects and aggregates performance data for virtual servers and workloads. Users view these reports through the ensemble-management Hardware Management Console (HMC).

GPMP is part of the z/OS BCP and is also called Management Center Agent (MCA) or z/OS agent. z/OS integrates with this new management environment. This new agent, GPMP, in z/OS V1R12 communicates with z/OS WLM and provides basic data (such as system resource utilization, system delays, and paging delays) back to the zEnterprise Unified Resource Manager over the INMN network. The zEnterprise Unified Resource Manager can add additional workload relationships from the ensemble components to your z/OS workload; for example, linking a transaction that started on the zBX back to DB2 on z/OS data.

### Intranode management network and intraensemble data network

With the z196 two CHPID types are introduced, as follows:

#### ► OSM for the intranode management network

The intranode management network (INMN) is one of the ensemble's two private and secure internal networks. INMN is used by the Unified Resource Management functions in the primary HMC. This private internal network provides the connections necessary to monitor and control components of the node, such as virtual servers or physical switches. System z firmware, and the intercomponent communication is performed over the trusted INMN.

#### ► OSX for the intraensemble data network

The intraensemble data network (IEDN) is the ensemble's other private and secure internal network. IEDN is used for communication across the virtualized images (LPARs and virtual machines on z/VM, and blades). The public Internet communications could be isolated across one VLAN in the IEDN to which no other virtual servers are allowed access. VLAN isolation is considered to be as secure as physical isolation by many networking industry groups. This is the network for system and application data communications within the ensemble. This network connects all nodes of the ensemble, including all z196 and zBX frames.

As shown in Figure 28-3 on page 586, the first rack (Rack B) in the zBX is the primary rack where one or two BladeCenter chassis reside. Four Top of Rack (TOR) switches are included in Rack B for intranode management network (INMN), and intraensemble data network (IEDN) connectivity. The other three racks (C, D, and E) are expansion racks with one or two BladeCenter chassis each.

### TOR switches

The z196 introduces the OSA-Express for Unified Resource Manager (OSM) CHPID type. The OSM connections are via the Bulk Power Hubs (BPHs) in the z196. The BPHs are also connected to the INMN TOR switches in the zBX. The INMN requires two OSA Express3 1000BASE-T ports from separate features. The IEDN provides private and secure 10 GbE high-speed data paths between all elements of an ensemble node through the IEDN TOR switches in the zBX. The OSA-Express for zBX (OSX) CHPID type supports connectivity and

access control from the z196 to the zBX. All BladeCenters are connected to these TOR switches to provide redundant access to the internal networks.

**Note:** Optionally, as part of the ensemble, any OSA-Express2 or OSA-Express3 features (with CHPID type OSD, as shown in Figure 28-3) in the z196 can connect to the client-managed data network, which can also be connected to the IEDN TOR switches in the zBX.

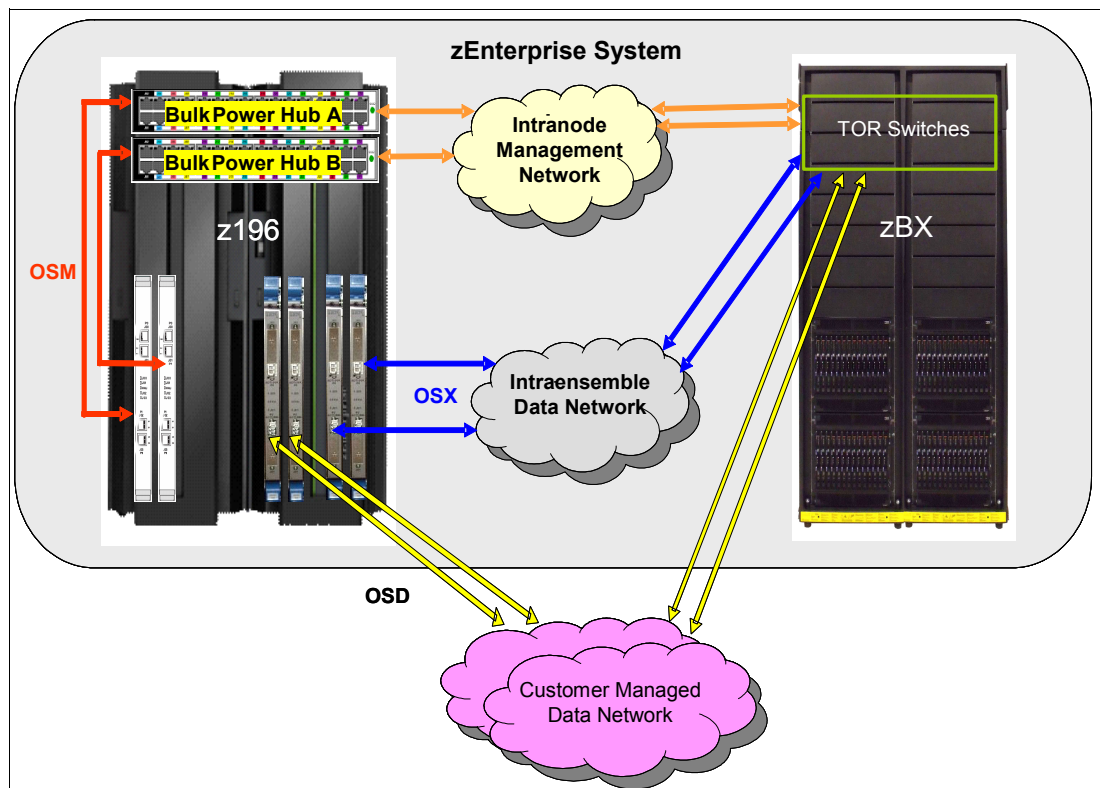


Figure 28-3 INMN, IEDN, and client-managed data networks in an ensemble

### 28.3.1 GPMP and WLM

Starting with zEnterprise System, the guest platform management provider (GPMP) is the interface between the z/OS Workload Manager (WLM) and the IBM zEnterprise Unified Resource Manager. The GPMP supports platform management functions and provides policy information to WLM about the platform-wide performance goals of workloads in which the z/OS system is participating.

Within the service definition, you can indicate on which system in the sysplex you want WLM to activate the guest platform management provider. If activation is selected, WLM automatically starts each guest platform management provider and restarts it as needed. Otherwise, you must manually start and stop the guest platform management provider. By default, activation is not enabled.

The guest platform management provider (GPMP) on a z/OS system sends all data over the intranode management network (INMN) using an Open Systems Adapter-Express3 1000BASE-T Ethernet adapter (CHPID type OSD). Applications sending or receiving data over the INMN require z/OS Communications Server configuration to access this network. See "TCP/IP in an ensemble" in *z/OS Communications Server IP Configuration Guide*,



SC31-8775 for the steps required to allow TCP/IP to participate in the ensemble, and to allow the guest platform management provider (GPMP) access to the INMN.

## GPMP components

The GPMP component performs these tasks:

- ▶ Collects z/OS utilization data and transaction response time for ARM instrumented applications.
- ▶ Classifies the incoming transaction to appropriate service classes based on the performance policy provided from the Support Element (SE).

GPMP provides policy information to WLM about the platform-wide performance goals of workloads in which the z/OS is participating. WLM supports GPMP by the following functions:

- ▶ Configuring the GPMP with the WLM ISPF application.
- ▶ Managing the GPMP address space (start, stop, and restart).
- ▶ Displaying GPMP status information.
- ▶ Collecting and aggregating performance measurements.

**Note:** The IBM zEnterprise 196 (z196) systems will carry some EWLM technologies into the new hybrid environments.

## HMC support

Especially the concept of cross-platform performance management (built into the HMC) will be a significant differentiator against other distributed solutions. The platform workload management policy is organized around so called “Workloads” (not to be confused with existing z/WLM policy workloads), which are basically groups of partitions or virtual services that support the same business applications. The platform management policy is defined at the HMC.

To support the IBM zEnterprise 196 (z196) with ensemble management, WLM adapts the existing EWLM support code:

- ▶ The concept of transaction classes is dropped.
- ▶ WLM ISPF Application function to define GPMP settings is enabled.
- ▶ Management of the GPMP address space (start, stop, and restart) is provided along with additional command and messages.
- ▶ New performance data APIs are made available in order to provide data aggregation at the z/OS level.

### 28.3.2 GPMP settings and related commands

The GPMP is the interface between the intranode management network (INMN) and the z/OS Workload Manager (WLM). It provides policy information to WLM about the platform-wide performance goals of workloads in which the z/OS systems are participating.

For z/OS V1R12, the D WLM command has some new information, as shown in the highlighted rows in Figure 28-4.

```

To display the name of the active service policy, enter:
D WLM
The system responds with:
IWM025I 17.40.54 WLM DISPLAY 913
ACTIVE WORKLOAD MANAGEMENT SERVICE POLICY NAME: WEEKDAY
ACTIVATED: 2009/08/12 AT: 12:55:57 BY: USER01 FROM: SYS2
DESCRIPTION: Weekday policy with ResGrp
RELATED SERVICE DEFINITION NAME: COEFFS
INSTALLED: 2009/08/12 AT: 12:55:51 BY: IBMUSER FROM: SYS2
WLM VERSION LEVEL: LEVEL025
WLM FUNCTIONALITY LEVEL: LEVEL025
WLM CDS FORMAT LEVEL: FORMAT 3
STRUCTURE SYSZWLM_WORKUNIT STATUS: CONNECTED
STRUCTURE SYSZWLM_EBAE2097 STATUS: CONNECTED
STATE OF GUEST PLATFORM MANAGEMENT PROVIDER (GPMP): ACTIVE

```

Figure 28-4 New information with the D WLM command

The GPMP option can be displayed with the operator command shown in Figure 28-5.

```

D WLM[,SYSTEM=sysname|,SYSTEMS]
[,APPLENV=app|envname|*]
[,DYNAPPL=app|envname|*[,SNODE=nodename]
[,SNAME=subsystemname]
[,STYPE=subsystemtype]]
[,SCHENV=schenvname[,SYSTEM=sysname|,SYSTEMS]]
[,RESOURCE=resourcename[,SYSTEM=sysname|,SYSTEMS]]
[,AM[,ALL]]
[,SYSTEMS[,GPMP]]
[,L={a|name|name-a}]

```

Figure 28-5 Display WLM operator command

### WLM ISPF primary menu option

The WLM Definition menu is considered the home base panel. With z/OS V1R12, Option 11 is the new GPMP selection option, as shown in Figure 28-6 on page 589.

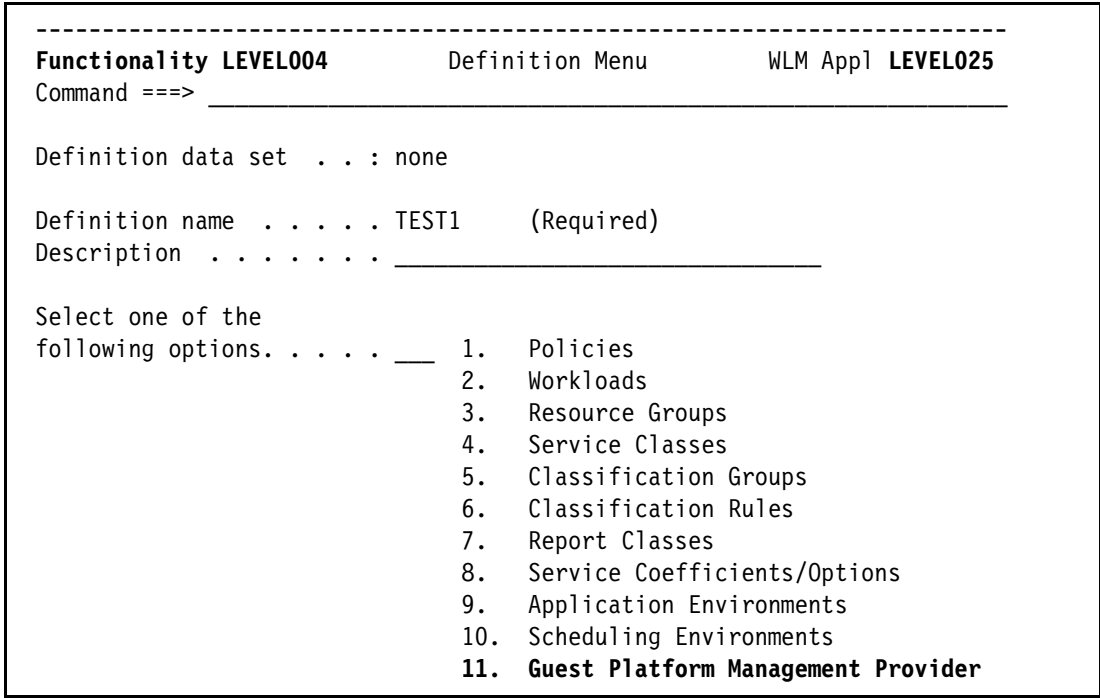


Figure 28-6 WLM ISPF Primary Menu panel

When you select Option 11, shown in Figure 28-6, PanelID IWMAPBD is displayed, as shown in Figure 28-7.

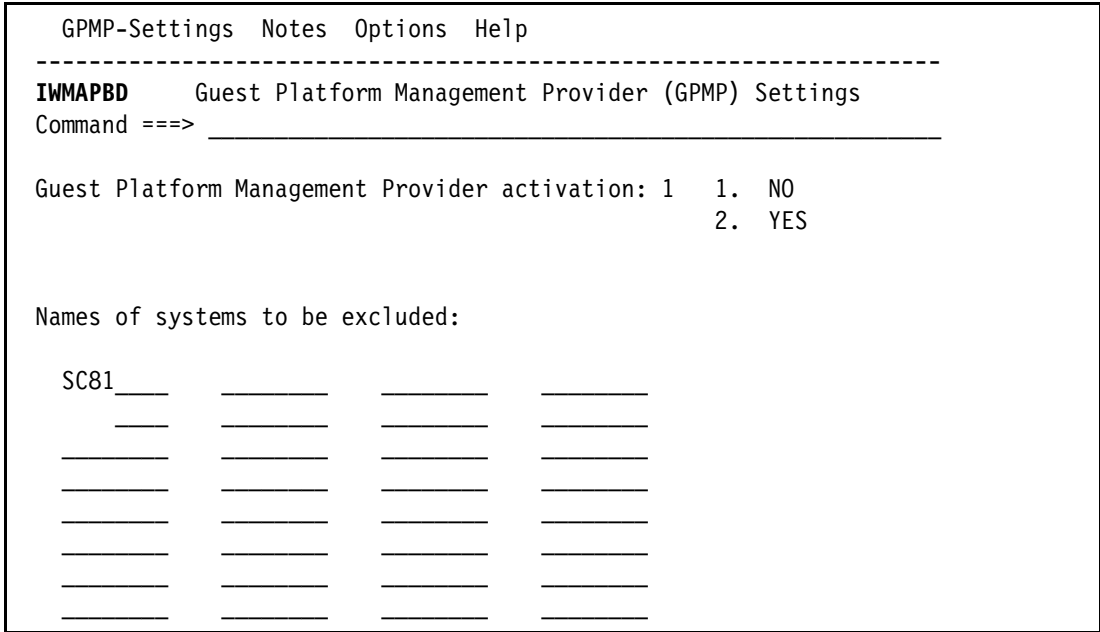


Figure 28-7 Specify systems to exclude with GPMP

### 28.3.3 Configuring the GPMP settings

GPMP settings are done through an ISPF panel (actually a new Option 11 in the general WLM panel), depicted in Figure 28-6 on page 589, and using this panel you can do the following:

- ▶ Choosing the activation as Option 2 (YES) indicates that you want WLM to manage the guest platform management provider on each system in the sysplex except the ones you specify in the exclude list. If YES is selected, WLM automatically starts each guest platform management provider and restarts it as needed. Otherwise, you must manually start and stop the guest platform management provider. By default, activation is not enabled; NO is selected.
- ▶ Excluded systems specifies the systems in the sysplex for which you do not want WLM to activate the guest platform management provider. You can enter the name of up to 32 systems. The system names can contain up to eight of the following characters: alphanumeric characters (A-Z, 0-9) and special characters (\$ # @).

#### Classification rules

Classification of end-to-end work towards WLM goals is modified in z/OS V1R12 (WLM application level LEVEL025).

- ▶ The set of valid Work Qualifiers for the Classification Rules of the EWLM Subsystem Type has changed in z/OS V1R12.

**Note:** Work qualifier ETC (EWLM transaction class name) is no longer supported. Although z/OS V1R12 simply disregards ETC classification rules, you will have to delete them the next time you modify the EWLM subsystem type.

Message IWMAM726, ETC is not a recognized qualifier type, is displayed when pressing F3=Exit: rows with ETC rules have to be deleted first.

- ▶ The new valid qualifier is ESC (EWLM service class name), which correlates EWLM service classes by name with WLM service or report classes.

When the service definition is stored in ISPF tables, the GPMP settings are saved in table SDCTAB in the following variables:

- ▶ SDCMSA defines activation YES or NO.
- ▶ SN1 to SN32 define the excluded systems.

The service definition with functionality LEVEL025 is stored in XML format.

#### New performance data APIs

With z/OS VR12, for GPMP support, two new functions are provided:

- ▶ `ewlm_get_serviceClassData()`  
Returns data accumulated since last call.
- ▶ `ewlm_get_appIData()`  
Returns application environment-specific data, for example, application name and group name.
- ▶ `ewlm_get_applResourceData()`  
Returns application environment-specific resource metrics.

## 28.3.4 Installation considerations

GPMP must be authorized to run as a trusted started task:

- ▶ You can use sample job HVEENV provided in SYS1.SAMPLIB to define the GPMP security setup. This job does the following:
  - Creates a RACF group and the associated group (GID) for Unix System Services. The GID can be any number (example is 888888).
  - Creates a RACF user ID, HVEMCA1, with z/OS UNIX with superuser capabilities (UID(0)). UID(0) is required to make sure that UNIX fork() operations do not result in a creation of a new address space. Instead it will allow UNIX processes to share the same MVS address space, which is required for correct operation of the ZGPA.
  - Authorizes HVEMCA to be run as trusted started task.
- ▶ Configure the execution environment for the JCL of GPMP STEP02 in modifying variables of Figure 28-8 according to your specific environment.

<b>Please update the following variables for your specific environment:</b>	
<b>DATA_ROOT</b>	- Directory on USS where MZGPA stores the configuration and diagnostics data.
<b>JREBIN_ROOT</b>	- Directory on USS where the java executable is located.
<b>LOGFILE_ROOT</b>	- Directory where ZGPA startup related diagnostics messages will be stored.
<b>MCA_USER</b>	- The User created in STEP01 of this JCL
<b>MCA_GROUP</b>	- The Group created in STEP01 of this JCL
<b>MCA_JAVADUMPS</b>	- Optional USS directory where ZGPA Java, Heap, Snap, CEE dumps will be created.
<b>ENV_PROFILE</b>	- The USS Profile to be used by ZGPA. This should contain environmental information such as TimeZone, Locale.

Figure 28-8 GPMP environment variables

- ▶ Run the JCL from an authorized user ID with UID=0.
- ▶ Note that STEP01 must be modified when running GPMP in a SECLABEL environment.

### Hardware dependencies

For GPMP to work properly, the IBM zEnterprise 196 (z196) system with ensemble management should be available.

**Note:** Guest platform management provider on z/OS cannot be started on z9 or z10 servers.

If GPMP is started on non-IBM zEnterprise 196 (z196) servers, the following message is issued on the console:

```
IWM078E GUEST PLATFORM MANAGEMENT PROVIDER CANNOT BE STARTED, FUNCTION NOT AVAILABLE
```

## Software dependencies

The GPMP function is in z/OS V1R12 with APAR OA30928, which is true also for z/OS V1R10 and V1R11.

## GPMP commands

Use the F WLM,GPMP command to start, stop, and modify the guest platform management provider. The syntax of the command is:

```
F WLM,GPMP,START
$HASP373 HVEMCA STARTED
IEF403I HVEMCA - STARTED - TIME=16.55.59
```

If GPMP is already active, message IWM077E is issued. If application resource measurement (ARM) is disabled or the hardware does not support GPMP, message IWM078E is issued.

To stop GPMP, issue the following command:

```
F WLM,GPMP,STOP
IEF404I HVEMCA - ENDED- TIME=17.03.39
$HASP395 HVEMCA ENDED
```

These messages indicate that WLM stops the currently active GPMP instance. WLM does not automatically restart GPMP even if the WLM policy contains a valid GPMP configuration with activate=yes and the system is not in the excluded list. GPMP is restarted only if the F WLM,GPMP,START command is issued from the MVS console. Even in case of an activation of a new policy that contains a valid GPMP configuration, it is not restarted. The D WLM,SYSTEMS command indicates that situation with a status of STOPPED.

## GPMP trace

When GPMP terminates unexpectedly or when too many errors occur, the GPMP state in the D WLM,SYSTEMS command is set to FAILED.

The GPMP trace can also be activated through the following command, which allows you to change the GPMP internal tracing level “on the fly” and to change the destination of the trace (file or memory):

```
F WLM,GPMP,TRACE=NONE|LOW|MEDIUM|HIGH,DEST=FILE|MEMORY
```

Consider the following situations with the GPMP TRACE command:

- ▶ If no GPMP instance is active, message IWM079E is issued.
- ▶ If this command is completed, messages FEW0613I and FEW0614I are issued. For example:

The F WLM,GPMP,TRACE=HIGH command can trigger either of the following messages:

- FEW0613I Trace destination unchanged.
- FEW0614I Trace level changed from low to high.

To display GPMP and systems status, issue the D WLM,SYSTEMS,GPMP command as shown in Figure 28-9 on page 593.

**D WLM,SYSTEMS,GPMP**

```

IWM025I 16.40.07 WLM DISPLAY 280
ACTIVE WORKLOAD MANAGEMENT SERVICE POLICY NAME: POLICY1
ACTIVATED: 2010/06/25 AT: 13:56:37 BY: RGREEN FROM: SC81
DESCRIPTION: test policy
RELATED SERVICE DEFINITION NAME: wlmdef1
INSTALLED: 2010/06/25 AT: 13:56:22 BY: RGREEN FROM: SC81
WLM VERSION LEVEL: LEVEL025
WLM FUNCTIONALITY LEVEL: LEVEL004
WLM CDS FORMAT LEVEL: FORMAT 3
STRUCTURE SYSZWLM_WORKUNIT STATUS: CONNECTED
STRUCTURE SYSZWLM_3BD52817 STATUS: CONNECTED
STATE OF GUEST PLATFORM MANAGEMENT PROVIDER (GPMP): UNAVAILABLE
*SYSNAME* *MODE* *POLICY* *WORKLOAD MANAGEMENT STATUS*
SC80 GOAL POLICY1 ACTIVE
SC81 GOAL POLICY1 ACTIVE
*SYSNAME* *GPMP STATUS*
SC80 UNAVAILABLE
SC81 UNAVAILABLE

```

Figure 28-9 D WLM,SYSTEMS,GPMP command

The output of the DISPLAY,WLM command indicates the different GPMP states, shown in Figure 28-10.

<b>UNKNOWN</b>	Status of GPMP cannot be retrieved
<b>INACTIVE</b>	GPMP was not started
<b>UNAVAILABLE</b>	Functionality not available. Hardware support missing
<b>STARTED</b>	GPMP started but not connected to WLM
<b>ACTIVE</b>	GPMP started and connected to WLM but has not established a connection to the INMN
<b>FAILED</b>	WLM attempted to start the GPMP but the agent did not connect to the INMN within the timeframe allotted
<b>SEVFAILED</b>	GPMP terminated twice within 10 minutes
<b>CONNECTED</b>	GPMP active and connected to WLM and to the INMN
<b>STOPPED</b>	GPMP was manually stopped
<b>DISABLED</b>	The ARM component of WLM is disabled
<b>EARLY-IPL</b>	USS or TCP/IP not running yet
<b>SHUTDOWNx</b>	Termination of GPMP is currently in progress, where x is a number
<b>PGMERROR</b>	Internal error. Contact IBM support
GPMP = guest platform management provider	
INMN = intranode management network	

Figure 28-10 GPMP states

To know whether ARM is enabled or not, use the D WLM,AM command shown in Figure 28-11 on page 594.

```

D WLM,AM
IWM075I 17.16.04 WLM DISPLAY 638
      ARM SERVICES ARE ENABLED
      NO GUEST PLATFORM MANAGEMENT PROVIDER IS CURRENTLY CONNECTED
      NO GPMP POLICY IS CURRENTLY ACTIVE
      NUMBER OF REGISTERED PROCESSES=1, APPLICATIONS=1

```

Figure 28-11 The D WLM,AM command

### WLM command considerations with GPMP

The existing F WLM,AM=DISABLE or ENABLE commands are not changed, but the current logic for disable and enable will be changed in the following ways:

- ▶ Disabling Application Response Measurement (ARM) terminates a running GPMP.
- ▶ Manually starting GPMP (using the F WLM,GPMP,START command) when ARM is disabled, results in message IWM078I.
- ▶ Activating a WLM policy that contains valid GPMP settings does not result in the start of a GPMP instance, if ARM is disabled.
- ▶ The state of GPMP is displayed as DISABLED, if ARM is disabled.
- ▶ If ARM is enabled again, the state of GPMP changes to STOPPED. To start GPMP again, it has to be started manually.

### GPMP messages

With this support, several messages are being added.

#### IWM077E GPMP IS ALREADY ACTIVE

This message is issued as a response to the MODIFY WLM,GPMP,START command, if an instance of a guest platform management provider is already active on this system. Processing continues. No other guest platform management provider is started in addition to the one that is currently active.

#### IWM078E GPMP CANNOT BE STARTED, reason

This message is issued as a response to the MODIFY WLM,GPMP,START command, where *reason* can be one of the following:

- ▶ ARM IS DISABLED: The processing is currently disabled because the MODIFY WLM,AM=DISABLE command has been issued before.
- ▶ TCP/IP OR USS NOT ACTIVE: The TCP/IP subsystem or UNIX Systems Services is not available.
- ▶ ASCRE FAILED: The creation of the guest platform management provider address space (MVS service ASCRE) has failed.
- ▶ FUNCTION NOT AVAILABLE: The function is not available. Hardware support is missing.

Processing continues. The guest platform management provider is not started. Operator action can be:

- ▶ ARM IS DISABLED: Make sure that ARM is enabled before the guest platform management provider is started.
- ▶ TCP/IP OR USS NOT ACTIVE: Make sure that the TCP/IP subsystem and UNIX Systems Services are available.



- ▶ **ASCRE FAILED:** Retry the action by manually starting the guest platform management provider (MODIFY WLM,GPMP,START). If that fails again, take a standalone dump and contact the IBM Support Center.

### **IWM079E GPMP IS NOT ACTIVE**

This message is issued as a response to the MODIFY WLM,GPMP,STOP command or the MODIFY WLM,GPMP,TRACE= command, if no GPMP is currently active.

### **IWM080E TIMEOUT OCCURRED DURING STARTUP OF GPMP**

This message is issued as a response to the F WLM,GPMP,START command, or if the guest platform management provider is implicitly started as a result of a policy activation with a valid guest platform management provider configuration in the service definition. The guest platform management provider address space has been started, but the guest platform management provider has failed to connect to z/OS WLM within a given time range.

A system programmer may verify that the guest platform management provider address space (HVEMCA) runs with sufficient priority. By default (no rule in the WLM Service Definition that matches the HVEMCA started task), the guest platform management provider address space is classified into system service class SYSSTC or SYSTEM. Ensure that WLM classification rules in your WLM policy do not direct the HVEMCA address space into a service class that does not have the appropriate attributes. If the guest platform management provider has the appropriate attributes, check the guest platform management provider logs and messages to find out why guest platform management provider has not connected.

### **IWM081E GPMP HAS TERMINATED ABNORMALLY**

This message is issued when an active guest platform management provider terminates unexpectedly (without calling ewlm\_disconnect()).

### **IWM082E GPMP HAS FAILED, HEARTBEAT IS MISSING**

This message is issued when an active guest platform management provider stops communicating with the z/OS operating system and no longer collects transaction completions. This problem might be caused by internal errors in the guest platform management provider, or might be due to insufficient dispatch priority given to the guest platform management provider address space (HVEMCA). WLM will automatically recycle the guest platform management provider once before the status of the guest platform management provider changes to SEVFAILED.

A system programmer can consult the guest platform management provider logs and messages for problem determination. Verify that the guest platform management provider address space runs with sufficient priority. IBM recommends to have the guest platform management provider address space classified into system service class SYSSTC. Ensure that WLM classification rules in your WLM policy do not direct the guest platform management provider address space into a service class that does not have the appropriate attributes.

### **IWM083E GPMP COULD NOT BE STOPPED, CANCEL COMMAND ISSUED**

This message is issued when an active guest platform management provider cannot be stopped or restarted through the normal interface. Diagnostics information has been captured, and the guest platform management provider address space (HVEMCA) will be cancelled by WLM.

## **IWM084I MODIFY COMMAND IGNORED, GPMP IS BUSY**

This message is issued when an operator has issued a MODIFY WLM,GPMP,START, STOP or TRACE request. The guest platform management provider is currently busy processing another request and cannot accept additional requests.

### **GPMP initialization and IPL considerations**

To start GPMP, issue this command:

```
F WLM,GPMP,START
```

after the following events:

- ▶ During IPL when service policy specifies GPMP activation.
- ▶ On WLM policy activation when service policy specifies GPMP activation.

WLM checks for any hardware dependencies, such as whether the Ensemble Communication Facility is installed. If it is, WLM starts GPMP by creating the HVEMCA address space.

- ▶ As a default, the HVEMCA procedure stored in SYS1.PROCLIB is used.
- ▶ An undocumented OPT parameter, MCAPROC, can be used to overwrite the procedure name.

Specify MCAPROC=NONE to disable start of GPMP.

Once GPMP is started, WLM takes care that it is restarted each time the HVEMCA address space terminates abnormally.

## **28.4 GPMP performance monitoring**

z/OS Workload Manager (WLM) provides the following services and routines:

- ▶ Application Response Measurement (ARM) services
- ▶ Sampling and reporting routines that provide feedback on ARM-instrumented applications and transactions
- ▶ Query services to return server configuration and high-level performance statistics collected on z/OS
- ▶ Classification and management of end-to-end work towards WLM goals

The guest platform management provider (GPMP):

- ▶ Obtains the aggregated transaction response time and resource data for the ARM-instrumented applications.
- ▶ Sends the data to the the IBM zEnterprise 196 (z196) HMC for end-to-end performance monitoring.



## SDSF enhancements

SDSF provides a powerful and secure way to monitor and manage your z/OS sysplex. SDSF's easy-to-use interface lets you control:

- ▶ Jobs and output
- ▶ Devices, such as printers, readers, lines, and spool offloaders
- ▶ Checks from IBM Health Checker for z/OS
- ▶ System resources, such as WLM scheduling environments, the members of your MAS, and JES job classes
- ▶ System log and action messages

In this chapter we describe the enhancements to SDSF in z/OS V1R12.

The following topics are discussed:

- ▶ Support for calling SDSF functions using a new Java API
- ▶ Enhancements to SDSF REXX regarding SYSLOG browsing
- ▶ Updates and changes in SDSF panels and commands
- ▶ New support for JES3 in SDSF panels
- ▶ Search help more quickly and easily using the new SEARCH command
- ▶ SDSF coexistence with earlier releases

## 29.1 Java API

With the new z/OS V1R12 SDSF Java API, you can access SDSF panel data and function through a Java program. The following capabilities are provided:

- ▶ Accessing panels and panel data

Each of the panels that you work with when using SDSF interactively (DA, O, PR and so on) has an associated Java interface that describes the returned data and the available methods. Panel data is represented by lists, with each element in a list corresponding to a row on the panel. You access column data within a list element by referencing column values by column name.

- ▶ Processing system log and issuing commands

You can retrieve records from the system log (SYSLOG) and search for specific messages or events. You can also issue free-form system commands and receive their responses in a manner similar to using the SDSF slash (/) command.

- ▶ Retrieving job output

You can allocate the spool data sets for a job and read them using standard utilities.

- ▶ Taking action

You use methods to perform functions similar to action characters and fields that can be overtyped, for example, to cancel a job or change the print destination for job output.

- ▶ Filtering data

For best performance, you should limit the data that a request returns to the minimum that is required. You do this with request settings, which allow you to specify:

- Filters of various kinds. The same filters that are available when you use SDSF interactively are available with request settings. They include filters by job name, owner and destination, like the PREFIX, OWNER, and DEST commands, or any column, like the FILTER command.
- The list of columns to process. Specify columns by column name.
- Whether to include columns with delayed access. Because gathering the data for these columns can take a significant amount of time, they are not included unless you request them explicitly.

- ▶ Viewing results

You can access messages and return codes that describe the completion of a request through a results object. SDSF messages and system messages, if any, issued in response to commands are contained in lists, with each element corresponding to a message. Return codes from SDSF functions are available both in the results object and as return codes on most methods.

- ▶ Controlling access

Standard SDSF authorization checking occurs for all requests and for attempts to modify the row represented by a returned object.

## 29.1.1 Getting documentation

The principal source of information for using Java with SDSF is the Javadoc supplied with SDSF. To use the Javadoc:

1. Use FTP, or a similar program, to download the `/usr/include/java_classes/isfjcallDoc.jar` file, in binary mode, to an empty directory on your workstation.
2. If you have the Java SDK installed on your workstation, use this command:

```
jar -xf isfjcallDoc.jar
```

Otherwise, use another utility available on your workstation to unzip the file.

3. Navigate to the `index.html` file and open it with a web browser. Once the file is displayed, links allow you to navigate to specific classes or topics, such as:

<b>Overview</b>	Display an overview to using SDSF with Java
<b>Package</b>	Display a list of classes
<b>Tree</b>	Display a hierarchical view of classes
<b>Index</b>	Display an index to the Javadoc

## 29.1.2 Enabling your application to use the SDSF Java API

Your application must have the SDSF Java classes accessible to it. To do this, add the SDSF API `.jar` file to the `CLASSPATH` or your Java application and modify your application `LIBPATH`. The syntax for doing this varies based on how your application is invoked. General guidelines are:

### ► CLASSPATH

The SDSF JAR file (`isfjcall.jar`) must be included on the `CLASSPATH`. The `CLASSPATH` can be included on the Java command (using the `-cp` keyword) that invokes your application, or through the `CLASSPATH` environment variable. For example, to invoke an application from the z/OS Unix System Services (z/OS Unix) shell, you might have the following statement:

```
export CLASSPATH=/usr/include/java_classes/isfjcall.jar:$CLASSPATH
```

### ► LIBPATH

The `LIBPATH` references a path containing the SDSF native library. There is one library for 31-bit Java and one for 64-bit Java. You must point to the appropriate library based on the version of Java you are running. Note that the `LIBPATH` references a path and not a specific file, whereas the `CLASSPATH` references a specific `.jar` file. This example assumes SDSF has been installed in the default directories and 31-bit Java is being used:

```
export LIBPATH=/usr/lib/java_runtime:$LIBPATH
```

If you are using 64-bit Java, the `LIBPATH` would be similar to the following:

```
export LIBPATH=/usr/lib/java_runtime64:$LIBPATH
```

### ► Java level

SDSF Java API requires any of the following Java levels, or higher:

- IBM 31-bit SDK for z/OS, Java Technology Edition, V6
- IBM 64-bit SDK for z/OS, Java Technology Edition, V6

To access Java, update your PATH environment variable to point to the level of Java you need (either 31-bit or 64-bit). Assuming Java has been installed in the default path, you would use a command similar to the following for 31-bit Java:

```
export PATH=/usr/lpp/java/J6.0/bin:$PATH
```

If you are using 64-bit Java, the PATH would be similar to the following:

```
export PATH=/usr/lpp/java/J6.0_64/bin:$PATH
```

### 29.1.3 Verifying that SDSF Java is installed correctly

SDSF provides several sample Java classes to show you how to use the SDSF Java API. The samples are installed by default under the `/usr/lpp/sdsf/java/samples` path. The available samples are listed in Table 29-1.

Table 29-1 SDSF Java API samples

Java class	Sample	Description
ISFGetJobsSample	Get list of jobs	Accesses the ST panel and displays the properties of selected jobs.
ISFChangeJobPrioritySample	Change job priority	Changes the priority of jobs.
ISFBrowseSample	Browse job output	Allocates the spool data sets for a job and browses them.
ISFHealthCheckSample	List exception checks and their output	Finds all exception checks and lists the check output.
ISFSearchSyslogSample	Search SYSLOG for string	Reads the last day of SYSLOG and searches for one or more strings.
ISFSlashCommandSample	Issue MVS commands	Issues one or more system commands.
ISFWhoCommandSample	Issue WHO command	Issues the SDSF WHO command to obtain user attributes.

You can use the provided samples to verify that SDSF Java is correctly installed in your environment. Compiled versions of the classes are available in the SDSF .jar file (`isfjcall.jar`). Once the .jar file is added to your CLASSPATH, you should be able to invoke the compiled samples.

For example, in our environment we used Java 1.6.0 64-bit. Figure 29-1 on page 601 shows the Java version and CLASSPATH settings in our environment.

```

java -version
java version "1.6.0"
Java(TM) SE Runtime Environment (build pmz6460sr7-20091215_02(SR7))
IBM J9 VM (build 2.4, JRE 1.6.0 IBM J9 2.4 z/OS s390x-64
jvmmz6460sr7-20091214_4
9398 (JIT enabled, AOT enabled)
J9VM - 20091214_049398
JIT - r9_20091123_13891
GC - 20091111_AA)
JCL - 20091202_01
echo $CLASSPATH
/usr/include/java_classes/isfjcall.jar:./usr/lpp/java/J6.0_64/lib/classes.zip

```

Figure 29-1 Java settings for SDSF Java API in our environment

We then simply invoke the ISFWhoCommandSample SDSF sample class as shown in Figure 29-2.

```

java ISFWhoCommandSample
ISF444I USERID=PELEG
ISF444I PROC=EXTERNAL
ISF444I TERMINAL=SC38TCB4
ISF444I GRPINDEX=3
ISF444I GRPNAME=ISFUSER
ISF444I MVS=z/OS 01.12.00
ISF444I JES=z/OS1.12
ISF444I SDSF=HQX7770
ISF444I ISPF=N/A
ISF444I RMF/DA=NOTACC
ISF444I SERVER=YES
ISF444I SERVERNAME=SDSF
ISF444I JESNAME=JES2
ISF444I MEMBER=SC74
ISF444I JESTYPE=JES2
ISF444I SYSNAME=SC74
ISF444I SYSPLEX=PLEX75
ISF444I COMM=NOTAVAIL
ISF767I Request completed.

```

Figure 29-2 Running an SDSF sample class

## 29.1.4 Writing an SDSF Java application

A basic SDSF Java application might do the following:

1. Create a *runner* that corresponds to the panel you want to work with. A runner is a Java class that provides access to SDSF and contains a results object describing completion of the request. Runners are described in “Runners and request settings” on page 603.
2. Create *request settings* and associate them with the runner to limit the results that are returned (This is optional but recommended). Request settings are described in “Runners and request settings” on page 603.
3. Invoke SDSF to create a list of objects and check the results object for SDSF completion messages.

4. Process the returned object list and obtain column values for each row.
5. Invoke methods on a row object to retrieve additional information or modify the object.

We suggest to always test the return codes from SDSF functions. These are available in the results object and as return codes on most methods. SDSF and system messages describing the completion of a request are also contained in the results object.

### A simple example

The code snippet in Figure 29-3 requests job-related data from the Status (ST) panel. The settings object is used to restrict the returned data to a subset of jobs with the indicated job name prefix (in this case, all job names) and owner (IBMUSER). This is not a complete example. It is shown here to help you get familiarized with the terms described in the rest of the chapter.

```
// Create optional settings object
ISFRequestSettings settings = new ISFRequestSettings();
settings.addISFPrefix("***"); // Set job name prefix
settings.addISFOwner("ibmuser"); // Set job owner

// Get a runner used to access SDSF ST panel
ISFStatusRunner runner = new ISFStatusRunner(settings);

List<ISFStatus> statObjList = null;

try {
    statObjList = runner.exec();
} catch (ISFException e) {
    // Process exception here
} finally {
    // Print SDSF messages related to request
    results.printMessageList(System.err);
}

// List job properties
if (statObjList != null) {
    for (ISFStatus statObj : statObjList) {
        System.out.println(statObjList.toVerboseString());
    }
}
```

Figure 29-3 SDSF Java code snippet

### Obtaining column values

Request column values by column name using the `getValue` method. The value can be returned as a formatted string or as a byte array for processing by the application. Column names are different from the column titles that are displayed when you use SDSF interactively. Use the SDSF COLSHELP command to list the column names recognized by the `getValue` method. Column names are not case sensitive. Some classes include convenience methods for obtaining common values such as job name. The fixed field (the first column on a panel when you use SDSF interactively) can also be obtained using the `getFixedField` method.

The code snippet in Figure 29-4 on page 603 shows how to obtain column values using a previously created `ISFStatus statObj` object.



```
// Get job name and owner
String jobname = statObj.getValue("jname");
String owner = statObj.getValue("ownerid");

// Get fixed field (jobname)
String fixedField = statObj.getFixedField();
```

Figure 29-4 Obtain column values with SDSF Java API

## Actions and overtypes

The available methods for an object are defined by the interface for the object. The method names are similar to the descriptions for action characters that you can display with the SET ACTION LONG command when using SDSF interactively.

The Figure 29-5 snippet shows how to cancel a job and list the command responses on the console.

```
// Cancel job without a dump
statObj.cancel();
// List the command responses
results.printResponseList(System.out);
```

Figure 29-5 Perform an action against a job with SDSF Java API

You can change column values, in a manner similar to overtyping a column, with the **requestPropertyChange** method. This method takes an array of column names to change and an array of values to change. The Figure 29-6 code snippet shows how to change the class of a job.

```
// Build column name array
String propName = { "jclass" };
// Build column value array
String propValue = { "a" };
// Change the job class
statObj.requestPropertyChange(propName, propValue);
// Print response list
results.printResponseList(System.out);
```

Figure 29-6 Changing a column value with SDSF Java API

## Runners and request settings

A runner is a Java class that provides access to SDSF in a means similar to using SDSF commands to access panels. To access SDSF, you create an instance of a runner for the desired panel and then use methods in the runner class to obtain the requested data. For functions that are not panel-related, such as issuing system commands, you use a special runner.

You can optionally provide request settings that are associated with the runner. You create an instance of the ISFRequestRunner class and add the desired settings to it. The settings correspond to SDSF settings such as job name prefix, job owner, and destination name filters. In addition, you can provide sort criteria for the returned data, as well as more complex filtering using all the capabilities of the SDSF FILTER command.

The request settings object contains all possible SDSF settings, although not all of them apply to the request being processed. SDSF ignores settings that are not appropriate for the function being performed, so you do not need to remove them.

The runner provides a *constructor* that is used to associate the request settings with the runner. However, you can always associate a settings object after the runner is created. Note that the settings take effect the next time SDSF is invoked. You can also remove settings after the runner is created, in which case SDSF uses the default settings when processing the request.

You can use the same runner for the duration of your application and modify the request settings between each request. Note that when invoking methods on previously obtained objects (for example, invoking the cancel method on a job) SDSF uses the request settings to verify that the object still exists. As a result, use caution when changing the request settings after a row object has been obtained since the new settings may prevent SDSF from rederiving the object.

After a request has been processed, the runner contains a reference to the ISFRequestResults object that describes the completion of the request. This object contains SDSF messages, system responses or return codes that were generated by SDSF. You should check the return codes to ensure that your request has been processed successfully.

### Selecting the appropriate runner

You select the runner based on what rows, columns or other SDSF capabilities your application needs. For example, if you need information about active jobs, you would use the ISFActiveRunner because it provides access to the SDSF DA panel. Similarly, if you needed to enter MVS system commands, you would use the ISFRunner class because it enables use of the SDSF slash command.

For the complete list of runners, see the SDSF Javadoc or refer to *SDSF Operation and Customization, SA22-7670*.

### A complete example

For a complete example of using the SDSF Java API, refer to Appendix A, "SDSF Java API example" on page 803.

## 29.2 SDSF REXX enhancements

In z/OS V1R11, JES2 introduced support for browsing the SYSLOG as a single logical data set. With this support in JES2, SDSF uses new indexes to the SYSLOG JQEs that JES2 maintains on the spool and does not require the HASPINDEX data set allocation. This allows browsing the SYSLOG as a single logical data set from SDSF. The logical SYSLOG data set can also be accessed using existing spool browse access methods. The data set name is `sysname.SYSLOG.SYSTEM`. Similar support is also available in JES3.

To accommodate the support for the logical SYSLOG data set, SDSF REXX has been enhanced in z/OS V1R12 with a new SDSF host environment command, ISFLOG. The new command simplifies SYSLOG access using REXX scripts and eliminates the need to separately allocate SYSLOG data sets in order to read them. The ISFLOG host environment command is supported by SDSF running under either JES2 or JES3.

The syntax of the ISFLOG host environment command is shown as follows:

```
Address SDSF "ISFLOG ALLOC|READ TYPE(log-type) (option)"
```

The ALLOC option indicates that the logical SYSLOG is to be allocated for use with a utility such as EXECIO. The SYSLOG is allocated with the FREE=CLOSE option so that the file is automatically de-allocated when closed. Use ALLOC with these special stem variables:

**ISFDDNAME** Contains the ddname that is returned

**ISFDSNAME** Contains the data set name that is returned

The READ option indicates that the logical SYSLOG is to be read into a special stem variable, ISFLINE. In addition, optionally you can use the following stem variables:

**TYPE(log-type)** This variable is optional and names the type of log to be used. For log-type, specify SYSLOG. Use the special variable ISFSYSID to indicate the member to be processed.

**option** This variable is currently the option VERBOSE, which adds diagnostic messages to the ISFMSG2 stem variable. The messages describe each row variable created by SDSF.

### Special variables

There are a number of special variables that you can use with the ISFLOG host environment command, to limit the number of variables to create in the special stem variable ISFLINE, and to define the start and end date for the SYSLOG. Following is a list of the special variables supported by the ISFLOG host environment command:

**ISFDATE** Specifies the date format, including the separator character, for special variables that take a date as input. It does not affect the date format for records returned in stem variables. The data format is specified as with the SET DATE command.

**ISFLINE** Contains the log records. It is a stem variable. ISFLINE.0 contains the number of variables.

**ISFLINELIM** Limits the number of ISFLINE stem variables that may be created. The valid values are 0-99999999. The default is no limit. This can be requested explicitly with a value of 0 for ISFLINELIM.

**ISFLOGSTARTTIME** Specifies the starting time for records returned by the ISFLOG command, in hh:mm:ss.th format. Only hh:mm is required. This is the local time corresponding to the first record to be returned. The default is 00:00:00.00. It must be less than the ending time.

**ISFLOGSTARTDATE** Specifies the starting date for records returned by the ISFLOG command. The format is either the current date format (see the ISFDATE special variable) or yyyy.ddd or yy.ddd. The default is the current day. It must be less than the ending date.

**ISFLOGSTOPTIME** Specifies the ending time for records returned by the ISFLOG command, in hh:mm:ss.th format. Only hh:mm is required. This is the local time corresponding to the last record to be returned. The default is 23:59:59.99.

**ISFLOGSTOPDATE** Specifies the ending date for records returned by the ISFLOG command. The format is either the current date format (see the ISFDATE special variable) or yyyy.ddd or yy.ddd. The default is the current day.

**ISFSYSID** Names the member to be processed by the ISFLOG command.

**Note:** When these special variables are used, SDSF positions the SYSLOG as near as possible to the requested record. However, due to the precision used for time stamps and the time the record is actually written to SYSLOG, it is possible that this may be several lines away from the desired record.

## 29.2.1 ISFLOG examples

The example in Figure 29-7 shows how to allocate the SYSLOG using the ISFLOG ALLOC command. The SYSLOG data set is then read using the EXECIO utility. In the example, we simply print some of the SYSLOG lines. However, ISFLOG ALLOC may be useful to scan the SYSLOG for messages.

```
/* REXX */  
  
rc = isfcalls('on')  
  
address SDSF "isflog alloc"  
  
say '# of DDs:' isfddname.0  
say 'DD name:' isfddname.1  
say 'DSN name:' isfdsname.1  
  
do ix = 1 to isfddname.0  
  "execio 10000 diskr" isfddname.ix "(finis stem syslog."  
  do jx = 1 to syslog.0  
    say syslog.jx  
  end  
end  
  
rc = isfcalls('off')
```

Figure 29-7 ISFLOG ALLOC example

Figure 29-8 shows sample output from the above example REXX exec. Note that only a single data set is allocated. This is because the ISFLOG command processes the logical SYSLOG data set. You can also see that the SYSLOG data set name matches the JES2 logical SYSLOG data set name.

```
# of DDs: 1  
DD name: SYS00089  
DSN name: SC74.SYSLOG.SYSTEM  
N 0000000 SC74 2009090 09:52:38.97 00000290 IEF196I IEF237I JES2  
ALLOCATED TO SYSLOG06  
X 0000000 SC74 2009090 09:52:38.98 SYSLOG 00000000 IEE042I SYSTEM LOG DA  
TA SET INITIALIZED  
N 0000000 SC74 2009090 09:52:38.98 00000290 IEF196I IEF285I +MA  
STER+.SYSLOG.STC03153.D0000106.? SYSOUT  
N 4000000 SC74 2009090 09:52:38.98 00000090 IEE043I A SYSTEM LOG  
DATA SET HAS BEEN QUEUED TO SYSOUT CLASS A  
...
```

Figure 29-8 ISFLOG ALLOC example output

The example in Figure 29-9 shows how to use ISFLOG READ to browse the SYSLOG. Since ISFLOG READ acts according to special variables, which were defaulted in the example, this REXX exec prints SYSLOG lines from the beginning of the current day. As described above, the SYSLOG lines are returned in the ISFLINE special stem variable.

```

/* REXX */

rc = isfcalls('on')

address SDSF "isflog read"
do ix = 1 to isfline.0
    say isfline.ix
end

rc = isfcalls('off')

```

Figure 29-9 ISFLOG READ basic example

In Figure 29-10 we show a more complex example of using ISFLOG READ. The example sets the specials variables in order to read the last 24 hours of SYSLOG. The messages in the ISFMSG2 special stem variable are printed to show you how SDSF REXX processed the ISFLOG READ command.

```

/* REXX */

rc = isfcalls('on')

isfdate          = "mddyyy /"
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"
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 29-10 ISFLOG READ special variables use example

The sample output is shown in Figure 29-11 on page 608. The content of the ISFMSG2 special stem variable is in bold.

```

M 4040000 SC74      2010123 10:32:27.04 STC07102 00000090 *HZS0003E CHECK(IBMxcf
,XCF_CDS_SPOF): 783
E
                                783 00000090 IXCH0242E One or more
couple data sets have a single point of failure.
N 0000000 SC74      2010123 10:32:32.14          00000290 IEE252I MEMBER BPXPRM
2A FOUND IN SYS1.PARMLIB
N 0000000 SC74      2010123 10:32:32.17          00000290 IEF196I IEF285I  SYS
1.PARMLIB
                                KEPT
N 0000000 SC74      2010123 10:32:32.17          00000290 IEF196I IEF285I  VOL
SER NOS= BH5CAT.
...
ISF754I Command 'SET DATE MDDYYYY /' generated from associated variable ISFDAT
E.
ISF757I Variable ISFLINELIM being processed with value '1000'.
ISF757I Variable ISFLOGSTARTTIME being processed with value '10:25:52'.
ISF757I Variable ISFLOGSTARTDATE being processed with value '05/03/10'.
ISF757I Variable ISFLOGSTOPTIME being processed with value '10:25:52'.
ISF757I Variable ISFLOGSTOPDATE being processed with value '05/04/10'.
ISF770W Request limit 1000 from variable ISFLINELIM reached, processing stopped
.
ISF767I Request completed.

```

Figure 29-11 ISFLOG READ special variables use example output

## 29.3 Updates to SDSF panels and commands

Existing SDSF panels have been enhanced to support new features and functions in z/OS V1R12, such as the Console Auto-Reply function and updated JES subsystem API function codes. In addition, SDSF keeps extending the support for JES3.

## 29.4 Health Check History panel

The Health Checker provides an option for saving Health Checks to a log stream for historical purposes. In previous releases, SDSF could only present you with the current status of health checks via the SDSF Health Checker display.

Starting with z/OS V1R12, SDSF provides a new L action character on the SDSF Health Checker display to enable you to view the history of a health check. When L is issued next to a health check, the Health Check History (CKH) panel is displayed, listing historic health checks runs. This allows you, for example, to see exactly when a health check started reporting an exception, or when it stopped.

Figure 29-14 on page 610 shows how to access the CKH panel by typing the L action character next to a health check. In support of the new SDSF Health Checker History display there is a new column on the SDSF Health Checker display titled LogStream. This column shows the current logstream to which the IBM Health Checker for z/OS is connected, from which history data will be gathered.

## 29.4.1 Defining the logstream

Figure 29-12 shows a Coupling Facility and logstream structure definition in the LOGR policy using the administrative data utility, IXCMIAPU.

```
DEFINE STRUCTURE NAME(HZS_HEALTHCHKLOG)
LOGSNUM(1)
MAXBUFSIZE(65532)
AVGBUFSIZE(1024)

DEFINE LOGSTREAM NAME(HZS.HEALTH.CHECKER.HISTORY)
DESCRIPTION(HEALTH_CHECK_RPT)
STRUCTNAME(HZS_HEALTHCHKLOG)
STG_DUPLEX(NO)
LS_DATACLAS(NO_LS_DATACLAS)
LS_MGMTCLAS(NO_LS_MGMTCLAS)
LS_STORCLAS(NO_LS_STORCLAS)
LS_SIZE(4096)
AUTODELETE(YES)
RETPD(14)
HIGHOFFLOAD(80)
LOWOFFLOAD(0)
DIAG(NO)
```

Figure 29-12 Define a structure and logstream for the health checker history

For more information about the Health Checker, see *z/OS IBM Health Checker for z/OS Users Guide*, SA22-7994.

**Note:** Access to the logstream is required for this function. Contact your security administrator for authorization to the logstream.

### Properties to specify with the PROPERTY statement

The PROPERTY statement defines customized values for certain SDSF properties. It provides an alternative to writing user exit routines to customize those properties. A user exit routine that customizes the same property as a PROPERTY statement overrides the value on the PROPERTY statement.

**Panel.CKH.DefaultCKLim** Specify 1–999999 to set the default maximum number of instances for a check for IBM Health Checker for z/OS that will be read from the logstream for the CKH panel. Users can override this with the SET CKLIM command.

**Default:** 10

**Note:** By default, SDSF will collect the last 10 iterations of a check. You can override this default with the new SET CKLIM command.

- ▶ The minimum number of checks can be 1 and the maximum is 999,999.
- ▶ You can also override the SDSF default of 10 iterations using the new Panel.CK.DefaultCKLim custom property.
- ▶ You can override this default with the SET CKLIM command

Optionally, customize columns on the panel for your installation using the CKHFLDS and CKHFLD2 parameters and FLD statements in ISFPARMS. Review customized field lists for the CK panel for changes related to the new LogStream column.

## 29.4.2 New health check panel in SDSF

In support of the new SDSF Health Checker History display there is a new column on the SDSF Health Checker display titled LogStream, as shown in Figure 29-13. This column shows the current logstream to which the IBM Health Checker for z/OS is connected, from which history data will be gathered.

Furthermore, the CK panel is enhanced in z/OS V1R12 to include the name of the log stream the Health Checker is connected to and uses to collect health check history. To display the LogStream column, type M on the command line followed by the F11 key.

```

Display Filter View Print Options Search Help
-----
SDSF HEALTH CHECKER DISPLAY SC74                INVALID COMMAND
COMMAND INPUT ===>                               SCROLL ===> HALF
PREFIX=SYSLOG DEST=(ALL) OWNER=* SYSNAME=SC74
NP  NAME                                           LogStream
    ASM_LOCAL_SLOT_USAGE                          HZS.HEALTH.CHECKER.HISTORY
    ASM_NUMBER_LOCAL_DATASETS                     HZS.HEALTH.CHECKER.HISTORY
    ASM_PAGE_ADD                                   HZS.HEALTH.CHECKER.HISTORY
    ASM_PLPA_COMMON_SIZE                           HZS.HEALTH.CHECKER.HISTORY
    ASM_PLPA_COMMON_USAGE                           HZS.HEALTH.CHECKER.HISTORY

```

Figure 29-13 LogStream column on the CK panel

The Health Check History (CKH) panel support is invoked as follows:

- ▶ The new L (ListHistory) action on the SDSF Health Checker display, as shown in Figure 29-14.
- ▶ This action is valid for checks that have logstream associated with them.

```

Display Filter View Print Options Search Help
-----
SDSF HEALTH CHECKER DISPLAY SC74                LINE 1-24 (158)
COMMAND INPUT ===>                               SCROLL ===> HALF
PREFIX=SYSLOG DEST=(ALL) OWNER=* SYSNAME=SC74
NP  NAME                                           LogStream
    ASM_LOCAL_SLOT_USAGE                          HZS.HEALTH.CHECKER.HISTORY
    ASM_NUMBER_LOCAL_DATASETS                     HZS.HEALTH.CHECKER.HISTORY
    ASM_PAGE_ADD                                   HZS.HEALTH.CHECKER.HISTORY
    ASM_PLPA_COMMON_SIZE                           HZS.HEALTH.CHECKER.HISTORY
    L ASM_PLPA_COMMON_USAGE                           HZS.HEALTH.CHECKER.HISTORY

```

Figure 29-14 The L action character on the CK panel



## SDSF check history panel

Using the default of 10, when you press Enter for Figure 29-15, you will see 10 history files for the health check.

```
Display Filter View Print Options Search Help
-----
SDSF CK HISTORY  ASM_PLPA_COMMON_USAGE                LINE 1-10 (10)
COMMAND INPUT ==> SET CKLIM 5                        SCROLL ==> HALF
PREFIX=SYSLOG  DEST=(ALL) OWNER=*  SYSNAME=SC74
NP    COUNT    CheckOwner    Status    Result Diag1    Diag2
     244      IBMASM      SUCCESSFUL    0 00000000 00000000
     243      IBMASM      SUCCESSFUL    0 00000000 00000000
     242      IBMASM      SUCCESSFUL    0 00000000 00000000
     241      IBMASM      SUCCESSFUL    0 00000000 00000000
     240      IBMASM      SUCCESSFUL    0 00000000 00000000
     239      IBMASM      SUCCESSFUL    0 00000000 00000000
     238      IBMASM      SUCCESSFUL    0 00000000 00000000
     237      IBMASM      SUCCESSFUL    0 00000000 00000000
     236      IBMASM      SUCCESSFUL    0 00000000 00000000
     235      IBMASM      SUCCESSFUL    0 00000000 00000000
```

Figure 29-15 Health Checker History

From the Health Checker History display shown in Figure 29-15, you can:

- ▶ Browse a specific run of a check
- ▶ Print a specific run of a check

The information provided is for each run of a check and includes:

- ▶ Check owner
- ▶ Status
- ▶ Result
- ▶ Diagnostic data
- ▶ Start and stop time
- ▶ System and sysplex name
- ▶ Check name which is also seen on the panel title line

By default, SDSF collects the last 10 iterations of a health check. To override the default use the new SET CKLIM command, (SET CKLIM 5, as shown in Figure 29-15), which changes the number of history files displayed to 5, as shown in Figure 29-16 on page 612. The minimum number of health check iterations is 1 and the maximum is 999,999.

To override the SDSF default of 10 iterations, use the new **Panel.CK.DefaultCKLim** custom property in ISFPRMxx parmlib member.

Display Filter View Print Options Search Help						
SDSF CK HISTORY ASM_PLPA_COMMON_USAGE				SET COMMAND COMPLETE		
COMMAND INPUT ==>				SCROLL ==> HALF		
PREFIX=SYSLOG DEST=(ALL) OWNER=* SYSNAME=SC74						
NP	COUNT	CheckOwner	Status	Result	Diag1	Diag2
	244	IBMASM	SUCCESSFUL	0	00000000	00000000
	243	IBMASM	SUCCESSFUL	0	00000000	00000000
	242	IBMASM	SUCCESSFUL	0	00000000	00000000
	241	IBMASM	SUCCESSFUL	0	00000000	00000000
	240	IBMASM	SUCCESSFUL	0	00000000	00000000

Figure 29-16 Result of the SET CKLIM 5 command

## 29.5 Enhanced JES2 and JES3 printer support

In z/OS V1R12, the SDSF PR panel no longer requires the use of WebSphere MQ to display sysplex-wide printer information. Instead, SDSF uses the JES SSI function code 83 to obtain device information from all members in the MAS or JES-plex. Both JES2 and JES3 are enhanced in z/OS V1R12 to support SSI function code 83. WebSphere MQ is still required by SDSF to display sysplex-wide data in other device panels.

Moreover, starting with z/OS V1R12, the PR panel supports JES3 printers. Many new fields are added to the PR panel in support of JES3 printers.

For the complete list of fields, see *z/OS SDSF Operation and Customization*, SA22-7670. An example of the PR panel with JES3 printers is shown in Figure 29-17.

Display Filter View Print Options Search Help							
SDSF PRINTER DISPLAY SC75					LINE 1-6 (6)		
COMMAND INPUT ==>					SCROLL ==> PAGE		
NP	PRINTER	Status	Group	SForms	SClass	JobName	JobID
	PRTWA2	OFF	TCPIP	STD	J		
	NS	OFF	TCPIP	VTAM	J		
	PRTWAY	OFF	TCPIP	STD	J		
	IPDPOK	OFF	IPDS	STD	I		
	IPDWAY	OFF	IPDS	STD	K		
	IAZFSS	AV	LOCAL	STD	CD		

Figure 29-17 PR panel, showing printers

### 29.5.1 New action characters on the PR panel

The PR panel now uses SSI 83 to obtain printer information. JES2 can present information about printers on all members across the SSI, without SDSF sysplex communication active, if all MAS members are at z/OS V1R11 or higher.

#### Extended printer name field length

The PRINTER field is a fixed-length field indicating the JES printer name. In SDSF V1R12, the length of the fields is extended to 10 characters for both JES2 and JES3. If you wish to

revert to the previous 8-character length, set the new `Panel.PR.DevnameAlwaysShort` property in `ISFPRMxx` parmlib member.

### JES2 support

The fixed field (device name) is increased to 10 characters for JES2 printers, as follows:

`Rnnnnn.PRn` for high remotes rather than `RnnnnnPn`

### JES3 support

Printers can now be displayed under JES3. New columns and overtypes for both JES2 and JES3 are now available. JES3 printers are controlled using commands such as `*START`, `*RESTART` and `*CALL,WTR`. These commands accept many parameters and can be very long. As shown in Figure 29-17 on page 612, new action characters are added to support JES3 device commands.

Since there are so many options, even the `SET ACTION LONG` command displays the action characters in a new abbreviated way. The term `E+ADHJLMRTX-RestartOptions`, for instance, means specify `E` qualified by one or more of the characters following the `+` sign. For example, specifying `EADHR` results in the command `*R device,A,D,HOLD,RSCD`.

**Note:** Notice that the `E`, `S`, and `X` parameters are abbreviated in the display. If all the combinations and permutations allowed for these had been listed, they would not fit on the panel.

Display Filter View Print Options Search Help										
-----										
SDSF PRINTER DISPLAY SC75					LINE 1-6 (6)					
COMMAND INPUT ==>					SCROLL ==> PAGE					
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,										
<b>ACTION=E+ADHJLMRTX-RestartOptions</b> ,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	FSSName	FSSProc	SysName	JESN	JESLevel	Type	DSPName	DevTyp	
	PRTWA2	PRINTWA2	PRINTWAY	SC75	JES3	z 1.12.0			PRTAFP	
	NS	PRINTWAY	PRINTWAY	SC75	JES3	z 1.12.0			PRTAFP	
	PRTWAY	PRINTWAY	PRINTWAY	SC75	JES3	z 1.12.0			PRTAFP	
	IPDPOK	IPDSWAY	IPDSWAY	SC75	JES3	z 1.12.0			PRTAFP	
	IPDWAY	IPDSWAY	IPDSWAY	SC75	JES3	z 1.12.0			PRTAFP	
	IAZFSS	WTRIAZF	WTRIAZF	SC75	JES3	z 1.12.0			PRTAFP	

Figure 29-18 JES3 printers using the PR command

### Action characters

`E` (restart), `S` (start), and `X` (call) can be qualified by multiple additional characters. For example `EADHR` results in the following command:

`*R device,A,D,HOLD,RSCD`

The action qualifiers can optionally be separated by commas, as follows:

(EA,D,H,R)

For the SDSF SET ACTION command, more combinations and permutations than can be listed on the SET ACTION display are possible. To display valid action characters with a description, use the SET ACTION command. SET ACTION LONG is shown in Figure 29-18 on page 613. SET ACTION SHORT is shown in Figure 29-19.

```

Display Filter View Print Options Search Help
-----
SDSF PRINTER DISPLAY SC75                                SET COMMAND COMPLETE
COMMAND INPUT ===>                                       SCROLL ===> PAGE
ACTION=//,=,+ ,BC,BCn,BCnP,BD,BN,BNn,BNnP,C,CG,CJ,CP,CT,D,DL,E,EA,ED,EH,EJ,EL,
ACTION=EM,ER,ET,EX,Fn,FC,FCn,FCnP,Fn,FnN,FnnP,K,L,LD,S,SA,SD,SM,ST,SX,V,VF,X,
ACTION=XD,XR,XT,XX
NP    PRINTER      SForms  SClass      JobName JobID   Owner   Rec-Cn
      PRTWA2       STD     J
      NS          VTAM    J
      PRTWAY      STD     J
      IPDPOK      STD     I
      IPDWAY      STD     K
      IAZFSS      STD     CD

```

Figure 29-19 SET ACTION SHORT action characters display

### Action character required with fields that can be overtyped

In JES3, most device characteristics are changed using the \*START and \*RESTART commands. Therefore, when trying to change device characteristics in the PR panel by overtyping one of the fields, the error message ACTION REQUIRED is displayed. This message means an action character, such as S or E, must also be specified against the device in order for the change to take effect. This is shown in Figure 29-20.

```

Display Filter View Print Options Search Help
-----
SDSF PRINTER DISPLAY SC75                                NOT AUTHORIZED FOR PRT
COMMAND INPUT ===>                                       SCROLL ===> PAGE
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      SForms  SClass      JobName JobID   Owner   Rec-Cn
S   PRTWA2       STD     X
      NS          VTAM    J
      PRTWAY      STD     J
      IPDPOK      STD     I
      IPDWAY      STD     K
      IAZFSS      STD     CD

```

Figure 29-20 Action Required message on the PR panel

## 29.6 New fields on the INIT panel

In z/OS V1R12, the SDSF INIT panel does not require the use of WebSphere MQ any more to display sysplex-wide initiator information. Instead, SDSF uses JES SSI function code 82 to obtain initiator information from all members in the MAS or JES-plex. Only initiators running on z/OS V1R12 and up are displayed. An example is shown in Figure 29-21.

```

Display Filter View Print Options Search Help
-----
SDSF INITIATOR DISPLAY SC74                               Cursor not on choice
COMMAND INPUT ==>                                         SCROLL ==> HALF
PREFIX=SYSLOG DEST=(ALL) OWNER=* SYSNAME=SC74
NP   ID Status      Classes JobName Stepname ProcStep JobID   C ASID ASID
   1  INACTIVE     ABCDEFG1
   2  INACTIVE     AB
   3  INACTIVE     ABC
   4  INACTIVE     ABC
   5  INACTIVE     ABC
   6  INACTIVE     ABC
   7  DRAINED      A
   8  DRAINED      A
   9  DRAINED      JX
  10  DRAINED      A
WLM ACTIVE                PELEG14                JOB07286 L 78 004E

```

Figure 29-21 INIT panel with WLM initiators

### New fields on the INIT panel

In addition, new fields are added to the INIT panel to include information about WLM-managed initiators. The new fields displayed by scrolling to the right are shown in Figure 29-22 on page 616.

- SrvClass**      Service class of the active job for a JES-managed initiator, or service class the initiator is running for a WLM-managed initiator
- Mode**            Initiator Mode, either JES or WLM
- Group**            Group name (WLM, JES2 or JES3 group name)
- ResType**         Resource Type

Display Filter View Print Options Search Help										
-----										
SDSF INITIATOR DISPLAY SC74							LINE 1-10 (10)			
COMMAND INPUT ==>							SCROLL ==> HALF			
PREFIX=SYSLOG DEST=(ALL) OWNER=* SYSNAME=SC74										
NP	ID	SysName	SysID	JESN	JESLevel	SecLabel	SrvClass	Mode	Group	ResType
	1	SC74	SC74	JES2	z/OS1.12			JES	JES2	INIT
	2	SC74	SC74	JES2	z/OS1.12			JES	JES2	INIT
	3	SC74	SC74	JES2	z/OS1.12			JES	JES2	INIT
	4	SC74	SC74	JES2	z/OS1.12			JES	JES2	INIT
	5	SC74	SC74	JES2	z/OS1.12			JES	JES2	INIT
	6	SC74	SC74	JES2	z/OS1.12			JES	JES2	INIT
	7	SC74	SC74	JES2	z/OS1.12			JES	JES2	INIT
	8	SC74	SC74	JES2	z/OS1.12			JES	JES2	INIT
	9	SC74	SC74	JES2	z/OS1.12			JES	JES2	INIT
	10	SC74	SC74	JES2	z/OS1.12			JES	JES2	INIT
	<b>WLM</b>	<b>SC74</b>	<b>SC74</b>	<b>JES2</b>	<b>z/OS1.12</b>		<b>DFLT_MG</b>	<b>WLM</b>	<b>WLM</b>	<b>INIT</b>

Figure 29-22 New fields on the INIT panel

## 29.7 New actions on the SR panel

z/OS V1R12 supports specifying an automatic reply policy to system WTORs in a PARMLIB member. Sometimes, the operator may want to tell the system not to automatically reply to an outstanding WTOR and take manual action later on. To allow this, a new AI action character is added on the SR panel. When you issue AI next to a reply ID on the SR panel, the system is instructed to ignore automatic replies for that reply ID.

More specifically, SDSF generates the following command when AI is issued against a reply id, as follows:

```
SETAUTOR IGNORE,RID=reply-id
```

The AI action character requires read access to the **ISFSR.REPLY.system.jobname** SAF resource.

## 29.8 New SEARCH command

Starting with SDSF in z/OS V1R12, it is possible to search the SDSF help from any panel using the new SEARCH command. The syntax of the SEARCH command is:

```
SEARCH phrase
```

Where phrase is one to four words. If you omit phrase, SEARCH displays the Search pop-up, on which you can type the search phrase. To search for the complete phrase, enclose it in single quotation marks. Figure 29-23 on page 617 shows the results of the following SEARCH command:

```
search ziip
```

```

Command ==>                               Search Help                               Row 1 to 15 of 24

Search for: zIIP                               Show titles

Select a search result to display the help panel.
zIIP-Time Accumulated zIIP service time, in seconds (RMF)
zIIP-NTime Normalized zIIP service time, in seconds (RMF)
zIIP-Use% Percent of the total zIIP time used by the address
zIIP-Time Accumulated zIIP service time, in seconds
zIIP-NTime Normalized zIIP service time, in seconds
zIIP-Time Accumulated zIIP time, in seconds
zIIP-NTime Normalized zIIP time, in seconds
SzIIP% zIIP view of CPU use for the system, in the
was eligible for a zIIP, in seconds (RMF)
GCPU-Use%, zAAP-Use and zIIP-Use% columns: Values are
general CPU, zAAP or zIIP time used by the job
general CPU, zAAP or zIIP time used by all jobs
zIIP-Time and zICP-Time are not normalized.
zIIP-Time source field is R791TSUP
zIIP-NTime is normalized to the slower CP, to facilitate
F1=Help      F3=Cancel      F7=Backward      F8=Forward      F12=Cancel

```

Figure 29-23 Search results pop-up

**Tip:** To search for information about the SEARCH command, issue: SEARCH search

## 29.9 Coexistence with earlier releases

If you share the ISFPRMxx parmlib member with SDSF V1R10 or V1R11 systems, the toleration PTFs associated with APAR PK86390 have to be applied:

- ▶ For SDSF V1R10, apply PTF UK90017. If running SDSF with JES2, also apply PTF UK90022.
- ▶ For SDSF V1R11 apply PTF UK90018. If running SDSF with JES2, also apply PTF UK90023.







## JES2 enhancements

JES2 is that component of MVS that provides the necessary functions to get jobs into, and output out of, the MVS system. It is designed to provide efficient spooling, scheduling, and management facilities for the MVS operating system.

JES2 maintains copies of its data sets that contain job and output queues (that is, lists of jobs to be processed by z/OS) on direct access storage devices (DASD). Future work is added to these queues and JES2 selects work for processing from them. These data sets and queues must remain current and accurate to maintain system integrity and performance. This chapter offers a discussion of the JES2 spool and checkpoint data sets and the processing JES2 uses to maintain them.

The following topics are described:

- ▶ EAV support for spool and checkpoint data sets
- ▶ Enhancements to possible \$S SPOOL error messages during dynamic allocation
- ▶ Better support for multiple TSO/E logons
- ▶ Mean time to restart improvements
- ▶ Other internal enhancements of JES2 processing

## 30.1 EAV support for spool and checkpoint data sets

Starting with z/OS V1R12, DFSMS supports allocation of all data set types in the extended addressing space (EAS) of extended address volumes (EAV). To complement this support, JES2 is enhanced in z/OS V1R12 to support allocation of the spool and checkpoint data sets in EAS. The new support provides greater flexibility when allocating spool data sets and potentially requires less spool volumes with larger spool data sets.

### 30.1.1 Introduction to spool addressing

Prior to z/OS V1R7, JES2 used a spool addressing scheme known as module track record (MTTR). The MTTR is a 4-byte address, in which:

- M** 1 byte is used to address spool extents.
- TT** 2 bytes are used to address spool tracks.
- R** 1 byte is used to address records within a track.

This addressing scheme forms a limitation of 255 spool volumes, 65,535 tracks per spool volume and 255 records per track.

#### Support for large format sequential data sets

The MTTtr addressing scheme was designed to complement the z/OS support for large sequential data sets, also introduced in z/OS V1R7. The MTTtr addressing scheme raises the maximum number of tracks per spool volume to 1,048,575 (x'FFFFFF'). This is because MTTtr uses 20 bits to address spool tracks instead of 2 bytes (16 bits). This support is also known in JES2 as LARGEDS support. The \$T SPOOLDEF command is used to control whether large spool data sets are allowed in your JES2 configuration. To instruct JES2 to use MQTRs to address spool volumes, issue the command \$T SPOOLDEF,LARGEDS=ALLOWED. In z/OS V1R7, JES2 introduced a new spool addressing scheme known as MTTtr, in which:

- M** 1 byte is used to address spool extents.
- TTt** 20 bits are used to address spool tracks.
- r** 4 bits are used to address records within a track.

**Note:** The change from MTTR to MTTtr decreases the number of supported records per track from 255 to 15. This is because in MTTtr, only 4 bits are used to address records, instead of 8. However, this does not pose a big limitation since the recommended spool buffer size for a 3390 volume is 3992 bytes. Using a buffer size of 3992 bytes requires only 12 records per track. The minimal buffer size supported with MTTR is 2944 bytes.

#### Support for EAS-eligible sequential data sets

JES2 is continuing its migration from the 4-byte MTTR to a 6-byte MQTR. This removes the architectural limit of 1 M tracks per spool data set. There are two major changes in z/OS V1R12 in this area:

- ▶ The first is the completion of the support for variable length PDDBs. There has been a runtime length field (PDBSIZE) in the PDDb for years. But the assembly time length PDBLENG has also existed. With this release, JES2 has completed the work needed to support variable size PDDBs. So at this time, we are deleting the compile time PDDb length field (PDBLENG). The size of the PDDb in z/OS V1R12 is constant, but IBM suggests not using the compile time PDDb length. You should be using services to access the data in the PDDb or services to step through the PDDb.

- ▶ The second change is additional migration of pointers in spool control blocks from MTTRs to MQTRs. The control blocks affected always have MQTRs in memory (they are translated when read or written from spool by \$CBIO). If you still access any of these SPOOL addresses directly, you will either need to migrate to the MQTR form of the SPOOL address or use other services to get the information you need.

Starting with z/OS V1R7 and over the next z/OS releases, JES2 internally started implementing support for a new spool addressing scheme known as module quad track record (MQTR). MQTR is planned to completely replace MTTR in the future. The MQTR is a 6-byte address, in which:

- M** 1 byte is used to address spool extents.
- QT (or TTTT)** 4 bytes are used to address spool tracks.
- R** 1 byte is used to address records within a track.

### EAV volumes

z/OS V1R10 introduced support for extended address volumes (EAV), which removes the architectural limitation of 65,520 cylinders per DASD volume. Starting with z/OS V1R10, the new maximum number of cylinders per volume is 268,434,453 (about 223 GB). However, in z/OS V1R10 and z/OS V1R11, sequential data sets were not eligible for allocation in the extended addressing space (EAS) of an EAV.

In z/OS V1R12, DFSMS allows all types of data sets to be allocated in EAS space. EAS is also known as cylinder-managed space. JES2 is updated in z/OS V1R12 to support allocation of the spool and checkpoint data sets in cylinder-managed space.

**Note:** Spool data sets can be placed anywhere on an EAV in JES2 z/OS V1R12. However, this support cannot be larger than 1,048,575 tracks, which is the JES2 architectural limit.

## 30.1.2 Spool addressing changes in z/OS V1R12

In z/OS V1R12, JES2 continues its migration from the 4-byte MTTR to the 6-byte MQTR. This is in order to eventually remove the architectural limit of 1,048,575 tracks per spool data set. Two major changes are made in this area in z/OS V1R12:

- ▶ The first change is the completion of the support for variable length peripheral data definition blocks (PDDB). The PDDB contains or points to all characteristics, known at the time of creation of the PDDB, of each subsystem data set known to JES2. PDDBs are contained in the Input/Output Table (IOT), which is a spool-resident JES2 job control block. There is a PDDB for each instance of a spool data set.

The field PDBSIZE in the PDDB indicates the runtime length of the PDDB. The field PDBLENG indicates the assembly time length. In z/OSV1R12, JES2 has completed the support for variable size PDDBs. The PDDB assembly length field (PDBLENG) is deleted.

Even though the size of the PDDB in z/OS V1R12 is still constant, we highly suggest not using the compile time PDDB length. Use JES2 services to access the data in the PDDB or services to step through the PDDB.

- ▶ The second change is the additional migration of pointers in spool control blocks from MTTRs to MQTRs. The affected spool control blocks are:

- CHK** FSI checkpoint record
- JCT** Job control table
- NHSB** Network header/trailer spool block
- OCT** Output control table

The affected control blocks always use MQTRs in memory. The MQTRs are translated by \$CBIO, if needed, during spool read or write. If you still access any of these spool control block addresses directly, you need to migrate to the MQTR form of the spool address or use other JES2 services to get the needed information.

## 30.2 Using spool data sets in EAV cylinder-managed space

JES2 exploitation of the EAV support added in z/OS V1R12 allows you to define EAS-eligible data sets for use as SPOOL extents and checkpoint data sets. The data sets can be placed anywhere on an EAV volume but they cannot be larger than the 1,048,575 track JES2 architectural limit.

The ability to create or use a spool and checkpoint data set that is in EAS storage is controlled by the new `CYL_MANAGED` keyword on the `SPOOLDEF` statement. By default, `CYL_MANAGED` is set to `FAIL`. This allows down-level members (prior to z/OS V1R12) to continue to access the spool and checkpoint volumes. To activate the support for EAS data sets, you first must complete your migration to z/OS V1R12 (once you activate the support you cannot warm start down-level members into the MAS), and ensure that `LARGEDS` on the `SPOOLDEF` statement is not set to `FAIL`. Given those two requirements, you can then use an operator command to set `SPOOLDEF CYL_MANAGED=ALLOWED`.

**Note:** IBM suggests stabilizing on z/OS V1R12 before considering setting `CYL_MANAGED=ALLOWED`.

### Setting the proper environment for EAV support

Use the `$T SPOOLDEF,CYL_MANAGED=ALLOWED` command to enable both spool and checkpoint data set allocation in EAV cylinder-managed space. Before issuing the command, two preconditions must be met:

- ▶ All MAS members must be at JES2 z/OS V1R12.
- ▶ `LARGEDS` must be set to `ALLOWED` or `ALWAYS`, not `FAIL`.

The default, `CYL_MANAGED=FAIL`, disables this support. Note that the command `$T SPOOLDEF,CYL_MANAGED=FAIL` is not allowed when:

- ▶ A spool or checkpoint data set currently resides in EAV storage.
- ▶ Or a start spool command is pending.

**Important:** Once `CYL_MANAGED=ALLOWED` is set, any MAS members at a release prior to JES2 z/OS V1R12 are not allowed to join the MAS. Even if `CYL_MANAGED=FAIL` is later set, members prior to JES2 z/OS V1R12 are not allowed to join the MAS. `CYL_MANAGED=ALLOWED` triggers additional exploitation of MQTRs, which are only compatible with JES2 z/OS V1R12.

To better identify the format of a spool volume, a new `ATTRIBUTE` keyword was added to the `$D SPOOL(xxxxx),UNITDATA` command. The possible values are `ABSOLUTE`, `RELATIVE`, `LARGEDS`, or `CYL_MANAGED`. Use the `$D SPOOL,UNITDATA` command to check whether a spool data set is currently allocated in cylinder-managed space. The command is enhanced to indicate data set placement, as shown in Figure 30-1 on page 623.

```

$dspool (spool1),unitdata
$HASP893 VOLUME(SPOOL1)
$HASP893 VOLUME(SPOOL1) UNITDATA=(EXTENT=00,TRKRANGE=(0001,
$HASP893 020D),BASETRAK=00000086,RECMAX=12,
$HASP893 TRKPERCYL=15,ATTRIBUTE=CYL_MANAGED)
$HASP646 6.2857 PERCENT SPOOL UTILIZATION

```

Figure 30-1 \$D SPOOL,UNITDATA command with cylinder-managed spool data set

### 30.2.1 Allocating a spool data set in EAV cylinder-managed space

The JCL in Figure 30-2 shows how to allocate the maximum size spool data set (1,048,575 tracks, or 69,905 cylinders) currently supported. We use the EATTR=OPT JCL parameter to indicate that this data set is eligible to reside in cylinder-managed space. Since the data set is larger than 65,520 cylinders, it must be allocated, at least in part, in cylinder-managed space.

```

//ALLOCSPL JOB ACCNT#,MSGLEVEL=(1,1),NOTIFY=&SYSUID
//*
//ALLOCATE EXEC PGM=IEFBR14
//SPOOLEAS DD DISP=(NEW,KEEP),DSN=SYS1.HASPACE,
// DCB=(DSORG=PSU),SPACE=(TRK,1048575,,CONTIG),
// DSNTYPE=LARGE,EATTR=OPT,UNIT=3390,VOL=SER=BH5SP2
//

```

Figure 30-2 Allocating a spool data set with EATTR=OPT

### 30.2.2 Dynamically adding a cylinder-managed spool data set to JES2

Figure 30-3 on page 624 shows our initial JES2 spool environment, before adding a new spool volume. We have no cylinder-managed spool data sets and CYL\_MANAGED is set to FAIL. We do have LARGEDS support turned on.

```

$D ACTIVATE
$HASP895 $DACTIVATE
$HASP895 JES2 CHECKPOINT MODE IS CURRENTLY Z11
$HASP895 CURRENT CHECKPOINT SIZE IS 409 4K RECORDS.
$HASP895 CURRENT NUMBER OF BERTS IS 2100.
$HASP895 PERCENT BERTS UTILIZED IS 15 PERCENT.
$D SPL,UNITDATA
$HASP893 VOLUME(BH5SP1)
$HASP893 VOLUME(BH5SP1) UNITDATA=(EXTENT=00,TRKRANGE=(0001,
$HASP893 C2D3),BASETRAK=000000D1,RECMAX=12,
$HASP893 TRKPERCYL=15,ATTRIBUTE=RELATIVE)
$HASP646 31.9097 PERCENT SPOOL UTILIZATION
$DSPoolDEF
$HASP844 SPOOLDEF
$HASP844 SPOOLDEF BUFSIZE=3856,DSNAME=SYS1.HASPACE,
$HASP844 FENCE=(ACTIVE=NO,VOLUMES=1),GCRATE=NORMAL,
$HASP844 LASTSVAL=(2005.182,17:42:57),LARGEDS=ALLOWED,
$HASP844 SPOOLNUM=32,CYL_MANAGED=FAIL,TGFSIZE=60,
$HASP844 TGSPACE=(MAX=97728,DEFINED=9975,ACTIVE=9975,
$HASP844 PERCENT=31.9097,FREE=6792,WARN=80),TRKCELL=3,
$HASP844 VOLUME=BH5SP

```

Figure 30-3 Our initial spool configuration

We use the \$T SPOOLDEF command to enable our environment for cylinder-managed spool data sets, as shown in Figure 30-4. All members in our MAS are at z/OS V1R12.

```

$T SPOOLDEF,CYL_MANAGED=ALLOWED
$HASP844 SPOOLDEF 606
$HASP844 SPOOLDEF BUFSIZE=3856,DSNAME=SYS1.HASPACE,
$HASP844 FENCE=(ACTIVE=NO,VOLUMES=1),GCRATE=NORMAL,
$HASP844 LASTSVAL=(2005.182,17:42:57),LARGEDS=ALLOWED,
$HASP844 SPOOLNUM=32,CYL_MANAGED=ALLOWED,TGFSIZE=60,
$HASP844 TGSPACE=(MAX=97728,DEFINED=9975,ACTIVE=9975,
$HASP844 PERCENT=32.4210,FREE=6741,WARN=80),TRKCELL=3,
$HASP844 VOLUME=BH5SP

```

Figure 30-4 Changing spool definition to support data sets in cylinder-managed space

Then we dynamically add the spool data set to our spool configuration, using the \$S SPOOL command. This is shown in Figure 30-5 on page 625. The spool data set was previously allocated, as shown in Figure 30-2 on page 623. Note that adding such large spool volumes requires sufficient number of track groups defined to JES2. The number of track groups can be updated using the \$T SPOOLDEF,TGSPACE=(MAX=nnnn) command.

```

$$ SPOOL(BH5SP2)
$HASP893 VOLUME(BH5SP2) STATUS=INACTIVE,COMMAND=(START)
$HASP646 34.3258 PERCENT SPOOL UTILIZATION
IEF196I IEF237I DD65 ALLOCATED TO $BH5SP2
$HASP423 BH5SP2 IS BEING FORMATTED
IEF196I $HASP423 BH5SP2 IS BEING FORMATTED
...
$HASP630 VOLUME BH5SP2 ACTIVE      0 PERCENT UTILIZATION
$DSPOOL,UNITDATA
$HASP893 VOLUME(BH5SP1) 300
$HASP893 VOLUME(BH5SP1) UNITDATA=(EXTENT=00,TRKRANGE=(0001,
$HASP893                      C2D3),BASETRAK=000000D1,RECMAX=12,
$HASP893                      TRKPERCYL=15,ATTRIBUTE=RELATIVE)
$HASP893 VOLUME(BH5SP2) 301
$HASP893 VOLUME(BH5SP2) UNITDATA=(EXTENT=01,TRKRANGE=(00000001,
$HASP893                      000FFFFF),BASETRAK=00000176,RECMAX=12,
$HASP893                      TRKPERCYL=15,ATTRIBUTE=CYL_MANAGED)
$HASP646 3.5118 PERCENT SPOOL UTILIZATION

```

Figure 30-5 Dynamically adding a spool volume

We let JES2 format the newly added spool volume. Since the spool volume is very large, it was not added immediately. In our environment, it took JES2 13 minutes and 10 seconds to format the new spool volume, before it was added.

### 30.3 Enhanced \$\$ SPOOL error messages

Currently, when a \$\$ SPOOL command fails because of dynamic allocation, all that is displayed is a return code that allocation failed. There is no indication as to why the allocation failed. To address this shortcoming, the HASP443 message was enhanced to display the messages that are returned by the DYNALLOC request. These are generally IKJ messages normally associated with TSO ALLOCATE command failures.

The \$HASP443 error message is used to explain the dynamic allocation error for a started spool volume. Dynamic allocation errors can especially happen when using the \$\$ SPOOL command to allocate a new spool volume. Support for allocating a new spool data set using the \$\$ SPOOL command was added in z/OS V1R11, using the new SPACE= parameter.

#### \$\$ SPOOL example

Starting a spool volume and there is not enough space on the volume, as follows:

```

$sspl(spool2),space=(cyl,30000)
$HASP893 VOLUME(SPOOL2) STATUS=INACTIVE,COMMAND=(START)
$HASP646 4.3809 PERCENT SPOOL UTILIZATION
$HASP443 SPOOL2 NOT ALLOCATED
DYNAMIC ALLOCATION FAILURE, RC=04
IKJ56245I DATA SET SYS1.HASPACE NOT ALLOCATED, NOT ENOUGH SPACE ON VOLUME
IKJ56245I USE DELETE COMMAND TO DELETE UNUSEDDATA SETS

```

### 30.3.1 z/OS V1R12 enhancement

In z/OS V1R12, the \$HASP443 message is enhanced to include the dynamic allocation error messages that explain the reason for dynamic allocation failure. Dynamic allocation messages are generally prefixed with IKJ. An example is shown in Figure 30-6.

```
$ssp1 (spool2) , space=(cy1,30000)
$HASP893 VOLUME(SPOOL2) STATUS=INACTIVE,COMMAND=(START)
$HASP646 4.3809 PERCENT SPOOL UTILIZATION
$HASP443 SPOOL2 NOT ALLOCATED
          DYNAMIC ALLOCATION FAILURE, RC=04
          IKJ56245I DATA SET SYS1.HASPACE NOT ALLOCATED, NOT
          ENOUGH SPACE ON VOLUME
          IKJ56245I USE DELETE COMMAND TO DELETE UNUSED
          DATA SETS
```

Figure 30-6 Enhanced \$HASP443 error message

### 30.3.2 Improved HASP414 message

During a \$S SPOOL command, JES2 uses the OBTAIN system service to get information about the specified spool data set. In case of an error with OBTAIN, message \$HASP414 is used to return the error information.

#### z/OS V1R12 enhancement

When preparing to allocate either a checkpoint or spool data set, JES2 uses the OBTAIN supervisor service to get information about the specified data set. When OBTAIN fails, JES2 issues the HASP414 with simply the OBTAIN return code. The return code alone is not adequate to diagnose the problem.

To address this, descriptive text was added to the HASP414 message to describe the meaning of the return code.

In z/OS V1R12, message \$HASP414 is also enhanced to contain the error message, and not just the error return code. An example is shown in Figure 30-7.

```
$ssp1 (spool3)
$HASP893 VOLUME(SPOOL3) STATUS=INACTIVE,COMMAND=(START)
$HASP646 1.3333 PERCENT SPOOL UTILIZATION
$HASP414 MEMBER IBM1 -- OBTAIN FAILED FOR DATASET=SYS1.HASPACE ON
          VOLSER=SPOOL3 WITH CC 24 -- DATA SET RESIDES IN CYLINDER
          MANAGED STORAGE AND SPOOLDEF,CYL_MANAGED=FAIL
$HASP443 SPOOL3 NOT ALLOCATED
          OBTAIN FAILURE, RC=01
```

Figure 30-7 Enhanced \$HASP414 message

## 30.4 Better support for multiple TSO/E logons

Starting with z/OS V1R12, JES2 officially supports multiple TSO/E logons of the same user ID in the MAS. Even though it was possible to log on to TSO/E multiple times in the same MAS before this release, JES2 did not officially support it. The problem was when JES2 had



to notify a TSO/E user ID about a job. Before z/OS V1R12, there was no way to guarantee that the user ID that submitted the job would be the one to receive the JES2 notification. In many cases, a job was submitted by a user ID running in one system in the MAS, but the JES2 notify message was issued back to a user ID running in a different system.

In z/OS V1R12, when deciding where to send a notify message, JES2 prefers the member number of the submitting JES2. If the notify user ID is logged on to that member, then that is where JES2 sends the notify message. This applies even if the notify user ID is not the user ID that originally submitted the job (if, for example, the user logged off and then back on). If the notify user ID is not on the same NJE node as the submitter of the job, then the old processing is used, as before z/OS V1R12.

The NOTIFY= parameter on the JCL job card continues to work as before. This means that there is currently no way to ask that the notify message be directed to a specific member in the MAS. However, the notify SSI, which actually sends the message, does support specifying which member to notify, and that member is used when provided.

## 30.5 Mean time to restart improvements

IEATEDS is a new system service provided in z/OS V1R12 that allows users to record events to a Timed Event Data Table and provide information that will help determine flow and performance. Each event is time stamped and includes data provided by the caller and additional data collected by the service.

In z/OS V1R12, JES2 adds support to collect the new timed event records during initialization, using the IEATEDS service. These records are meant to be used as part of an overall effort to understand and eventually reduce JES2 restart delays. JES2 registers with the service using the JES2 subsystem name. For example, if the name of your JES2 subsystem is HASP, then that is the component name JES2 will use to record events using IEATEDS. If you run a poly-JES, then that subsystem name will be the component for the poly-JES records.

Currently, JES2 records the following information:

- ▶ Initialization phase times, similar to the information provided by the \$D PERFDATA(INITSTAT) command
- ▶ Delays associated with HASP709 messages
- ▶ Exit delays
- ▶ The time to complete warm start

## 30.6 Stop using captured UCBs for spool volumes

In previous versions of JES2, recommendations were made to users not to move the UCB for spool volumes above the line. This is because JES2 captures the UCB for the spool volume to common storage for use in the DEB data area that JES2 builds to access the spool. However, if the user does move the UCB above the line (many users move all their UCBs above the line), they are exposed to an undiagnosed field problem with the captured UCB. The problem surfaces as an error trying to access a spool volume on one system. This is because the UCB that the JES2 DEB points to has been unexpectedly uncaptured. What is suspected is that the capture count for the page with the UCB has gone bad (too many uncaptures of a different UCB on the same page) and the page gets uncaptured prematurely. This is difficult to diagnose since the error could have occurred hours or days before the JES2 symptom.

## JES2 V1R12 implementation

In the z/OS V1R12 DFSMS changes, there is a change to the DASD extension to the DEB that supports a 31-bit UCB address. JES2 will delete the code that captures the UCB for spool and instead build the DASD DEB extension that supports 31-bit DEB addresses.

Starting with z/OS V1R12, JES2 no longer uses captured UCBs for spool volumes. JES2 makes use of the XTIO allocation enhancements, also introduced in z/OS V1R12. Instead of capturing a 31-bit UCB address in 24-bit storage, JES2 builds a DASD DEB extension that supports 31-bit DEB addresses.

This allows JES2 to do away with the UCB capture for spool volumes.

## 30.7 SSI updates

With z/OS V1R12, SSI 83 with the SSOB extension IAZSSJD, is introduced to obtain information on local and remote printers. This information is available on z/OS V1R12 from any z/OS V1R11 MAS member and later. Only local and remote printers are currently supported via the SSI (though the IAZSSJVD data area does define additional devices). The information is provided via data gather code that was available in z/OS V1R11. To fully support requests from z/OS V1R12, you must apply the data gatherer APARs OA31703 and OA32712 on your z/OS V1R11 system.

These APARs are a prerequisite for the JES2 toleration APAR OA28532. The initiator subfunction of the JES property SSI was enhanced to support requesting information from other members. This requires the target member to be running z/OS V1R12.

Several enhancements to the SSI are available in z/OS V1R12:

- ▶ SSI function code 83 - This is a new SSI in z/OS V1R12. It is used to obtain information about local and remote JES printers. This information is available on z/OS V1R12 from any z/OS V1R11 MAS member and later. Only local and remote printers are currently supported via the SSI, even though the IAZSSJVD data area does define additional devices.
- ▶ The initiator subfunction of the JES properties SSI (SSI function code 82) is enhanced to support requesting information from other members.
- ▶ SYSOUT ENF processing is enhanced to support requesting ENFs for data sets selected by SAPI. This is useful if a SAPI application is requesting SYSOUT information and returning it for later processing, but still wants to track the status of the data set.

## 30.8 Serviceability updates

A number of serviceability updates were made in this release:

- ▶ Improved formatting of the existing SAPI JES2 \$TRACES. The current traces dump pure hex data. To locate a specific field, you must know its offset. This makes searching a trace much more difficult.

With the improved formatting, all fields are formatted as field=value. This significantly increases the size of the output, but makes processing the records easier.

- ▶ The standard JES2 \$SAVE/\$RETURN trace IDs 11, 12, 18, and 19 have been updated to include the SSOB and extension when tracing entry and exit to any SSI routine. The data is currently dumped in raw hex.

- ▶ The IPCS formatter \$MODLOC (locates which JES2 module a passed address is in) has been updated to format the last APAR/PTF applied to the module.
- ▶ The \$HASP473 message (\$CPOOL Build failure) has been updated to include the JOBNAME that was trying to build the \$CPOOL.

## 30.9 Migration and coexistence considerations

The recommended migration path to z/OS V1R12 is from z/OS V1R10 or z/OS V1R11. APAR OA28532 must be installed for MAS coexistence with z/OS V1R12 JES2. It is also highly recommended on down-level releases for support.

However, there is no coexistence support, and fallback will result in problems with SYSLOG data sets printing on JES mode or RJE printers (due to changes made in z/OS V1R11).

### **From JES2 z/OS V1R7, V1R8, V1R9**

You do an all member warm start of z/OS V1R12 JES2. There is no coexistence support. Therefore, there are fallback implications. Some new data structures are created by z/OS V1R11 JES2 that may result in problems in z/OS V1R8 and prior releases.

### **From JES2 z/OS V1R10 or z/OS V1R11**

APAR OA28532 is needed on z/OS V1R10 or z/OS V1R11 to coexist in the MAS with z/OS V1R12. The PFTs are:

PTFs - HJE7750 - UA52850, HJE7760 - UA52851

Also recommended for fallback are the following APARS for the new and updated z/OS V1R12 SSI interfaces.

APARs OA31703 and OA32712 are required to correct functional problems

**Note:** A direct migration from releases prior to z/OS V1R7 is not recommended.





## JES3 enhancements

JES3 runs on either one processor or up to thirty-two processors in a sysplex. A sysplex is a set of MVS systems communicating and cooperating with each other through certain multisystem hardware components and software services to process client workloads.

In a sysplex, your installation must designate one processor as the focal point for the entry and distribution of jobs and for the control of resources needed by the jobs. That processor, called the *global processor*, distributes work to the other processors, called *local processors*.

It is from the global processor that JES3 manages jobs and resources for the entire complex, matching jobs with available resources. JES3 manages processors, I/O devices, volumes, and data. To avoid delays that result when these resources are not available, JES3 ensures that they are available before selecting a job for processing.

JES3 keeps track of I/O resources, and manages workflow in conjunction with the workload management component of MVS by scheduling jobs for processing on the processors where the jobs can run most efficiently. At the same time, JES3 maintains data integrity. JES3 will not schedule two jobs to run simultaneously anywhere in the complex if they are going to update the same data.

This chapter describes the following enhancements to z/OS V1R12 JES3:

- ▶ JES3 EAV support for the spool, JCT data sets, and checkpoint
- ▶ SAPI enhancements

## 31.1 EAV support for spool, checkpoint and JCT data sets

Starting with z/OS V1R12, DFSMS supports allocation of all data set types in the extended addressing space (EAS) of extended address volumes (EAV). To complement this support, JES3 is enhanced in z/OS V1R12 to support allocation of the spool, checkpoint, and JCT data sets in EAS. The new support provides greater flexibility when allocating JES3 data sets and potentially requires less spool volumes with larger spool data sets.

Managing spool space is not changed in z/OS V1R12. All JES3 interfaces that handle spool information are updated to support large track numbers (up to 28-bit track numbers):

- ▶ Job validation SNAP output
- ▶ The following JES3 Monitoring Facility (JMF) reports:
  - Spool data set description
  - Single track table space allocation snapshot
- ▶ JES properties (SSI function code 82) support for spool volume information
- ▶ FORMAT, TRACK and BADTRACK initialization statements
- ▶ The \*MODIFY Q command
- ▶ JES3 messages
- ▶ The JES3 JCT utility (IATUTJCT)

### 31.1.1 Allocating a spool data set in EAS

The JCL in Figure 31-1 shows how to allocate a very large spool data set (120,000 cylinders) in a single extent. We use the EATTR=OPT JCL parameter to indicate that this data set is eligible to reside in EAS. Since the data set is larger than 65,520 cylinders, it must be allocated, at least in part, in EAS.

```
//ALLOCS P3 JOB ACCNT#,MSGLEVEL=(1,1),NOTIFY=&SYSUID
//ALLOCATE EXEC PGM=IEFBR14
//SPOOLEAS DD DISP=(NEW,CATLG),DSN=SYS1.JES3SPL2,
// DCB=(RECFM=U,BLKSIZE=4084),SPACE=(CYL,120000,,CONTIG),
// DSNTYPE=LARGE,EATTR=OPT,UNIT=3390,VOL=SER=BH5JS4
//
```

Figure 31-1 Allocating a spool data set in EAS

### 31.1.2 Adding the spool data set to JES3

After allocating the new spool data set, as shown in Figure 31-1, we update the JES3 procedure to allocate it as DDNAME SPOOL2. We also update the JES3 inish deck to format it. The inish deck statement is shown in Figure 31-2.

```
FORMAT,DDNAME=SPOOL2
```

Figure 31-2 JES3 inish deck statement to format a spool data set

Formatting a new spool data set requires a JES3 warm start. Note that a JES3 warm start requires an IPL. So once the inish deck is updated, we IPL our system and start JES3 with a warm start. Figure 31-3 on page 633 shows the messages issued during JES3 initialization.

```

S JES3,PARM=NOREQ
IAT6360 CHECKPOINT DATA SET <CHKPNT2> UNAVAILABLE - CHKPNT2 DD NOTDEFINED.
IAT6359 *** WARNING *** ONLY ONE JES3 CHECKPOINT DATA SET IN USE.
IAT3040 STATUS OF JES3 PROCESSORS IN JESXCF GROUP WTSCPLX4
IAT3040 SC74 ( ), SC70 ( ), SC75 +UP+
IAT3011 SPECIFY JES3 START TYPE
029 IAT3011 (C, L, H, HA, HR, HAR, W, WA, WR, WAR, OR CANCEL)
R 29,W
030 IAT3012 SELECT JES3 INISH ORIGIN (N OR M=), AND OPTIONAL EXIT PARM
(,P=) OR CANCEL
R 30,M=GL
IEE600I REPLY TO 030 IS;M=GL
IAT4038 FORMATTING BYPASSED FOR PREVIOUSLY FORMATTED SPOOL DATA SET "SPOOL1 "
IAT4031 FORMATTING OF SPOOL DATA SET "SPOOL2 " IN PROGRESS
IAT6397 FCT JSS (NODEVICE) HAS BEEN DISPATCHED CONTINUOUSLY FOR:
IAT6398 00000 HOURS 00 MINUTES 30 SECONDS
IAT6415 at 0002B70A in LOADMOD=IATDMVR , EPNAME=IATDMVR ,EPADDR=0002AA
68,LEN=001290
031 IAT6418 Reply 'JES3 ' to fail JES3 INITIALIZATION or 'nnn' to ask later
R 31,999
IEE600I REPLY TO 031 IS;999
IAT4032 FORMATTING OF SPOOL DATA SET "SPOOL2 " COMPLETE, NO ERRORS
IAT4030 0002 SPOOL DATA SETS IN USE
IAT4075 MAXIMUM NUMBER OF JOBS IS LIMITED TO 0021461 BY JCT DATA SET SIZE
6. IAT6365 CHECKPOINT WRITE FAILED DUE TO INSUFFICIENT SPACE - PTC ,CHKPNT
7. IAT6368 CHECKPOINT SPACE PROBLEM - DATA SET <CHKPNT > HAS INSUFFICIENT
SPACE.
8. IAT6365 CHECKPOINT WRITE FAILED DUE TO INSUFFICIENT SPACE - PTC ,CHKPNT
9. 032 IAT4142 ERROR WRITING THE PARTITION TAT CKPT RECORD, RC=12,
10. (CONTINUE OR CANCEL)
11. R 32,CONTINUE
...

```

Figure 31-3 JES3 startup messages with the new EAS spool data set

Since the new spool data set is very large, it took JES3 in our environment over 17 minutes to format it. If this is an unacceptable startup time in your environment, we suggest that you format the spool data set using the IEBDG utility prior to restarting JES3 with a warm start.

### Looking at spool volume attributes with SDSF

We use the SDSF spool volumes (SP) panel to look at the EAS spool data set attributes. The spool data set is allocated using the SPOOL2 DD name. Figure 31-4 on page 634 shows the low track numbers for SPOOL2. The low track number is x'E1' and the high track number is x'1B7820'. The difference between the two is x'1B773F', which is equal to 1,800,000 in decimal. This is the number of tracks in SPOOL2, which is equal to 120,000 cylinders used in the JCL to allocate SPOOL2. The JCL is shown in Figure 31-1 on page 632.

Furthermore, the TGPct (spool utilization in percents) column shows that the new spool volume is only 1% full and contains 1,800,000 track groups. Use the SDSF ARRANGE command to make the TGNum column wide enough to clearly display the number of track groups. In our environment, we set it to 7 characters wide using the ARRANGE TGNUM 7 command.

Display Filter View Print Options Search Help											
-----											
SDSF SPOOL DISPLAY SC75			1%	ACT	1772K	FRE	1766K	LINE	1-5	(5)	
COMMAND INPUT ==>						SCROLL ==> HALF					
NP	NAME	Status	TGPct	TGNum	TGUse	Ext	LoCyl	LoTrk	LoHead	HiCyl	HiTrk
	DRAINED	INACTIVE	100		0	0	00	00000000	00000000000000000000	00000000	00000000
	UNAVAIL	INACTIVE	100		0	0	00	00000000	00000000000000000000	00000000	00000000
	JES3PART	ACTIVE	1	1815000	6524	00		00000000	00000000000000000000	00000000	00000000
	SPOOL1	ACTIVE	38	15000	5706	02		00000014	000000000000000012C	00000000	000003FB
	SPOOL2	ACTIVE	1	1800000	818	03		0000000F	0000000000000000E1	00000000	0001D4CE
											00000000001B7820

Figure 31-4 SDSF SP panel with EAS spool

### Checking available pool space

In Figure 31-5, we show the amount of available pool space using three similar JES3 commands. The command output shows that JES3 has two spool data sets in the default JES3 spool partition, JES3PART. Spool space is estimated to be 100% free.

```

*I Q S
IAT8530 1,815K GRPS, 1,809K LEFT (100%); 0 UNAVAIL, 0 DRAINED
*I Q DD=ALL
IAT8513 SPOOL1 JES3PART 15,000 GRPS, 9,294 LEFT ( 62%), STT
IAT8513 SPOOL2 JES3PART 1,800K GRPS, 1,799K LEFT (100%), STT
IAT8611 INQUIRY ON SPOOL DATA SET STATUS COMPLETE
*I Q SP=ALL
IAT8980 DRAINED HAS NO SPOOL DATA SETS
IAT8980 UNAVAIL HAS NO SPOOL DATA SETS
IAT8509 JES3PART: 1,815K GRPS, 1,809K LEFT (100%); MIN 5%, MRG 5%,
DEF, INIT
IAT8607 INQUIRY ON SPOOL PARTITION STATUS COMPLETE

```

Figure 31-5 Using JES3 commands to check available spool space

### 31.1.3 Allocating a checkpoint data set in EAS

During JES3 initialization, JES3 issued messages IAT6365 and IAT6368, indicating that the checkpoint data set is too small for the new, very large, spool data set. This is shown in Figure 31-3. You can choose to allow JES3 to complete initialization, but then return it to allocate a larger checkpoint data set.

Figure 31-6 shows the JCL to allocate a new JES3 checkpoint data set. We use the EATTR=OPT JCL parameter to indicate that the checkpoint data set is also EAS-eligible.

```

//ALLOCCK3 JOB ACCNT#,MSGLVL=(1,1),NOTIFY=&SYSUID
//ALLOCATE EXEC PGM=IEFBR14
//CKPTEAS DD DISP=(NEW,CATLG),DSN=SYS1.JES3CKP2,
// DCB=(DSORG=PS,RECFM=U,BLKSIZE=32760,LRECL=32760),
// SPACE=(CYL,15,,CONTIG),EATTR=OPT,UNIT=3390,VOL=SER=BH5JS4
//

```

Figure 31-6 Allocating a checkpoint data set in EAS



### 31.1.4 Adding the checkpoint data set to JES3

After allocating the EAS-eligible checkpoint data set, we update the JES3 procedure to allocate the new checkpoint using DD name CHKPNT2. Then we restart JES3. Figure 31-7 shows the messages issued during JES3 initialization with the new checkpoint. We start JES3 with a hot start.

```
IAT6363 JES3 CHECKPOINT I/O ERROR -  
          CHKPNT2,BH5JS4,DD65,CMAP,41,06,0D40,0000,001D4CF  
IAT6361 CHECKPOINT DATA SET <CHKPNT2> IS INVALID OR NEW, TRACK MAP INITIALIZED  
IAT3040 STATUS OF JES3 PROCESSORS IN JESXCF GROUP WTSCPLX4  
IAT3040 SC74 ( ), SC70 ( ), SC75 + +  
IAT3011 SPECIFY JES3 START TYPE  
040 IAT3011 (C, L, H, HA, HR, HAR, W, WA, WR, WAR, OR CANCEL)  
R 40,H  
IEE600I REPLY TO 040 IS;H
```

Figure 31-7 JES3 startup messages with the new EAS checkpoint data set

We then repeat the process, this time reallocating the first checkpoint data set, so both JES3 checkpoint data sets are now large enough for the new spool.

### 31.1.5 The JES3 JCT utility (IATUTJCT)

In order to allow a JCT data set to be copied without a cold start, JES3 provides a program called the JCT utility called IATUTJCT. IATUTJCT is updated in z/OS V1R12 to allow migrating from existing JCT and checkpoint data sets to EAS-eligible data sets.

### 31.1.6 BADTRACK enhancements

The BADTRACK initialization statement is used to identify defective tracks on a spool volume. The \*MODIFY,Q command can be used to add a BADTRACK element for a defective track in a spool data set. Both the initialization statement and command have been updated to allow 1-digit to 7-digit hexadecimal values for the cylinder specification.

#### **BADTRACK (bypass defective tracks)**

Use the BADTRACK statement to identify defective tracks on a spool volume. JES3 dynamically adds an entry to the BADTRACK table when a defective track is discovered and issues a message to the console operator that identifies the defective track. If possible, add a BADTRACK statement to your initialization stream at that time so that JES3 keeps a record of the defective track across a warm or cold start. If you cannot add a BADTRACK statement immediately, make sure that you add a BADTRACK statement before the next warm or cold start.

### 31.1.7 JES3 data sets on EAV

Other JES3 data sets, which are physical sequential data sets, may also reside anywhere on EAV volumes. The data sets include, but are not limited to:

- ▶ SYSABEND
- ▶ SYSUDUMP
- ▶ PROCLIBs

User data sets include:

- ▶ JES3IN
- ▶ JES3ABEND
- ▶ JES3DRDS
- ▶ JES3SNAP

Other JES3 data sets are accessed using standard DFSMS access methods, versus EXCP for spool, checkpoint, and JCT data sets. Based upon DFSMS support for the data set type, these data sets may also reside anywhere on an EAV volume.

### 31.1.8 Migration considerations for EAV support

Care must be taken when specifying EATTR=OPT for data sets that are not EAS-eligible in z/OS V1R10 or z/OS V1R11. Basic and large format sequential data sets will not be allowed to be opened from z/OS V1R10 or z/OS V1R11 if they were allocated in z/OS V1R12 with extended attributes.

#### IEC144I message

The JES3 global and all locals must be at z/OS V1R12 with z/OS V1R12 JES3. For earlier-levels of z/OS and JES3, JES3 initialization fails with system completion code 313 accompanied by system message IEC144I and return code 0C.

```
IEC144I 313-0C,IFG0194D,JES3,JES3,CHKPNT,0340,DJEAV ,SYS1.CMSCKPT1
```

**Explanation:** The error occurred during processing of an OPEN macro instruction for a data set on a direct access device.

**0C** - During an attempt to open a data set, open encountered an extended attribute DSCB (format-8) for a data set that is not eligible to have extents above 65,520 cylinders. This is invalid for this type of data set.

For z/OS V1R12, earlier-level JES3 initialization fails with system completion code 113 accompanied by system message IEC142I and return code 44.

Care must be taken when specifying EATTR=OPT for JES3 data sets that cannot be opened with z/OS V1R10 and z/OS V1R11 JES3 (even if z/OS V1R12 is IPLed). These JES3 releases cannot open data sets if they were allocated in z/OS V1R12 with extended attributes.

## 31.2 SAPI enhancements

In V1R12 JES3 has added a new SAPI disposition flag to request ENF58 signals for any selected SYSOUT data sets using the SAPI PUTGET (SSS2PUGE) option. This allows clients to monitor and process the progress of SYSOUT data sets throughout the system for the lifetime of the data sets.

#### ENF58 support

Each ENF58 signal includes a description of the data set's change in status. ENF58 broadcasting requires every listener to monitor for every signal they are interested in, and to take appropriate action based on the ENF58 subtype.

For information about how to use the ENFREQ macro to listen to these events, see *MVS Programming: Authorized Assembler Services Reference, Volume 2*.

JES3 has also added a new output field for the extended status verbose output section to display whether or not ENF58 signaling is enabled for a SYSOUT data set.

### **User applications**

A typical user of this capability would be an application that uses SAPI to gather SYSOUT data sets for processing at a later time. The application would maintain an inventory database of selected data sets' tokens returned by SAPI via SSS2DSTR as an output of a PUT/GET request.

The application would then set up an ENF58 listener using ENFREQ macro support. Because it receives an ENF58 event, it can then decide an appropriate action based on the event subtypes to update its inventory data base.

You can now monitor status changes for any SYSOUT data sets using the SAPI PUT/GET interface. Both the application's system and the JES3 global must be at z/OS V1R12 or higher.

If an application uses a data set token in order to select a single data set for the bulk modify request, then the request can receive a return code of SSS2CLON. This return code (in SSOBRETN) will be given if the single data set selected by token is part of an output group containing more than one data set and requiring homogeneous processing (flag SSS2DSH is on). The function is workable only if the provided token has been returned by SAPI (SSI 79) in a field pointed to by SSS2DSTR or returned by extended status (SSI 80) in field STSTCTKN.





## SMF enhancements

System management facilities (SMF) collects and records system- and job-related information that you can use for:

- ▶ Billing users.
- ▶ Reporting reliability.
- ▶ Analyzing the configuration.
- ▶ Scheduling jobs.
- ▶ Summarizing direct access volume activity.
- ▶ Evaluating data set activity.
- ▶ Profiling system resource use.
- ▶ Maintaining system security.

SMF formats the information that it gathers into system-related records (or job-related records). System-related SMF records include information about the configuration, paging activity, and workload. Job-related records include information about the processor time, SYSOUT activity, and data set activity of each job step, job, APPC/MVS transaction program, and TSO/E session.

In this chapter these SMF enhancements in z/OS V1R12 are described:

- ▶ SMF message flood automation.
- ▶ SMF flooding automation. Now it is possible to specify an SMF record flood policy, including actions to be taken when the arrival rate of records changes according to some behavior patterns. Flood policies can detect abnormal SMF behavior and allow for automated actions to be taken.
- ▶ NOBUFFS and BUFUSEWARN options can be used to manage SMF log stream buffers.

These functions will provide a more robust environment for SMF recording.

## 32.1 Message Flood Automation

Message Flood Automation was introduced in z/OS V1R9 and is designed to detect and react to a message flood before those consequences have had an opportunity to occur. A defined installation policy allows installations to tailor Message Flood Automation to an individual environment by deciding how quickly the Message Flood Automation should react to a potential message flood and then, what actions should be taken if a message flood occurs. Depending on the policy that is established, Message Flood Automation can prevent message buffers from filling, console queues from becoming overly long, and console displays from becoming unreadable.

### z/OS V1R11 Message Flood Automation improvement

With z/OS V1R11, Message Flood Automation processing is now done after the MPFLSTxx process, which allows message processing to not overlook messages and cause Message Flood Automation to be removed from the IEAVMXIT exit, as shown in Figure 32-1. Since the processing is now done before any specific MPF exit processing, this allows an MPF exit to override the changes made by Message Flood Automation.

**Note:** Message Flood Automation message processing runs before any MPF exit processing. Therefore, MPF exits can override the RETAIN®, AUTO, and SUP specifications set by Message Flood Automation.

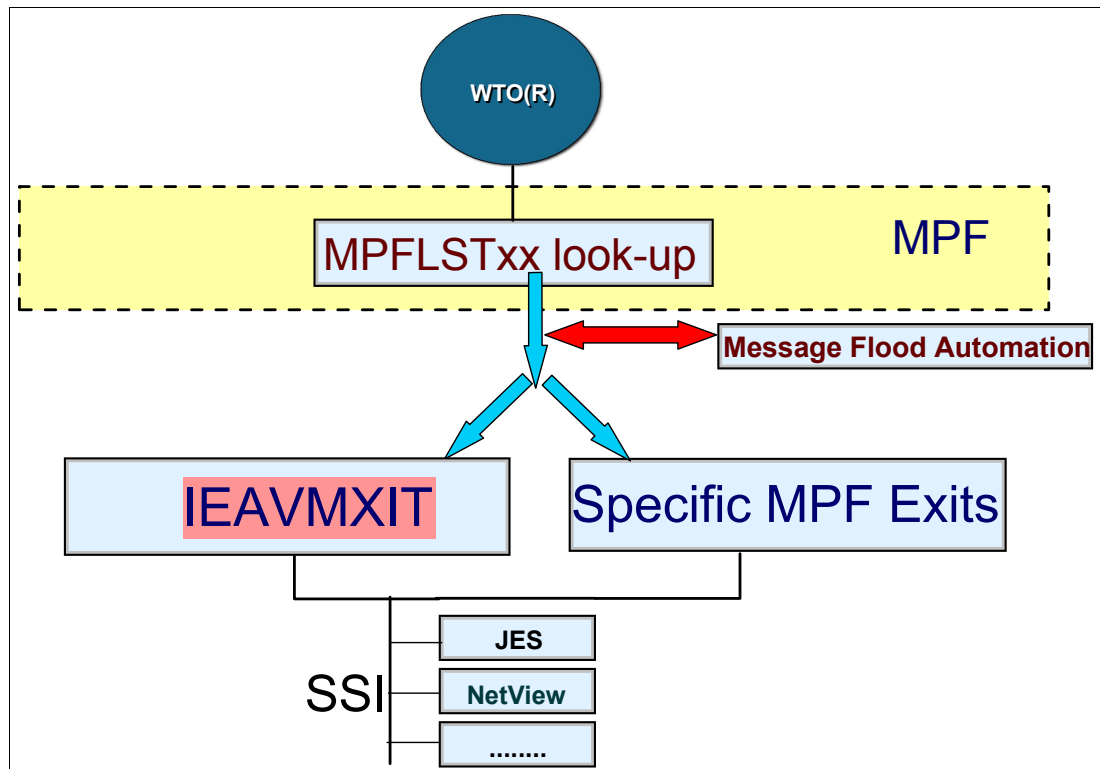


Figure 32-1 Message processing change in z/OS V1R11

## 32.2 SMF records to log stream in z/OS V1R9

z/OS V1R9 also introduced an additional capability to write SMF records to log streams managed by System Logger. With this capability you can define several log streams for several groups of SMF records. You can define one log stream to write just the RMF record types, so that during the post processor they are already isolated. When you define one log stream you have to define the SMF record types that will be written to this log stream. You can define a default log stream to write all the remaining SMF record types not defined to a specific log stream.

When recording to log streams, as shown in Figure 32-2, subsystems or user programs still invoke the SMFEWTM macro to take the user record and invoke SMF code. However, instead of locating a buffer in SMF private storage, SMF locates a dataspace corresponding to the user's record type and log stream where the record will be written. A buffer with space to hold the record is located and the record is copied there. When the record is full, a writer task is posted.

Unlike the scheduled approach in SMF data set recording, this task is already started and ready to write. In addition, writes to System Logger are done at memory-to-memory speeds, with System Logger accumulating many, many records to write out, resulting in improved access not possible with current SMF data set recording.

Using a dataspace to hold the records for a given log stream allows a full 2 GB of pageable memory to be used for buffering records in the event of delays in log stream writing in System Logger. This allows more data to be buffered than SMF data set recording, which is limited to the amount of available private storage in the SMF address space.

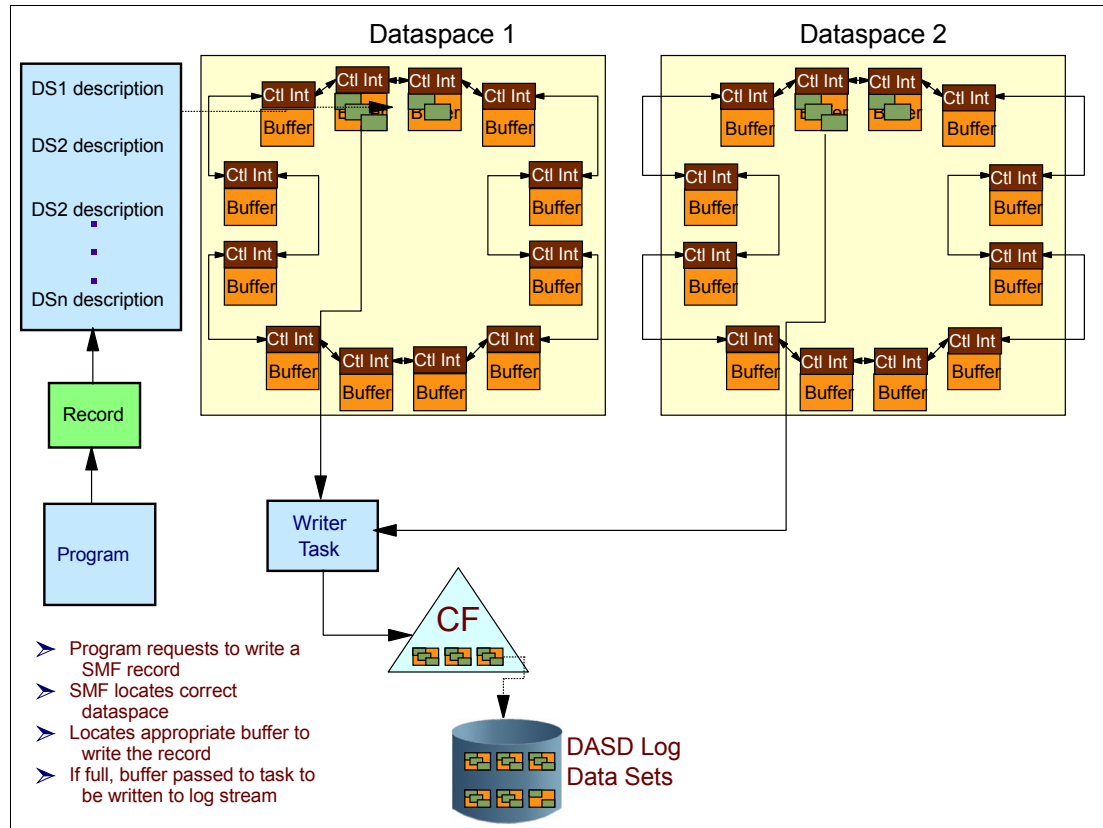


Figure 32-2 SMF data flow using log streams

The benefit in all this is that you can write more data faster, with more functionality. The System Logger was created to handle large volumes of data. With minimal SMF switch processing and no record numbering schemes to maintain, this eliminates the switch SMF command bottleneck.

**Note:** The use of log streams for SMF data is optional. The existing SYS1.MANx function continues to exist for installations satisfied with this functionality.

### Processing of SMF records

There are two SMF dump programs to process SMF records:

- ▶ Dumping SMF log streams - IFASMF DL
- ▶ Dumping the SMF data sets - IFASMF DP

## 32.3 z/OS V1R12 enhancements for flood of SMF records

When it became possible to record SMF records in log streams using System Logger, it also became more difficult to prevent system outages due to an unexpected increase in SMF record arrival rates. When using SMF data set recording, it was more obvious that something unexpected could happen, because the SYS1.MANxx data sets filled up more quickly. Indicators from the Logger subsystem may happen too late to allow time for correcting the issue. With z/OS V1R12, you have the ability to set up rules for matching a flood of SMF data records and either issue a warning message or begin dropping that record type.

### SMF flooding statistics

The IFASMF DP and IFASMF DL dump programs have been updated to allow the gathering of SMF flooding statistics. You can use the FLDSTATS options in these SMF log stream dump programs to display flood-related statistics. The SMF data set dump program creates an SMF flood statistics report to display statistics for each record type that matched the specified record type filters. This report is generated using the new FLDSTATS(xxxx) keyword and prints out a table with statistics related to record flooding. Each line of the report is for a given record type. Statistics are gathered based on the xxxx value. This value indicates the number of records that make a single interval for the statistical calculations.

### 32.3.1 Using the log stream dump program

Figure 32-3 shows an example to get statistics information for type 90 and 101 records with the FLDSTATS parameter. This job gathers individual statistics for type 90 and 101 and also for the combination of the two. Each interval consists of 1000 records.

```
//RUNSMFDL JOB MSGLEVEL=(1,1),NOTIFY=&SYSUID
//SMFDMP EXEC PGM=IFASMF DL
//DUMP04 DD DUMMY
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
LSNAME(IFASMF.SOME.LOGSTREAM,OPTIONS(DUMP))
OUTDD(DUMP04,TYPE(90,101))
FLDSTATS(1000)
```

Figure 32-3 Sample JCL for dumping an SMF data set with the FLDSTATS option



## Return codes

The possible return codes from the dump utility of the SMF log stream dump program are:

- 00** The dump was successful; no errors were encountered.
- 04** The dump was successful; one or more errors were detected but processing continued.
- 08** The dump was not successful; an error terminated processing.

## GRS serialization

Serialization of the SMF log stream dump program IFASMF DL obtains an enqueue for each log stream that is being processed. The major name is SYSZSMFL; the minor name is the name of the log stream. Table 32-1 describes the mode in which the enqueue is obtained, depending on the type of request and the type of log stream.

Table 32-1 Mode in which the enqueue is obtained

Log stream type	DELETE	ARCHIVE	DUMP
DASD only	System exclusive	System exclusive	System SHARED
CF based	System exclusive	System exclusive	System SHARED

During initialization, IFASMF DL attempts to obtain all the required enqueues. If any enqueues cannot be obtained, the obtained enqueues up to that point are released and message IFA838I is issued. IFASMF DL retries every 30 seconds to obtain the enqueues and reissues message IFA838I every 5 minutes when it is waiting.

Because IFASMF DL is not waiting on the enqueues, the contention is not shown with a D GRS,C command. Also because IFASMF DL is releasing and attempting to reobtain the enqueues, the order of execution of two IFASMF DL jobs may change from the order in which they were submitted.

## Sample flood statistics report

The SMF data set dump program creates a flood statistics report to display statistics for each record type that matched the specified filters. This report is generated using the FLDSTATS(XXXX) keyword. Table 32-2 on page 644 shows the fields that appear on an SMF flood statistics report, for record types 4 and 5 passed through filtering. The report is generated, separately, for each z/OS image processed and includes statistics for each record type encountered. You can specify an SMF flood policy based on the information shown in the flood statistics report.

Table 32-2 on page 644 shows an example of the flood statistics report the SMF data set dump program creates. The meaning of each heading is as follows:

- START DATE-TIME** Indicates the date and time of the earliest record read, excluding record types 2, 3, and those greater than 127. Under most circumstances this date and time will not equal the date and time of the first record written.
- END DATE-TIME** Indicates the date and time of the latest record read, excluding record types 2, 3, and those greater than 127. Under most circumstances this date and time will not equal the date and time of the last record written.
- RECORD TYPE** Indicates the record type for this record.
- TOTAL INTERVALS** Indicates the number of complete intervals for all record types read as specified by the FLDSTATS option. This value may not equal the sum of each individual record type's number of intervals.

- LOW INTERVAL** Indicates the lowest time it took for a complete interval. This is the interval with the fastest velocity, in tenths of a second.
- HIGH INTERVAL** Indicates the highest time it took for a complete interval. This is the interval with the slowest velocity, in tenths of a second.
- AVERAGE INTERVAL** Indicates the average of all the intervals. This value is in tenths of a second.
- STANDARD DEVIATION** Indicates the standard deviation for the set of complete intervals, in tenths of a second.

This text appears at the top of Table 32-2:

**START DATE-TIME 05/28/2009-10:45:57**

**END DATE-TIME 05/28/2009-11:16:47**

*Table 32-2 Sample SMF flood statistics report*

RECORD TYPE	TOTAL INTERVALS	LOW INTERVAL	HIGH INTERVAL	AVERAGE INTERVAL	STANDARD DEVIATION
4	11	50	5,286	642	1,404
5	1350 <sup>TM</sup>	50	5,236	546	1,308
6	14	50	5,176	506	1,263
7	15	40	5,116	383	1,226
8	17	40	5,055	340	1,146
9	19	40	5,015	307	1,082
10	2	40	5,452	2,746	2,706

## 32.4 Specifying SMF record flood options

You can use the FLOOD and FLOODPOL parameters to specify SMF record flood options using one of the following:

- ▶ Issue the SETSMF FLOOD(ONIOFF) command. No policy changes are performed as a result of this command.
- ▶ Update an SMFPRMxx parmlib member with FLOOD(ONIOFF) to enable or disable the SMF record flood automation facility. Turning the facility on and off resets all counts and any active flood situations.

### 32.4.1 FLOOD parameter

The syntax of the FLOOD parameter is:

FLOOD(ON|OFF)

This specifies whether SMF record flood support is active. The default value, when this parameter is omitted, is OFF.

## 32.4.2 SMF flood policy definition - FLOODPOL

The SMFPRMxx parmlib member FLOODPOL parameter defines an SMF flood policy. The syntax is FLOODPOL(ffff), and this option must be set to ON to become activated. Each of the options below must be present in the filter (ffff).

**Note:** FLOODPOL cannot be specified on the SETSMF command; it must be specified in the SMFPRMxx parmlib member.

The FLOODPOL parameter specifies a flood policy filter (ffff). The FLOOD SMFPRMxx parmlib member option must be set to ON for this option to become activated. Each of the following options must be present in the filter (ffff):

```
FLOODPOL (
  {TYPE({aa,bb})
  {aa,bb:zz
  {aa,bb:zz,...} } },
  RECTHRESH(xxxx),
  INTVLTIME(xxxx),
  MAXHIGHINTS(xxxx),
  ENDINTVL(xxxx),
  ACTION({MSG|DROP})
)
MAXEVENTINTRECS(nn)
```

Where:

- |                                 |  |
|---------------------------------|--|
| <b>TYPE({aa,bb} {aa,bb:zz})</b> | Specifies the records for this filter.<br>Value range: 0-255 (SMF record types)<br>Default: None   |
| <b>RECTHRESH(xxxx)</b>          | Specifies the number of records in an interval for this filter.<br>Value range: 1-9999<br>Default: None  |
| <b>INTVLTIME(ssss)</b>          | Specifies a flood interval time value, given in tenths of seconds.<br>Value range: 1-9999<br>Default: None   |
| <b>MAXHIGHINTS(xxxx)</b>        | Specifies the number of intervals, defined by <b>INTVLTIME</b> , that must occur at or above the flood rate before action is taken.<br>Value range: 1-9999<br>Default: None                              |
| <b>ENDINTVL(ssss)</b>           | Specifies the amount of time, in tenths of a second, that must pass before the <b>RECTHRESH</b> records are generated to determine that a flood state has ended.<br>Value range: 1-9999<br>Default: None |
| <b>ACTION({MSG DROP})</b>       | Specifies the action to be taken when a flood state is entered. MSG - Issue warning message IFA780A at the start of the flooding state. Message IFA781I is issued when the flood has stopped.            |

DROP - Issue message IFA782A at the start of the flood state and also begin dropping records. At the end of the flood state, message IFA783I is issued that indicates the number of records that have been dropped.

Default: None

When the MSG filter is specified, message IFA780A is issued at the start of the flood to warn you. Message IFA781I is issued when the flooding has stopped. When the DROP filter is specified, message IFA782A is issued at the start of the flood and dropping of records begins. Any attempts to write a record through the SMFEWTM or SMFWTM macro will get a return code= 52. At the end of the flooding, message IFA783I is issued with the number of records dropped.

### New flood messages

```
IFA780A SMF RECORD FLOOD MSG FILTER FOR TYPE xx EXCEEDED AT TIME=hh.mm.ss  
IFA781I SMF RECORD FLOOD MSG FILTER FOR TYPE xx RETURNED TO NORMAL AT  
TIME=hh.mm.ss
```

```
IFA782A SMF RECORD FLOOD DROP FILTER FOR TYPE xx EXCEEDED AT TIME=hh.mm.ss
```

**Explanation:** This message is issued when a flood is triggered and the action for the flood filter is DROP processing.

```
IFA783I SMF RECORD FLOOD DROP FILTER FOR TYPE xx RETURNED TO NORMAL AT  
TIME=hh.mm.ss, RECORDS DROPPED=yyyyyyyy
```

**Explanation:** This message is issued when a DROP filter has detected the end of a flood.

### Example of a message flood policy

In the example shown in Figure 32-4 on page 647, two filters are set up:

- ▶ The first filter sets up a monitor for both record types 4 and 5. It will detect when 1000 records have been generated within 5 seconds, and if records continue to be generated at this rate for 15 consecutive 1-second intervals, message IFA780A is issued. The flood state ends when it takes more than 12 seconds to generate 1000 records. Message IFA781I is issued when the flood state ends.
- ▶ The second filter for record type 102 is a two-part filter. The first part that issues the warning message is triggered when 5000 records are generated in less than 1 second, and records continue to be generated at that rate for more than 15 consecutive 1-second intervals. The flood state ends when it takes more than 10 seconds to generate 5000 records. Message IFA781I is issued when the flood ends.
- ▶ After the message is issued, and if the flood state persists, the DROP filter becomes active and begins dropping records if 5000 type 102 records are generated within 1 second over 15 consecutive times. Records will stop being dropped once it takes more than 5 seconds to generate 5000 type 102 records.

```

FLOOD(ON)
FLOODPOL(TYPE(4,5),RECTHRESH(1000),INTVLTIME(50),MAXHIGHINTS(15),ENDINTVL(120),
ACTION(MSG))
FLOODPOL(TYPE(102),RECTHRESH(5000),INTVLTIME(10),MAXHIGHINTS(15),ENDINTVL(100),
ACTION(MSG))
FLOODPOL(TYPE(102),RECTHRESH(5000),INTVLTIME(10),MAXHIGHINTS(15),ENDINTVL(50),A
CTION(DROP))

```

Figure 32-4 SMF flood policy

### SMF record type 7

SMF record type 7 was updated to indicate when the dropping condition starts and how many records are lost due this condition. Bit SMF7DRP indicates that this record was created by SMF flooding automation.

### Coexistence with earlier versions of z/OS

If you need to share SMFPRMxx parmlib members between z/OS V1R12 and z/OS V1R10 or V1R11 systems, you must apply APAR OA30848. This APAR ignores the new SMFPRMxx parmlib member options.

## 32.5 SMF log stream buffer management

With z/OS V1R12, a new feature allows management of log stream buffers, in the same way that it was possible with SMF data sets. Now you can use the NOBUFFS and BUFUSEWARN options for managing SMF log stream buffers. BUFSIZMAX is not supported for log streams, a 2 GB buffer is always used. Each log stream can be configured with its own BUFUSEWARN and NOBUFFS policies, allowing for log streams with important data to be managed in different ways. You can use the BUFUSEWARN and NOBUFFS parameters to specify SMF buffering options.

**Note:** BUFUSEWARN and NOBUFFS apply to SMF log streams and data sets; BUFSIZMAX does not apply to log streams, because for log streams the buffer size is fixed at 2 GB.

You can specify NOBUFFS and BUFUSEWARN on the LSNAME and DEFAULTLSNAME keywords. The global NOBUFFS and BUFUSEWARN are applied to any log stream that does not have any other options specified. Each log stream can now be targeted with a unique buffer management setup. Using the options on the LSNAME or DEFAULTLSNAME keyword will override the global values.

### SMFPRMxx parmlib member

You can specify NOBUFFS and BUFUSEWARN on the LSNAME and DEFAULTLSNAME keywords. The global NOBUFFS and BUFUSEWARN are applied to any log stream that does not have any other options specified. BUFSIZMAX is not supported for log streams, a 2 GB buffer is always used.

```
NOBUFFS( {HALT} |{MSG})
```

#### **NOBUFFS**

Specifies the system action when the log stream specified on this LSNAME parameter buffer area is full. This option can also be specified as a global option setting. It overrides the globally specified NOBUFFS option for the log stream specified on this

LSNAME parameter only. For a detailed description of this parameter, see the parameter option description for the global specification of NOBUFFS.

MSG - Specifies that the system is to issue a message and continue processing; SMF data is lost until buffer storage is again available.

HALT - Specifies that the system is to enter a restartable wait state. HALT means that no SMF data is lost.

**BUFUSEWARN( {nn})** BUFUSEWARN specifies the overall buffer warning level percentage (nn) when SMF starts to issue warning message IFA785E for the log stream specified on this LSNAME parameter. This option can also be specified as a global option setting that applies to all log streams defined by the LSNAME and DEFAULTLSNAME keywords. This option overrides the globally specified BUFUSEWARN option for the log stream specified on this LSNAME parameter only. For a detailed description of this parameter, see the parameter option description for the global specification of BUFUSEWARN.

```
DEFAULTLSNAME(logstreamname,NOBUFFS(MSG|HALT), BUFUSEWARN(nn))
LSNAME(logstreamname,TYPE({aa,bb}|{aa,bb:zz}|{aa,bb:zz,...})
,NOBUFFS(MSG|HALT), BUFUSEWARN(nn))
```

Figure 32-5 BUFUSEWARN and NOBUFFS options when defining log streams

## System actions to prevent data loss

You can use the combination of the BUFSIZMAX and BUFUSEWARN parameters to prevent SMF data loss conditions. After setting the SMF buffer options, if you still cannot prevent data loss conditions, use the SMF NOBUFFS parameter to specify what the system is to do in this situation. The following are the possible system actions:

- ▶ Continue processing with the loss of SMF data and if parmlib option NOBUFFS(MSG) is in effect, message IEE979W, shown in Figure 32-6 on page 649, is issued. All records presented to SMF are lost until buffers become available. When no buffers are available for a given SMF log stream and the NOBUFFS(MSG) action is specified, message IFA786W, shown in Figure 32-6 on page 649, is issued. SMF records have been lost and more might be lost until SMF can write the buffered records to the SMF log stream.
- ▶ Enter a restartable wait state. Note: When the BUFSIZMAX parameter is specified, if the amount of storage requested is not available during initialization, SMF takes the specified value, decreases it by 20%, and attempts to get that amount. If unsuccessful, SMF retries this four more times. Message IFA723E, shown in Figure 32-6 on page 649, is issued with the amount of storage that SMF was finally able to obtain. When the BUFSIZMAX parameter is specified via a SETSMF or SET SMF=xx command, if the new larger area cannot be obtained, the current value remains in effect. Message IFA723E is issued to the console where the SETSMF or SET SMF=xx command was entered.

```

IFA785E SMF HAS USED nn% OF AVAILABLE BUFFER SPACE FOR LOGSTREAM lsname
IFA786W SMF DATA LOST - NO BUFFER SPACE AVAILABLE TIME=hh.mm.ss FOR LOGSTREAM
lsname
IFA787E WAIT STATE 'D0D-02'X - NO SMF BUFFERS FOR lsname
IFA723E REQUESTED SMF BUFSIZE MAX STORAGE AMOUNT xxxxx COULD NOT BE ALLOCATED,
USING yyyyy OF STORAGE FOR SMF BUFFERS
IEE979W SMF DATA LOST - NO BUFFER SPACE AVAILABLE TIME=hh.mm.ss

```

**Notes:**

- ▶ Message IFA786W takes the place of existing message IFA713I and is equivalent to IEE979W for data set recording.
- ▶ Message IFA787E is equivalent to IEE987E for data set recording.

Figure 32-6 Message IEE979W and new messages IFA723E, IFA785E, IFA786W, and IFA787E

**Note:** If present, the global NOBUFFS and BUFUSEWARN values are now honored for SMF log stream recording and applied to all log stream buffers.

### SMF record type 7

SMF record type 7 was updated to indicate when the dropping condition starts and how many records are lost, due to this condition. Bit SMF7LSD indicates that this record is the result of a log stream becoming full.

### Coexistence with earlier versions of z/OS

If you need to share SMFPRMxx parmlib members between z/OS V1R12 and z/OS V1R10 or V1R11 systems, you must apply APAR OA30848. This APAR ignores the new SMFPRMxx parmlib member options.

## 32.6 SMF processor capacity data

Following are the changes made to SMF in z/OS V1R12 to reflect processor capacity changes during a measurement interval, due to CBU or OOCUoD invocation. In previous releases, the SMF type 30 and type 89 interval records did not contain current processor capacity data, so changes in processor capacity were not recorded until the end of the regularly scheduled measurement interval. This could lead to some imprecision in capacity accounting. Now, SMF type 30 and type 89 records contain current processor capacity data.

### Processor capacity in SMF records 30 and 89

An expansion of SMF record types 30 and 89 includes processor capacity data and can be used for accounting purposes. When the processor capacity changes, during a measurement interval, additional event-driven interval records are created, to reflect this change.

The new fields of these SMF records are shown in Table 32-3 on page 650. Note that the same fields appear on both 30 and 89 records, changing only the labels, from 30 to 89.

Table 32-3 New fields in records 30 and 89 contain processor capacity data

Field	Contents
SMF30(89)_Capacity_Change_Cnt	The number of processor capacity changes that occurred since the previous interval or event interval. This number is greater than 1 when the number of processor capacity changes exceeded the number specified in the MAXEVENTINTRECS parmlib option.
SMF30(89)_RCTPCPUA_Actual	Physical processor adjustment factor (this is the adjustment factor for converting processor time to equivalent service, in basic mode with all processors online). Based on model capacity rating.
SMF30(89)_RCTPCPUA_Nominal	Physical processor adjustment factor (this is the adjustment factor for converting processor time to equivalent service in basic-mode with all processors online). Based on nominal model capacity rating.
SMF30(89)_RCTPCPUA_scaling_factor	Scaling factor for SMF30(89)_RCTPCPUA_actual and SMF30(89)_RCTPCPUA_nominal.
SMF30(89)_Capacity_Adjustment_Ind	When: 0 - The indication is not reported. 1-99 - Some amount of reduction is indicated. 100 - The machine is operating in normal capacity. Primary processors and all secondary-type processors are similarly affected.
SMF30(89)_Capacity_Change_Rsn	Indicates the reason that is associated with the present value contained in SMF30(89)_Capacity_Adjustment_Ind. The bit values of this field correspond to those described in RMCTZ_Capacity_Adjustment_Indication of the IRARMCTZ mapping macro. (See <i>MVS Data Areas</i> )
SMF30(89)_Capacity_Flags	Bit meaning when set: <b>0 SMF30(89)_Event_Driven_Intvl_Rec</b> Meaning: When on, indicates that the current interval record was generated as a result of an event, rather than as a result of standard interval expiration based on time. <b>1 SMF30(89)_RQSVSUS_Err</b> Meaning: When on, indicates that an error occurred while collecting the data for SMF30SUS following a change in processor capacity. If this bit is found to be on when the record is being written, an additional attempt to collect the data from SRM is made. If that attempt is successful, the data is filled in at that time and the SMF30PIN error bit will be off. <b>2 SMF30(89)_Capacity_Data_err</b> Meaning: When on, indicates that an error occurred while collecting the processor capacity data; therefore, the following fields are unreliable: SMF30(89)_RCTPCPUA_Actual SMF30(89)_RCTPCPUA_Nominal SMF30(89)_RCTPCPUA_scaling_factor SMF30(89)_Capacity_Adjustment_Ind <b>3 SMF30(89)_PCD_Rsvd_Exists</b> When on, indicates running on z/OS V1R10 or later.

### SMF record type 90 - subtype 34

A subtype 34 record is generated when a processor capacity change occurs, triggered by an ENF41 signal, qualified by WLMENF12 along with a change in processor capacity. This record has a Processor Capacity Change section. For a description of the fields present in this subtype, see *z/OS MVS System Management Facilities (SMF)*, SA22-7630.



## Event-driven interval recording activation

To activate this function, set the MAXEVENTINTRECS option in SMFPRMxx. The syntax is:

```
MAXEVENTINTRECS(nn)
```

This option is used to indicate the maximum number of event-driven SMF type 30 and type 89 interval records that are allowed during a regular interval cycle. Extra SMF Type 30 and Type 89 interval records can be generated when a processor capacity change occurs. The additional interval records are generated such that, when a processor changes capacity, the current interval is expired and a new, event-driven interval begins. The event-driven interval expires at the regularly scheduled end of the current interval, or when the processor changes capacity again, whichever occurs first.

**Value Range:** 0-60

**Default:** 0

For detailed information, see *z/OS MVS Initialization and Tuning Reference*, SA22-7592.

## IFAURP enhancements

IFAURP was enhanced in z/OS V1R12 to calculate MSUs using the processor capacity data found in the SMF type 89 record, when the data is present (fallback to static values for pre-SMF-support records). Event-driven interval data is aggregated into the hourly calculations accordingly. Several IFAURP reports were restructured, such that processor data is represented in terms of type and serial number, rather than type, model, and serial number. Processors with multiple model numbers are shown as part of the same processor.

## Processor Capacity Audit report

In z/OS V1R12, IFAURP generates a new report, Processor Capacity Audit report, which displays a new line for each change that occurred in processor capacity (type, serial, model, or capacity data).

To generate this report, include PARM='AUDIT' in the EXEC JCL statement for IFAURP, and a DD statement for file PCAUDIT must be included, as shown in Figure 32-7. For a detailed description of these parameters, and SORT requirements for the SMF input data set, refer to *z/OS MVS Product Management*, SA22-7603.

```
//STEPRPT EXEC PGM=IFAURP,PARM='PCAUDIT'  
//STEPLIB DD DSN=SYS1.SIFALIB,DISP=SHR  
//SMFDATA DD DSN=&&SORT89,DISP=(OLD,PASS)  
//SYSUDUMP DD DUMMY  
//SYSMSG DD SYSOUT=*  
//PCAUDIT DD SYSOUT=*  
//SYSPRINT DD SYSOUT=*  
//SYSUSAGE DD SYSOUT=*  
//SYSHIN DD DUMMY,DCB=BLKSIZE=4096  
//SYSHOUT DD DUMMY  
//SYSIN DD *  
CUSTOMER(NAME(*TEST SYSTEM*))  
/*
```

Figure 32-7 Sample JCL to generate the Processor Capacity Audit Report

**Note:** Users who attempt to share the SMFPRMxx parmlib member, with MAXEVENTINTRECS other than zero in effect, with earlier versions of z/OS, should review APARs OA30563, OA31279, and OA40968.

## 32.7 Initiator processor time

z/OS V1R12 now accounts for the processor time used by the initiator, on a step basis, both in TCB and SRB modes, while executing job steps; see Table 32-4. In earlier versions of z/OS, the timing fields were cleared at job termination, prior to merging SMF type 30 step hold data into the SMF type 30 job end data. The time spent in this effort, as well as the time spent during step initialization, was reflected in the SMF30ICU and SMF30ISB fields for the subsequent step.

Table 32-4 SMF record type 30 initiator processor time

Field	Content
SMF30ICU	Initiator processor time under the task control block (TCB), in hundredths of a second. This field is set at step termination. SMF30ICU = SMF30ICU_STEP_INIT (for this step) + SMF30ICU_STEP_TERM (from the previous step).
SMF30ISB	Initiator processor time under the service request block (SRB), in hundredths of a second. This field is set at step termination. SMF30ISB = SMF30ISB_STEP_INIT (for this step) + SMF30ISB_STEP_TERM (from the previous step)

For additional information about this function, see *z/OS MVS System Management Facilities (SMF)*, SA22-7630.

## 32.8 SMF dump program exits

In z/OS V1R12, you can enable SMF dump programs to call user exits. To do this, define the exits to the system using the following keywords:

- ▶ SMFDLEXIT allows you to specify exits for the IFASMF DL program.
- ▶ SMFDPEXIT allows you to specify exits for the IFASMF DP program.

Both keywords have options USER1, USER2, and USER3, which allow you to specify multiple exits for each user exit point in the respective dump program. After you specify that an exit is using USER1, USER2, or USER3, it remains active until you redefine it in the SMFPRMxx parmlib member or use the SETSMF command (not to be confused with the SET SMF command).

For example, to have user exit SAMPEXIT available as a USER2 exit for IFASMF DP, you code the following in the active SMFPRMxx member:

```
SMFDPEXIT(USER2(SAMPEXIT))
```

If you enter a SETSMF command with the following parameters, it affects only USER1:

```
SMFDPEXIT(USER1(SAMPEXIT2))
```

The USER2 you previously specified remains in effect.

For additional details, see *z/OS MVS Initialization and Tuning Reference*, SA22-7592 and *z/OS MVS System Management Facilities (SMF)*, SA22-7630.





## 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 that update data, for example, need exclusive access to that data; if several users tried to update the same data at the same time, the result would be a data integrity exposure, that is, 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 system 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 in z/OS V1R12:

- ▶ GRSQ SDUMP performance
- ▶ Unauthorized ISGENQ using ECB
- ▶ GRS CTRACE improvements

## 33.1 GRSQ SDUMP performance

Dump times when the SDATA option GRSQ is specified take an excessive amount of time to gather the GRS-related data. This is due to unnecessary interface overhead and paged-out GRS-private storage that was often irrelevant.

A previous related enhancement added the CONTENTION keyword for GRSQ:

```
GRSQ(ALL|LOCAL|CONTENTION) in GRSCNFxx
```

The GRS health check recommends the CONTENTION default. This reduces the amount of data gathered by internal QScan for Star mode and significantly shortens GRSQ SDump times. In previous releases the ISGENQ service did not allow unauthorized requesters to specify WaitType=ECB. With z/OS V1R12, the ISGENQ service's functionality is more general purpose.

**Note:** IBM suggests using the ISGENQ service over ENQ/DEQ/RESERVE.

### GRS hot queues

With increasing system images there is a corresponding increase in ENQ activity. The latest problem in GRSQ SDump times deals primarily with paged-out blocks of GRS private storage that are being captured at dump time. "Hot queues" were introduced in z/OS V1R6 as a performance enhancement to ENQ processing, but the queues were never compressed. If ENQ activity spiked, unused blocks would be paged out and would subsequently adversely affect dump times. This was particularly true for large DB2 images with its management of tens of thousands of data sets.

### 33.1.1 GRS contention notification with z/OS V1R12

Resource contention can cause poor system performance, and resource contention that lasts over time can result in program starvation or deadlock conditions. Global resource serialization provides APIs, modified in z/OS V1R12, to query for ENQ contention information:

- ▶ ISGECA - Obtains waiter and blocker information about SYSTEM and SYSTEMS scope ENQ resources.
- ▶ ISGQUERY - Obtains the status of resources and the requesters of STEP, SYSTEM, and SYSTEMS scope ENQ resources.

Global resource serialization also issues an event notification facility (ENF) signal 51 to notify monitoring programs to track contention for SYSTEM and SYSTEMS scope ENQ resources. This is useful in determining contention bottlenecks, preventing bottlenecks, and potentially automating correction of these conditions.

### Ring mode

In ring mode, each system in the global resource serialization complex knows about the complex-wide SYSTEMS scope ENQs that allow each system to issue the appropriate ENF 51 contention notification event. However, in star mode, each system only knows about the ENQs that are issued by the system. To ensure that the ENF 51 notification for SYSTEMS scope ENQs are issued on all systems in proper sequential order, one system in the sysplex is appointed as the sysplex-wide SYSTEMS contention notification system (CNS). All SYSTEMS scope ENQ contention events are sent to the notifying system, which then issues a sysplex-wide ENF 51. During IPL or if the notification system can no longer perform the

duties, any system in the sysplex can act as the contention notification system. You can determine the current contention notification system with the DISPLAY GRS command.

GRS is now an ENF listener as well as a signaler. GRS issues ENF 51 signals for events such as RNL suspension and ENQ contention. But as a listener of ENF 55 events, GRS is able to react to shortages in REAL or AUX storage.

Message ISG376, shown in Figure 33-1, is issued by GRS when there is a spike of ENQ activity and compression alleviates the time required for a subsequent SDump.

```
ISG376I GLOBAL RESOURCE SERIALIZATION FREEING STORAGE BUFFERS DUE TO reason
DIAG= diag1

reason is one of the following:

REAL STORAGE SHORTAGE
The system is constrained on real storage as indicated by the ENF 55 signal.

AUX STORAGE SHORTAGE
The system is constrained on auxiliary storage as indicated by the ENF 55
signal.

BUFFER THRESHOLD EXCEEDED
The system exceeded one or more thresholds associated with the internally
managed storage buffers as determined by GRS.
```

Figure 33-1 Message ISG376I

**Note:** GRS ENQ performance may be temporarily affected by the compression of the buffers, but GRS will automatically replenish those buffers over time.

### Software support for prior releases

This new function is provided for prior z/OS releases via OA29329, which closed in November 2009. The PTFs are available for z/OS V1R9 to z/OS V1R11—UA51183, UA51184, and UA51185, respectively.

This provides the “Hot Queue” compression support for buffer threshold exceeded conditions but does not include the RSM and SDUMP interface updates.

### Installation considerations for z/OS V1R12

This is a usability enhancement for ISGENQ users. There can be environments where the requester does not want to be suspended by GRS at that point in the code, or the requester may want to wait on a list of ECBs.

The syntax of the ISGENQ service is unchanged, as follows:

```
CONTENTIONACT=WAIT, WAITTYPE=ECB, ECB@=ecb@
```

**Note:** With z/OS V1R12, the ECB must be in one of the following locations:

- ▶ The home address space of the caller.
- ▶ Must be common space.
- ▶ Must be for unauthorized requesters, in the same storage key as the requester.

The reason code ISGENQRsn\_NotAuthorizedForECB, xxxx0820, is no longer used.

### **OwningTToken keyword**

To use OwningTToken, ENQMAX, or when the specified QNAME is one of the authorized QNAMEs, authorization must be one of the following: Supervisor state, PSW key 0-7, or APF authorized.

The OwningTToken keyword is still restricted to authorized requesters because unauthorized requesters cannot alter another task's resource ownership.

**Note:** The ENQ service is unchanged. Unauthorized requesters cannot specify ECB= on the ENQ service.

## **33.2 GRS CTRACE improvements**

Prior to z/OS V1R12, GRS did not provide component tracing for its latch processing and the GRS CTRACE maximum buffer size was limited.

In z/OS V1R12, GRS provides component tracing for latches. The GRS CTRACE buffer, which is used for both ENQs and latches, has moved above-the-bar and IOCS ANALYZE now includes new time stamps and latch IDs. This imposes GRS serviceability significantly, especially for latch services.

### **Installation considerations**

It is a big diagnostic improvement to now provide CTRACE support for latches. FLOWA is recommended but optional.

It is also significant to move GRS' buffers above-the-bar, providing much more room for data before it is forced to wrap. You can change options while a SYSGRS trace is running. However, to change the buffer size, you have to stop the trace and restart it with the new buffer size.

The values for the OPTIONS parameter for the CTnGRSxx parmlib member and reply for a TRACE command are listed below. The suboptions on the CONTROL, REQUEST, MONITOR, SIGNAL, and FLOW, allow you to refine the set of events traced for the major option. When you select the major option, all events pertaining to that option are traced. However, you can select one or more of the suboptions instead of the major option and thus limit the trace to only those events included in the suboptions specified. A major option, such as MONITOR, and all of its suboptions (in this case MONITOR0, MONITOR1, and MONITOR2 through MONITORF) is referred to as an option group. A new option in z/OS V1R12 is FLOWA.

FLOWA is added for latch processing and traces GRS Latch Manager processing only. This produces CTRACE data for:

- ▶ Entry point entry and exit



- ▶ Before and after pause/release points
- ▶ Schedule of SRBs
- ▶ Unexpected paths

Moving the GRS CTRACE buffers above the bar allows much more data to be recorded before the CTRACE output it is forced to wrap.

The allowable CTRACE buffer size for GRS has changed, as follows:

- ▶ Default buffer size changed from 128 K to 16 M
- ▶ Maximum buffer size changed from 16 M to 2047 M.

**Note:** 2047 M is the maximum allowed by CTRACE.

### CTRACE options

The CTRACE options are specified via the CTnGRSxx parmlib member. You can change options while a SYSGRS trace is running. However, to change the buffer size you must stop the trace and restart it with the new buffer size.

Latch identity (LID) support was added in z/OS V1R11 with RRS and System Logger as the first exploiters. It provides the ability for a latch user to assign a descriptive name to one or more latches of the latch set.

### IPCS support

IPCS ANALYZE now displays more latch diagnostics. These include:

- ▶ Request time
- ▶ Grant time
- ▶ Latch identity (LID) information, if provided, as follows:
  - Up to the first 255 characters displayed.
  - T follows the 255th character for truncation.

## 33.3 GRS XCF enhancements

GRS, as a critical component of the sysplex, can suffer sympathy sickness with regard to the serialization services it provides. With z/OS V1R12, GRS is the first exploiter of several new XCF/XES enhancements to help prevent sympathy sickness.

The sysplex failure management (SFM) policy, which is optional, defines how MVS is to manage sysplex resources. With z/OS V1R12, GRS exploits both XCF and XES critical member support and indicates a TERMLEVEL of SYSTEM. With the existing availability attributes of the z/OS platform, the enhancement supports the platform's RAS characteristics even further. GRS serialization services, particularly those regarding global ENQs, are a critical part of the z/OS installation and are therefore part of this effort.

XCF/XES with z/OS V1R12 provides new functionality for GRS to utilize:

- ▶ Extending XCF heart beating for critical members
- ▶ Extending XCF "stalled member" processing for critical members
- ▶ Providing XES structure exit hang detection

## **z/OS V1R12 implementation**

With this new support, most of the potential problems may have never been seen in installations. However, these potential problems represent critical sections of GRS processing that can now be monitored and automatically handled through XCF/XES and SFM, as follows:

- ▶ GRS Star signalling sickness, which is the inability of GRS to respond to XCF signals in this mode
- ▶ GRS Star space sickness, when there is an inability of GRS to schedule SRBs that complete in a timely manner
- ▶ GRS Star Global and QScan sickness, when there is an inability of these tasks to process requests
- ▶ GRS Ring signalling sickness, when there is an inability of GRS to respond to XCF signals in this mode
- ▶ GRS rebuild hang detection, when there is an inability of GRS to respond to rebuild signals for ISGLOCK

Ultimately, XCF/XES wait states the system of an “impaired” critical member. A Sysplex Failure Management Policy is necessary for rebuild hangs because a sysplex takes no action against the hung rebuild without it. With z/OS V1R12, great care has been taken to avoid “false positives” in XCF/XES determining that GRS as a critical member is impaired. If the sickness persists, wait stating a system is preferable for the overall health of the sysplex.



## 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 (the programmer) define to XCF. XCF maintains information about the parts you define, and provides notification of changes. Again, these parts can be on the same MVS system or different MVS systems.
- ▶ A way to design your program for high availability, such 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, to design your application for high availability by allowing it to be restarted automatically when it, or the system it is running on, fails.

This chapter describes the following changes with z/OS V1R12:

- ▶ XCF heart beating for critical system members
- ▶ SFM-based structure hang recovery
- ▶ XES/XCF health checks
- ▶ CF sublist notification enhancements
- ▶ Mean time to recovery (MTTR) for XCF
- ▶ Partitioning action by GDPS controlling system

## 34.1 XCF heart beating for critical system members

A system is deemed unresponsive if it stops sending XCF signals and stops updating its status in the sysplex Couple Data Set (CDS). However, these indications of activity do not necessarily mean that a system is able to accomplish useful work. An apparently active system could be causing sympathy sickness because critical components are unable to perform their intended function.

### XCF status monitoring

New XCF function in z/OS V1R12 extends XCF system status monitoring to incorporate status information about critical system components such as GRS. The goal of this support is to resolve the sympathy sickness by expeditiously partitioning a sick system out of the sysplex whenever any critical XCF member on that system is deemed unresponsive.

This new function:

- ▶ Monitors XCF group members of system critical components.
- ▶ Terminates the system if a critical component is impaired.

### XCF status monitoring actions

In addition this new function extends XCF member status monitoring to take action when a member indicates that it is “status missing”. XCF will now externalize via messages that the member is “impaired”. Furthermore, for members that are deemed to be “critical”, XCF will terminate the member if the impaired state persists long enough.

This additional new function:

- ▶ Enhances monitoring of XCF group members.
- ▶ Reports members that become impaired.
- ▶ Terminates impaired members identified as critical to the application.

The benefits include:

- ▶ Reduced sympathy sickness
- ▶ Improved diagnosis for status missing conditions
- ▶ Improved mean time to recovery (MTTR)

### 34.1.1 New function terminology

The following terminology is used to describe this new function:

<b>Critical member</b>	Any member that specifies CRITICAL=YES when invoking IXCJOIN to become an active member of the group.
<b>TermLevel</b>	Scope at which member is to be terminated. This can be task, space, or system. Specified by member when invokes IXCJOIN
<b>Confirmed impaired</b>	Any member that reports itself to be “status missing” via its member status exit routine.
<b>Deemed impaired</b>	Any member that XCF deems to be impaired because it is “stalled” and exhibits no signs of activity.

## 34.1.2 Enhancements to status monitoring

A “system critical member” is one that specifies CRITICAL=YES and TERMLEVEL=SYSTEM when invoking IXCJOIN to become an active member of an XCF group. XCF honors the attributes for application that specifies them. It is up to the exploiter to determine the applicability of “system critical”.

Prior to z/OS V1R12, XCF member status monitoring interacted with the member’s status exit routine to determine whether the member is operational or not. XCF monitors the member’s “status field”. If the content of the status field changes within the “status interval”, the member is deemed to be operational. If the status field does not change, XCF schedules an SRB to the member’s address space to call the member “status exit” routine. The status exit returns to XCF with a code to indicate whether the member is operational or not. If the status exit does not return in a timely fashion, XCF deems the member to be non-operational. Changes in status from operational to non-operational (or vice versa) are propagated to the other members of the group via the group exit.

With z/OS V1R12, XCF may now take additional actions. XCF will surface the “not operational” state via messages. And for critical members, XCF will terminate the member if the impairment condition persists long enough.

In addition XCF monitors the member group exit and signal exit processing for stall conditions. With this new function, XCF also monitors the status exit routine. If the member is stalled and does not have any signs of “user activity”, XCF deems the member to be impaired.

## 34.1.3 Installation considerations

The support is included in the base with z/OS V1R12. It performs the following functions:

- ▶ XCF member status monitoring reports impaired members.
- ▶ If a member declares itself to be “critical”, it is subject to being terminated if it becomes impaired and stays that way long enough.
- ▶ New and updated messages to document impaired members.
- ▶ The **D XCF,G,grpname,membername** command is enhanced to report additional member attributes and status information.

**Note:** GRS is an exploiter of this new function because it is declared as “system critical.”

The amount of time that XCF will allow a critical member to persist in an impaired state before it gets terminated is controlled by the MEMSTALLTIME specification in the SFM policy. If MEMSTALLTIME(NO) is specified, either explicitly or by default, XCF will instead wait the system failure detection interval or two minutes (whichever is longer) and then take action.

### New messages

The following new messages have been added to support this new function:

- ▶ IXC633I - “member is impaired”
- ▶ IXC634I - “member no longer impaired”
- ▶ IXC635E - “system has impaired members”
- ▶ IXC636I - “impaired member impacting function”

**Note:** Message IXC635E is likely the key message that would be used to trigger the relevant automation and/or operational procedures.

### Updated messages

The following messages have been updated to support this new function:

- ▶ IXC431I - “member stalled” (includes status exit)
- ▶ IXC640E - “going to take action”
- ▶ IXC615I - “terminating to relieve impairment”
- ▶ IXC333I - “display member details”
- ▶ IXC101I, IXC105I, IXC220W - “system partitioned via wait-state 0A2 reason code 194”

## 34.1.4 XCF processing for member impairment condition

The chart in Figure 34-1 summarizes how XCF deals with member impairment conditions. It attempts to combine both “stalled member” monitoring and “member impairment” monitoring because both monitors need to interact in an appropriate fashion. For clarity, messages IXC430E, IXC440E, IXC631I, IXC632I, and IXC640E as they relate to signalling sympathy sickness conditions are omitted from the chart.

**Note:** A member can be “impaired” either because XCF “deems” it to be so, or because the member “confirms” itself to be impaired.

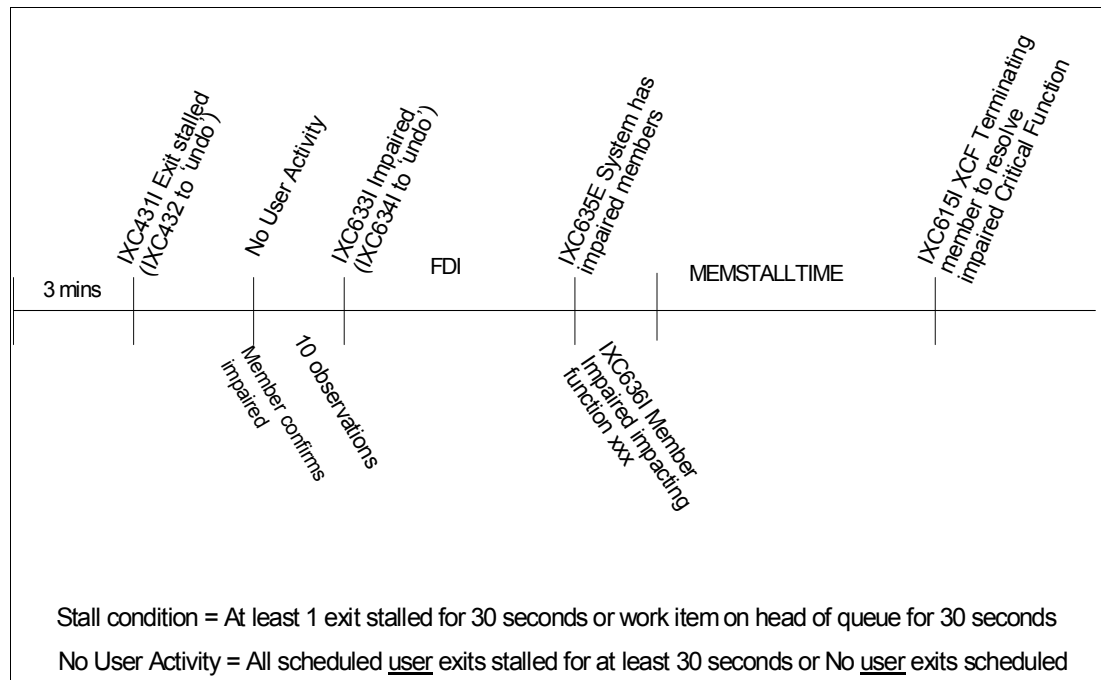


Figure 34-1 XCF processing for member impairment

### XCF monitoring

In the past, XCF monitoring detected “stall conditions” where one or more exit routines (group or signal) have not made progress, or work items pending for these exits have not been processed. With this new function the monitor is extended to consider the member status exit

routine as well. If the stall condition persists long enough (approximately 3 minutes), XCF issues message IXC431I to document the “stalled member”. Once the stall condition is externalized, the monitor looks to see whether the member should be deemed impaired. To qualify, XCF must find no signs of “user activity” for a 30 second period. Each time a new user exit is scheduled and/or each time XCF observes an exit makes progress, XCF restarts the “no user activity” timer. If the 30-second timer expires, the member is deemed to be impaired.

### **Impaired members**

A member is “confirmed impaired” if its status exit routine reports that the member is “status missing”. If a member confirms itself to be impaired, the state of the exit routines is mostly irrelevant. The member is responsible for determining its own status correctly. If the member indicates that it is impaired, then XCF will take action.

At the point the member is deemed impaired or confirmed impaired, XCF continues to observe the member. If the impairment condition persists for 10 observations (seconds), XCF issues message IXC633I to document that the member is impaired. After the message is issued, XCF continues its monitoring. If the impairment condition persists for the system failure detection interval (FDI, aka INTERVAL specification on the COUPLE statement in the COUPLExx parmlib member), XCF issues message IXC636I to document the impaired member and the function that it is being impacted. This message also indicates whether the member is “critical” and thus subject to being terminated if the condition persists. XCF also issues message IXC635E to the console to alert the operator. DISPLAY XCF,GROUP commands can be used to determine more information about the impaired member.

After issuing IXC636I for the member, XCF determines whether the impaired member is critical, and if so, whether the member is “deemed impaired” or “confirmed impaired”. If “confirmed impaired”, XCF redrives the status exit routine to have the member confirm its status one more time. If the critical member confirms “status missing”, or if the member is “deemed impaired”, XCF issues IXC640E to the console to indicate that impaired members are impacting the sysplex. The message indicates when XCF (SFM) intends to take action to alleviate the situation.

After issuing the IXC640E message, XCF waits MEMSTALLTIME seconds. If MEMSTALLTIME(NO) is in effect, XCF waits FDI seconds or 120 seconds, whichever is greater. For brevity, we just say XCF waits MEMSTALLTIME seconds. If the member is still impaired at approximately MEMSTALLTIME minus 30 seconds, XCF issues abend 00C reason 020F000D and takes a dump for diagnosis. If at the end of the MEMSTALLTIME interval, the member is still impaired, XCF terminates the member per the TERMLEVEL specification from the member’s IXCJOIN invocation. XCF issues message IXC615I just before it initiates termination.

## **34.1.5 Migration and coexistence considerations**

The SFM policy MEMSTALLTIME specification needs to be evaluated and adjusted if desired.

If your active SFM policy specifies MEMSTALLTIME(n) where n is some integral number of seconds, that value n will determine the number of seconds that SFM waits before it terminates:

- ▶ A stalled member causing signalling sympathy sickness between systems
- ▶ An impaired critical member that is deemed to be impacting its function (and thus the system, and thus the sysplex).

**Note:** The existing MEMSTALLTIME specified to deal with sympathy sickness conditions is likely valid for critical member support as well.

If you do not currently have an active SFM policy, or your SFM policy does not specify MEMSTALLTIME, MEMSTALLTIME(NO) is the default. With MEMSTALLTIME(NO), SFM will terminate an impaired critical member after the system failure detection interval (FDI) or two minutes, whichever is greater.

**Note:** The default FDI on a z/OS V1 R12 system is 165 seconds and can be displayed via the DISPLAY XCF,COUPLE command.

### Using MEMSTALLTIME

If you currently use MEMSTALLTIME(NO) and you do not want SFM to terminate impaired critical members (that is, a wait-state system if GRS becomes impaired), then you can prevent XCF taking action for an impaired member by:

- ▶ Creating an SFM policy if you do not already have one.
- ▶ Specifying MEMSTALLTIME(64800). This is the maximum of 18 hours and the problem should be resolved long before then.

Toleration APAR OA31619 for systems running z/OS V1R109 and z/OS V1R11 should be installed before IPLing z/OS V1R12. OA31619 allows the down-level systems to understand the new sysplex partitioning reason that is used when the z/OS V1R12 system removes itself from the sysplex because a system critical component was impaired.

**Note:** If OA31619 is not installed, the content of the IXC101I and IXC105I messages will be incorrect, but there is no effect on XCF functions.

## 34.1.6 Messages related to an impaired system critical member

This section shows how the D XCF,G command can be used to display a system critical member that has become impaired.

**Note:** The messages are the result of a system critical XCF member that was caused to become impaired, and observing the new XCF function actions.

The output from the D XCF, G command in Figure 34-2 shows that member C0000002 is flagged with an asterisk and so has a problem.

```
D XCF,G
IXC331I 15.33.46 DISPLAY XCF 872
* INDICATES PROBLEM, ! INDICATES SEVERE PROBLEM
GROUPS(SIZE): COFVLFNO(2)      CTTXNGRP(2)      *C0000002(2)
                ISTCFS01(2)     ISTXCF(2)        IXCL002B(2)
                SYSCNZMG(2)     SYSDAE(5)        SYSENF(2)
                SYSGRS(2)       SYSGRS2(1)      SYSIEFTS(2)
```

Figure 34-2 Message IXC331I for D XCF,G



We now issue a D XCF,G,C000002 command to get the list of members in group C000002, as shown in Figure 34-3.

```
D XCF,G,C000002
IXC332I 15.33.46 DISPLAY XCF 873
* INDICATES PROBLEM, ! INDICATES SEVERE PROBLEM
GROUP C000002: MEMBER1 *MEMBER2
```

Figure 34-3 Message IXC331I for D XCF,G,C000002

We see that MEMBER2 is flagged with an asterisk, signifying that member MEMBER2 of group C000002 appears to be having problems.

The DISPLAY XCF,GROUP,C000002,MEMBER2 command shown in Figure 34-4 provides detailed information about MEMBER2.

```
D XCF,G,C000002,MEMBER2
IXC333I 15.33.46 DISPLAY XCF 874
INFORMATION FOR GROUP C000002
* INDICATES PROBLEM, ! INDICATES SEVERE PROBLEM
MEMBER NAME: SYSTEM: JOB ID: STATUS:
*MEMBER2 SY4 XCBMEZ57 CONFIRMED STATUS MISSING

INFO FOR GROUP C000002 MEMBER MEMBER1 ON SYSTEM SY4

FUNCTION: TESTCASE XCJMEZ57
MEMTOKEN: 0200000F 00130002 ASID: 002E SYSID: 02000024
INFO: CURRENT COLLECTED: 03/03/2010 15:33:46.328662

ATTRIBUTES JOINED: 03/03/2010 15:33:41.195378
JOIN TASK ASSOCIATION
CRITICAL MEMBER
LOCAL CLEANUP NOT NEEDED
TERMLEVEL IS SYSTEM
MEMSTALL RESOLUTION IS SYSTEM TERMINATION AFTER 120 SECONDS
EXITS DEFINED: MESSAGE, GROUP, STATUS

MONITOR SERVICE
STAT INTERVAL: 3 STATUS: CONFIRMED MISSING
STAT DETECTED: 03/03/2010 15:33:45.196442
LAST VERIFIED: 03/03/2010 15:33:45.196442
EXIT 026B9550: 03/03/2010 15:33:45.194492 SM 00:00:00.001864
```

Figure 34-4 Message IXC331I for D XCF,G,C000002,MEMBER2

### 34.1.7 z/OS V1R12 message text enhancements

The summary status of the member has been enhanced with new text to indicate whether the member is impaired. In this example, the member status exit has indicated that the member is “status missing”, thus XCF reports that the member has “confirmed status missing”.

#### ATTRIBUTES section

The ATTRIBUTES section of the display is new for z/OS V1R12. We now include the TOD when the member joined the group. This TOD is useful for knowing how long the member has

been defined to XCF. The remaining lines appear if they apply to a member. So, for example, the text CRITICAL MEMBER will appear if the member specified CRITICAL=YES on the IXCJOIN invocation that caused the member to join the group. TERMLEVEL IS SYSTEM lets us know that the system will be removed from the sysplex if this member needs to be terminated by XCF.

**Note:** This method of termination would be used whenever XCF has occasion to terminate the member, not just for impaired members.

The MEMSTALL RESOLUTION line indicates whether XCF will take action against this member in order to relieve various conditions (signal sympathy sickness, member impairment), and if so, which action and how quickly. For example, if this critical member becomes impaired, XCF will remove the system from the sysplex if the member does not become operational within two minutes.

### **MONITOR SERVICE section**

The STAT DETECTED TOD indicates when XCF most recently saw the member status change. In this case, it would be when the member status became “missing”.

The LAST VERIFIED TOD indicates when XCF last checked the status of the member. For example, the XCF member status monitor will periodically drive the member status exit routine to inquire as to the state of the member. The status exit may confirm that the status is still the same.

The last EXIT line indicates the TOD when the member status exit was called (or scheduled). The two-letter code SM indicates that XCF last inquired as to whether the member was “status missing”. One might also see SR if XCF inquired to see if the member had resumed normal operation. Following the SM, there will either be text (such as RUNNING) to indicate the state of the status exit routine itself, or as in this case, a delta to indicate how long it took for the status exit routine to return to XCF.

## **34.1.8 XCF reports impairment**

At this point the member is now “confirmed impaired” and so XCF issues message IXC633I, as shown in Figure 34-5 on page 669.

Message IXC636I indicates what function is being impacted. Because the member is a critical member, the impacted function is reported as being critical. The “function” description is provided by the member when it invokes the IXCJOIN macro to become a member of the group. It is up to the exploiter to make the description meaningful.

Message IXC635E is issued to the console to alert the existence of impaired members.

```

IXC633I GROUP C0000002 MEMBER MEMBER2 JOB XCBMEZ57 ASID 002E
CONFIRMED IMPAIRED AT 03/03/2010 15:33:45.196442 ID: 1.1
  LAST MSGX:                                0 STALLED      0 PENDINGQ
  LAST GRPX:                                0 STALLED      0 PENDINGQ
  LAST STAX: 03/03/2010 15:34:04.086335    0 STALLED

IXC636I GROUP C0000002 MEMBER MEMBER2 JOB XCBMEZ57 ASID 002E
  IMPAIRED, IMPACTING CRITICAL FUNCTION TESTCASE XCJMEZ57

IXC635E SYSTEM SY4 HAS IMPAIRED XCF GROUP MEMBERS

```

Figure 34-5 Messages ICXC633I and IXC635E indicating an impaired member

### 34.1.9 IXCJOIN macro

The IXCJOIN macro places a cross-system Coupling Facility (XCF) member in the active state, associating it with an XCF group. In the active state, the member can use the monitoring and signalling services of XCF.

With z/OS V1R12, the IXCJOIN macro has new keywords to support, as follows:

**CRITICAL=(YES | NO)** Use this input parameter to indicate whether the member is a critical member. A critical member is one whose function is so critical to the normal operation of the group (and probably the system) that the member should be terminated if it appears to be impaired.

**NO** - The member does not designate itself as a critical member. This is the default.

**YES** - The member designates itself as a critical member. XCF will monitor the health of the member. If the member becomes impaired for too long, XCF will terminate the member according to the TERMLEVEL specification.

To determine whether the member is impaired, XCF can use two techniques to monitor the health of a critical member:

- ▶ XCF can use the status supplied by the member.
- ▶ XCF can monitor the member's ability to process its XCF related work.

If the member requests status monitoring by coding the STATFLD, STATEXIT, and INTERVAL keywords, XCF will use both monitoring methods. If the member does not request status monitoring at join time, XCF will use the second method only.

**FUNCTION=** Use this input parameter to describe the function, service, or application associated with the member. This description will appear in various XCF messages that provide information about the member.

**NO\_FUNCTION** is the default.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 24-character input field that contains the description of the function, service, or application associated with the member. The string can contain any alphanumeric (A-Z),

national (@,#,\$), or special character (underscore or blank). Leading blanks and all blank descriptors are not permitted.

**Note:** The FUNCTION keyword is available on the IXCJOIN macro to any member. Critical members are required to code the FUNCTION keyword in an attempt to make it reasonably clear what is being impacted should the member become impaired, and then subsequently terminated.

The FUNCTION keyword is required if the IXCJOIN macro uses a parameter list of version 2 or higher. That is to say, it is required if any of the following keywords are used: CRITICAL, TERMLEVEL, and RECOVERYMGR.

**RECOVERYMGR=**

**(YES | NO)** - Use this input parameter to indicate whether the member is a recovery manager. A recovery manager is a group member that coordinates a sysplex-wide recovery process.

**NO** - The member does not designate itself as a recovery manager.

**YES** - The member designates itself as a recovery manager.

**TERMLEVEL=**

**(MEMASSOC | ADDRSPACE | SYSTEM)** - Use this input parameter to specify the first member-specific termination action the system is to take against this member when it needs to be terminated. For example, XCF may terminate a member that is determined to be causing signalling sympathy sickness, or a critical member that becomes impaired.

**MEMASSOC** - The system will use the MEMASSOC keyword specification to determine the task or address space with which the member is associated for termination purposes.

- ▶ For MEMASSOC=TASK, the task from which the member invoked the IXCJOIN macro will be terminated.
- ▶ For MEMASSOC=JOBSTEP, the job step task from which the member invoked the IXCJOIN macro will be terminated.
- ▶ For MEMASSOC=ADDRSPACE, the address space from which the member invoked the IXCJOIN macro will be terminated.

**ADDRSPACE** - The system will terminate the address space from which the member invoked the IXCJOIN macro.

**SYSTEM** - The system will terminate the system on which the member resides. The system will enter wait state and be removed from the sysplex.

**Note:** The termination will cause any other members associated with the relevant task, space, or system to be terminated too.

The setting on TERMLEVEL is honored whenever XCF needs to terminate the member, for example, when IXCTERM is used to terminate a target member.

**LOCALCLEANUP=**

**(NEEDED | CONTINUE)** - Use this input parameter to indicate whether the member needs time to perform cleanup processing

when the system on which the member resides is being removed from the sysplex. XCF calls the member group user exit routine with the “system being removed from sysplex” event (GEPLTYPE=GESYSVRT) to notify the member that its system is being removed from the sysplex.

**NEEDED** - The member needs to perform cleanup when the system on which it resides is being removed from the sysplex. XCF will wait no longer than the installation-defined CLEANUP interval for the member to accomplish the cleanup before putting the system in a wait state. The member must inform XCF when the cleanup is completed by invoking either the IXCLEAVE or IXCQUIES macro. System termination will continue when the CLEANUP interval expires, or when all members that need to do cleanup have indicated that their cleanup is finished.

**CONTINUE** - The member does not need time to perform additional cleanup after its group exit is presented with the “system being removed from the sysplex” event (GEPLTYPE=GESYSVRT). XCF may wait for other members to finish their cleanup, but it need not wait for this member to perform cleanup. With respect to this member, XCF can immediately continue with system termination.

**Note:** The system defaults to LOCALCLEANUP=CONTINUE if a group exit is not provided. XCF may also proceed as if LOCALCLEANUP=CONTINUE is specified if the group exit routine fails when presented with the “system being removed from sysplex” event (GEPLTYPE=GESYSVRT).

## Detecting problems

The D XCF,G and D XCF,G,C0000002 commands shown in Figure 34-6 now indicate a “severe problem”. A critical member, in group C0000002, is impaired, thus the function it provides is likely not operational. The display output now flags the group and member with an exclamation point.

```

D XCF,G
IXC331I 15.34.41 DISPLAY XCF 918
  * INDICATES PROBLEM, ! INDICATES SEVERE PROBLEM
  GROUPS(SIZE):  COFVLFO(2)      CTTXNGRP(2)      !*C0000002(2)
                  ITCFS01(2)    ISTXCF(2)        IXCLO02B(2)
                  SYSCNZMG(2)   SYSDAE(5)        SYSENF(2)
                  SYSGRS(2)     SYSGRS2(1)      SYSIEFTS(2)

D XCF,G,C0000002
IXC332I 15.34.41 DISPLAY XCF 921
  * INDICATES PROBLEM, ! INDICATES SEVERE PROBLEM
  GROUP C0000002:  MEMBER1      !*MEMBER2
  
```

Figure 34-6 Messages IXC331I and IXC332I indicating a severe problem

The detailed member status via the D XCF,G,C0000002,MEMBER2 command shown in Figure 34-7 on page 672 also flags the member with an exclamation point indicating a status of confirmed impaired and the member status is confirmed missing.

```

D XCF,G,C000002,MEMBER2
IXC333I 15.34.41 DISPLAY XCF 924
  INFORMATION FOR GROUP C000002
  * INDICATES PROBLEM, ! INDICATES SEVERE PROBLEM
  MEMBER NAME:      SYSTEM:    JOB ID:    STATUS:
  !*MEMBER2        SY4        XCBMEZ57  CONFIRMED IMPAIRED

INFO FOR GROUP C000002 MEMBER MEMBER2 ON SYSTEM SY4

FUNCTION: TESTCASE XCJMEZ57
MEMTOKEN: 0200000F 00130002      ASID: 002E      SYSID: 02000024
  INFO: CURRENT      COLLECTED: 03/03/2010 15:34:41.527918
...
MONITOR SERVICE
  STAT INTERVAL:      3  STATUS: CONFIRMED MISSING
  STAT DETECTED: 03/03/2010 15:33:45.196442
  LAST VERIFIED: 03/03/2010 15:34:38.785000
  EXIT 026B9550: 03/03/2010 15:34:38.725594 SR 00:00:00.059303

```

Figure 34-7 Message IXC333I displays a member that has status of confirmed impaired

**Note:** The status missing condition was detected at 15:33:45.196442 and that XCF most recently confirmed this status at 15:34:38.785000. XCF most recently asked the status exit whether the member had resumed normal operation (SR). However, the status exit continued to report the status as “missing”.

The member remained impaired for FDI seconds, so XCF issues IXC640E shown in Figure 34-8 to indicate that XCF is going to take action to deal with the impaired member.

```

IXC640E IMPAIRED XCF GROUP MEMBERS ON SYSTEM SY4 IMPACTING SYSPLEX
SFM POLICY SFMPOL01 WILL TAKE ACTION AT 03/03/2010 15:36:46

IXC633I GROUP C000002 MEMBER MEMBER2 JOB XCBMEZ57 ASID 002E
CONFIRMED IMPAIRED AT 03/03/2010 15:33:45.196442 ID: 1.2
  LAST MSGX:                0 STALLED      0 PENDINGQ
  LAST GRPX: 03/03/2010 15:34:48.983239  0 STALLED      0 PENDINGQ
  LAST STAX: 03/03/2010 15:35:07.830721  0 STALLED

```

Figure 34-8 Message IXC640E and IXC633I for XCF taking action for impaired member

### Refresh impaired member

XCF also issues a “refresh” of the member’s impairment notification to show the current state of the member. The ID of 1.2 indicates that this message refers to impairment #1, and that this is the second IXC633I message to be issued for that condition. The message indicates when the member became impaired (15:33:45.196442) and provides information about the various exit routines that the member is using. In this case, we see that there are no stall conditions and there is no XCF work pending for any of the exit routines. The other TODs indicate when the relevant exit routine last returned to XCF.

## XCF takes action for an impaired system critical member

Approximately 30 seconds before the member is to be terminated, XCF issues abend 00C rsn 020F000D to create a logrec entry and generate an SVC dump to document the problem, as shown in Figure 34-9. This data may help with problem diagnosis.

```
IEA045I AN SVC DUMP HAS STARTED AT TIME=15.36.17 DATE=03/03/2010 005
FOR ASIDS(0006,0001,002E)
ERROR ID = SEQ00052 CPU81 ASID0001 TIME15.36.17.6
QUIESCE = YES

IEA794I SVC DUMP HAS CAPTURED: 008
DUMPID=005 REQUESTED BY JOB (*MASTER*)
DUMP TITLE=COMPON=XCF,COMPID=5752SCXCF,ISSUER=IXCM2REC,MODULE=I
XCS1DCM,ABEND=S000C,REASON=020F000D
```

Figure 34-9 XCF schedules an SVC dump for the impaired member

**Note:** The abend 00C-020F000D does not impact the application because the processing is done in the XCF address space.

The member remains impaired for MEMSTALLTIME seconds. When that interval expires, XCF takes action to alleviate the problem, as shown in Figure 34-10. In this case, the critical member specified TERMLEVEL=SYSTEM. This results in:

- ▶ Message IXC615I to indicate that it is terminating the member.
- ▶ Message IXC640E being reissued to indicate that XCF is taking action (now!) to resolve the impaired member.
- ▶ Wait state 0A2, and reason code 194 are issued to indicate that the member was removed as a result of XCF's attempt to resolve an impaired critical member.

```
IXC615I GROUP C000002 MEMBER MEMBER2 JOB XCBMEZ57 ASID 002E
SFM TERMINATING SYSTEM TO RELIEVE IMPAIRMENT CONDITION

IXC640E IMPAIRED XCF GROUP MEMBERS ON SYSTEM SY4 IMPACTING SYSPLEX
SFM POLICY SFMPOL01 IS TAKING ACTION

IXC220W XCF IS UNABLE TO CONTINUE: WAIT STATE CODE: 0A2 REASON CODE:
194,
SYSTEM HAS AN IMPAIRED CRITICAL MEMBER

IXC101I SYSPLEX PARTITIONING IN PROGRESS FOR SY4 REQUESTED BY
XCFAS. REASON: SYSTEM HAS AN IMPAIRED CRITICAL MEMBER
...
IXC105I SYSPLEX PARTITIONING HAS COMPLETED FOR SY4
- PRIMARY REASON: SYSTEM HAS AN IMPAIRED CRITICAL MEMBER
- REASON FLAGS: 800015
```

Figure 34-10 Processing for impaired member after MEMSTALLTIME expires

On a peer system in the sysplex, the sysplex partitioning messages (IXC101I and IXC105I) will indicate that the system was partitioned in order to resolve a critically impaired member.

**Note:** Down-level systems would need to have the compatibility PTF for APAR OA31619 installed in order to issue these messages with the correct insert.

### 34.1.10 Messages for members confirmed impaired

XCF issues messages for a situation where the member's exits become stalled. The XCF monitor detects the stall condition, sees it persist with no signs of user activity, and so deems the member to be impaired. The following actions can then be taken:

- ▶ The member then resumes activity with respect to its exits, so the member is no longer "deemed impaired", but then the member uses its status exit routine to indicate that it is "status missing".
- ▶ Eventually XCF reports the member as "confirmed impaired". The member then resumes normal status, and terminates.
- ▶ One of its exit routines is still stalled though. Figure 34-11 shows the output from the D XCF,G and D XCF,G,group commands for the impaired member.
- ▶ Groups and members are flagged with an asterisk if there is a relevant member that is impaired or stalled.

```
D XCF,G
IXC331I 12.18.02 DISPLAY XCF 994
* INDICATES PROBLEM, ! INDICATES SEVERE PROBLEM
GROUPS(SIZE):  COFVLFN0(2)      CTTXNGRP(2)      *GROUP1(2)
                 ISTCFS01(2)      ISTXCF(2)        IXCL002B(2)
                 SYSCNZMG(2)      SYSDAE(5)        SYSENF(2)
                 SYSGRS(2)        SYSGRS2(1)       SYSIEFTS(2)

D XCF,G,GROUP1
IXC332I 12.18.02 DISPLAY XCF 995
* INDICATES PROBLEM, ! INDICATES SEVERE PROBLEM
GROUP GROUP1:  MEMBER1          *MEMBER2
```

Figure 34-11 Messages IXC331I and IXC332I for an impaired member

The output from the D XCF,G,GROUP1,MEMBER2 command in Figure 34-12 on page 675 shows that the member status is reported as DETECTED STATUS MISSING. This means that the status exit was deemed unresponsive. As a result, XCF assumes that the member is "status missing". However, "detected missing" will necessarily cause XCF to flag the member as impaired. XCF must either get confirmation from the status exit for it to be "confirmed impaired", or the stalled exit monitoring must discover that the member is stalled with no user activity.



```

D XCF,G,GROUP1,MEMBER2
IXC333I 12.18.02 DISPLAY XCF 996
  INFORMATION FOR GROUP GROUP1
  * INDICATES PROBLEM, ! INDICATES SEVERE PROBLEM
  MEMBER NAME:      SYSTEM:      JOB ID:      STATUS:
*MEMBER2           SY4           XCBC6Z06    DETECTED STATUS MISSING

  INFO FOR GROUP GROUP1 MEMBER MEMBER2 ON SYSTEM SY4
  * INDICATES STALLS

FUNCTION: TESTCASE XCJC6Z06
MEMTOKEN: 02000012 00130002      ASID: 002E      SYSID: 0200002E
  INFO: CURRENT      COLLECTED: 03/04/2010 12:18:03.444140

ATTRIBUTES      JOINED: 03/04/2010 12:17:18.953386
  JOIN TASK ASSOCIATION
  LOCAL CLEANUP NOT NEEDED
  TERMLEVEL IS TASK
  MEMSTALL RESOLUTION IS NO ACTION
  EXITS DEFINED: MESSAGE, GROUP, STATUS

GROUP SERVICE
  EVNT RECEIVED:      1  PENDINGQ:      0

  *EXIT 02690020: 03/04/2010 12:17:19.787841 02 RUNNING

MONITOR SERVICE
  STAT INTERVAL:      3  STATUS: DETECTED MISSING
  STAT DETECTED: 03/04/2010 12:17:26.013930
  LAST VERIFIED: 03/04/2010 12:18:02.816921
  *EXIT 02690198: 03/04/2010 12:17:22.867676 SM RUNNING

```

Figure 34-12 Message IXC333I for D XCF,G,GROUP1,MEMBER1

**Note:** The criteria XCF uses for “detected missing” versus “stalled exit” are different. The “detected missing” criteria are satisfied much sooner than the “stalled exit” criteria: three seconds for “detected missing” versus 30 seconds for “stalled exit”.

The asterisk on the EXIT lines, shown in Figure 34-12, indicates that XCF considers the respective exit routines to be stalled. In this case, the fact that the exits are RUNNING tells us that XCF called the exit routine at the indicated TOD, but as of the point in time when the display output was collected, the exit routine had not yet returned to XCF.

**Note:** The display command output was issued at 12:18:03.45. We can compare that time or the “time when collected” to the times reported for the “running” exits to determine how long the exits have been stalled.

### XCF reports the member is stalled

XCF eventually issues message IXC431I to indicate that the member is stalled shown in Figure 34-13 on page 676. The stalled at TOD indicates when XCF determines the stall started.

```

IXC431I GROUP GROUP1 MEMBER MEMBER2 JOB XCBC6Z06 ASID 002E 997
  STALLED AT 03/04/2010 12:17:19.787841 ID: 0.1
  LAST MSGX:                0 STALLED        0 PENDINGQ
  LAST GRPX:                1 STALLED        0 PENDINGQ
  LAST STAX:                1 STALLED
IXC430E SYSTEM SY4 HAS STALLED XCF GROUP MEMBERS

```

Figure 34-13 Messages IXC431I and IXC430E for a stalled member

This TOD matches the TOD when the group exit was called in the display output in Figure 34-12 on page 675. The ID 0.1 says that the stall/impairment condition number 0 is being reported and this is the first IXC431I message issued for this particular condition.

After a while, XCF deems the member to be impaired because none of its exit routines are showing any activity, as shown in Figure 34-14.

```

IXC633I GROUP GROUP1 MEMBER MEMBER2 JOB XCBC6Z06 ASID 002E 006
DEEMED IMPAIRED AT 03/04/2010 12:17:22.867676 ID: 1.1
  LAST MSGX:                0 STALLED        0 PENDINGQ
  LAST GRPX:                1 STALLED        1 PENDINGQ
  LAST STAX:                1 STALLED
IXC636I GROUP GROUP1 MEMBER MEMBER2 JOB XCBC6Z06 ASID 002E
  IMPAIRED, IMPACTING FUNCTION TESTCASE XCJC6Z06
IXC635E SYSTEM SY4 HAS IMPAIRED XCF GROUP MEMBERS

```

Figure 34-14 Messages IXC633I and IXC635E for a member deemed impaired

In this case both the group exit and the status exit stalled on the very first call. As no one has sent any signals to this member, the signal exit has never been driven. If a signal arrived XCF would revise the “deemed impaired” state because the scheduling of the signal exit routine would be seen as “user activity”.

**Note:** The member impairment is a new condition and so ID 1.1 indicates that the IXC633I message is reporting the status of condition one for the first time.

Because the member is not “critical,” it will not be terminated. XCF reports the impairment conditions that it detects:

- ▶ Message IXC636I is issued because the member is impaired.
- ▶ Message IXC635E is issued to the console to alert the operator.

### XCF reports that the member is no longer impaired

The status exit finally returns to XCF and so XCF no longer deems the member to be impaired because recent activity has been observed. Because the group exit is still stalled, the member remains stalled. This status is shown in Figure 34-15 on page 677.

```

IXC634I GROUP GROUP1 MEMBER MEMBER2 JOB XCBC6Z06 ASID 002E
NO LONGER IMPAIRED.
ACTIVITY OBSERVED AT 03/04/2010 12:19:06.474031 ID: 1

IXC431I GROUP GROUP1 MEMBER MEMBER2 JOB XCBC6Z06 ASID 002E 074
STALLED AT 03/04/2010 12:17:19.787841 ID: 0.2
LAST MSGX:                0 STALLED        0 PENDINGQ
LAST GRPX:                1 STALLED        1 PENDINGQ
LAST STAX: 03/04/2010 12:19:11.730106  0 STALLED

```

Figure 34-15 Message IXC634I and IXC431I for a member that is no longer impaired

Message IXC634I is issued to indicate that the member is no longer impaired and indicates when activity was observed. This is TOD when the status exit returned.

**Note:** Message IXC634I is associated with ID 1 indicating that it is “closing out” impairment condition number one.

Because the state of the exits has changed, the stalled exit monitor issues message IXC431I to refresh the information about the stall condition. ID 0.2 indicates that this is the second issuance of message IXC431I for the stall condition number zero. You can see that the status exit is no longer stalled and has now been observed to have returned to XCF. The TOD when the stall condition started has not changed because the group exit which was the first exit to stall remains stalled.

The status exit runs in a timely fashion and reports that the member is impaired. XCF issues message IXC633I, shown in Figure 34-16, to indicate that the member is confirmed impaired.

```

IXC633I GROUP GROUP1 MEMBER MEMBER2 JOB XCBC6Z06 ASID 002E 077
CONFIRMED IMPAIRED AT 03/04/2010 12:19:05.570133 ID: 2.1
LAST MSGX:                0 STALLED        0 PENDINGQ
LAST GRPX:                1 STALLED        1 PENDINGQ
LAST STAX: 03/04/2010 12:19:14.877502  0 STALLED

IXC636I GROUP GROUP1 MEMBER MEMBER2 JOB XCBC6Z06 ASID 002E
IMPAIRED, IMPACTING FUNCTION TESTCASE XCJC6Z06

```

Figure 34-16 Message IXC633I and IXC636I for a confirmed impaired member

The impairment condition lasts long enough for XCF to issue message IXC636I to indicate that the impaired member is impacting the indicated “function”.

**Note:** The ID 2.1 on message IXC633I indicates that we have now started a new condition number 2.

The group exit is still stalled and now also has an event pending which was reported via IXC431I in Figure 34-15 on page 677.

When the member leaves the XCF group, XCF recognizes that the member is going away and issues the messages shown in Figure 34-17 on page 678.

```

IXC432I GROUP GROUP1 MEMBER MEMBER2 JOB XCBC6Z06 ASID 002E
RESUMED AT 03/04/2010 12:19:55.804639 ID: 0

IXC634I GROUP GROUP1 MEMBER MEMBER2 JOB XCBC6Z06 ASID 002E
NO LONGER IMPAIRED.
TERMINATING AT 03/04/2010 12:19:56.451206 ID: 2

```

Figure 34-17 Message IXC432I and IXC634I for a member that is no longer impaired

- ▶ Message IXC432I is issued to indicate that the member is no longer stalled.
- ▶ Message IXC634I similarly indicates that the member is no longer deemed to be impaired because the member is terminating.

In the previous displays the IXC432I message has ID=0, which matches the initial IXC431I member stalled message. The IXC634I message has ID=2, which matches the most recent IXC633I message that was issued because the member confirmed that it was impaired. The messages with ID=1 have already matched. Those messages were issued to bracket the period when XCF deemed the member to be impaired because none of its exit routines exhibited any activity.

## 34.2 SFM-based structure hang recovery

The XES hang detect function was introduced in OS/390 V1R8 (HBB6608) to report cases when an expected response to a structure-related event is not received in a timely manner. After 2 minutes without a response XES issues IXL040E or IXL041E to identify the unresponsive connector, the associated structure, the event, and the affected process. Installations often fail to react to these messages, or worse, react by terminating the wrong connector.

An application failure to respond to structure-related events can cause sysplex-wide sympathy sickness. The existing hang detect function reports overdue responses but installations often do not react appropriately or in a timely manner.

Structure-related processes require coordinated processing across all connected users of a CF structure. These include:

- ▶ Structure rebuild or duplexing
- ▶ Connector failure or disconnect recovery
- ▶ User synch points

Timely and correct participation from all of these users is essential. If they are unable to participate in the processing (for example due to a “sick but not dead” kind of problem), these processes will hang. This in turn can lead to sysplex-wide sympathy sickness because the affected CF structure may become unavailable for use by any of the exploiters while the process remains hung indefinitely.

With the existing hang detect support:

- ▶ After 2 minutes without a response, XES issues either message IXL040E or IXL041E.
- ▶ The message remains outstanding until an expected response is provided or is no longer required due to process or connector termination.
- ▶ Clearing of hang conditions is reported by message IXL042I or IXL043I.

## Operator automated responses

Often the IXL040E and IXL041E messages are missed, ignored, or misinterpreted. This new function in z/OS V1R12 continues the recent strategy of eliminating the need for operator response by automating responses to system-detected problems, particularly those with sysplex-wide implications. When the installation authorizes XES by updating the SFM policy, XES will not only report a delinquent response but take action to relieve the resultant hang. System actions may include stopping the affected process or terminating the unresponsive connector.

## SFM policy CFSTRHANGTIME keyword

The new SFM policy specification CFSTRHANGTIME is provided to allow the system to take automatic action to relieve a hang. This terminates hangs in a timely manner to prevent an unresponsive application from affecting the entire sysplex and relieves installations of the responsibility for determining appropriate action.

The new function is enabled by specifying the CFSTRHANGTIME keyword in the SFM policy.

- ▶ CFSTRHANGTIME(NO) = system is not to take automatic action to relieve a hang (default value)
- ▶ CFSTRHANGTIME(interval) = number of seconds after hang is recognized before system takes action
  - Range 0 – 1800 seconds
  - CFSTRHANGTIME(0) => system takes immediate action after recognizing hang
  - Initial recommendation 300 seconds

## CFSTRHANGTIME health check

The XCF\_SFM\_CFSTRHANGTIME health check described in “XCF\_SFM\_CFSTRHANGTIME” on page 689 can be customized to change default parameters if desired.

Figure 34-18 shows an example of the IXCMIAPU policy utility using the new keyword CFSTRHANGTIME for the SFM policy.

```
DEFINE POLICY NAME(WCNTST) CONNFAIL(YES) REPLACE(YES)
      SYSTEM NAME(*)
      ISOLATETIME(0)
      WEIGHT(10)
      CFSTRHANGTIME(300)
      SYSTEM NAME(MANUAL)
      PROMPT
      WEIGHT(10)
      CFSTRHANGTIME(NO)
```

Figure 34-18 SFM policy using the new CFSTRHANGTIME keyword

## 34.2.1 Structure hang recovery processing

The CFSTRHANGTIME interval begins when the hang is reported by issuing message IXL040E or IXL041E. This is about 2 minutes after the event requiring a response is delivered to the connector. The escalation hierarchy begins with the least disruptive actions and progresses to more disruptive actions.

The initial hang action taken at expiration of CFSTRHANGTIME interval can be:

- ▶ Rebuild stop is initiated if rebuild is in progress.
- ▶ Path stop if it is an XCF signaling structure and no rebuild occurs.
- ▶ Connector termination at TERMLEVEL level.

### Hang recovery actions

If hang persists, structure hang recovery escalates to more aggressive actions:

- ▶ Stop rebuild.
- ▶ Stop signaling path (XCF signaling only).
- ▶ Force disconnect (XCF signaling only).
- ▶ Terminate connector task.
- ▶ Terminate connector address space.
- ▶ Partition connector system.

**Note:** Each system acts against its own connectors and so no system will take action against any other system.

Often there are multiple events requiring a response outstanding simultaneously. For example, the application's protocol may incorporate user synch points into its rebuild quiesce processing (for example, XCF signaling), or a connector may fail during a rebuild. XES makes no attempt to determine whether the disc/fail response is blocking the rebuild-related response, or the rebuild response is blocking the disc/fail response, or both are blocked by some external factor completely invisible to XES, such as ENQ contention. XCF simply proceeds down the hierarchy, as stated previously.

### Connector termination

An application could be structured in such a way that recovery for the connect task is insufficient to protect application or system resources. The application might rely instead on address-space-level resource managers or on group notifications to clean up sysplex-wide resources.

An example of a case where task termination would complicate diagnosis is GRS. Terminating the GRS connect task would produce a wait state 0A3-090, which would obscure the fact that the task was terminated for failure to respond to some structure-related event.

A connector is normally terminated by terminating the connect task, but an application may have special considerations:

- ▶ Task termination may leave resources in an unusable state, whereas it may have address-space-level recovery.
- ▶ Task termination could complicate diagnosis.

Connectors can now use the IXLCONN TERMLEVEL keyword to specify that initial action should be task, address space, or system-level termination:

- ▶ TERMLEVEL(TASK)
- ▶ TERMLEVEL(ADDRSPACE)
- ▶ TERMLEVEL(SYSTEM)

**Note:** If TERMLEVEL is not specified, the system will default to terminating the connector task, which should be the appropriate response for the great majority of applications. As of z/OS V1R12, GRS connects specifying IXLCONN TERMLEVEL(SYSTEM).

## New and updated messages

There are a number of new messages and updated messages associated with structure hang recovery, as listed in Figure 34-19. These messages are explained in the following sections.

Updated Messages	New Messages
IXC220W	IXL047I
IXC307I	IXL048I
IXC357I	IXL049E
IXC360I	IXL050I
IXC467I	
IXC522I	
IXC614I	

Figure 34-19 New and updated messages

- ▶ IXC220W updated to include new wait state reasons:
  - TERMINATING STRUCTURE CONNECTOR TO RESOLVE HANG
  - TERMINATING STRUCTURE CONNECTOR DUE TO INTERNAL ERROR
- ▶ IXC307I updated to include CONNECTOR HANG RESOLUTION as a reason for stopping a path
- ▶ IXC467I updated to include CONNECTOR HANG as a reason for stopping a path
- ▶ IXC522I updated to include CONNECTOR HANG as a reason for stopping a rebuild
- ▶ IXC614I updated to state that the CFSTRHANGTIME specification applicable to the current system after an SFM policy is activated

The new IXL049E and IXL050I messages are shown in Figure 34-20 on page 682.

```

IXL049E HANG RESOLUTION ACTION FOR CONNECTOR NAME: conname
TO STRUCTURE strname, JOBNAME: jobname, ASID: asid:
actiontext

where actiontext is one of:
SFM POLICY NOT ACTIVE, MANUAL INTERVENTION REQUIRED.
SFM POLICY REQUIRES MANUAL INTERVENTION.
SYSTEM IS TAKING ACTION.
SYSTEM WILL TAKE ACTION AT termdate termtime
SYSTEM ACTION UNSUCCESSFUL, MANUAL INTERVENTION REQUIRED

IXL050I CONNECTOR NAME: conname TO STRUCTURE strname,
JOBNAME: jobname, ASID: asid
HAS NOT PROVIDED A REQUIRED RESPONSE AFTER noresponsetime SECONDS.
TERMINATING termtarget TO RELIEVE THE HANG.

where termtarget is one of:
REBUILD
SIGNAL PATHS (ATTEMPT 1)
SIGNAL PATHS (ATTEMPT 2)
SIGNAL PATHS (ATTEMPT 3)
CONNECTION
CONNECTOR TASK
CONNECTOR SPACE (WITH RECOVERY)
CONNECTOR SPACE (NO RECOVERY)
CONNECTOR SYSTEM

```

*Figure 34-20 IXL049E and IXL050I messages*

### **New messages IXL047I and IXL048I**

IXL047I and IXL048I replace IXL042I and IXL043I respectively, and are shown in Figure 34-21.

**Important:** These new messages introduce a potential migration action for installations who might have automated the original messages.



```

IXL047I THE RESPONSE REQUIRED FROM CONNECTOR NAME: conname
TO STRUCTURE strname, JOBNAME: jobname, ASID: asid
responsetype
IS NO LONGER EXPECTED.
REASON: reason

where responsetype is one of the following:
FOR THE REBUILD QUIESCE EVENT
FOR THE REBUILD CONNECT EVENT (IXLCONN REBUILD)
FOR THE REBUILD CONNECT EVENT (IXLREBLD REQUEST=COMPLETE)
FOR THE REBUILD SWITCH EVENT
FOR THE REBUILD CLEANUP EVENT
FOR THE REBUILD STOP EVENT
FOR THE STRUCTURE TEMPORARILY UNAVAILABLE EVENT
FOR THE USER SYNC POINT EVENT
AFTER CONNECTING DURING THE REBUILD QUIESCE PHASE
AFTER CONNECTING DURING REBUILD CONNECT (IXLCONN REBUILD)
AFTER CONNECTING DURING THE REBUILD CONNECT PHASE
AFTER CONNECTING DURING DUPLEX ESTABLISHED (IXLREBLD REQUEST=COMPLETE)
AFTER CONNECTING DURING REBUILD SWITCH (IXLCONN REBUILD)
AFTER CONNECTING DURING REBUILD SWITCH (IXLREBLD REQUEST=DUPLEXCOMPLETE)
AFTER CONNECTING DURING A USER SYNC POINT
AFTER CONNECTING DURING THE REBUILD STOP PROCESS
AFTER AN IXLCONN REBUILD PRIOR TO THE REBUILD CONNECT EVENT

IXL048I THE RESPONSE REQUIRED FROM CONNECTOR NAME: conname
TO STRUCTURE strname, JOBNAME: jobname, ASID: asid
FOR THE event
FOR SUBJECT CONNECTION subjectconname IS NO LONGER
EXPECTED.
REASON: reason

event is one of the following:
DISCONNECTED/FAILED CONNECTION EVENT
REBUILD CONNECT FAILURE EVENT

Both messages
reason is one of the following:
CONNECTOR HAS PROVIDED THE REQUIRED RESPONSE
CONNECTOR HAS DISCONNECTED/FAILED
REBUILD STOPPED
REBUILD COMPLETION

```

Figure 34-21 IXL047I and IXL048I messages

### Updated message IXC357I - D XCF,COUPLE output

The CFSTRHANGTIME value for the current system from the SFM policy is now displayed with the output from the D XCF,COUPLE command in the IXC357I message shown in Figure 34-22 on page 684.

```

IXC357I  hh.mm.ss
INTERVAL  OPNOTIFY  MAXMSG  CLEANUP  RETRY  CLASSLEN
interval  opnotify  maxmsg   cleanup  retry  classlen

SSUM ACTION  SSUM INTERVAL  SSUM LIMIT  WEIGHT MEMSTALLTIME
ssumaction  ssuminterval  ssumlimit  weight memstalltime

CFSTRHANGTIME
cfstrhangtime

```

Figure 34-22 IXC357I message

### Updated message IXC360I - D XCF,STRUCTURE output

The updated IXC360I message for the D XCF,STRUCTURE command output is displayed in Figure 34-23.

```

IXC360I  hh.mm.ss
STRNAME: strname
  STATUS: XES INITIATED REBUILD STOP:
    CONNECTOR HANG
...

CONNECTION NAME: conname
  ID           : id
...
ALLOWAUTO     : allowauto
SUSPEND       : suspend
TERMLEVEL     : termlevel

```

Figure 34-23 IXC360I message

## 34.2.2 Migration and coexistence considerations

Messages IXL042I and IXL043I are replaced by IXL047I and IXL048I. This requires a migration action for installations that previously automated on the superseded messages.

APAR OA30880 provides toleration support to allow z/OS V1R10 and R11 systems to use the IXCMIAPU utility to display CFSTRHANGTIME specifications in a policy created by a z/OS V1R12 system.

**Note:** This APAR does not permit downlevel systems to define a policy that includes CFSTRHANGTIME, nor does it enable any of the functions represented by the CFSTRHANGTIME parameter.

Since downlevel systems are unaffected by a CFSTRHANGTIME policy specification, there was no update to the DISPLAY XCF,COUPLE command for downlevel systems.

### SYS1.SAMPLIB update

A sample SFM policy utility job in SYS1.SAMPLIB(IXCSFMP) is updated with CFSTRHANGTIME examples.

## 34.3 New XES/XCF health checks

The following XES/XCF health checks are added in z/OS V1R12:

- ▶ XCF\_CF\_PROCESSORS
- ▶ XCF\_CF\_MEMORY\_UTILIZATION
- ▶ XCF\_CDS\_MAXSYSTEM
- ▶ XCF\_SFM\_CFSTRHANGTIME
- ▶ XCF\_CFRM\_MSGBASED
- ▶ XCF\_CF\_STR\_POLICYSIZE

Using Coupling Facility processor configuration for production and mission critical applications is based on the Coupling Facility architected function level (CFLEVEL). See the detailed description of the IXCH0912I report for the criteria that the check uses to determine successful or exception conditions based on CFLEVEL.

### 34.3.1 XCF\_CF\_PROCESSORS check

XCF\_CF\_PROCESSORS is a local system health check to check that Coupling Facilities in use by the system are configured for the best performance and throughput based on the Coupling Facility CFLEVEL. This check provides a warning when a Coupling Facility processor configuration is not consistent with IBM expected results and may result in degraded response time and throughput possible for Coupling Facility requests as compared to Coupling Facilities configured for the best performance and throughput based on the Coupling Facility architected function level (CFLEVEL).

This check provides a parameter to enable the installation to specify names of CFs whose processor configurations should be excluded from the determination of the overall status of the check. The installation may wish to exclude non-production CFs or CFs that are running in “test” mode with shared processors.

#### **EXCLUDE parameter**

EXCLUDE is a required parameter with which you can specify a list of CFNames that the check should not consider in its verification processing. A CF named in the EXCLUDE list indicates that the check should not include the processor configuration for that CF in determining the overall check status. Processor configurations for excluded CFs will be reported in message IXCH0912I and the report will indicate that the check results for the excluded CFNAME were not factored into the overall check status. Specifying EXCLUDE() will exclude no Coupling Facilities from the check verification process.

**Default:** EXCLUDE()

**Note:** This check is disabled in a VM environment because Coupling Facility processors cannot be configured as dedicated processors in a VM environment.

## POLICY statement in the HZSPRMxx parmlib member

To update this health check, Figure 34-24 is an example specification.

```
UPDATE CHECK(IBMxcf,XCF_CF_PROCESSORS)
SEVERITY(MED) INTERVAL(004:00) DATE (20090707)
PARM(EXCLUDE(CFName, CFName, ...))
REASON('Coupling facility should have dedicated processors')
```

Figure 34-24 Sample HZSPRMxx parmlib member

Figure 34-25 displays the XCF\_CF\_PROCESSORS health check and shows sample output.

```
CHECK(IBMxcf,XCF_CF_PROCESSORS)
START TIME: 06/01/2010 11:29:41.499784
CHECK DATE: 20090626 CHECK SEVERITY: MEDIUM
CHECK PARM: EXCLUDE()

* Medium Severity Exception *

IXCH0444E Coupling facility processor configurations in use by the
local system may result in degraded response time and throughput
for coupling facility requests.

IXCH0912I This report summarizes the processor configuration for
coupling facilities in use by the local system.

CFNAME                = The coupling facility name.

Processor Configuration = The configuration of dedicated and shared
                        processors for the coupling facility

Check Results          = The check result for the individual
                        coupling facility

An asterisk (*) before a Check Result for a coupling facility
indicates that at least one dedicated processor is in
the coupling facility configuration, but the number of shared
```

CFNAME	Processor Configuration	Check Results
CF7B	1 SHARED AND 0 DEDICATED PROCESSORS	EXCEPTION
CF7A	1 SHARED AND 0 DEDICATED PROCESSORS	EXCEPTION

```
END TIME: 06/01/2010 11:29:41.521403 STATUS: EXCEPTION-MED
```

Figure 34-25 XCF\_CF\_PROCESSORS health check and sample output

### 34.3.2 XCF\_CF\_MEMORY\_UTILIZATION

The current memory utilization for a Coupling Facility is determined by the amount of space allocated for structures in the Coupling Facility and the amount of space reserved for dumping structures allocated in the Coupling Facility. The reason for a check is that the percentage of

memory use in a Coupling Facility should not approach an amount high enough to keep it from allocating new structures or expanding existing structures.

This check provides a warning when a Coupling Facility processor configuration is not consistent with IBM recommendations and may result in degraded response time and throughput possible for Coupling Facility requests as compared to Coupling Facilities configured for the best performance and throughput based on the Coupling Facility architected function level (CFLEVEL).

This check provides a parameter to enable the installation to specify names of CFs whose processor configurations should be excluded from the determination of the overall status of the check. The installation may wish to exclude non-production CFs or CFs that are running in “test” mode with shared processors.

### **MAXUTILIZATION parameter**

MAXUTILIZATION is a required parameter indicating the threshold percentage that the Coupling Facility memory utilization should not exceed. The number must be an integer in the range of 1 to 99. Specifying a percent (%) sign is optional (for example, MAXUTILIZATION(60%)). It is possible that system-initiated alter processing for a structure may start and increase the memory utilization percentage for a Coupling Facility before the check executes or raises an exception. This check may issue an exception during reconfiguration actions or during maintenance windows when the percentage of memory use exceeds the specified check parameter.

### **POLICY statement in the HZSPRMxx parmlib member**

To update this health check, Figure 34-26 is an example specification.

```
UPDATE CHECK(IBMxcf,XCF_CF_MEMORY_UTILIZATION)
SEVERITY(MED) INTERVAL(001:00) DATE (20090707)
PARM('MAXUTILIZATION(60)')
REASON('Coupling facility memory should not be over utilized')
```

*Figure 34-26 Sample HZSPRMxx parmlib member*

Figure 34-27 on page 688 displays the XCF\_CF\_MEMORY\_UTILIZATION health check and shows sample output.

```

CHECK(IBMxcf,XCF_CF_MEMORY_UTILIZATION)
START TIME: 06/01/2010 12:30:02.765623
CHECK DATE: 20090706 CHECK SEVERITY: MEDIUM
CHECK PARM: MAXUTILIZATION(60%)

IXCH0455I Coupling facility memory utilization for all coupling
facilities in use by the local system are below the defined owner
maximum memory utilization for the check.

IXCH0914I This report summarizes the coupling facility memory
utilization for CHECK(XCF_CF_MEMORY_UTILIZATION).

CFNAME                = The coupling facility name.

MEMORY UTILIZATION    = The current memory utilization for the
                        coupling facility. Current memory utilization
                        is determined by the amount of space
                        allocated for structures in the coupling
                        facility and the amount of space reserved for
                        dumping structures allocated in the coupling
                        facility.

TOTAL SPACE           = Total amount of storage available in a
                        coupling facility.

UTILIZATION EXCEEDED = Whether the coupling facility memory utilization
                        exceeds the defined maximum memory utilization
                        percentage for the check.

```

CFNAME	MEMORY UTILIZATION	TOTAL SPACE	UTILIZATION EXCEEDED
CF7B	3%	935M	NO
CF7A	26%	930M	NO

```

END TIME: 06/01/2010 12:30:02.772428 STATUS: SUCCESSFUL

```

Figure 34-27 XCF\_CF\_MEMORY\_UTILIZATION health check and sample output

### 34.3.3 XCF\_CDS\_MAXSYSTEM

This check provides a warning when a function CDS (any CDS other than the sysplex CDS) is formatted with a MAXSYSTEM value that is less than the MAXSYSTEM value associated with the primary sysplex CDS.

This is a sysplex-wide check (GLOBAL), which means that it runs on one system but reports on sysplex-wide values and practices. A global check shows up as disabled for all systems in the sysplex, except for the one where it is actually running.

The reason for this check is that it is recommended that each couple data set defined to the sysplex be formatted with a MAXSYSTEM value that is at least equal to the value defined in the primary sysplex CDS. If a function CDS has a smaller MAXSYSTEM value, then a system joining the sysplex with a higher slot number will not be able to use the function provided by that function CDS.

## User override of IBM values

Figure 34-28 shows keywords you can use to override check values on either a POLICY statement in the HZSPRMxx parmlib member or on a MODIFY command. This statement may be copied and modified to override the check defaults.

```
UPDATE CHECK(IBMxcf,XCF_CDS_MAXSYSTEM)
SEVERITY(MED) INTERVAL(ONETIME) DATE (20090707)
REASON('CDS MAXSYSTEM value across all CDS types should be at least equal to
the value in the primary sysplex CDS')
```

Figure 34-28 Sample HZSPRMxx parmlib member

Figure 34-29 displays the XCF\_CDS\_MAXSYSTEM health check and shows sample output.

```
CHECK(IBMxcf,XCF_CDS_MAXSYSTEM)
START TIME: 05/28/2010 07:29:43.047447
CHECK DATE: 20090725 CHECK SEVERITY: MEDIUM

IXCH0402I All function couple data sets were formatted with a MAXSYSTEM
value that is at least equal to the MAXSYSTEM value associated with the
primary sysplex couple data set.

The MAXSYSTEM value associated with the primary sysplex couple data set
is 4.

END TIME: 05/28/2010 07:29:43.055945 STATUS: SUCCESSFUL
```

Figure 34-29 XCF\_CDS\_MAXSYSTEM health check and sample output

### 34.3.4 XCF\_SFM\_CFSTRHANGTIME

This check monitors the current setting of the CFSTRHANGTIME value in the sysplex failure management (SFM) policy to make sure it has not changed from the setting most desirable for your installation. It does this by comparing the SFM CFSTRHANGTIME value with the check parameter CFSTRHANGTIME value that you specify, issuing an exception message if the two are not consistent. The CFSTRHANGTIME SFM parameter specifies the amount of time you are willing to wait before the system takes automatic action to relieve hangs caused when a connector fails to respond to structure-related events in a timely manner.

The reason for this check is that when the CFSTRHANGTIME parameter is specified in the policy, the system automatically takes action to relieve hangs in CF structure-related processes caused by a connector's failure to respond to structure-related events in a timely manner. The CFSTRHANGTIME SFM policy attribute applies to any system using SFM, whether or not SFM is in use throughout the sysplex. This check applies to z/OS V1R12.

The following parameter is accepted:

```
PARM('CFSTRHANGTIME(NO | seconds)'):
```

- NO** Indicates that the system should not take automatic action to relieve a hang in a structure-related process.
- seconds** Specifies the time interval, in seconds, that a Coupling Facility structure connector can remain unresponsive before the system takes action to relieve a hang in a structure-related process.

**Default:** 300 is the default value for this check parameter.

This check generates an exception if the check parameter you specify is not consistent with the existing SFM policy CFSTRHANGTIME parameter. For instance, the check generates an exception if:

- ▶ The SFM policy specifies or defaults to CFSTRHANGTIME(NO), but the check parameter specifies a value other than NO.
- ▶ The SFM policy specifies a CFSTRHANGTIME decimal number of seconds, but the check parameter is either NO or a value less than the value specified in the SFM policy.

The CFSTRHANGTIME SFM policy attribute applies to any system using SFM, whether or not SFM is in use throughout the sysplex. The CFSTRHANGTIME setting must be NO or a decimal value between 0 and 1800 seconds, inclusive.

**Note:** XCF\_SFM\_CFSTRHANGTIME is a local system health check that monitors the current setting of the CFSTRHANGTIME value in the sysplex failure management (SFM) policy to make sure it has not changed from the recommended setting.

Figure 34-30 displays the XCF\_SFM\_CFSTRHANGTIME health check and shows sample output.

```
CHECK(IBMxcf,XCF_SFM_CFSTRHANGTIME)
START TIME: 06/01/2010 11:29:41.499455
CHECK DATE: 20090701 CHECK SEVERITY: MEDIUM
CHECK PARM: CFSTRHANGTIME(300)

* Medium Severity Exception *

IXCH0531E The Sysplex Failure Management (SFM) policy specification
for the time a connector response to a coupling facility (CF)
structure-related event is allowed to remain overdue is not
consistent with the owner recommendation.

END TIME: 06/01/2010 11:29:41.522947 STATUS: EXCEPTION-MED
```

Figure 34-30 XCF\_SFM\_CFSTRHANGTIME and sample output

### 34.3.5 XCF\_CFRM\_MSGBASED

The IBM-recommended event processing protocol is message-based (MSGBASED) rather than policy-based (POLBASED). XCF\_CFRM\_MSGBASED is a global sysplex health check to ensure that CFRM is enabled to use the specified event processing protocol when a structure is defined in the CFRM active policy in a multisystem-capable sysplex.

CFRM message-based event management improves recovery time for users of CF structures. The CFRM message-based event management protocol was introduced in z/OS V1R8 and can be used for CFRM event management of structures other than XCF signaling.

This check is to see if CFRM is enabled to use the specified event processing protocol when a structure is defined in the CFRM active policy in a multisystem-capable sysplex. This is a sysplex-wide check (GLOBAL), which means that it runs on one system but reports on sysplex-wide values and practices. A global check shows up as disabled for all systems in the sysplex, except for the one where it is actually running.



**Note:** IBM suggests that, once all systems are at z/OS V1R8 and there is no intention of falling back to a lower level of z/OS, message-based event processing should be enabled because of the large performance, availability, and scalability benefits that it can provide in some Parallel Sysplex environments.

## Parameters

The parameters that apply are:

**MSGBASED** Specifies that the check should issue an exception if the sysplex is not using message-based CFRM processing.

**POLBASED** Specifies that the check should issue an exception if the sysplex is not using policy-based CFRM processing.

**Default:** MSGBASED

Figure 34-31 displays the XCF\_CFRM\_MSGBASED health check and shows sample output.

```
CHECK(IBMxcf,XCF_CFRM_MSGBASED)
START TIME: 06/01/2010 11:29:45.845769
CHECK DATE: 20090707 CHECK SEVERITY: MEDIUM
CHECK PARM: MSGBASED

IXCH0254I The CFRM structure event management protocol is message-based.
This is consistent with the owner specification.

END TIME: 06/01/2010 11:29:46.046903 STATUS: SUCCESSFUL
```

Figure 34-31 XCF\_CFRM\_MSGBASED health check and sample output

## 34.3.6 XCF\_CF\_STR\_POLICYSIZE

This check makes sure that structures in the CFRM active policy do not have too large a difference between the value specified for INITSIZE and the value specified for SIZE. All specifications of INITSIZE in the active or pending CFRM policy should indicate an initial structure size of at least half the maximum structure size (as determined by the SIZE specification). The policy should not specify an initial structure size less than the maximum structure size when altering of the structure size is not supported (as determined by this check).

This is a sysplex-wide check (GLOBAL), which means that it runs on one system but reports on sysplex-wide values and practices. A global check shows up as disabled for all systems in the sysplex, except for the one where it is actually running.

### Implementation considerations

When you specify different INITSIZE and SIZE values, this provides flexibility to dynamically expand the size of a structure for workload changes, but too large a difference between INITSIZE and SIZE may waste Coupling Facility space or prevent structure allocation.

When allocating the structure initially, whether INITSIZE is specified or not, the system attempts to build all control structures that will be required to support the maximum size of the structure. These control structures are built in the control storage allocation of the structure. For structures whose users do not allow structure alter, the control storage allocated to

accommodate larger sizes is wasted. An INITSIZE value substantially smaller than the SIZE value might cause the following:

- ▶ It might be impossible to allocate a structure at a size of INITSIZE, because the amount of control storage required to support the SIZE value might actually be larger than INITSIZE.
- ▶ If the allocation succeeds, it might result in a structure with a proportionally large amount of its storage allotted to structure controls, leaving too few structure objects to be exploited usefully by the associated application.

**Suggestion:** IBM suggests that the INITSIZE and SIZE specification for structures be determined by the CfSizer (Coupling Facility Structure Sizer) tool:

<http://www.ibm.com/systems/support/z/cfsizer/>

Any INITSIZE specification for a structure should be at least half of the SIZE specification. If structure alter is not allowed by users of a structure, INITSIZE should not be specified for that structure.

Figure 34-32 displays the XCF\_CF\_STR\_POLICYSIZE health check and shows sample output.

```
CHECK(IBMxcf,XCF_CF_STR_POLICYSIZE)
START TIME: 06/01/2010 11:29:45.845665
CHECK DATE: 20090707  CHECK SEVERITY: MEDIUM

IXCH0256I No CFRM policy structure specification has too large a
difference between the INITSIZE and SIZE values.

END TIME: 06/01/2010 11:29:46.012811  STATUS: SUCCESSFUL
```

Figure 34-32 XCF\_CF\_STR\_POLICYSIZE health check and sample output

### VERBOSE support

This check has VERBOSE support. All structures in the CFRM active policy are included in an IXCH0923I report when the check is run in verbose mode. A check can be put into verbose mode using the following methods:

- ▶ Specify the UPDATE, filters, VERBOSE=YES parameter either on the MODIFY command or in a POLICY statement in an HZSPRMxx parmlib member.
- ▶ Overwrite the NO value with the YES value in the VERBOSE column of the SDSF CK command display.

**Note:** When VERBOSE=NO is specified, only structures that have SIZE too large compared to INITSIZE are listed in the IXCH0923I report. When VERBOSE=YES is specified, all defined structures are listed in the IXCH0923 report.

## 34.4 CF Sublist Notification enhancements

The CFRM policy information describes the Coupling Facilities (CF) and structures (STRUCTURE) that can be used in the sysplex once an administrative policy is activated.

The SETXCF START,POLICY,TYPE=CFRM command is used to activate an administrative policy. If no policy is active for CFRM, then the specified administrative policy is activated immediately. Otherwise a transition to the newly activated policy is required. For an allocated structure, some subparameters can take effect immediately. STRUCTURE subparameters which take effect immediately include:

- ▶ ALLOWAUTOALT
- ▶ FULLTHRESHOLD
- ▶ REBUILDPERCENT
- ▶ DUPLEX
- ▶ ALLOWREALLOCATE
- ▶ SITE
- ▶ SUBNOTIFYDELAY - (New with z/OS V1R12)

### **SUBNOTIFYDELAY subparameter in z/OS V1R12**

This Sublist Notification Delay support was provided in z/OS V1R10 and requires CFLevel 16 for CF Keyed List Structures. The sublist notification delay time was set to 5 milliseconds and there was no support to modify this function or to disable this function. The sublist monitoring delay function would automatically apply to any CF List Structure allocated with Primary Keys with monitoring connectors.

New function in z/OS V1R12 provides enhancements to the current Sublist Notification Delay function to allow you to explicitly specify and modify the sublist notification delay time.

This new function will benefit and be of general interest to Parallel Sysplex clients, particularly those who make use of the existing sublist notification function for CF Keyed List Structures. Exploiters of sublist monitoring, particularly those that use a *hot standby* configuration can transparently reduce their scheduling overhead with the enhanced sublist notification mechanism.

### **Using SUBNOTIFYDELAY (delaytime)**

The Sublist Notification Delay function enhancements support modification of the delay time value. A new CFRM policy parameter on existing STRUCTURE definition statements will allow the sublist notification delay time to be specified in microseconds:

```
SUBNOTIFYDELAY(delaytime)
```

This parameter specifies the number of microseconds for sublist notification delay time (SLND time). This value refers to delay between the time when a single selected shared message queue exploiter instance is notified of sublist transition and the time when the other instances are notified. It can be that the other instances are never notified, depending on the processing done by the initial exploiter. See *z/OS MVS Programming: Sysplex Services Guide*, SA22-7617 for an explanation of the sublist monitoring function.

The value specified for SUBNOTIFYDELAY can be 1 to 7 decimal digits in a range of 0 to 1,000,000 microseconds. SUBNOTIFYDELAY only applies to structures meeting certain criteria and will be ignored for all other structures.

The structure criteria for the SUBNOTIFYDELAY parameter are:

- ▶ The structure must be allocated as a keyed list structure.
- ▶ The structure must have EMCs allocated.
- ▶ The structure must be allocated in a CFLEVEL 16 or higher.

SUBNOTIFDELAY takes effect immediately with policy activation. The SUBNOTIFYDELAY value cannot exceed the CF model-dependent Subsidiary List Notification Delay Limit (SLNDL) value determined by the CF where the structure is allocated. Therefore, the SUBNOTIFYDELAY value that will take effect will be the smaller of the SUBNOTIFYDELAY value and the CF-dependent SLNDL value.

The delay time will take effect immediately at policy activation time. The default sublist notification delay time will remain 5 milliseconds (set to 5,000 microseconds) when SUBNOTIFYDELAY is not specified.

### Using the Administrative Data Utility

For an allocated structure, some subparameters can take effect immediately. STRUCTURE subparameters that take effect immediately include:

- ▶ ALLOWAUTOALT
- ▶ DUPLEX
- ▶ ALLOWREALLOCATE
- ▶ SUBNOTIFYDELAY

SUBLISTNOTIFYDELAY only applies to the following structures and will be ignored for all others:

- ▶ Keyed List Structures
- ▶ Monitoring established by connectors
- ▶ Allocated in CFLevel 16 CF

Figure 34-33 shows additional information for sublist notification delay time displayed by the D XCF command.

```
IXC360I hh.mm.ss  DISPLAY XCF
.
POLICY INFORMATION:
    POLICY SIZE:      policysize
.
    SUBNOTIFYDELAY:  subnotifdelay
ACTUAL SUBNOTIFYDELAY: actualsubnotifdelay
```

Figure 34-33 Additional information via D XCF

In most cases the actual and the policy SUBNOTIFYDELAY value will be the same. They may differ if the DISPLAY XCF command is issued after a policy change has processed but before the system has updated the SUBNOTIFYDELAY value in the CF.

### Hardware and software dependencies

This new function requires CFlevel 16 and will be rolled down to z/OS V1R19, V1R10, and V1R11 via APAR OA30994.

### Migration and coexistence considerations

To benefit from this enhancement, the sysplex environment needs to meet the following conditions:

- ▶ The monitored list structure is allocated in a Coupling Facility at CFLEVEL16.

- ▶ The z/OS system support for Sublist Monitoring has been installed on the particular system in the sysplex.

For full benefit, all z/OS systems in the sysplex need to support Sublist Monitoring, as follows:

- ▶ z/OS V1R12
- ▶ z/OS V1R9 – z/OS V1R11 with PTFs for OA30994 installed

### **Installation considerations**

The PTFs for APAR OA30994 must be installed on all systems in the sysplex for this support to be fully effective.

The CFRM CDS must be formatted with the new SUBNOTIFYDELAY keyword to specify the number of microseconds for the sublist notification delay time value using administrative data utility IXCMIAPU.

The CFRM Policy should be activated via the SETXCF command. The sublist notification delay time value will take effect immediately with policy activation.

## **34.5 Mean Time To Recovery (MTTR) for XCF**

XCF initialization during system IPL or recovery contained delays to ensure successful joining of systems and serialization of cross-sysplex resources. These delays were deemed inefficient and increased sysplex recovery time by many seconds per system.

New function in z/OS V1R12 results in a reduction of delays and repetitive processing to decrease XCF initialization duration and overall system MTTR.

- ▶ Joining system initialization requires coordinating with existing systems in the sysplex. The use of interval polling instead of a hard coded delay reduced IPL time by an average of 4.5 seconds.
- ▶ Eliminating redundant CDS XCF Group Record writes showed measurable MTTR improvements

## **34.6 Partitioning action by GDPS controlling system**

New function was added via z/OS V1R11 to allow a GDPS controlling system (K-SYS) to continue running after the loss of the sysplex time source. This new function allows the K-SYS to continue running in LOCAL timing mode without issuing IEA015A or IEA394A for up to 80 minutes and requires:

- ▶ GDPS V3R6
- ▶ z/OS V1R11 or
- ▶ z/OS V1R9 or z/OS V1R10 with:
  - – APAR OA28323 for supervisor timer
  - – APAR OA26085 for XCF

When there is a temporary failure such as a momentary link failure that causes a loss of time source, the production systems will issue the loss of time source WTOR - IEA015A for ETR or IEA394A for STP. With this new function the K-SYS will continue running in LOCAL timing mode.

### **New function APAR OA32236**

In the case of a momentary failure of the time source, the K-SYS will detect that the server is again synchronized and will transition back to ETR or STP timing mode while the production systems wait for a reply to IEA015A or IEA394A.

The problem now is that the production systems are not updating their heart beat and the K-SYS will take partitioning action after the sysplex failure detection interval (FDI) expires. This is not desirable because the production systems would continue running if the reply of RETRY was given to IEA015A or IEA394A in time.

APAR OA32236 introduces new function that prevents the GDPS controlling system from taking partitioning action after it switches to local timing mode for 80 minutes. This does not require the GDPS K-SYS to stay in local timing mode. The 80-minute period starts when the GDPS K-SYS switches to local timing mode and stays in effect even if the GDPS K-SYS transitions back to ETR or STP timing mode.



## z/OS HCD V1R12

The process of defining and selecting an I/O configuration is through the Hardware Configuration Definition (HCD) element of z/OS. It also introduces the concept of dynamic I/O configuration, which is a means of selecting or changing the I/O configuration on a running system.

This chapter describes enhancements provided in z/OS V1R12 for managing an I/O configuration. Most of them are provided through a new level of HCD, HCD V1R12.

The following topics are covered:

- ▶ Enhancements to HCD
  - I/O autoconfiguration
  - Automatic generation of D/R configuration
  - Support of over-defined CIB channels
  - Support of 3rd subchannel set
  - New hardware support
  - Enhancement of the CSS/OS device compare report
  - Free format IPL parameter
  - Messages handling
- ▶ New Vary CU command
- ▶ New Display IOS control unit group information.

## 35.1 Enhancements to hardware configuration definition

With z/OS V1R12, hardware configuration definition (HCD) has been enhanced with new functionalities as well as with the support of new hardware.

In particular, as shown in Figure 35-1, as compared with z10, the z196 system is enhanced as follows:

- ▶ It supports an additional subchannel set.
- ▶ The maximum number of ESCON channel paths has been decreased.
- ▶ CBP and FCV channel paths are no longer supported.
- ▶ The maximum number of CIB channel paths has been increased to 128,
- ▶ The maximum number of IQD channel paths has been increased from 16 to 32.

Processor type	2817
Models	M15, M32, M49, M66, M80
Number of CSSs	4
Number of LPs	60
Subchannel sets	SS 0, SS 1, SS 2
Number of ESCON CHPIDs (CNC, CTC, CVC, CBY)	360
Number of FC/FCP/FCV CHPIDs	336/256/-
Number of CIB/CFP/CBP/ICP CHPIDs	128/48/-/32
Number of OSA CHPIDs (OSD, OSE, OSC, OSN, OSM, OSX)	48
Number of IQD CHIDs	32

Figure 35-1 z196 configuration

### 35.1.1 I/O autoconfiguration

With the I/O autoconfiguration function, HCD discovers undefined FICON storage devices (DASD and tape) connected to the processor via a switch.

According to user-defined policies or to HCD/HCM provided defaults, HCM can automatically define the control units and devices of discovered controllers. You can either accept the proposed definitions without changes, or you can update the proposed definitions before committing them to your specified target work IODF.

Accordingly, HCD provides:

- ▶ A dialog to define autoconfiguration policies
- ▶ A dialog to perform the discovery and definition process

This function is only available on z196 processors.



With the I/O autoconfiguration function, you can use HCD to discover undefined FICON storage devices (DASD and tape) on a switch and to automatically define them in a work IODF that is based on the active IODF.

In a new dialog, invoked from the Define, Modify, or View Configuration Data selection, HCD shows the discovered controllers and proposes definitions for control units and devices of the selected controllers, based on user-controlled policies. You can either accept the proposed definitions or perform updates to them.

User-controlled autoconfiguration policies can be managed using the Edit profile options and policies dialog.

The existing “Edit profile options” dialog is enhanced to allow for the setting and modification of autoconfiguration policies. These policies are a set of new profile options used to tailor autoconfiguration discovery and propose processing.

In addition, the “Edit profile options and policies” dialog is the entry to the definition dialogs for two new types of IODF objects:

- ▶ Autoconfiguration LP groups, which are collections of logical partitions. During autoconfiguration, processing an LP group, selected by policy, defines the set of partitions to which discovered devices are autodefined.
- ▶ Autoconfiguration OS groups, which are collections of operating system configuration names. During autoconfiguring an OS group, selected by policy, determines which operating systems get access to autodefined devices.

### **35.1.2 Automatic generation of D/R configuration**

For exploitation in a GDPS environment with peer-to-peer remote copy (PPRC) controlled DASD devices, HCD provides a function to automatically generate the OS configuration of the disaster/recovery (D/R) site; see Figure 35-2 on page 700.

This function can be enabled by:

- ▶ Specifying the name of the D/R site OS configuration as an attribute of the primary site OS configuration.
- ▶ Indicating the DASD devices that are used by PPRC with a PPRC usage type.

The D/R site OS configuration is generated automatically during the Build production I/O definition file or the Build validated work I/O definition file processing as a copy of its primary site OS configuration, whereby DASD devices, which are used in PPRC connections, are defined to the D/R site OS configuration with a reversed OFFLINE parameter.

For a given operating system configuration, HCD offers the possibility to specify the name of a disaster recovery (D/R) site OS configuration.

```

z/OS V1.12 HCD
Discovery and Autoconfiguration Options

Specify autoconfiguration options. Then, press Enter to start the
discovery process.

Autoconfiguration is based on 1 1. Active IODF
                               2. Currently accessed IODF

Show proposed definitions . . 1 1. Yes
                               2. No

Scope of discovery . . . . . 1 1. New controllers only
                               2. All controllers
                               3. Controller containing CU ____ +

Force full mode discovery . . 2 1. Yes
                               2. No

Target IODF name . . . . . 'SYS6.IODF31.WORK' +

F1=Help    F2=Split    F3=Exit    F4=Prompt    F5=Reset    F9=Swap
F12=Cancel

```

Figure 35-2 D/R autoconfiguration panel

In addition, it allows DASD devices to be classified as PPRC simplex, duplex, flashcopy, or utility devices.

At the time of the Build Production IODF or the Build validated work I/O definition file, the named D/R site OS configuration is generated as a copy of the reference OS configuration with the DASD device OFFLINE=YES parameter set to OFFLINE=NO and vice versa for each PPRC duplex device; see Figure 35-3.

```

z/OS V1.12 HCD
Autoconfiguration Policies
Row 1 of 9 More: >
Command ==> _____ Scroll ==> HALF

Edit or revise autoconfiguration policies.

HCD Profile :

Policy keyword      Value +
AUTO_MATCH_CU_DEVNUM  YES
AUTO_SS_ALTERNATE    1
AUTO_SS_DEVNUM_SCHEME PAIRING
AUTO_SUG_CU_RANGE     0001-FFFE
AUTO_SUG_DEV_RANGE    0001-FFFF
AUTO_SUG_DYN_CHPIDIS  6
AUTO_SUG_LPGROUP      _____
AUTO_SUG_OSGROUP      _____
AUTO_SUG_STAT_CHPIDIS 2
***** Bottom of data *****

F1=Help    F2=Split    F3=Exit    F4=Prompt    F5=Reset
F7=Backward F8=Forward  F9=Swap    F12=Cancel

```

Figure 35-3 Autoconfiguration policies

Different options for autoconfiguration can also be chosen from the HCD V1R12 panels, such as shown in Figure 35-4.

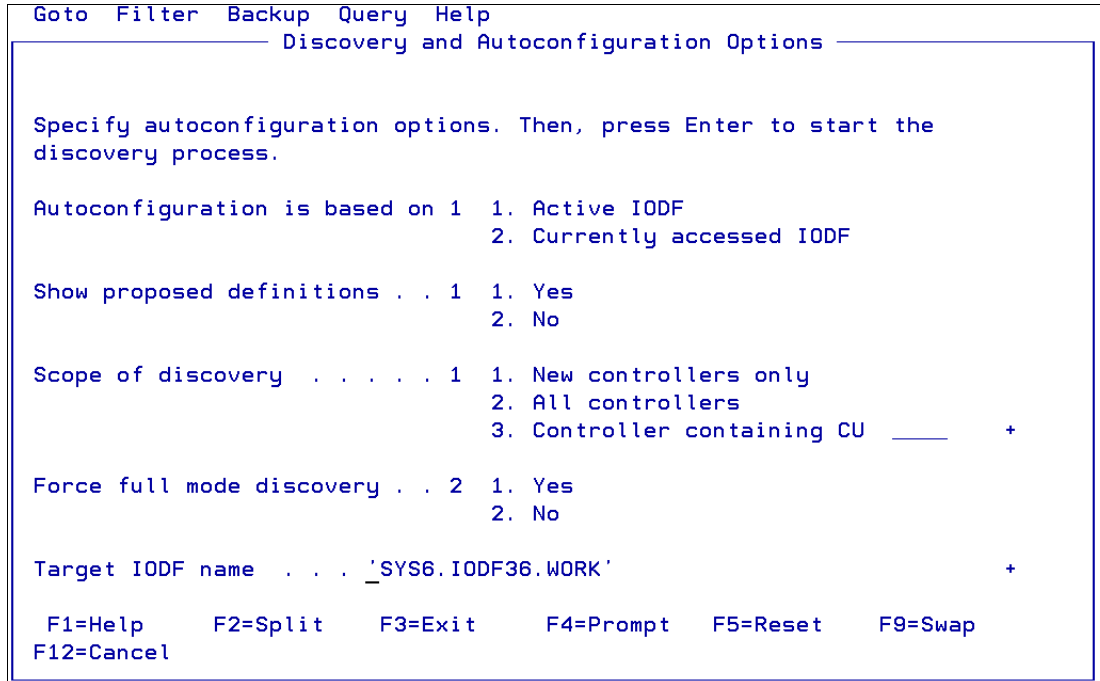


Figure 35-4 HCS V1R12 discovery and autoconfiguration options

In Figure 35-4, the options are as followed:

- ▶ Before starting the discovery process, a basing IODF must be identified. This IODF is copied to the selected target IODF and receives all changes made to the configuration during autoconfiguration processing.
  - With this option you can choose whether the currently accessed IODF or the active production IODF should be taken as base for new configuration definitions resulting from the discovery process.
  - Initially the active production IODF is selected as base. The option setting is saved across invocation of the dialog.
  - The active IODF can only be taken as base if it is fully dynamic capable.
- ▶ Specify YES (which is the initial setting) in the “Show proposed definitions” option if the dialog should display proposed definitions for possible configuration changes, thereby allowing for interactions and user-added changes.
  - Change the option to NO if you want to run the fast-path process of I/O autoconfiguration. It does not offer a possibility to revise proposals or to interact until the definitions are complete and saved in the IODF.
  - The option setting is saved across invocation of the dialog.
- ▶ The third selection offers the following choices:
  - All controllers
    - This indicates that a discovery is done for new as well as for changed controllers. Essentially, the definition proposal is done for all discovered new or changed controllers.

- New controllers only  
This indicates that discovery should be performed for new controllers only. Essentially, the definition proposal is done for the discovered new controllers only.
- Controller containing CU
  - This indicates that discovery should be performed on the controller containing the control unit with the specified number. The referenced control unit must be a DASD or TAPE control unit and must be defined in the IODF.
  - The control unit number is required input, if **Controller containing CU** is selected.
- ▶ Regarding the 4th option, “Force full mode discovery” shown in Figure 35-4 on page 701:
  - If set to YES, for each discovered controller, all possible logical control unit addresses (CUADD values) and unit addresses are checked for changes. If set to NO (which is the initial default), processing stops with the first possible CUADD value or unit address that does not exist.
  - The option setting is saved across discovery dialog invocation and redisplayed with the next call to it.

### 35.1.3 Support of over-defined CIB channels on an XMP processor

InfiniBand (CIB) channels have been introduced in z/OS V1R10 for supporting the then new hardware provided for both internal and coupling links.

#### Defining the InfiniBand coupling link

As depicted in Figure 35-5 on page 703, the support InfiniBand coupling link is invoked as follows:

- ▶ Defining the new channel path type CIB (Coupling using IB)
- ▶ Specifying the Host Channel Adapter (HCA) ID and port number

Channel path type CIB has the following characteristics:

- ▶ CIB channel path can be DED, REC, SHR, or SPAN.
- ▶ Up to 16 CHPIDs per HCA2-O.
- ▶ Maximum of 16 CHPIDs per Adapter ID.
- ▶ 16 CHPIDs can be shared across the two ports of the HCA2-O.
- ▶ No PCHID value.
- ▶ Point-to-point connections only.
- ▶ The target server identified by CSYSTEM on the CHIPID statement.
- ▶ The local server identified by LSYSTEM on the ID statement.

The CIB CHPID type is being defined in the IOCP as:

- ▶ Dedicated (DED), Reconfigurable (REC), Shared (SHR), or Spanned (SPAN)
  - AID (Adapter ID)
    - z10 EC 00-1F
    - z9 EC 00-1F
    - z9 BC 08-0F
  - Maximum of 16 CIB CHPIDs per AID

- ▶ PORT: port on the HCA where the CHPID is defined
  - Two ports per adapter (1 and 2)
- ▶ CSYSTEM: Target system
- ▶ CPATH: CSS and CHPID on the target system

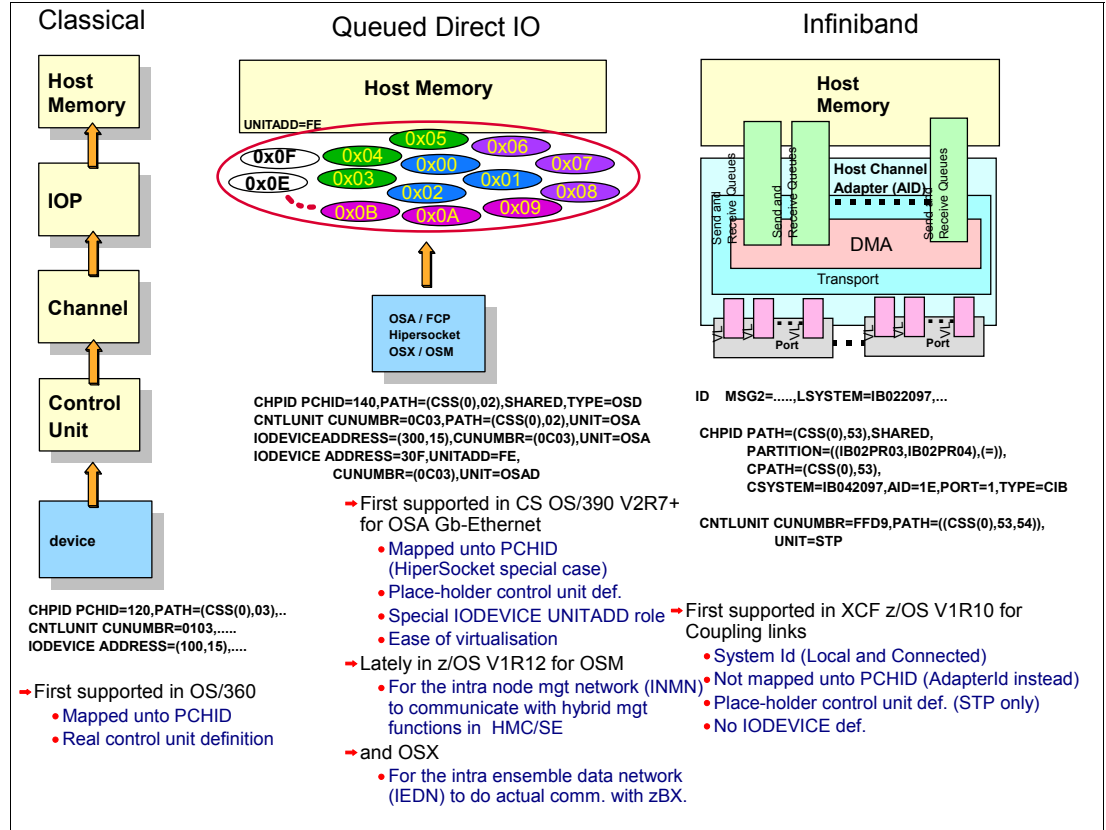


Figure 35-5 I/O definition comparison for different technologies

### Over-defining CIB channels

Over-defined CIB channel paths can be defined in order to avoid an outage of a standalone Coupling Facility when defining new CIB connections. An over-defined CIB path can be installed later and perform a dynamic activation of the configuration.

To distinguish an over-defined CHPID from a physically installed CHPID, a character \* should be used for the PCHID value or the HCA ID for CIB channels, when over-defining the CHPID. An over-defined CHPID must adhere to all validation rules.

When installing the channel path later, you must edit the CHPID and replace the \* with its valid PCHID or HCA ID.

Over-defined channel paths are not taken into account by an IOCDs download, by an IOCP build, and by a dynamic activation of an I/O configuration: if a control unit contains only CHPIDs with a PCHID value or HCA ID \*, the whole control unit (including any attached devices) is omitted from the configuration.

If a CHPID changed its PCHID or HCA ID from \* to a valid value during a dynamic activation, an Add CHPID request is generated. Correspondingly, if the PCHID or HCA ID is changed from a valid value to an \*, a Delete CHPID request is generated.

When building a CONFIGxx member, CHPIDs with a PCHID/HCA ID \* are skipped. Attached control units, including attached devices, are also omitted.

When copying a configuration or generating I/O configuration statements, channel path definitions with PCHID=\* and AID=\* are included.

When building a production IODF, HCD requires that the CIB channels be connected, even if they are over-defined. If the connection is within the same processor, a mix of over-defined and fully defined HCA IDs is not accepted. In this case, error message CBDG541I is issued and the production IODF is not built.

### 35.1.4 Support of third subchannel set

Starting with z196 processors (processor type 2817), you can define devices to a third subchannel set with ID 2 (SS 2). In this third subchannel set, you can configure a maximum of 64 K-1 devices.

Actually, the following can be defined into an alternate subchannel set, starting with:

▶ z9 EC processors

Each channel subsystem contains more than one subchannel set (SS 0, SS 1), where other devices can be placed. As mentioned in Figure 35-7 on page 706, devices in SS 0 can only be used for 4 x-digit device addresses. Alternate subchannel sets opened up the possibility to use 5 x-digit device addresses, but then the corresponding software has to be modified accordingly and the messages it issues, as well.

▶ z/OS V1R7 HCD

- PAV alias devices (types 3380A and 3390A) can be defined into an alternative subchannel set.
- In SS 0, 63.75 K (that is, 65,280) devices can be placed, while 64 K-1 (65,535) can be in SS 1.

▶ z/OS V1R10 HCD

- Parallel Access Volume (PAV) alias device types 3380A (3390A) of the 2105, 2107 and 1750 DASD control units
- PPRC secondary devices (3390D)
- DB2 data backup volumes (3390S)

Starting with z196 processors and z/OS V1R12, they can be defined to Subchannel Set (SS) 2.

The subchannel set ID for a device can be defined in field Subchannel Set ID in the panel shown in Figure 35-6 on page 705.

```

Goto Show/Hide Filter Backup Query Help
- Specify Subchannel Set ID -----
C                                     HALF
S Specify the ID of the subchannel set into which devices are
S placed, then press Enter.
C Subchannel Set ID 0 +
/                                     Available Subchannel Set IDs Row 1 of 3
/ Command ==>
/ F1=Help F2=Spl Select one.
- F12=Cancel
- Subchannel Set ID
- 0
- 1
- 2
- ***** Bottom of data *****
-
- F1=Help F2=Split F3=Exit
- F7=Backward F8=Forward F9=Swap
- F12=Cancel
-
- 1853 3390A
- 1854 3390A
- 1855 3390A
- 1856 3390A
- 1857 3390A
- 1858 3390A
- 1859 3390A
- 185A 3390A
- 185B 3390A
- 185C 3390A
- 185D 3390A
- 185E 3390A
- 185F 3390A
- 1860 3390A
- 1861 3390A
- F1=Help F2=Split
- F8=Forward F9=Swap

```

Figure 35-6 Subchannel Set ID for a device

**Note:** 3390D and 3390S devices cannot be defined in subchannel set SS 0.

HCD messages that refer to a device in a subchannel set with a subchannel set ID > 0 will display the device number in the format n-devnumber where n is the subchannel set ID.

For example, the device 1234 located in subchannel set 1 will show up as 1-1234. A device 4567 in subchannel set 0 will further on be shown as 4567.

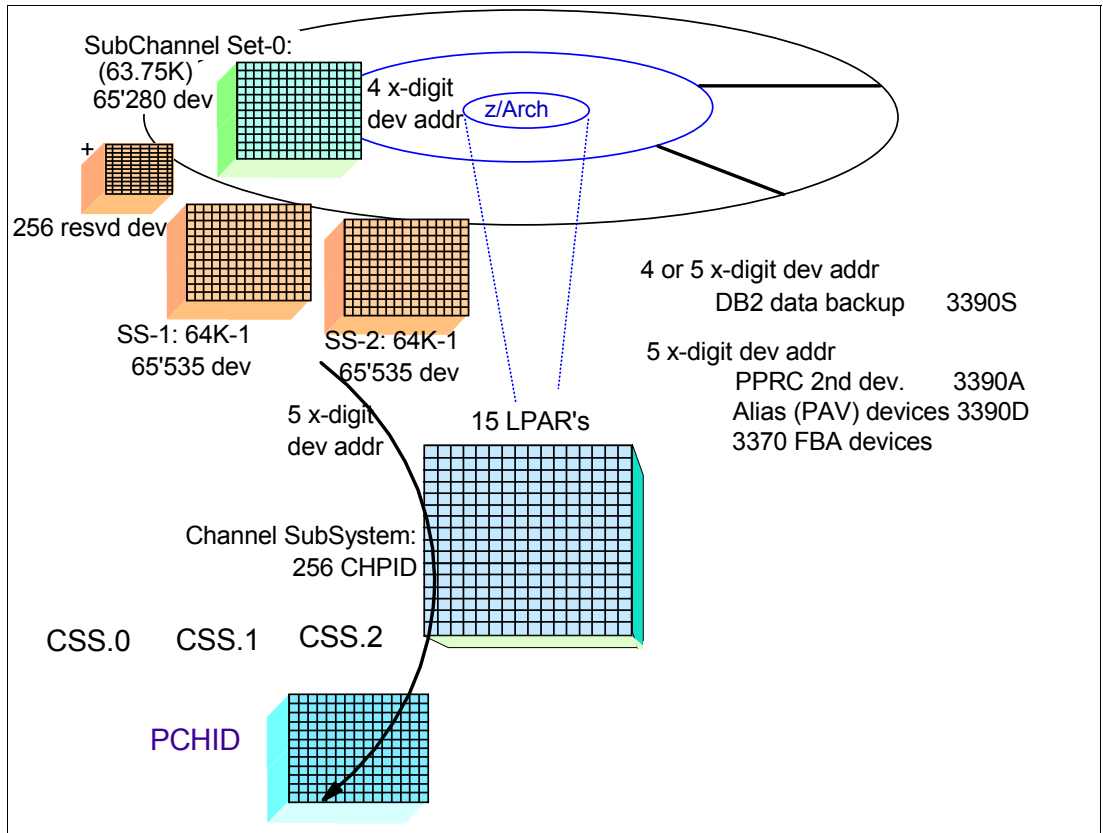


Figure 35-7 XMP Subchannel Set with ID 2 (SS 2)

### 35.1.5 New hardware support

With z/OS V1R12, HCD supports the IBM z196 processor family, as shown in Figure 35-8 on page 707.



```

CBDPM000          z/OS V1.12 HCD
C  Query Supported Hardware and Installed UIMs
  C BDPHW40
    Supported Processors
      C BDPSPR0          Row 293 of 586 More: >
      Command ==>      Scroll ==> 2

      Select one to view more details.

      Processor
      Type-Model Support Level
      2817-M15   XMP, 2817 support, SS 2
      # 2817-M15
      2817-M32   XMP, 2817 support, SS 2
      # 2817-M32
      2817-M49   XMP, 2817 support, SS 2
      # 2817-M49
      2817-M66   XMP, 2817 support, SS 2
      # 2817-M66
      2817-M80   XMP, 2817 support, SS 2
      # 2817-M80
      3000-A10   Parallel, ESCON (CBY), OSA, ISD channels
      F1=Help    F2=Split    F3=Exit    F7=Backward  F8=Forward
      F9=Swap    F12=Cancel
  
```

Figure 35-8 New hardware support

The new processor definition can be viewed in HCD V1R12 panels such as the one shown in Figure 35-9.

```

Goto Filter Backup Query Help
View Processor Definition
-
Processor ID . . . . . : SCZP301
Support level:
XMP, 2817 support, SS 2

Processor type . . . . . : 2817
Processor model . . . . . : M32
Configuration mode . . . . . : LPAR

Serial number . . . . . : 0B3BD52817
Description . . . . . :
Number of Channel Subsystems : 4

Processor token . . . . . :
SNA address: Network name . . : USIBMSC
                  CPC name . . . : SCZP301

Local system name . . . . . : SCZP301

ENTER to continue.
F1=Help  F2=Split  F3=Exit  F9=Swap  F12=Cancel
  
```

Figure 35-9 HCD V1R12 view of new processor definition

## New channel path types

As depicted in Figure 35-11 on page 709, in order to support the internal communication networks that connect a z196 processor to a BladeCenter Extension (zBX), two new channel paths are introduced with z/OS V1R12 and can be defined with HCD V1R12:

- ▶ OSX for the intra ensemble data network (IEDN) that is used by the software to do the actual communication with the zBX
  - An OSX channel path is defined like an OSD CHPID, that is, it may have device priority queuing enabled or disabled.
  - It can be attached to an OSX control unit and OSA-X devices. The control unit and device add panels contain the required and defaulted attributes for OSX channel paths.
  - An OSAD device at unit address FE can be defined.
  - On an OSX channel path there are a maximum of 480 valid subchannels if device priority queuing is enabled, and a maximum of 1920 valid subchannels if device priority queuing is disabled.

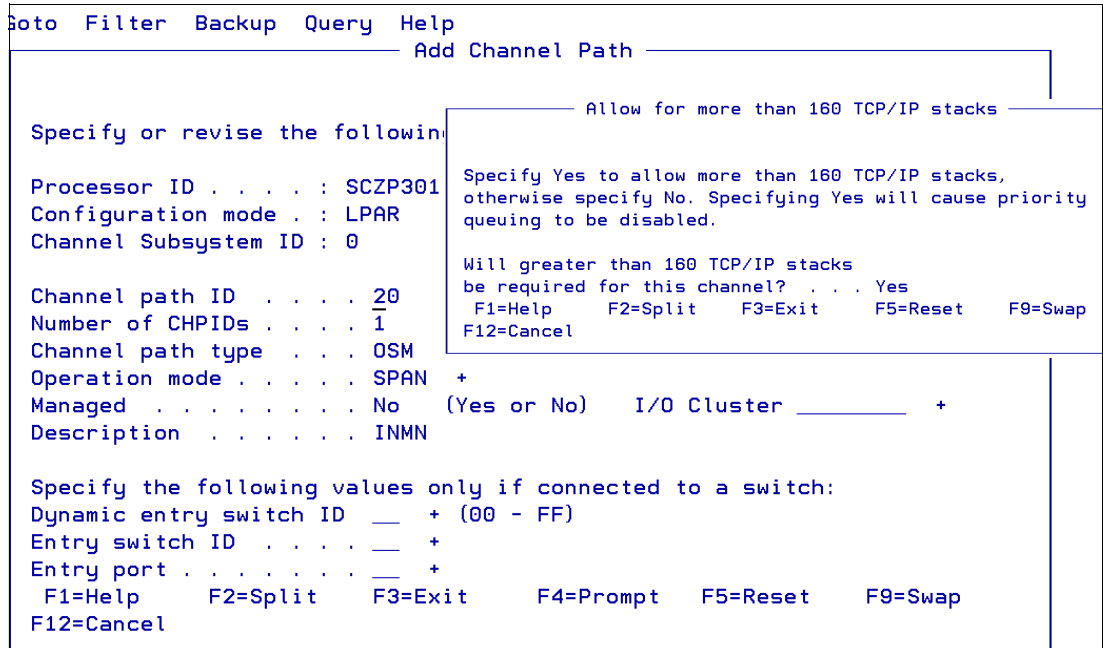


Figure 35-10 Adding an OSM channel path

- ▶ OSM for the intra node management network (INMN) that is used by management functions within the software to communicate with hybrid management functions in the HMC/SE.
  - As shown in Figure 35-10, an OSM channel path is defined like an OSD CHPID, except that device priority queuing must be disabled. The new OSA channel paths are defined in the normal HCD dialogs like other channel paths.
  - It can be attached to an OSM control unit and to OSA-M devices. The control unit and device add panels contain the required and defaulted attributes for OSM channel paths.
  - An OSAD device at unit address FE can be defined.
  - There is a maximum of 1920 valid subchannels on an OSM channel path.

From an HCD perspective, as depicted in Figure 35-5 on page 703, these new CHPID types follow the OSD rules. However, device priority queuing is always disabled for the management network.

### Defining more than 160 TCP/IP stacks

When defining or changing channel paths of type OSD, OSM, or OSX for processors with the corresponding support level, as shown in Figure 35-10, you can defined more than 160 TCP/IP stacks with this channel.

This choice, however, will disable priority queuing.

OSM channels must be defined to allow for more than 160 TCP/IP stacks.

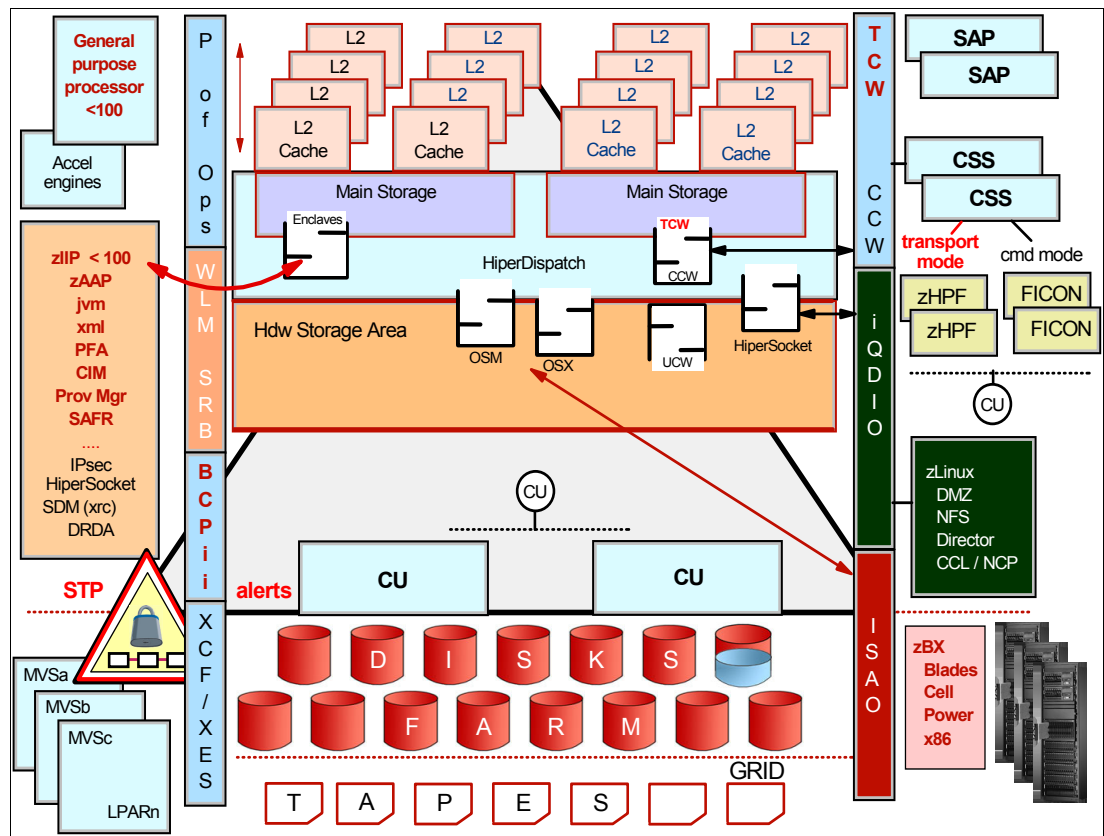


Figure 35-11 z196 hardware accelerators

### 35.1.6 Enhancement of the CSS/OS device compare report

As shown in Figure 35-12 on page 710, the CSS/OS device compare report has been enhanced in z/OS V1R12 to display the subchannel set ID for devices or device ranges which are located in a subchannel set with a subchannel set ID > 0.

OS Device Compare Report		TIME: 08:32 DATE: 1997-11-06 PAGE R - 1		
New IODF name: REDDE.IODF00.COMP1		Old IODF name: REDDE.IODF00.COMP2		
Limited to New Operating System Id: OS1		Old Operating System Id: OS1		
OS	Device, Range	New IODF	Old IODF	Description
OS1	0100	Actual Data	Old Data	
		9033	same	Device Type
		SWCH	same	Name of Generic
		Yes	same	Value(s) of Parameter OFFLINE
		Yes	same	Value(s) of Parameter DYNAMIC
OS1	0200,32	Actual Data	Old Data	
		3390	same	Device Type
		3390	same	Name of Generic
		No *	same	Value(s) of Parameter OFFLINE
		Yes	same	Value(s) of Parameter DYNAMIC
OS1	0101,4	Actual Data	Old Data	
		3390A	same	Device Type
		3390	same	Name of Generic
		0	1	Subchannel Set ID
		No	same	Value(s) of Parameter OFFLINE
		Yes	same	Value(s) of Parameter DYNAMIC
		...	...	...
SHARED	same	Feature		
...	...	...		

Figure 35-12 CSS/OS device compare report

### 35.1.7 Free format IPL parameter

With z/OS V1R12, an additional column is added to the IPL Attribute List, showing the parameters in an unformatted form. You can modify these unformatted values to specify non-z/OS IPL parameters, for example, for z/VM or Linux on IBM System z for the next IPL. The unformatted parameter values are checked according to the requirements for load parameters specified on the HMC.

### 35.1.8 Message handling

To distinguish the severity of warning messages when building the production IODF, HCD introduces a new severity classification S in the Message List (when working in the HCD dialog) or in the HCD message log (when working in batch processing).

This classification designates severe warning messages that you should carefully consider. Though these messages do not prevent the function to be terminated, the reason for these messages may lead to undesirable results.

## 35.2 Vary CU command

Recent technologies, such as subchannel sets and HyperPAV, have created scenarios where devices attached to a logical control unit may not reside in a consecutive device number range. With subchannel set capabilities on z9 processors, users were able to move aliases

into subchannel set 1, creating ranges of device numbers that were previously attached to one logical control unit to be attached to a different logical control unit. This has created holes in device ranges. With HyperPAVs, the number of required aliases can be reduced, allowing more base devices to be defined. Both of these enhancements have caused more complexity in operations when online and offline actions are required.

With z/OS V1R12, a new VARY command variant, VARY CU, simplifies the operational complexity by making the logical control unit number the point of control for vary device or vary path operations.

Using VARY CU, you can:

- ▶ Vary all devices attached to a control unit online or offline.
- ▶ Vary a path to all devices attached to a control unit online or offline.
- ▶ Control the AUTOSWITCH feature for all tape drives attached to a control unit.
- ▶ Allow or prevent allocation from using all offline tape devices attached to a control unit.

It thus simplifies operations in situations where the control unit is the point of control for device selection.

The new operator command uses a similar syntax as the VARY device and VARY path commands; see Figure 35-13.

One control unit number can be specified.

```

V CU {(cunumber)},{ONLINE[,UNCOND[,FORCE] | ,SHR | ,RESET]}
      {OFFLINE[,FORCE] }
      {AUTOSWITCH | AS [,ON | OFF] }
      {UNAVAILABLE | UNAVAIL} }
      {AVAILABLE | AVAIL} }
V CU {(cunumber,chipid)},{ONLINE[,FORCE] }
      {OFFLINE[,UNCOND | ,FORCE]}

```

Figure 35-13 VARY CU syntax

### VARY command resources

Two new resources are introduced to control authorization for usage of the V CU command.

If Force for **vary cu(xxx),offline** (no path specified) is specified, control access to resource MVS.VARYFORCE.CU is required, otherwise, control access to the MVS.VARY.CU resource is required. If the security product cannot determine authorization, master authority is required.

### Implementation considerations

The possible usages are:

- ▶ Vary all devices attached to control unit 400 offline:
 

```
V CU(400),OFFLINE
```
- ▶ Vary one path to all devices attached to control unit 400 online to easily unbox secondary devices:
 

```
V CU(400,51),ONLINE
```

This might be useful to easily unbox PPRC secondaries in SCHSET 1. The VARY CU path command variant, like the **vary path** command, will operate on secondary devices. The

VARY CU device command variant, like the **vary device** command, operates only on current primary devices. If a secondary device is attached to the specified control unit, the VARY CU device command is terminated.

- ▶ Define all tape drives attached to control unit 500 as auto switchable:

```
V CU(500),AUTOSWITCH,ON
```

- ▶ Prevent all offline tape drives attached to control unit 500 from being eligible for use by Allocation:

```
V CU(500),UNAVAILABLE
```

Note that when options applicable only to tape devices are used on the VARY CU command, as in the last two examples, only tape devices attached to the control unit are included for command processing. If a CU that does not have tape devices attached to it is specified on the command, the V CU command will find no devices eligible for the vary.

### Complementary system commands

The display matrix device command shows the number of the control unit to which a device is attached. This is the number that would be specified on the Vary CU command; see Figure 35-14.

```
----- LIST MCS COMMAND OUTPUT ----- Row 1 to 10 of 10
C => _                                     S => HALF
Mode: BOTH   SC74   LAFITTE
Display => Time   N System Y Job   N Hold => N (Y or N)
                                     15:20 10/05/10 Enter '?' for HELP
-----
SC74   IEE174I 15.20.15 DISPLAY M 633
SC74   DEVICE 8000 STATUS=OFFLINE
SC74   CHP                                52  56  5A  5E
SC74   ENTRY LINK ADDRESS                2B  2B  27  27
SC74   DEST LINK ADDRESS                 08  0C  25  25
SC74   PATH ONLINE                       Y   Y   Y   Y
SC74   CHP PHYSICALLY ONLINE             Y   Y   Y   Y
SC74   PATH OPERATIONAL                   Y   Y   Y   Y
SC74   PATHS NOT VALIDATED
      D M=DEV(8000)
***** Bottom of data *****
```

Figure 35-14 D M=DEV shows the CU number

The display matrix control unit command shows the devices and paths attached to the specified control unit, as shown in Figure 35-15 on page 713.

```

----- LIST MCS COMMAND OUTPUT ----- Row 1 to 20 of 20
C =>
Mode: BOTH SC74 LAFITTE S => HALF
Display => Time N System Y Job N Hold => N (Y or N)
15:15 10/05/10 Enter '?' for HELP
-----
SC74 IEE174I 15.15.44 DISPLAY M 624
SC74 CONTROL UNIT 8000
SC74 CHP 52 56 5A 5E
SC74 ENTRY LINK ADDRESS 2B 2B 27 27
SC74 DEST LINK ADDRESS 08 0C 25 25
SC74 CHP PHYSICALLY ONLINE Y Y Y Y
SC74 PATH VALIDATED Y Y Y Y
SC74 MANAGED N N N N
SC74 ZHPF - CHPID Y Y Y Y
SC74 ZHPF - CU INTERFACE N N N N
SC74 MAXIMUM MANAGED CHPID(S) ALLOWED = 0
SC74 DESTINATION CU LOGICAL ADDRESS = 00
SC74 CU ND = 002105.800.IBM.13.000000022513.0004
SC74 CU NED = NOT AVAILABLE
SC74 TOKEN NED = 002105.000.IBM.13.000000022513.0000
SC74 DEFINED DEVICES
SC74 08000-0808E
SC74 DEFINED PAV ALIASES
SC74 0808F-080FF
D M=CU(8000)
***** Bottom of data *****

```

Figure 35-15 D M=CU shows devices attached to a control unit

The non-alias devices are the devices on which the VARY CU command would operate.

In the output, new message IEE902I will be written to the console, as shown in Figure 35-16. This message lists the devices on which the Vary CU command will attempt to operate. Following this new message, you will see the usual messages that would be listed for a vary device or vary path command.

```

IEE902I VARY CU(400,11),OFFLINE: ACCEPTED
      DEVICE LIST: 00400,00410-00417

```

Figure 35-16 Message IEE902I

### 35.3 Displaying IOS control unit group information

z/OS V1R12 introduces the new DISPLAY IOS,CUGRP command to display information about one or more IOS control unit groups. A control unit group consists of one or more control units that share the same set of static CHPIDs. As is depicted in Figure 35-17 on page 714, each control unit group is assigned a group identifier that is the same as the lowest numbered control unit in the group.

When dynamic channel-path management (DCM) makes a change, it makes a change to all managed control units in a control unit group at the same time.

```

----- LIST MCS COMMAND OUTPUT ----- Row 1 to 13 of 13
C =>
Mode: BOTH SC74 LAFITTE
Display => Time N System Y Job N Hold => N (Y or N)
15:07 10/05/10 Enter '?' for HELP
-----
SC74 IOS632I 15.07.49 CU GROUP DATA 603
SC74 GROUP STATIC CHPIDS MAX #CUS DCM
SC74 6000 52 56 5A 5E 0 4 N
SC74 6100 53 57 5B 5F 0 4 N
SC74 8000 52 56 5A 5E 0 4 N
SC74 8100 53 57 5B 5F 0 4 N
SC74 C400 50 54 5A 5E 0 2 N
SC74 C500 51 55 5B 5F 0 2 N
SC74 C800 52 56 5A 5E 0 4 N
SC74 C900 53 57 5B 5F 0 4 N
SC74 D000 50 54 58 5C 2 7 Y
SC74 D100 51 55 59 5D 2 7 Y
D IOS,CUGRP
***** Bottom of data *****

```

Figure 35-17 D IOS,CUGRP





## Unicode enhancements

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 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—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 enhancements to UNICODE with z/OS V1R12:

Unicode on Demand

## 36.1 Unicode on Demand

Prior to z/OS V1R7, the only way to load conversions into the Unicode environment was via images. Customers were responsible for creating their own images in order to populate the Unicode environment.

In z/OS V1R6, Unicode Services provided enhancements to facilitate customer exploitation of DB2 Version 8 by making Unicode easier to use and configure. The following changes were made in z/OS 1.6 (and rolled back to previous releases):

- ▶ Providing a prebuilt image with all the DB2-required codepage conversion tables. The prebuilt image is built as a binary file (instead of a FB 80 file) and shipped with z/OS. The size of the prebuilt image is about 40 MB.
- ▶ Implementing a mechanism that automatically loads this prebuilt image into the Unicode environment for DB2. The prebuilt image is loaded automatically when the caller of Unicode Services runs in Key 7 (presumably DB2) and the Unicode environment is empty.

Starting in z/OS V1R7, the Unicode Services environment can be dynamically updated when a conversion service is requested. If the appropriate table needed for the service is not already loaded into storage, Unicode Services will load the table without requiring an IPL or disrupting the caller's request.

In V1R12 of z/OS, the prebuilt image CUNIDHC2 is eliminated from the Base for the following reasons:

- ▶ Unicode on Demand eliminates the need to have a pre-loaded image.
- ▶ Reduce the storage footprint.
- ▶ Improve serviceability.

**Note:** IBM recommends that you use the Unicode on Demand capability to load all tables.

### Installation considerations

Unicode Services will no longer ship the prebuilt image SYS1.SCUNIMG(CUNIDHC2) and no longer automatically load the prebuilt image. For equivalent support, you can use the conversion statements in SYS1.SAMPLIB(CUNSISM6) to create a regular image and specify the image name in CUNUNIxx parmlib member. The regular image can be loaded into the system during IPL or by using the SETUNI or SET UNI commands as currently supported.

## 36.2 Set up a Unicode image

You can use the SET UNI system command to dynamically set the Unicode environment from the console. Use the SETUNI command to add, delete, and replace tables in storage. Furthermore, you can set the UNI= parameter in your IEASYSxx parmlib member to set the Unicode system default image. In Figure 36-1 on page 717 you see console output from a SET UNI system command to activate an entire Unicode environment.

```

SET UNI=LK
IEE252I MEMBER CUNUNILK FOUND IN SYS1.PARMLIB
CUN3007I REALSTORAGE IS NOW DEFINED AS    12800 PAGES<    12800 PAGES>
CUN2020I START LOADING CONVERSION IMAGE CUNIMGT4
IEE252I MEMBER CUNIMGT4 FOUND IN SYS1.PARMLIB
IEE252I MEMBER CUNIMGT4 FOUND IN SYS1.PARMLIB
IEF196I IEF285I    SYS1.PARMLIB                                KEPT
IEF196I IEF285I    VOL SER NOS= BH8CAT.
IEF196I IEF285I    CPAC.ZOSR1B.PARMLIB                        KEPT
IEF196I IEF285I    VOL SER NOS= Z1BCAT.
IEF196I IEF285I    SYS1.IBM.PARMLIB                            KEPT
IEF196I IEF285I    VOL SER NOS= Z1CRC1.
CUN2022I LOADING CONVERSION IMAGE CUNIMGT4 FINISHED: 1490744 BYTES
LOADED
IEE536I UNI        VALUE LK NOW IN EFFECT

```

Figure 36-1 Activating a Unicode image

In Figure 36-2 we dynamically add a Unicode image to the system. The SETUNI system command can be used to define, delete, or replace Unicode images in your system.

```

SETUNI ADD, IMAGE (CUNIMGT4) , DSN(SYS1.PARMLIB)
CUN2020I START LOADING CONVERSION IMAGE CUNIMGT4
IEF196I IEF237I D24D ALLOCATED TO SYS00011
IEE252I MEMBER CUNIMGT4 FOUND IN SYS1.PARMLIB
IEE252I MEMBER CUNIMGT4 FOUND IN SYS1.PARMLIB
IEF196I IEF285I    SYS1.PARMLIB                                KEPT
IEF196I IEF285I    VOL SER NOS= BH8CAT.
CUN2022I LOADING CONVERSION IMAGE CUNIMGT4 FINISHED: 1490744 BYTES
LOADED
CUN3006I SETUNI COMMAND WAS SUCCESSFULLY EXECUTED

```

Figure 36-2 Activating a single Unicode table

## 36.3 Migration considerations

Remove any reference to Unicode Services' prebuilt image CUNIDHC2 on any point in your System. Delete any reference to data set SYS1.SCUNIMG because it has been removed from z/OS V1R12.

- ▶ Remove references to this obsolete data set from PARMLIB.
- ▶ Remove SYS1.SCUNIMG from LNKST and APF authorization lists.

After successful migration to z/OS V1R12, remove the following DDDEFs from your SMP/E environment because they are no longer used. Table 36-1 shows the obsolete DDDEF entries.

Table 36-1 Obsolete DDDEFs

Zone	DDDEF	Default data set
DLIB	ACUNIMG	SYS1.ACUNIMG
TLIB	SCUNIMG	SYS1.SCUNIMG

## 36.4 Build conversation image

To create and load a Unicode conversion image similar to the eliminated prebuilt image, do the following:

- ▶ Invoke the image generator program using the JCL provided in SYS1.SCUNJCL(CUNJIUTL). Use the conversion statements specified in SYS1.SAMPLIB(CUNSISM6).
- ▶ After building the image, specify the image name in the CUNUNlxx parmlib member.
- ▶ Specify the corresponding UNI=xx parameter in your IEASYSy parmlib member and re-IPL.



# 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 via policy-based automation.

In this chapter we explain the latest enhancements for the Capacity Provisioning Manager (CPM):

- ▶ The Provisioning Manager report in CPCC
- ▶ Command correlation
- ▶ Average performance index
- ▶ CICS/IMS support

## 37.1 Provisioning Manager report in CPCC

In prior Releases of z/OS, the Provisioning Manager status could only be inspected using the Provisioning Manager reports on the host. The Control Center only displayed the Processing Mode, Active Policy, and Active Configuration, but no details. Starting with z/OS V1R12, the statuses of Domain, Active Policy, and Active Configuration are now displayed in a detailed form on the Control Center. Now it is not necessary to switch to the host to inspect the Provisioning Manager status.

When the Control Center is connected to a Provisioning Manager, you can do the following:

- ▶ Display domain status
- ▶ Display active policy status
- ▶ Display active configuration status
- ▶ Refresh status information

The status information is cached, so that if the connection is lost, the information can still be displayed. Figure 37-1 demonstrates the interactions between all connection states.

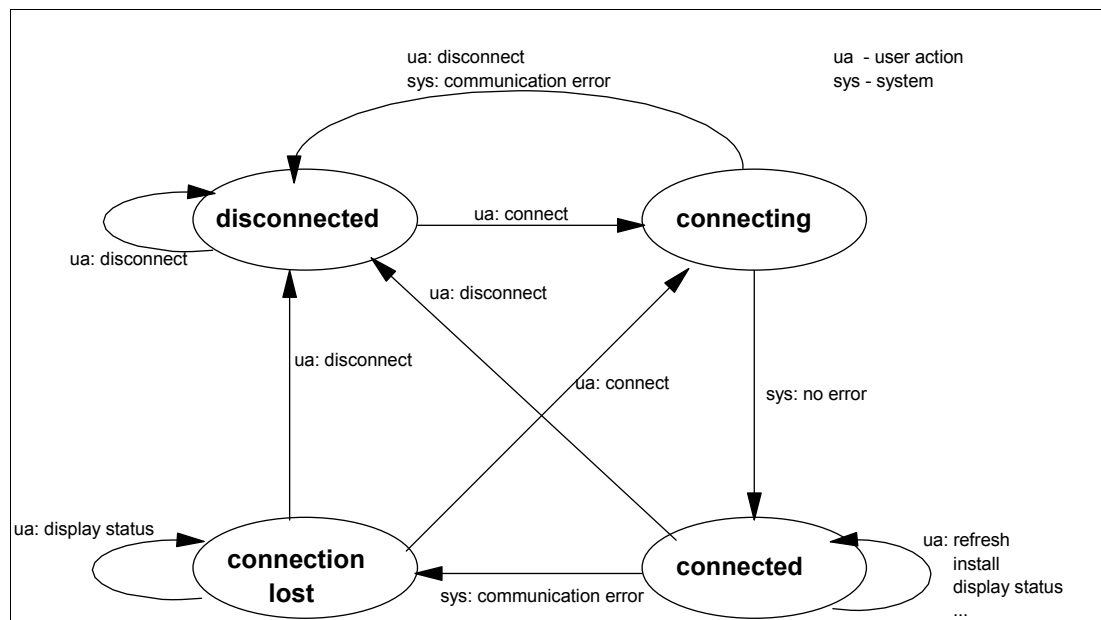


Figure 37-1 Connection States

### 37.1.1 Provisioning Manager

The Provisioning Manager is the component of Capacity Provisioning that monitors the domain and performs activation and deactivation requests for temporary capacity, based on your active policy and domain configuration. When the Control Center is connected to the Provisioning Manager, you can view the status of the Provisioning Manager and install provisioning policies and domain configurations.

#### Mainline logic

A first user connects to a Provisioning Manager (PM), which means that the Control Center sends an asynchronous request called `ConnectRequest` to the PM. After that the PM returns the following notifications:

- ▶ ConnectedNotification
- ▶ DomainReportNotification
- ▶ PolicyReportNotification
- ▶ ConfigurationReportNotification

Then the user selects an object to inspect, and its status and Status information is displayed in a corresponding panel.

**Note:** If a ConnectRequest or DisconnectRequest fails, an ErrorNotification is sent.

After a successful connection, the user clicks the Refresh button. This causes an asynchronous request to the Control Center which sends a ReportRequest to the PM. The PM returns one of the following notifications:

- ▶ A corresponding ReportNotification
- ▶ A ConnectionLostNotification

When the work is done, the user is disconnected from the PM.

- ▶ A DisconnectRequest is sent to the PM.
- ▶ The PM returns a DisconnectedNotification.

To be able to inspect the Provisioning Manager Status, you first have to define a connection to the PM and connect to it. If your connection was successful, you see a panel as shown in Figure 37-2 on page 722. To do so:

- ▶ Select **Provisioning Manager** in the tree view on the left.
- ▶ Click **Add Connection** and fill out the new entry in the Connections table.
- ▶ Click the connection you want to use in the Connections table.
- ▶ Click **Connect** and log in to the host where the PM for this domain runs.

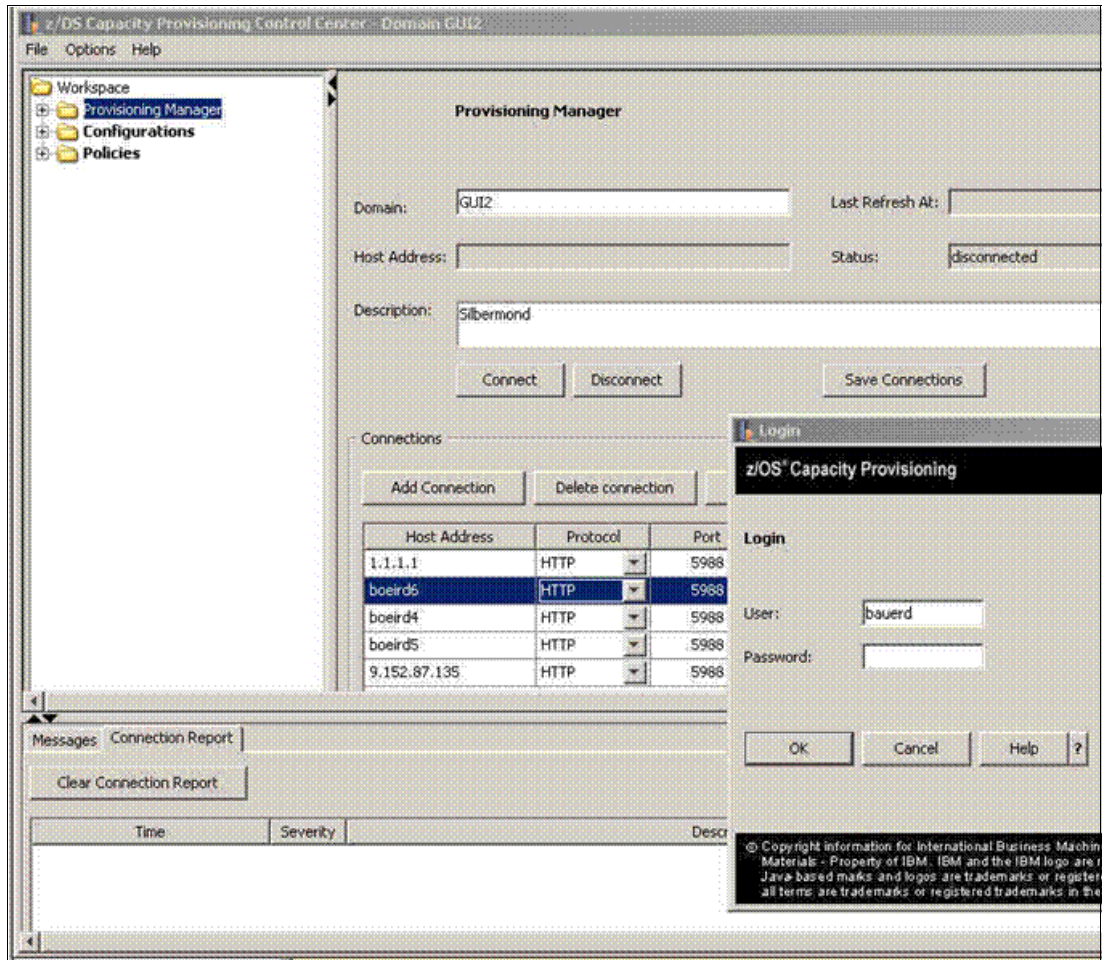


Figure 37-2 CPM Status panel

To inspect the Active Configuration, click **Active Configuration** in the Status folder of the tree view. This panel shows information about the active domain configuration and the status of its elements. An example of this panel is documented in Figure 37-3 on page 723.

The information about the Active Configuration is not refreshed automatically. To update the information, click **Refresh**. The Last Refresh At field will be updated to show the date and time the information was last gathered. To inspect the CPCs of an Active Configuration, click the **CPCs** tab. Select a CPC in the table to display the details of the selected CPC.



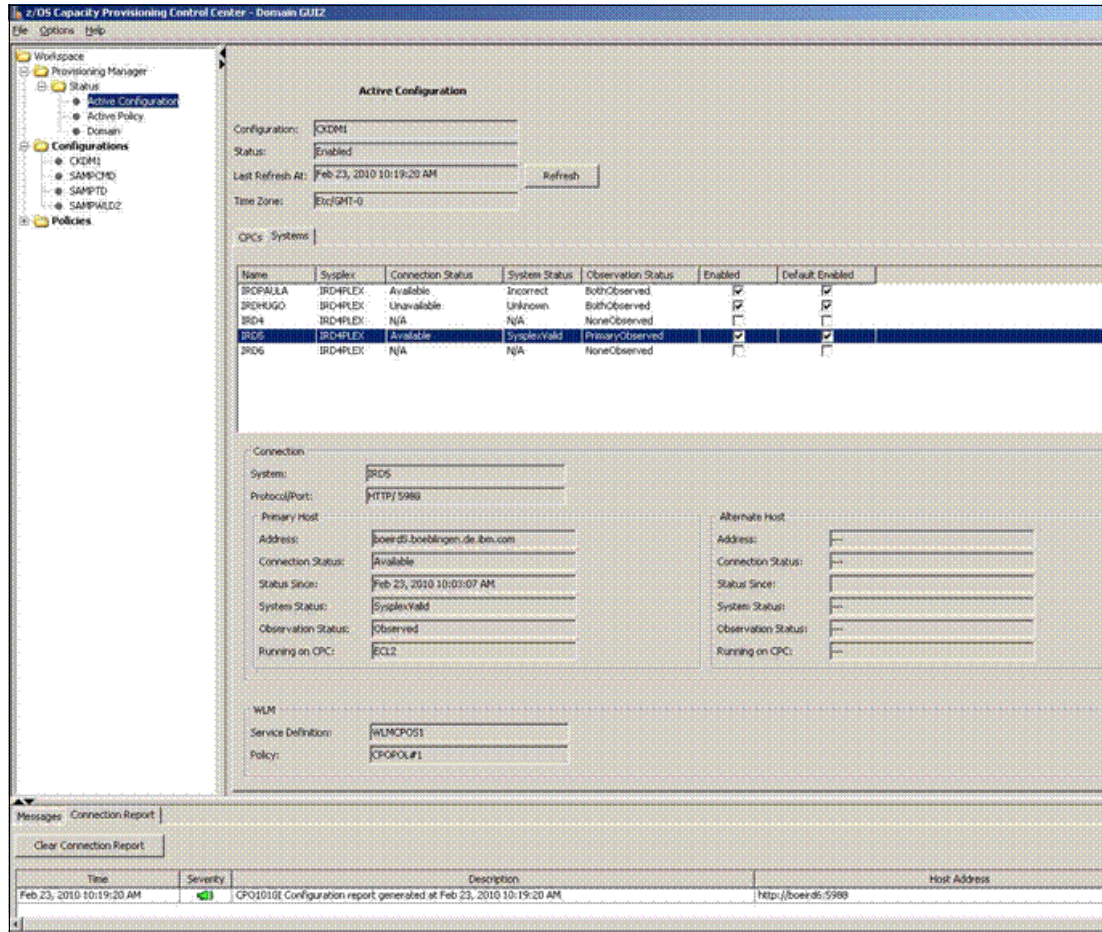


Figure 37-3 CPM CPC configuration

To inspect the systems of an Active Configuration, click the **Systems** tab you see in Figure 37-4 on page 724. Select a system in the table to display the details of the selected system.

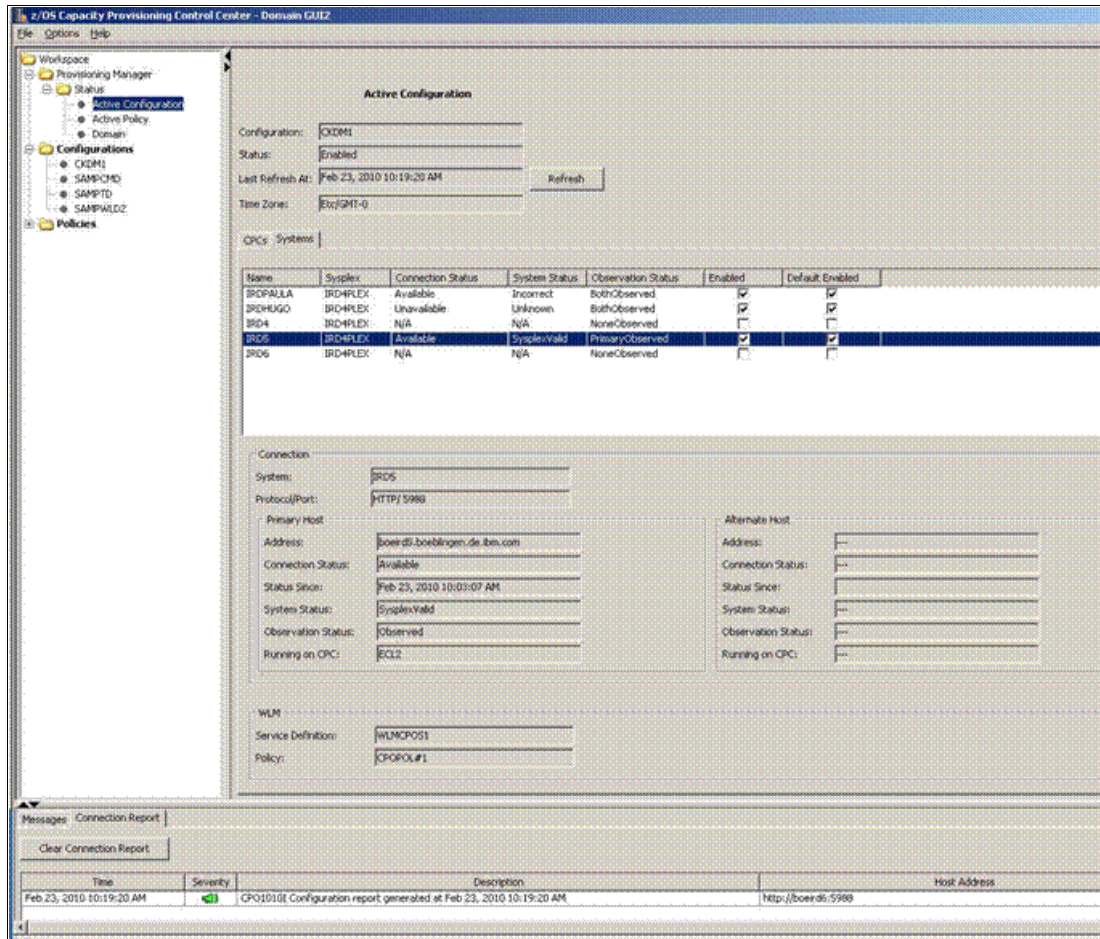


Figure 37-4 CPM Systems configuration

To inspect the Active Policy, click **Active Policy** in the Status folder of the tree view. This panel shows information about the active policy and the status of its elements, as shown in Figure 37-5 on page 725.

For each element, the table shows the following status information:

- ▶ The type of the element and its position in the hierarchy of the policy
- ▶ The name of the element
- ▶ The status of the element, whether it is Enabled, active, and so on
- ▶ Details of the policy element in short form

Select an element in the table to display the details of the policy element.

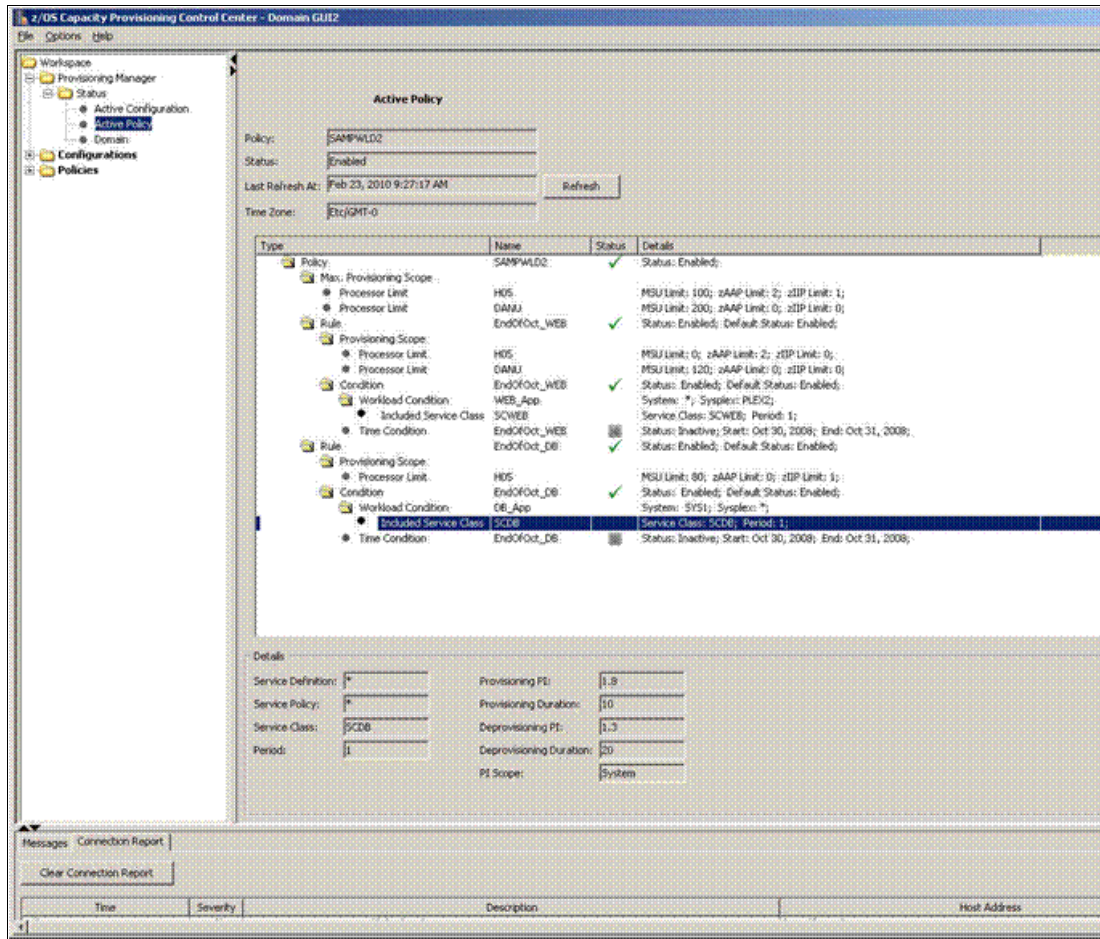


Figure 37-5 CPM active policy

To inspect the domain, select **Domain** in the Status folder of the Provisioning Manager folder in the tree and you should see a panel as shown in Figure 37-6 on page 726.

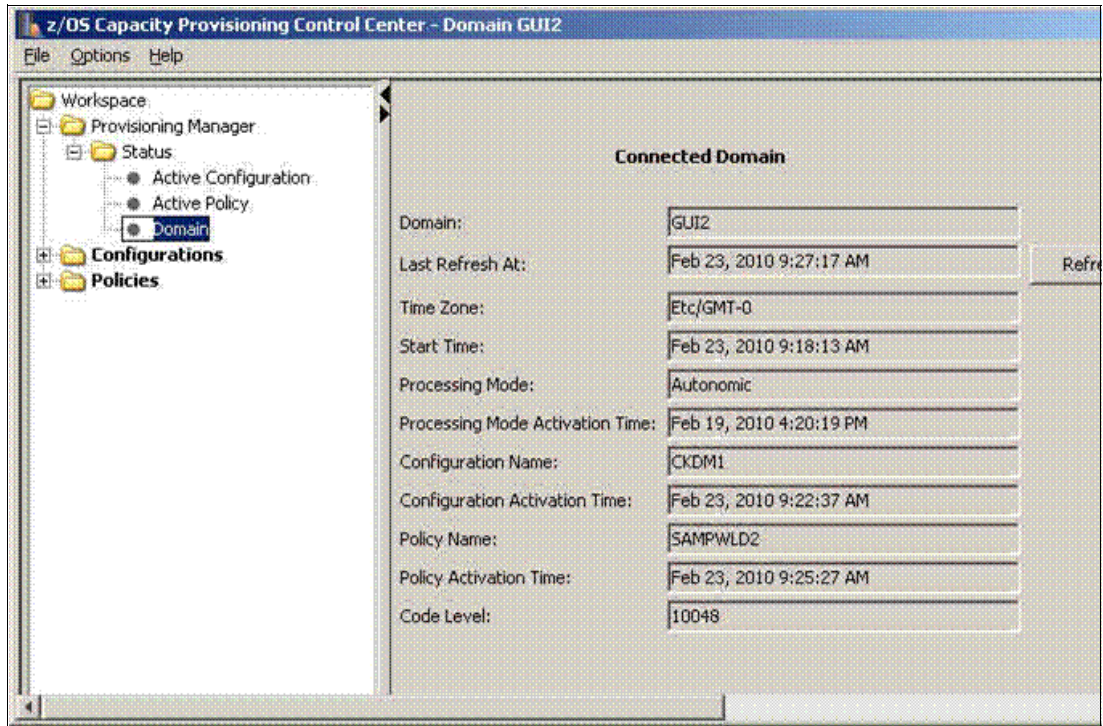


Figure 37-6 CPM Inspecting Domains

To analyze connection problems, have a look at the messages in the Connection Report you see in Figure 37-7 on page 727. This panel will show any kind of connection problems.



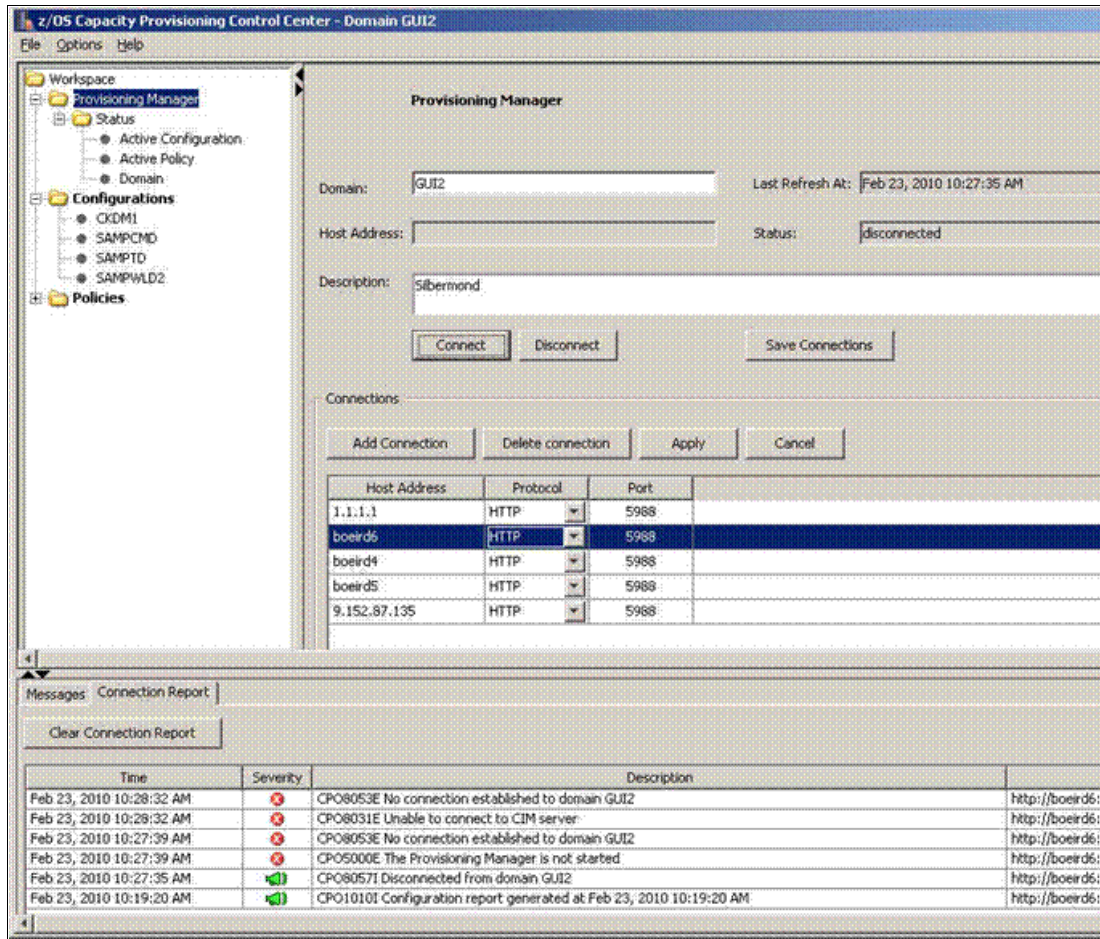


Figure 37-7 CPM connection problems

## 37.2 Command correlation

There can be many reasons why commands fail. In most cases, the Capacity Provisioning Manager (CPM) will not issue the commands. But due to timing issues and parallel work on the HMC, these situations sometimes arise. Sometimes, commands to the HMC/SE fail, for example:

- ▶ On/Off CoD record is locked.
- ▶ Someone is working in parallel on HMC/SE.

The Capacity Provisioning Manager did not detect failures and in the past asked the operator how to resolve, because commands to the hardware are processed asynchronously. Now CPM is able to detect command failures and start automatic recovery or retry the operation. Furthermore, CPM will now display detailed error messages that you can see in Figure 37-8 on page 728.

```
CP01042I Activity report generated at 01/26/2010 08:40:58
Number of activities between 11/24/2009 and 01/26/2010 was 2
Number of detected failed activities was 1...
Command failure detected at 01/18/2010 12:01:39 for CPC ECL2.
Reason=40
Activation for CPC ECL2 at 01/18/2010 12:01:05
Activation of model 724, 4 zAAPs and 2 zIIPs
Active resources before activation: model 722, 4 zAAPs, 2 zIIPs
Inducing policy element is policy POLICY, rule CP160106R,
provisioning condition CP160106C, time condition CP160106T
```

Figure 37-8 enhanced CPM error messages

In this example the command failure usually belongs to the activity for the same CPC.

**Note:** This support is available on R10 and R11 using APAR OA30496.

## 37.3 Average performance index

The Capacity Provisioning Manager uses performance index (PI) graphs to recognize suffering workloads because PI measurements could prevent provisioning. Starting with z/OS V1R12, the CPM can now control PI graphs, minimizing the effects of other values. You can enable or disable this feature and adapt smoothing factors to your needs.

The performance index of many workloads may 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 via 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 may be recognized more reliably.

### Analyzing workload bottlenecks

An analysis of a workload that is suffering from a bottleneck is done by you through the Capacity Provisioning policy. A criteria is that the workload needs to surpass a defined PI for a defined period of time (PI above 2.0 for at least 15 minutes).

A fluctuating workload does not meet the provisioning criteria because during the specified provisioning PI duration, the PI temporarily drops beneath the provisioning PI. 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 via an exponentially weighted moving average (EWMA) function. The calculation considers current PI observations (in the interval  $t$ ) as well as all preceding PI observations (as far as back to the first observed interval 0) of a continuous time series, and weights the values with a user-specified smoothing factor  $\omega$ .

If a workload exhibits fluctuating PI patterns of an observed workload, for example from transaction bursts, these short-term PI decreases would possibly prevent the Provisioning Manager from taking actions early, since the PI has not exceeded the limit of the Provisioning

PI for the entire provisioning PI duration. Nevertheless, you may want the provisioning criteria to be met because on average the PI is above the Provisioning PI.

A provisioning condition of a CP policy specifies that the PI must be completely above a specified PI limit for a given duration before the workload will be recognized as being *suffering*. Single outliers, as in the example shown in Figure 37-9, let the PI graph fall below that limit and therefore the minimum provisioning PI duration is not met. This will prevent fulfilling of the rule and provisioning of resources on behalf of that workload. This situation can happen with transactions where workload distribution is irregular, although the class itself is suffering in the long term.

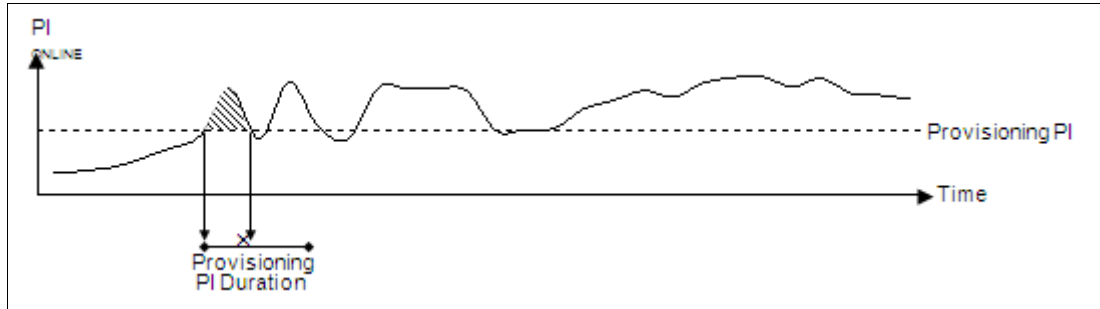


Figure 37-9 CPM old behavior of PI

Starting with z/OS V1R12, the z/OS CPM can optionally smoothen the PI graph, eliminating unwanted effects of single outlier values, as shown in Figure 37-10.

The graph resulting from such a moving average smoothing is characterized by a more even value pattern; this might allow for activations in situations that would not be taken into account without the additional smoothing.

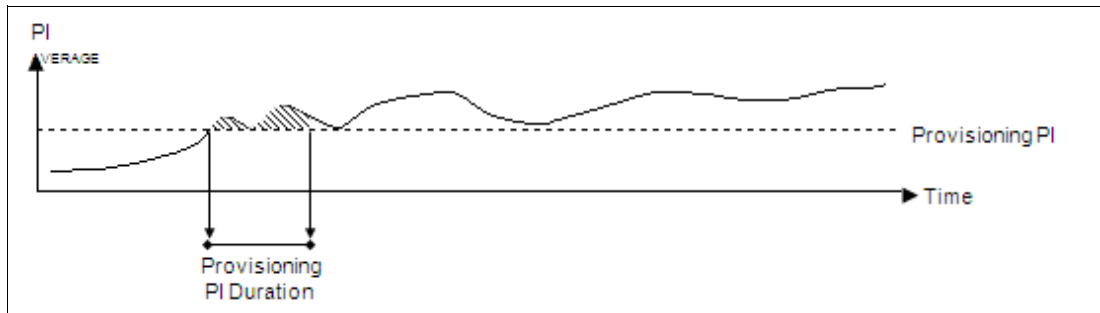


Figure 37-10 CPM smoothing PI

The smoothed PI graph fulfills the criteria specified in the Capacity Provisioning policy at an early stage.

**Note:** Smoothing is done with the exponentially weighted moving average (EWMA) algorithm.

The CPM can now recognize a suffering workload even if the PI graph is distorted by single measurements. Also, a simplified initial security setup for the CPM is shipped with release 12.

You can now specify an upper limit capping: With the applied smoothing, past measurements have effects on the calculated smoothed PI values of future points in time. The stronger the smoothing factor is, the more past measurements will affect calculated and smoothed PIs of

the future. Very high (single) outliers may lift the smoothed PI graph to higher levels. An upper limit capping prevents high outliers from distorting the smoothed graph. Everything (smoothing factor, upper limit capping) also applies to deprovisioning: the CPM will no longer be fooled by (high) single outlier values when recognizing a deprovisioning situation.

### 37.3.1 Invoke the support

To activate this support, you have to add keys in the parmlib member of the CPM configuration data set. These parameter keys are specified in `<CPMuser>.<Domainname>.PARM[PARAM]`. The weighting factor specifying the degree of smoothing is activated by:

```
SystemObservation.MovingAveragePiWeight=50
```

The capping value specifying an upper limit is activated by:

```
SystemObservation.MovingAveragePiCapping=5.5
```

The new external output is added. When support is activated, you see a message in addition in the Capacity Provisioning workload report as shown in Figure 37-11.

Workload is analyzed for 1 system(s) average PI management enabled: weight 50%, capping PI 5.5
--

Figure 37-11 Additional CPM messages

The weighting factor ranges from 1 to 100% (100%=smoothing disabled), which expresses how much the last measurement is being weighted for the smoothed PI. Low values give more importance to the past measurements, high values give more weight to the current measurements. The capping value prevents that higher PI measurements distort the smoothed PI graph in the long term.

### 37.3.2 Setup and security

While the *z/OS MVS Capacity Provisioning User's Guide*, SC33-8299 details the security definitions for many different scenarios, most users are interested in a simple straightforward setup. The provided samples in `SYS1.SAMPLIB` are based on best practices. The samples can be copied and changed by you as appropriate. Clients that stick to the standards, for example for naming conventions, have to make only very few changes in areas where defaults would be inappropriate or a security risk. Capacity provisioning categorizes z/OS images into runtime systems and observed systems.

A runtime system is a system where a CPM instance runs or could fail over to. Usually, runtime systems are part of the same sysplex. An observed system is a system that is monitored by CPM. It can be part of the same sysplex as the runtime system, or part of another sysplex. Two samples are delivered:

`SYS1.SAMPLIB(CPOSEC1)` contains all security definitions that are required on a runtime system that is also an observed system.

`SYS1.SAMPLIB(CPOSEC2)` contains all security definitions that are required on additional observed systems that are using another security data base.

This support can be invoked by copying the sample members from `SYS1.SAMPLIB` to a private library. The RACF commands in there can be adjusted to reflect the specific naming conventions that were chosen, or adopted RACF settings that are in effect in the installation.



**Important:** When using copy-and-paste from a pdf file, many special characters may not be handled correctly. Copying from a library member eliminates that problem.

## Sample CPOSEC1

This job contains sample security definitions that should be reviewed based on the documentation provided in *z/OS MVS Capacity Provisioning User's Guide*, SC33-8299. The z/OS RMF and CIM elements, including RMF Distributed Data Server and RMF CIM provider, should have been customized prior to the Capacity Provisioning customization. This job contains statements that must be completed with installation-specific data. It contains sample statements required to set up the runtime systems, for example the systems where a Capacity Provisioning Manager may run. It also includes the definitions for observed systems. If additional observed systems need to be set up that are using a different security database, refer to sample member CPOSEC2. The statements you see in Figure 37-12 are intended for use with z/OS Security Server (RACF). This job does not contain the required z/OS UNIX commands that are also required for complete security definitions.

**Important:** Replace all bold parameters in Figure 37-13 on page 732 and Figure 37-14 on page 733 with your installation defaults.

Figure 37-14 on page 733 contains security definitions that are only required when using the BCPii protocol. These commands are not necessary and can be deleted if the SNMP protocol is used instead of BCPii.

## Sample CPOSERV2

This job contains sample security definitions that should be reviewed based on your installation defaults. This Job is used to run only on observed system which are monitored by CPM.

```
ADDUSER (CPOSRV)
ALTUSER (CPOSRV) +
        NORESTRICTED +
        DFLTGRP(.....) +
        OMVS(HOME() UID(.....) ) +
        PASSWORD(.....)
SETROPTS CLASSACT(PTKTDATA)
SETROPTS RACLIST(PTKTDATA)
RDEFINE PTKTDATA CFZAPPL SSIGNON(KEYMASKED(.....)) +
        APPLDATA('NO REPLAY PROTECTION')
PERMIT CFZAPPL CLASS(PTKTDATA) ID(CPOSRV) ACCESS(READ)
SETROPTS RACLIST(PTKTDATA) REFRESH
RDEFINE PTKTDATA IRRPTAUTH.CFZAPPL.CPOSRV
PERMIT IRRPTAUTH.CFZAPPL.CPOSRV CLASS(PTKTDATA) ID(CPOSRV) +
        ACCESS(UPDATE)
SETROPTS RACLIST(PTKTDATA) REFRESH
RDEFINE FACILITY IRR.RTICKETSERV
PERMIT IRR.RTICKETSERV CLASS(FACILITY) ID(CPOSRV) ACCESS(READ)
SETROPTS RACLIST(FACILITY) REFRESH
```

Figure 37-12 RACF definitions for observed systems

```

ADDUSER (CPOSRV)
ALTUSER (CPOSRV) +
    NORESTRICTED +
    DFLTGRP(.....) +
    OMVS(HOME('/u/cposrv') UID(.....) ) +
    PASSWORD(.....)
RDEFINE STARTED CPOSERV.* STDATA(USER(CPOSRV))
SETROPTS RACLIST (STARTED) REFRESH
RDEFINE FACILITY IXCARM.SYSCPM.SYSCPO UACC(NONE)
PERMIT IXCARM.SYSCPM.SYSCPO CLASS(FACILITY) ID(CPOSRV) +
    ACCESS(UPDATE)

SETROPTS RACLIST(FACILITY) REFRESH
ADDGROUP CPOQUERY OMVS(GID(.....))
ADDGROUP CPOCTRL OMVS(GID(.....))
CONNECT CPOSRV GROUP(CPOQUERY) AUTH(USE)
CONNECT CPOSRV GROUP(CPOCTRL) AUTH(USE)
ADDSD 'CPO.DOMAIN1.*' GENERIC UACC(NONE)
PERMIT 'CPO.DOMAIN1.*' GENERIC ID(CPOSRV) ACCESS(UPDATE)
ADDSD 'CPOSRV.**' GENERIC UACC(NONE)
PERMIT 'CPOSRV.**' GENERIC ID(CPOSRV) ACCESS(CONTROL)
SETROPTS GENERIC(DATASET) REFRESH
RDEFINE FACILITY BPX.CONSOLE UACC(NONE)
PERMIT BPX.CONSOLE CLASS(FACILITY) ID(CPOSRV) ACCESS(READ)
SETROPTS RACLIST(FACILITY) REFRESH
SETROPTS CLASSACT(PTKTDATA)
SETROPTS RACLIST(PTKTDATA)
RDEFINE PTKTDATA CFZAPPL SSIGNON(KEYMASKED(.....)) +
    APPLDATA('NO REPLAY PROTECTION')
PERMIT CFZAPPL CLASS(PTKTDATA) ID(CPOSRV) ACCESS(READ)
SETROPTS RACLIST(PTKTDATA) REFRESH
RDEFINE PTKTDATA IRRPTAUTH.CFZAPPL.CPOSRV
PERMIT IRRPTAUTH.CFZAPPL.CPOSRV CLASS(PTKTDATA) ID(CPOSRV) +
    ACCESS(UPDATE)

SETROPTS RACLIST(PTKTDATA) REFRESH
RDEFINE FACILITY IRR.RTICKETSERV
PERMIT IRR.RTICKETSERV CLASS(FACILITY) ID(CPOSRV) ACCESS(READ)
SETROPTS RACLIST(FACILITY) REFRESH
CONNECT (CPCCUSR) +
    GROUP(CPOQUERY) AUTH(USE)
CONNECT (CPCCUSR) +
    GROUP(CPOCTRL ) AUTH(USE)
RDEFINE SURROGAT BPX.SRV.** UACC(NONE)
PERMIT BPX.SRV.** CLASS(SURROGAT) ID(CFZADM) ACCESS(READ)
SETROPTS GENERIC(SURROGAT) REFRESH
PERMIT CIMSERV CLASS(WBEM) ID(CPCCUSR) ACCESS(UPDATE)
SETROPTS RACLIST(WBEM) REFRESH
RDEF SURROGAT BPX.SRV.** UACC(NONE)
PERMIT BPX.SRV.** CLASS(SURROGAT) ID(CFZADM) ACCESS(READ)
SETROPTS GENERIC(SURROGAT) REFRESH

```

Figure 37-13 RACF statements for the CPM server

```

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
RDEFINE FACILITY HWI.APPLNAME.HWISERV UACC(NONE)
RDEFINE FACILITY HWI.TARGET.netname.cpc1 +
APPLDATA('.....') UACC(NONE)
RDEFINE FACILITY HWI.CAPREC.netname.cpc1.* UACC(NONE)
PERMIT HWI.APPLNAME.HWISERV CLASS(FACILITY) ID(CPOSRV) +
ACCESS(READ)
PERMIT HWI.TARGET.netname.cpc1 CLASS(FACILITY) ID(CPOSRV) +
ACCESS(CONTROL)
PERMIT HWI.CAPREC.netname.cpc1.* CLASS(FACILITY) +
ID(CPOSRV) ACCESS(READ)
SETROPTS RACLIST(FACILITY) REFRESH

```

Figure 37-14 Statements when using BCPII

## 37.4 CICS/IMS support

In prior releases of z/OS, Capacity Provisioning was unable to manage capacity on behalf of individual CICS/IMS transaction service classes. CICS/IMS subsystems use WLM execution delay monitoring facilities in the following way:

- ▶ Delays are aggregated to the service class of the server.
- ▶ The service class of the transaction reports no delays.

In z/OS V1R12, we are able to manage resources on behalf of a specific transaction service class, not on behalf of a transaction server service class serving a multitude of different transaction classes. RMF now relates delays accurately to the CICS/IMS transaction class they belong to. Capacity Provision can now be triggered by transaction classes.

**Important:** CICS/IMS support is available in R10 and R11 using APAR OA29641, with dependency to RMF APAR OA29771 and CIM APAR OA30040.





## 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 (RRS) 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.

z/OS, when requested, can coordinate changes to one or more protected resources, which can be accessed through different resource managers and reside on different systems. z/OS ensures that all changes are made or no changes are made. Resources that z/OS can protect include:

- ▶ A hierarchical database
- ▶ A relational database
- ▶ A product-specific resource

## 38.1 Performance issues

When a syncpoint coordinator is available and coordinating a transaction that could involve multiple resource managers, the transaction is known as a global transaction. When, instead, each resource manager involved is separately coordinating its own changes, and only its changes, the transaction is known as a local transaction.

For an RRS-coordinated global transaction, even with no enlisted resource manager interest, it would still go through a complete two-phase commit syncpoint processing, going through all syncpoint phases and state transactions. Typically, a global transaction goes through the following state transactions:

- ▶ In-flight
- ▶ In-state-check
- ▶ In-prepare
- ▶ In-doubt
- ▶ In-commit/backout
- ▶ In-end
- ▶ In-completion
- ▶ In-forget

Since there is no interest in this global transaction, during the course of the full syncpoint processing, RRS will not drive any resource manager's exits or write any log records. To minimize the transaction processing overhead, RRS will “fast path” a global transaction with no interest by optimizing the syncpoint processing by transitioning this transaction from in-flight to in-forgotten without rolling through full syncpoint processing—a transition from in-flight to forgotten, skipping all the states and processing in between.

### Storage initialization

The RRS modules ATRLMLLOG, ATRLMMIG, and ATRLMTKO use the INITSTACK parameter on the ASAEntry macro. This does an MVCL to clear the dynamic area. However, parts of that storage are cleared in recovery initialization and individual field initialization. The bottom line is, we are spending too many cycles clearing the same storage when it is not needed.

The ATRBCARR macro has methods GetArray and GetSerializedArray that does a storage obtain and then initializes a 20 k array storage area each time an array is created. This array is used to track items such as RM names and system names. They are used then released numerous times during RRS processing. So a 20 k array is obtained, cleared, a few entries used, and the storage is released. In typical user environments this behavior is very rarely used for the entire array. RRS spends too much time to initialize this array, which is probably never used. The processing is enhanced to initialize only the header at storage obtain and each entry prior to use.

## 38.2 Cascaded displays with z/OS V1R12

In prior releases of z/OS, RRS did not have to provide detailed debug information for transaction hangs. Starting with z/OS V1R12, IBM introduced a new DISPLAY RRS command that will be enhanced to help debug RRS transaction hang problems.

In Table 38-1 you see the required software service to implement the new functionality in prior releases of z/OS.

Table 38-1 Rollback support

Release	APAR	PTF
z/OS 1.8	0A29174	UA50141
z/OS 1.9	0A30351	HBB7740 UA52881 JBB774J UA52884
z/OS 1.10	0A30351	HBB7750 UA52882 JBB775J UA52885
z/OS 1.11	0A30351	HBB7760 UA52883 JBB776J UA52886
z/OS 1.12	n/a	part of the product

### 38.2.1 DISPLAY enhancements

In z/OS V1R12, the D RRS command is changed to display all the members for a cascaded UR. Furthermore, the Extend UR COMMENTS field can now contain new exceptions. The new EXCEPTION display lists URs that are waiting and the new STATUS display lists the number of URs in each state and the min/max duration time.

#### New parameter FAMILY

FAMILY or F - Indicates that the system is to display the units of recovery (UR) in the cascaded transaction specified as input.

In Figure 38-1 you see a UR family. From the information in the COMMENTS field, it can be determined that this is a sysplex cascaded UR family (S), the top-level UR is on SY2 (T), and the UR on SY1 is a child (C).

D RRS,UR,FAMILY,URID=C48C1CAC7DFD63D01010000						
SY1	ATR622I	12.53.23	RRS	UR	FAMILY	475
	URID		SYSTEM	GNAME	ST	COMMENTS
	C48C1CAC7DFD60001010000		SY2	PLEX1	FLT	TS
	C48C1CAC7DFD63D01010000		SY1	PLEX1	FLT	CS

Figure 38-1 Usage of the FAMILY option

### 38.2.2 Extend COMMENTS field

IBM added three new values to the COMMENTS column in the D RRS command:

- A** Waiting for the child/subordinate application to signal that it is complete (ready for syncpoint to be driven).
- E** Waiting for a response from the resource manager's exit.
- P** Coordinator UR waiting for a response from RRS on one or more remote systems.

In Figure 38-2 on page 738 we demonstrate some of the new values of the COMMENTS column.

```

SY1  ATR601I  10.31.33  RRS UR SUMMARY 290
  URID                      SYSTEM  GNAME  ST   TYPE  COMMENTS
C48C1CAC7E3ED3D01030000 SY2    PLEX1  FLT  Prot  CSA
C48C1CAC7DFD63D01010000 SY1    PLEX1  FLT  Prot  TE
C48C1CAC7DFD60001010000 SY1    PLEX1  PRP  Prot  TSP

```

Figure 38-2 Example of extended comments

### 38.2.3 EXCEPTION option on the DISPLAY command

The DISPLAY RRS command is used to display status information about RRS coordinated transactions to the system console and SYSLOG. You can also use automation to parse the output and trigger alerts.

In z/OSV1R12 there are two new parameters, as follows:

**URSTATUS or URST** Display unit of recovery information statistics for the local system or the systems specified on the SYSNAME= and GNAME= parameters. This parameter indicates that the system is to display unit of recovery statistics that include UR state, UR count, and UR duration time information for in-storage URs and not process any URs found in the log streams. The system is to use message output ATR623I for the information.

**UREXCEPTION or UREX** Display unit of recovery information for URs that are in exceptions on the local system or the systems specified on the SYSNAME= and GNAME= parameters. This parameter indicates that the system is to display unit of recovery (UR) information for URs that are in exceptions or waiting for a resource. The system is to use message output ATR624I for the information. If the DISPLAY RRS,UREXCEPTION command fails during processing, messages are issued indicating the problem. In such a situation, failure messages prevent the display of any valid information.

**Command Examples:** DISPLAY RRS,UREXCEPTION or DISPLAY RRS,UREX

Optionally, you can type SYSNAME and GNAME to get further information. In Figure 38-3 on page 739 you see the modified syntax of the DISPLAY RRS command.



```

D RRS[,UR[ ,SUMMARY|,SUM|,S|,DETAILED|,D]uroptions]
[,UR[,FAMILY|,F]urfamoptions]
[,RM[ ,SUMMARY|,SUM|,S|,DETAILED|,D]rmoptions]
[,URSTATUS|,URSTsysoptions]
[,UREXCEPTION|,UREXsysoptions]
[,L=a|name|name-a]
uroptions:
[,URID=ur-identifier]
[,STATE=FLT|SCK|PRP|DBT|CMT|BAK|END|OLA|CMP|FGT]
[,SYSNAME=system-name]
[,GNAME=logging-group-name]
urfamoptions:
[,URID=ur-identifier]
[,SYSNAME=system-name]
[,GNAME=logging-group-name]
rmoptions:
[,RMNAME=resource-manager-name]
[,SYSNAME=system-name]
[,GNAME=logging-group-name]
sysoptions:
[,SYSNAME=system-name]
[,GNAME=logging-group-name]

```

Figure 38-3 Enhanced syntax for D RRS

## Command examples

Figure 38-4 shows a sample output with the new EXCEPTION option in a hang situation.

**Note:** The E, A, P letters in the example do not appear in the message but were added to show the association between the new exception comments and the UREXCEPTION text output.

The first UR is waiting for job RM1DSP18 to reply to the commit exit. The second UR is waiting for job RM2DSP02 to signal application complete. The third UR is waiting for RRS to finish processing on SY1.

```

D RRS, UREX
SY1 ATR624I 12.52.04 RRS UR EXCEPTION 323
SYSTEM URID WAIT FOR
SY1 C498D44A7E0970001010000 RM1DSP18 COMMIT Exit E
SY1 C498D44A7E0973D01010000 RM2DSP02 Appl Comp A
SY1 C498D44A7E0973D01010000 RRS P

```

Figure 38-4 Example of the EXCEPTION parameter

### Restrictions:

- ▶ The DISPLAY RRS command is not a performance path.
- ▶ Trying to process more than 1500 URs may exceed the DISPLAY RRS internal work area. When the area is exceeded, the following conditions are expected:
  - D RRS,UR will terminate with:

```
ATR605I DISPLAY RRS COMMAND TRUNCATED, SOME DATA NOT AVAILABLE
```

- D RRS,UREX will process all the data in the work area issuing message ATR624I and terminate with:
  - \* Data truncated. DISPLAY RRS work area exceeded
  - or
  - \* Data truncated. DISPLAY RRS secondary work area exceeded
- D RRS,URST is not affected by the number of URs.

Trying to process lots of URs may exceed the CONSOLE buffer. Message CNZ3011I will be issued and the output will be truncated when the buffer is exceeded. The number of lines written before truncation is based on a buffer limit specified at IPL in the CONSOLxx parmlib member. However, the data will appear in the system log without being truncated.

### 38.2.4 STATUS display

The new DISPLAY RRS option indicates that the system is to display unit of recovery statistics that include UR state, UR count, and UR duration time information for in-storage URs.

**Important:** The new command does not process any URs found in the log streams.

The system is to use message output ATR623I for the information. Figure 38-5 shows an example output for the new URSTATUS option.

SY1	ATR623I	15.53.31	RRS UR STATUS	166		
SYSTEM	GNAME	STATE	NUM OF URS	MIN TIME	MAX TIME	
SY1	PLEX1	FLIGHT.....	3	000:01:05	000:03:13	
SY1	PLEX1	PRE PREPARE....	1	000:00:35	000:00:35	
. . .						
SY1	PLEX1	FORGET.....	1	000:03:13	000:03:13	
SY1	PLEX1	Total URS....	15			

Figure 38-5 Output of URSTATUS

#### Meaning of Min Time and Max Time

For a given state, RRS invokes exits for that state. RRS keeps track of the elapsed time, in a TOD format, that was used for the exit to do its processing. When an exit has not yet returned to RRS, its elapsed time will continue to increase until the exit returns. For a given unit of recovery (UR), in a given state, any number of exits can be driven based on the number of Resource Managers that have expressed interest in the UR. The duration for that UR is the largest elapsed time for all the Resource Managers associated with that UR. The largest duration displayed is 999:59:59, which equates to 41 days, 15 hours, 59 minutes, and 59 seconds. When this value is displayed, the actual duration is most likely more than that value.

For all the URs on a given system, Logging Group (Gname) and System Name (SysName), in a given state, the Max Time is the largest UR duration. Conversely, the Min Time is the smallest UR duration. Example:

SYSTEM	GNAME	STATE	NUM OF URS	MIN TIME	MAX TIME
SY1	PLEX1	COMMIT.....	15	000:00:24	000:23:35

At this point in time, on system SY1 and group name PLEX1, there are 15 URs in commit. Of all the URs, one has been in commit for 23 minutes and 35 seconds, while another has been there just 24 seconds. The other URs in commit have a duration in between those two times.

Issuing the D RRS,URSTATUS command again will probably have different results as transactions proceed to completion. If subsequent D RRS,URSTATUS commands indicate an increasing Max Time for a particular state, steps should be taken to identify the transaction that is not progressing. A suggestion would be D RRS,UREXCEPTION to determine what the resource manager is waiting for.





## Parallel subsystems initialization

A subsystem is a service provider that performs one function or many functions, but does nothing until it is requested. Although the term *subsystem* can be used in other ways, in this chapter a subsystem must be the master subsystem or be defined to MVS in one of the following ways:

- ▶ Processing the IEFSSNxx parmlib member during IPL

You can use either the keyword format or positional format of the IEFSSNxx parmlib member. It is best practice to use the keyword format, because this allows you to define and dynamically manage your subsystems.

- ▶ Issuing the IEFSSI macro
- ▶ Issuing the **SETSSI** system command

This chapter describes a contribution made by the z/OS V1R12 initialization process to reduce the time to complete the IPL, and thereby reduce the mean time to recovery (MTTR).

This enhancement speeds up subsystem initialization by having the subsystems be initialized as much in parallel as possible.

## 39.1 Mean time to recovery

Mean time to recovery (MTTR), as depicted in Figure 39-1, is defined as the time from when a problem is detected until the system resumes performing productive work.

Several enhancements to z/OS have been implemented which reduce the time it takes the operating system, subsystems, and middleware to go down and then recover. This reduction can be achieved by:

- ▶ Reducing initialization path lengths where possible
- ▶ Exploiting parallelism where possible

This approach reduces the cost of operating system services used by subsystems and middleware for initialization.

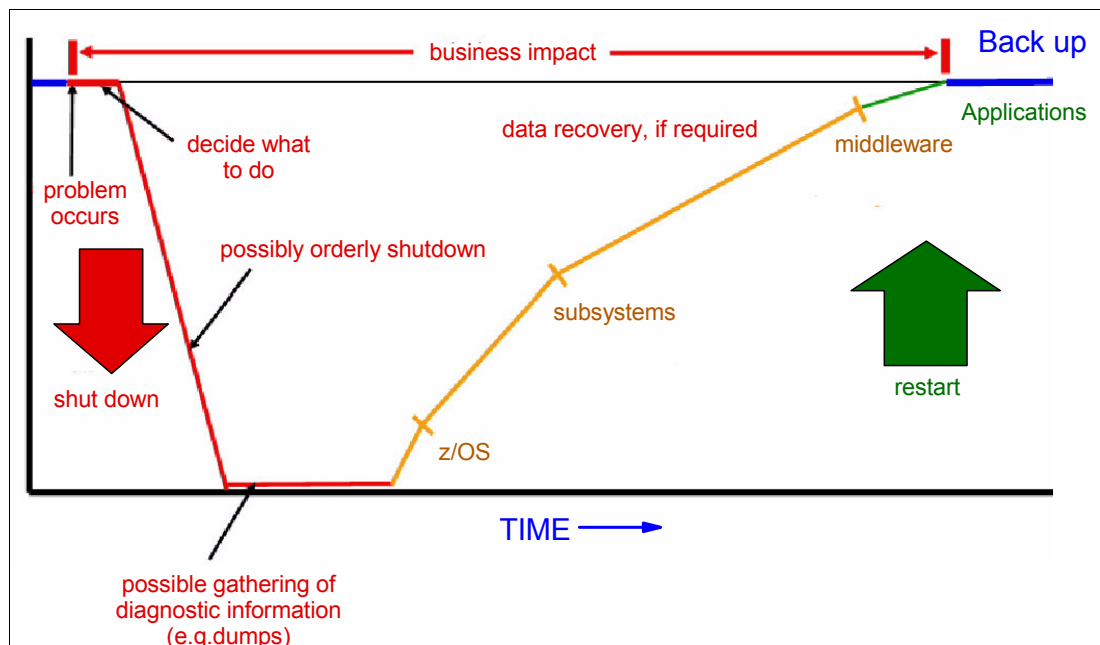


Figure 39-1 MTTR scope of the problem

## 39.2 Subsystem initialization routines

Prior to z/OS V1R12, the subsystem initialization routines specified in parmlib member IEFSSNxx were invoked in the sequence they appeared and under a task that never terminated. From z/OS V1R12, the initialization routines are invoked in parallel after the BEGINPARALLEL keyword in the IEFSSNxx parmlib member is processed, and no longer run under a permanent task when they are run in parallel.

Subsystem initialization routines have been identified as having potential MTTR issues. z/OS V1R12 allows SSI initialization routines to be run in parallel. As a result, overall subsystem initialization time can decrease and thereby the MTTR can also decrease.

### IEFSSNxx parmlib member

Subsystems are defined in the IEFSSNxx parmlib member, with optionally an initialization routine being specified with the subsystem definition itself.

Up to z/OS V1R12, initialization routines are invoked the way their IEFSSNxx parmlib member definitions are specified:

- ▶ Routines are invoked in the order specified in the IEFSSNxx parmlib member of the parmlib.
- ▶ Routines are invoked serially.
- ▶ Routines run under a never-ending task.

### **z/OS V1R12 enhancements**

To support parallelism in z/OS V1R12, the IEFSSNxx parmlib member now supports a new keyword (BEGINPARALLEL), with the following meanings:

- ▶ All subsystem initialization routines specified after BEGINPARALLEL will be invoked in parallel.
- ▶ Initialization routines specified before BEGINPARALLEL (or if keyword is not specified) will continue to be invoked serially.

The benefit of this enhancement depends on the following characteristics:

- ▶ Number of initialization routines
- ▶ Complexity of routines
- ▶ Serialization requirements of routines
- ▶ Available number of CPs

## **39.2.1 BEGINPARALLEL usage**

The BEGINPARALLEL keyword in the IEFSSNxx parmlib member is governed by certain rules, as follows:

- ▶ All initialization routines specified before the BEGINPARALLEL keyword are invoked serially. All initialization routines specified after the BEGINPARALLEL keyword are invoked in parallel.
- ▶ BEGINPARALLEL statement should be specified after the SMS definition. If not, message IEFJ009E is issued:

```
IEFJ009E BEGINPARALLEL KEYWORD SPECIFIED BEFORE SMS SUBSYSTEM DEFINITION
```

- ▶ For the z/OS Communications Server TNF and VMCF subsystems, subsystem definitions must be specified before you specify the BEGINPARALLEL statement if the INITRTN parameter is included with the subsystem definitions.
- ▶ If BEGINPARALLEL is specified multiple times or in different concatenated IEFSSNxx parmlib members, all but the first will be rejected with message ASA011I:

```
ASA011I ERROR IN PARMLIB MEMBER=IEFSSNR2 ON LINE 2, POSITION 1: DUPLICATE  
SPECIFICATION OF 'BEGINPARALLEL' FIRST SPECIFICATION IS USED. DETECTING  
MODULE IS IEFJPACK. INPUT LINE: BEGINPARALLEL
```

- ▶ The BEGINPARALLEL statement has no effect on subsystems that do not specify an initialization routine.
- ▶ If BEGINPARALLEL is not specified, the subsystem initialization routines run serially and no performance benefits are obtained.

As an example, Figure 39-2 on page 746 shows the specification of having RACF, LOGGER and DB2 subsystem initialization routines (respectively named IRRSSI00, IXGSSINT, and DSN3INI) run in parallel.

```

SUBSYS SUBNAME(SMS)
INITRTN(IGDSSIIN)
INITPARM(ID=ZX)
SUBSYS SUBNAME(JES2) /* JES2 AS PRIMARY SUBSYSTEM */
PRIMARY(YES) START(YES)
BEGINPARALLEL
SUBSYS SUBNAME(RACF) /* RACF SUBSYS */
INITRTN(IRRSSI00)
INITPARM('#,M')
SUBSYS SUBNAME(LOGR) /* LOGR */
INITRTN(IXGSSINT)
SUBSYS SUBNAME(RRS) /* RESOURCE RECOVERY SERVICES */
SUBSYS SUBNAME(ID9Y) /* IRLM DB2 9.1 FOR DB9Y */
SUBSYS SUBNAME(DB9Y) /* DB2 V910 DB9Y */
INITRTN(DSN3INI)
INITPARM('DSN3EPX,-DB9Y,S')

```

Figure 39-2 BEGINPARALLEL example

### 39.2.2 MSI dynamic exit routine

Some vendor products use the SSI initialization routines as an “exit point,” with the following results:

- ▶ These “exits” need to complete before true subsystems are initialized.
- ▶ Parallelization could cause these products to experience serious problems.

To keep these exit points out of the IEFSSNxx parmlib member process, a new dynamic exit point is provided during master scheduler initialization (MSI), after the security product is initialized and before the IEFSSNxx parmlib member is processed.

In z/OS V1R12, an MSI dynamic exit routine can be used in the following ways:

- ▶ It can be defined in the PROGxx parmlib member:
 

```
EXIT ADD EXITNAME(CNZ_MSIEXIT) MODNAME(xxx) PARAM(parm)
```
- ▶ PROGxx is enhanced to allow parameter specification to be passed to an exit routine.
- ▶ The code must reside in the LPA, LNKLST, or nucleus:
 

Dynamic allocation is not available at this point in MSI.
- ▶ No exit routine return codes are processed.
- ▶ After the exit routine is invoked, the exit point will become “undefined.”
- ▶ Storage occupied by the exit routine is released.

## 39.3 Migration considerations

With the parallel subsystem initialization provided in z/OS V1R12, some behavior might be different due to the following reasons:

- ▶ The execution order of initialization routines running in parallel is now unpredictable.
 

Consequently, routines must not have any execution order dependencies.
- ▶ When running in parallel, routines execute under a task that terminates.



- ▶ Subsystem owners should verify that initialization routines can run under a task that terminates. Specifically, look for:
  - Dataspaces that are created but not deleted (particularly CADs).
  - Task-related storage that is obtained but not released.
  - An ENQ that is obtained but not DEQed.
  - An ALESERV ADD that is issued without DELETE.
  - An ESTAE CREATE that is issued without DELETE.
  - Joining XCF groups without leaving.
  - Connections to Coupling Facility structures that are obtained but not released.
  - Task level Name Tokens that are created without deleting.

Consequently, before inserting the BEGINPARALLEL keyword in the IEFSSNxx parmlib member, consult subsystem configuration or installation documentation to identify any issues that can exist with running in parallel.

**Note:** BEGINPARALLEL only affects subsystem initialization routines. It has no effect on subsystems that do not specify a routine.





## 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 several 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 addresses the following topics:

- ▶ IBM z/OS Management Facility (z/OSMF)
- ▶ z/OSMF introduction to z/OS V1 R12
- ▶ Browser for z/OSMF
- ▶ z/OSMF components
- ▶ z/OSMF installation
- ▶ z/OSMF customization
- ▶ Focus areas for simplification in z/OSMF V1R12
- ▶ Problem determination in z/OSMF V1R12
- ▶ z/OSMF Workload Management task
- ▶ Resource Monitoring application plug-in
- ▶ z/OSMF administration
- ▶ z/OSMF installation considerations
- ▶ z/OSMF dependencies
- ▶ Migration and coexistence
- ▶ Installation considerations
- ▶ Preparing your workstation for z/OSMF

## 40.1 IBM z/OS Management Facility

Currently there is no central system management portal for z/OS. Instead, many interfaces are unfamiliar to users who are new to the System z platform, manual tasks require extensive documentation, and overall it requires years of z/OS experience to be fully productive.

IBM z/OS Management Facility (z/OSMF), which was a new product with z/OS V1R11 at no additional charge, simplifies, optimizes, and modernizes the z/OS system programmer experience in the following ways:

- ▶ z/OSMF delivers solutions in a task-oriented, web browser-based user interface with integrated user assistance.
- ▶ z/OSMF makes 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.

### 40.1.1 Introduction to the new faces of 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. The goal is to simplify and modernize z/OS management for zSeries IT professionals.

These efforts include the following developments:

- ▶ The creation of new user interfaces (UIs)
- ▶ The enablement of remote access technologies on z/OS
- ▶ Extensions to base operating system components to allow for such management

Marketplace forces, including the aging of the z/OS workforce, added to the difficulty of attracting and training new skills to this platform due to the perception that the management tools were dated, and demanded that the platform improve and modernize accessibility in a way that is common to the rest of the IT industry.

### 40.1.2 z/OS ease-of-use enhancements

To address the skill base for z/OS, a new approach has been developed to increase skills to help less experienced members of the zSeries IT community gain familiarity with the following areas.

#### **z/OS basics**

In collaboration with the ITSO, z/OS development is creating 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 system programmer or system analyst after transferring from another area in your organization, this new series of publications, *z/OS Basics*;, will help you develop your understanding of the various aspects of the z/OS system.

#### **Configuration and operations**

The 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. It is designed to help automate, eliminate, or 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 help 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 have 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 currently required for ordering and delivery of IBM PTFs using current methods.

## **Networking**

To be able to dramatically reduce the amount of time needed to create configuration files, with the z/OS Network Security Configuration Assistant GUI you can 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. You can download the GUI from the Communications Server family downloadable tools web page.

Through a series of wizards and online help panels, you can use the GUI 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 job opportunities. The goal is to increase student awareness of our platform and develop 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, system programmer assistance to instructors for setting up and teaching their classes on z/OS, access to zSeries course material, and opportunities for no-cost faculty education. A current focus of the program is working with zSeries clients to help them find schools to partner with to develop zSeries skills.

## **Library Center for z/OS**

IBM is providing an alternative way to navigate our z/OS library on the Internet. The Library Center for z/OS provides a Windows Explorer-like view of the contents of the entire z/OS and Software Products DVD Collection. The Library Center uses the new IBM Library Server (formerly BookManager® BookServer) with new advanced search functions to help you find exactly the z/OS information you need.

The Library Center contains documentation for z/OS elements, features, software products and selected z/OS-related IBM Redbooks. There are multiple library centers, one for each release, and also one for z/VM.

## Security

The following RACF-based products can help you manage security and monitor compliance:

- ▶ Partnership with Vanguard Integrity Professionals, Inc. includes the following products: Administrator, Advisor, Analyzer, Enforcer, and SecurityCenter
- ▶ IBM Tivoli administration for RACF - providing a lower-function alternative

### 40.1.3 z/OS V1R9 and the new faces of z/OS

With z/OS V1R9, 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 various functions in z/OS V1R9 with a set of new tools enhancing the “MVS face” of z/OS, as shown in Figure 40-1 on page 752, to provide the following new functions.

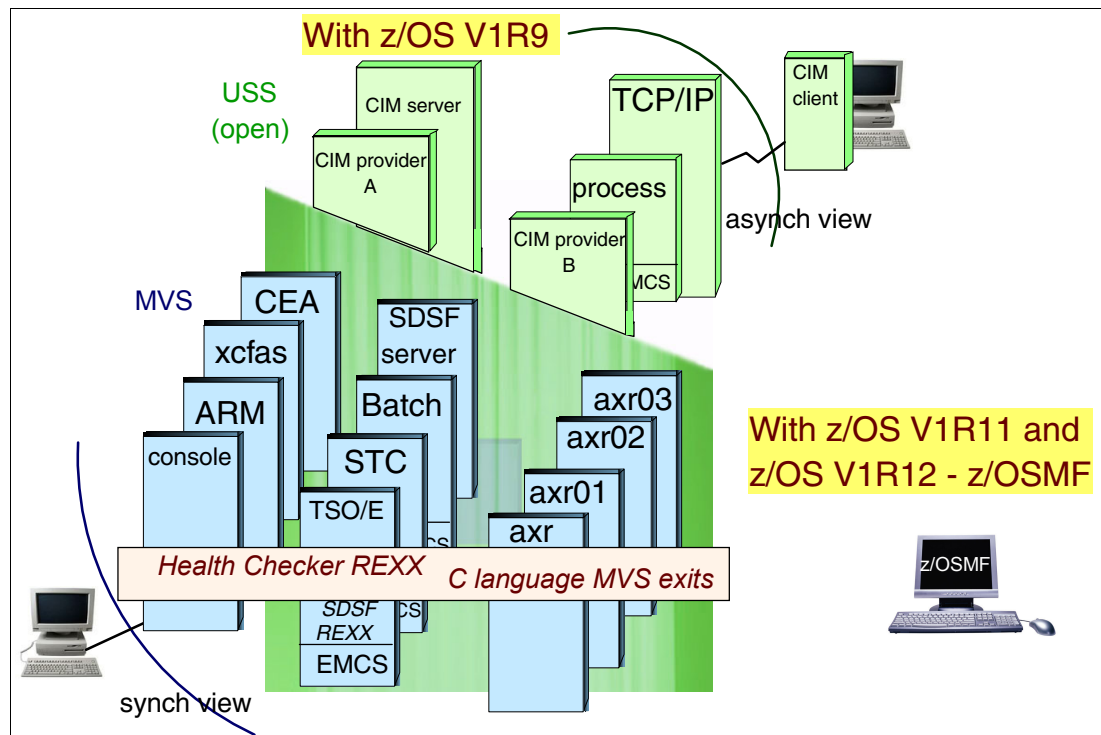


Figure 40-1 New face with z/OS V1R9 and z/OSMF

## 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 can 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.

Since z/OS V1R9, you can access SDSF function with the REXX programming language. Using REXX with SDSF 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 a SYSREXX facility made available with z/OS V1R9. This SYSREXX facility makes it easier for you to write your own 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. It must be up and running for the CIM server to properly operate. The CEA 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 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 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 vendors (including IBM) known as the Distributed Management Task Force (DMTF) as part of the Web Based Enterprise Management (WBEM) initiative. CIM was introduced in z/OS with z/OS V1R7.

The Web Based Enterprise Management (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.

## 40.2 z/OSMF introduction to z/OS V1R12

All the previously reviewed functionalities, introduced along the various z/OS releases (several since z/OS V1R7, others starting with z/OS V1R9), are forming the basis on which z/OSMF has been developed.

Figure 40-2 on page 756 illustrates how z/OSMF provides system management solutions in a task-oriented, web browser-based user interface with integrated user assistance. This allows both new and experienced system programmers to more easily manage the daily operations and administration of the mainframe z/OS systems.

z/OSMF with z/OS V1R12 provides you with a single point of control for:

- ▶ Performing common system administration tasks
- ▶ Defining and updating policies that affect system behavior
- ▶ Performing problem data management

Most functions in z/OSMF are now provided through optional plug-ins that you enable when you configure the product. In general, it is best practice to configure all of the available plug-ins during the z/OSMF configuration process. See “z/OSMF plug-ins” on page 761 for more information about this topic.

**Important:** Installing a new release of z/OSMF will preserve any installation-specific links you have defined. If you fall back from a z/OSMF V1R12 system to a z/OSMF V1R11 system, your link definitions are retained, but are displayed in the Links category, regardless of which categories were specified for the links.

### 40.2.1 z/OSMF system management tasks with V1R12

With z/OSMF, the following system management tasks are available.

- ▶ Configuration Assistant task

This task has new functions with z/OS V1R12 under the Configuration category and is enhanced with new functions:

- 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 task

This task has new functions with z/OS V1R12.

The Incident Log task under the Problem Determination category is enhanced with new functions:

- 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 freeform notes or comments for each incident.

The Incident Log task now 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 setup 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
- ▶ Monitoring Desktops task

This task is new with z/OS V1R12. It 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 task

This task is new with z/OS V1R12. It 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 task

This task is new with z/OS V1R12. It provides a shared location (Settings tab) where you can specify how long to keep the service definition history and 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.

## 40.3 Browser for z/OSMF

Figure 40-2 on page 756 shows a workstation with a browser that communicates with z/OSMF through HTTPs. z/OSMF is a Web 2.0-based solution. It incorporates a browser interface that communicates with the z/OS system. The browser can be located anywhere in the data center or around the world; you simply need a secure connection. Shown at the bottom of the figure is a window capture of the z/OSMF welcome page, which displays after you log into z/OSMF.

z/OSMF allows you to communicate with the z/OS system through a web browser, so you can access and manage your z/OS system from anywhere. z/OSMF manages z/OS from z/OS itself. It is not an external application, and it does not have an external client.

z/OSMF is an application on z/OS with direct access to z/OS data and information, and access is through the browser interface from the workstation. z/OSMF contains the GUIs and the application code. Everything is installed on the z/OS server, and there are no client-side installation requirements.



Figure 40-2 Structure for z/OSMF with z/OS V1R12

### Client machine and browsers

For the client machine (as mentioned, there are no client machine install requirements), you can use Windows XP, Windows Vista, and the Windows 7 operating system.

To access z/OSMF on the z/OS system, your workstation requires one of the following web browsers:

- ▶ Mozilla Firefox Version 3.5
- ▶ Mozilla Firefox Version 3.0 (with a minimum service level of 3.0.15)
- ▶ Microsoft Internet Explorer Version 7
- ▶ Microsoft Internet Explorer Version 8

**Note:** These are the only supported browsers with z/OSMF V1R12.

## 40.4 z/OSMF components

z/OSMF requires an application server and a runtime environment. The application server box shown on the rightmost side of Figure 40-3 on page 757 is really a special version of WebSphere Application Server OEM Edition V7R0 that is packaged together with z/OSMF along with scripts and documentation to make it easier to set up and configure this runtime on z/OS.

After WebSphere Application Server OEM Edition has been set up and installed, the z/OS Management Facility application itself is deployed into this runtime and this is where the application servlets and GUIs reside. z/OSMF uses the Dojo technology for GUIs, which uses JavaScript and helps improve performance overall because the GUI can perform all the graphics rendering in the browser on the workstation.

## z/OS components used by z/OSMF

This application stack communicates with z/OS components. The components can be whatever is applicable for that particular task; there are not technical limitations. The applications utilize the Dojo framework and JavaScript and run on an special version of WebSphere Application Server OEM Edition. The applications exploit functions provided by z/OS system components. Everything is installed on the z/OS server. There are no client-side installation requirements.

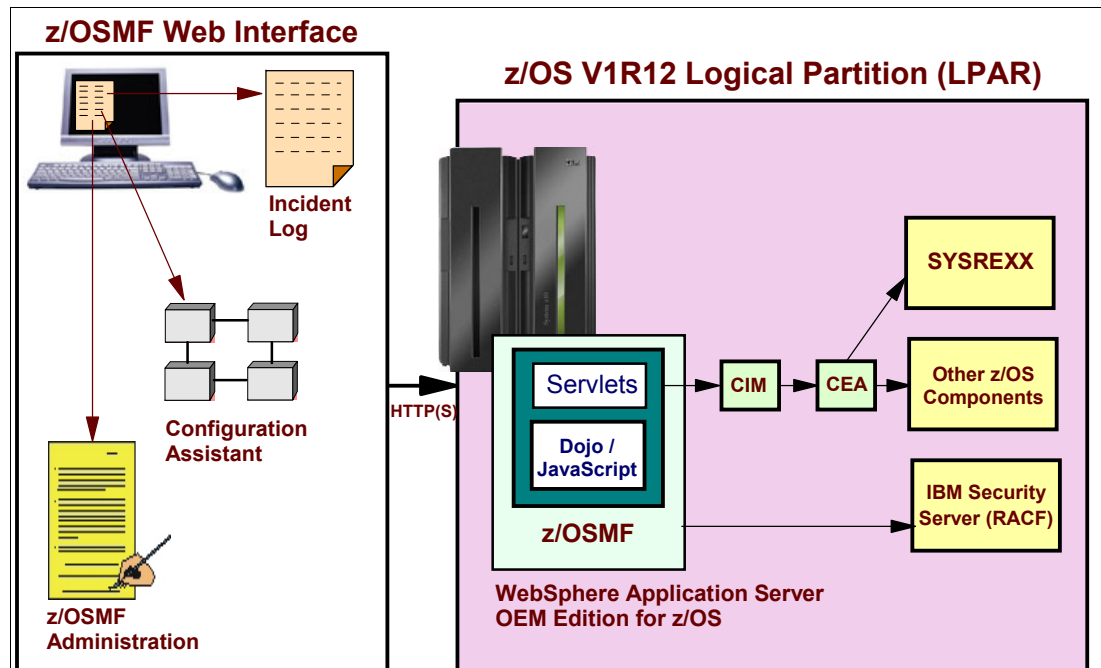


Figure 40-3 z/OSMF components

## z/OSMF and zIIPs

z/OS is updated so that z/OS CIM server processing is eligible to run on the System (zIIP); this includes CIM server and CIM provider workloads. Other CIM-related workloads (such as CIM client and CIM-enabled resource systems processing) are not eligible for zIIP processing. The parts of z/OSMF V1R11 (and V1R12) that use the z/OS CIM Server constitute a workload that is eligible for the zIIP.

## z/OSMF system components

The z/OSMF application is written in Java and is eligible for zAAPs. z/OSMF requires the following components:

- ▶ z/OS Communications Server
- ▶ Security definitions (SAF)
- ▶ System Logger
- ▶ Other components for specific z/OSMF applications

## 40.5 z/OSMF installation

Keep the following considerations in mind when customizing during initialization of z/OSMF.

## 40.5.1 z/OSMF administrator

The z/OSMF administrator ID is created with both TSO and OMVS segments, because you expect someone will eventually log on to TSO with this ID. The TSO logon processing sets up a user's environment, giving them access to the resources they need.

In your installation, you can review the existing logon procedure that system programmers use to logon to TSO, and then either use that procedure or customize a procedure based on it. The z/OSMF scripts or ServerPac jobs only reference the procedure and build the RACF ALTUSER statement with this procedure name. The scripts do not generate the procedure. If you are using another security product, you can use our commands as a template and translate them to the appropriate equivalent.

### **z/OSMF administrator ID**

The values for properties IZU\_ADMIN\_PROC and IZU\_ADMIN\_ACCOUNT are used in creating the z/OSMF administrator ID. These properties are used in the resultant RACF exec that is generated at the end of the -config step, as follows.

<b>IZU_ADMIN_PROC</b>	This specifies the name of the z/OSMF administrator's default logon procedure when logging on through the TSO/E logon panel.
<b>IZU_ADMIN_ACCOUNT</b>	This specifies the z/OSMF administrator's default TSO account number when logging on through the TSO/E logon panel.

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.

## 40.6 z/OSMF customization

In z/OSMF, the term *plug-in* refers to 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. Configuring a plug-in is optional. In general, it is best practice to configure all of the available plug-ins during the z/OSMF configuration process.

Although it is possible to configure z/OSMF without any plug-ins selected, doing so provides only the base functions of z/OSMF (referred to as *core functions* in this document), such as the Welcome task, the Links category, and the online help. The core functions of z/OSMF are always enabled when you configure the product.

z/OSMF V1R12 includes the following plug-ins for your use:

- ▶ The Configuration Assistant plug-in adds the Configuration Assistant task to the Configuration category.
- ▶ The Incident Log plug-in adds the Incident Log task to the Problem Determination category.
- ▶ The Resource Monitoring plug-in adds the Monitoring Desktops and Sysplex Status tasks to the Performance category.
- ▶ The Workload Management plug-in adds the Workload Management task to the Performance category.

**Note:** If you install z/OSMF as part of a ServerPac order using the full system replacement method of installation, this setup work is performed for you during the ServerPac post-installation process.

## 40.6.1 Customizing the Welcome panel for guest users

Your installation can customize the z/OSMF Welcome panel with its own information for guest users. You can do this, 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.

**Important:** To customize the Welcome panel, you can follow the steps we used in writing this book:

1. Copy from `/usr/lpp/zosmf/V1R12/samples/customWelcome.properties` to `/var/zosmf/data/customWelcome.properties`. (In our case, we used the default target location `/var/zosmf/data` when building z/OSMF.)
2. Edit `/var/zosmf/data/customWelcome.properties`. In our case, we updated the last two lines as shown:  

```
header=Welcome to z/OSMF on WTSC80  
footer=Brought to you by the ITS0
```
3. We copied the ITS0 logo .gif file to `/var/zosmf/data/cutsomLogo.gif`, as shown at the bottom of Figure 40-4 on page 760.

Regarding the size of the image, keep in mind that technically, you do not need to know the size of the area for the image to fit.

- ▶ If the image is smaller than the area, then it will remain 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.
- ▶ Because no one wants image distortion, this area is 120 x 40 pixels.

### ITSO customized Welcome window

Figure 40-4 on page 760 shows the Welcome window after being customized. The arrows point to the ITS0 customization changes, as just discussed.

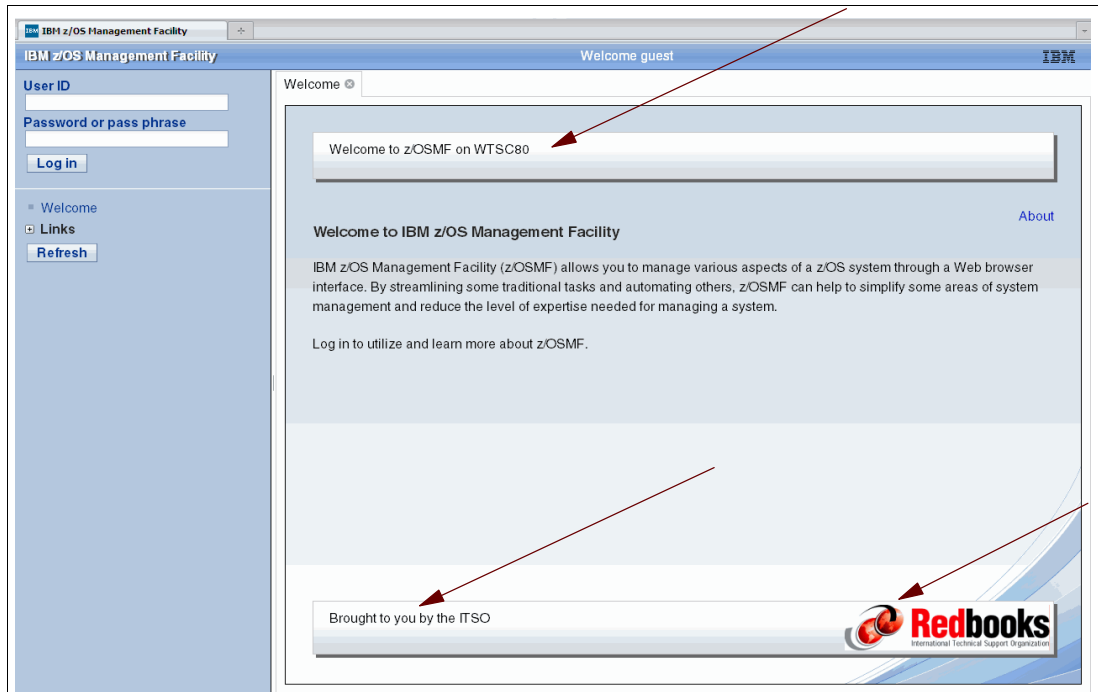


Figure 40-4 ITSO customized Welcome window

## 40.6.2 Accessing z/OSMF

After z/OSMF has been set up, configured, and started on a system, point your browser to the URL for the z/OSMF instance (which is basically the host name, the port name, and the context route for z/OSMF). The window shown in Figure 40-5 will display. Notice that this is the welcome window for a guest (at this point you are still a guest, because you have not logged in to the system yet).

### z/OSMF user access

z/OSMF allows you to communicate with the z/OS system through a web browser, so you can access and manage your z/OS system from anywhere. You can have only one instance of z/OSMF active in a system or sysplex at any given time.

Multiple users can log into the same instance of z/OSMF using various computers, various browsers, or multiple instances of the same browser. Even though you can have only one instance of z/OSMF active in a sysplex, you can create additional instances of z/OSMF on other systems. You might do this, for example, for testing purposes, or for backup in case of system failure.

Logging into z/OSMF requires a user ID and password (or pass phrase) to authenticate. All actions taken from that browser session after successful authentication are performed under the identity that was used to log in. The z/OSMF interface customizes this view, based on the user's role definition. As a result, each user's navigation area displays only those tasks for which the user is authorized.

z/OSMF can use RACF or equivalent security product IDs.

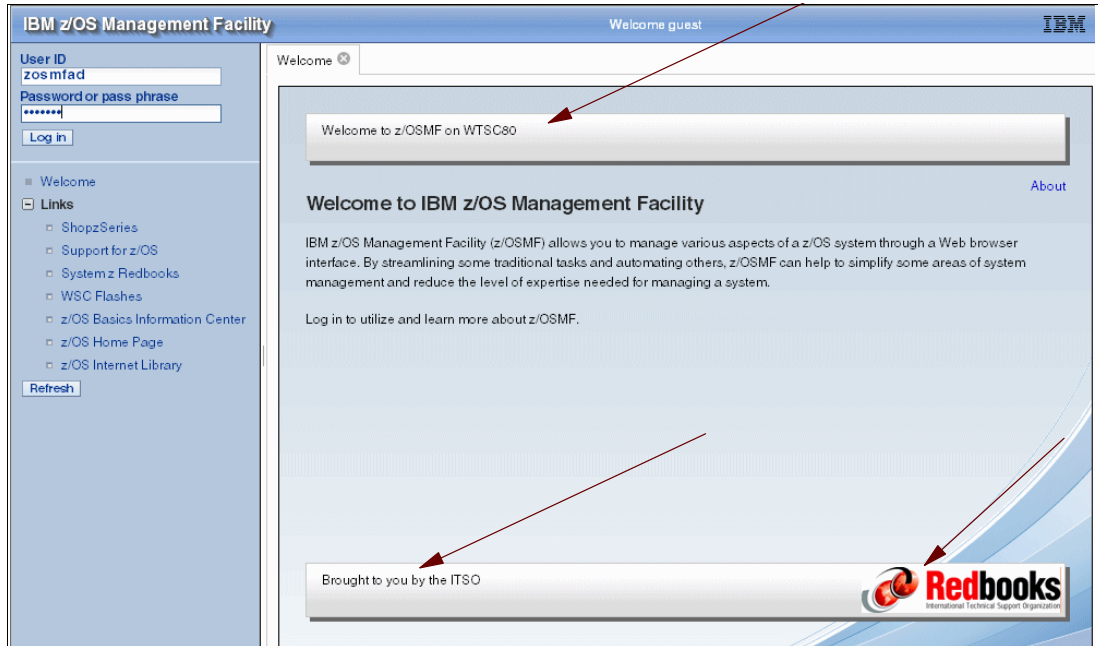


Figure 40-5 z/OSMF customized Welcome window for login

## 40.7 Focus areas for simplification in z/OSMF V1R12

Focusing on z/OS system programming, and drawing on feedback from clients, the initial focus areas for simplification for z/OSMF in V1R12 are problem management analysis and configuration. z/OSMF simplifies and modernizes the user experience and helps make pertinent information readily available and easily accessible.

Other areas, such as security administration, network administration, storage management, and workload management, might be addressed in the future as specialized and separate tasks are added to the z/OSMF infrastructure.

When configured, z/OSMF includes one user ID that is authorized to access z/OSMF as the product administrator. With this user ID, the administrator can log into z/OSMF and add more users and administrators as needed.

**Note:** If you are migrating to a new release of z/OSMF, you can reuse much of the customization from your current configuration.

### 40.7.1 z/OSMF plug-ins

As mentioned in 40.6, “z/OSMF customization” on page 758, in z/OSMF, a plug-in is a collection of one or more system management tasks. With z/OS V1R12, this feature allows you to add functions to z/OSMF by configuring optional plug-ins. Most functions in the z/OSMF navigation area are provided through plug-ins, which you enable when you configure the product. Which z/OSMF setup actions you need to complete depend on which plug-ins you choose to deploy in z/OSMF.

z/OSMF V1R12 includes the following plug-ins for your use:

- ▶ The Configuration Assistant plug-in adds the Configuration Assistant task to the Configuration category.
- ▶ The Incident Log plug-in adds the Incident Log task to the Problem Determination category.
- ▶ The Resource Monitoring plug-in adds the Monitoring Desktops and sysplex status tasks to the Performance category.
- ▶ The Workload Management plug-in adds the Workload Management task to the Performance category.

**Note:** Configuring a plug-in is optional. As mentioned, it is best practice to configure all of the available plug-ins during the z/OSMF configuration process.

In terms of system resources, the cost of a plug-in is negligible. If you add a plug-in after completing the configuration process, you are required to repeat most of the steps in configuring z/OSMF. If you add a plug-in later, you can use the `izusetup.sh` script to add the plug-ins. Here, you will configure plug-ins individually, repeating most of the steps you followed earlier to configure z/OSMF.

## 40.7.2 z/OSMF login

Figure 40-6 on page 762 shows the full scope of what z/OSMF provides in V1R12 after you log in. When z/OSMF is first set up, the first user ID to log in must always be an administrator; this is a requirement for setup. It allows the first person to get in and to add and enable others.

Notice that Figure 40-6 is the welcome window for a user or an administrator. The navigation panel on the left is populated with multiple categories, with tasks under each category.

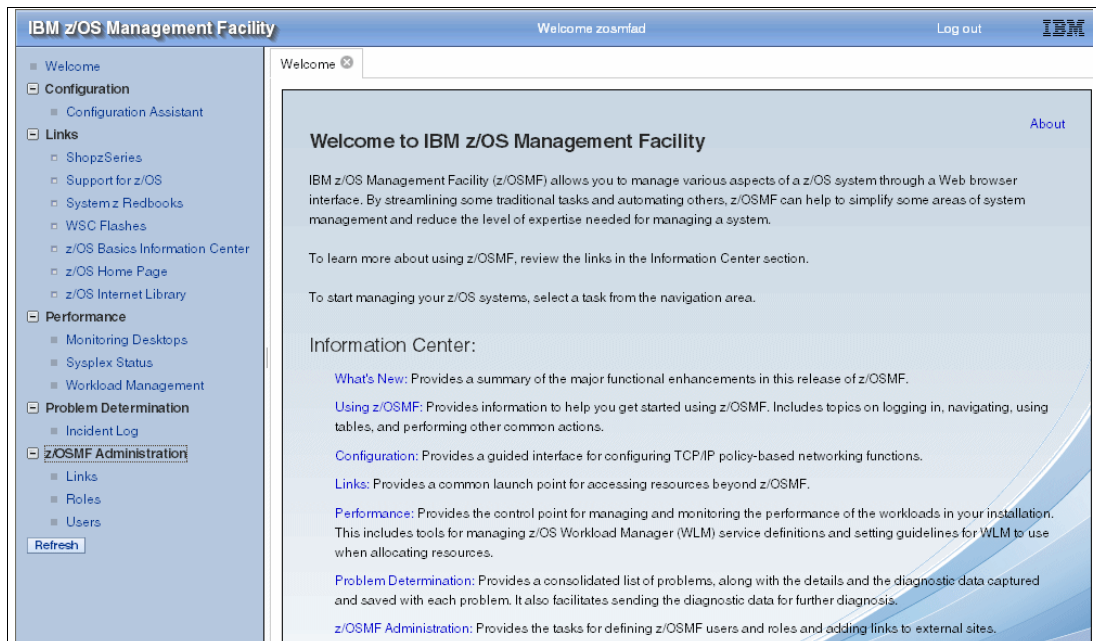


Figure 40-6 z/OSMF full scope view of Welcome window



### 40.7.3 Configuration Assistant

IBM provides a configuration graphical user interface (GUI) that you can use to generate configuration files for Communications Server for z/OS. Using 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.

The Configuration Assistant task under the Configuration category is enhanced with new functions in support of z/OS V1R12:

- ▶ 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

Selecting the Configuration Assistant button in Figure 40-6 on page 762 displays Figure 40-7, which allows you to define a simplified configuration and setup of TCP/IP policy-based networking functions.

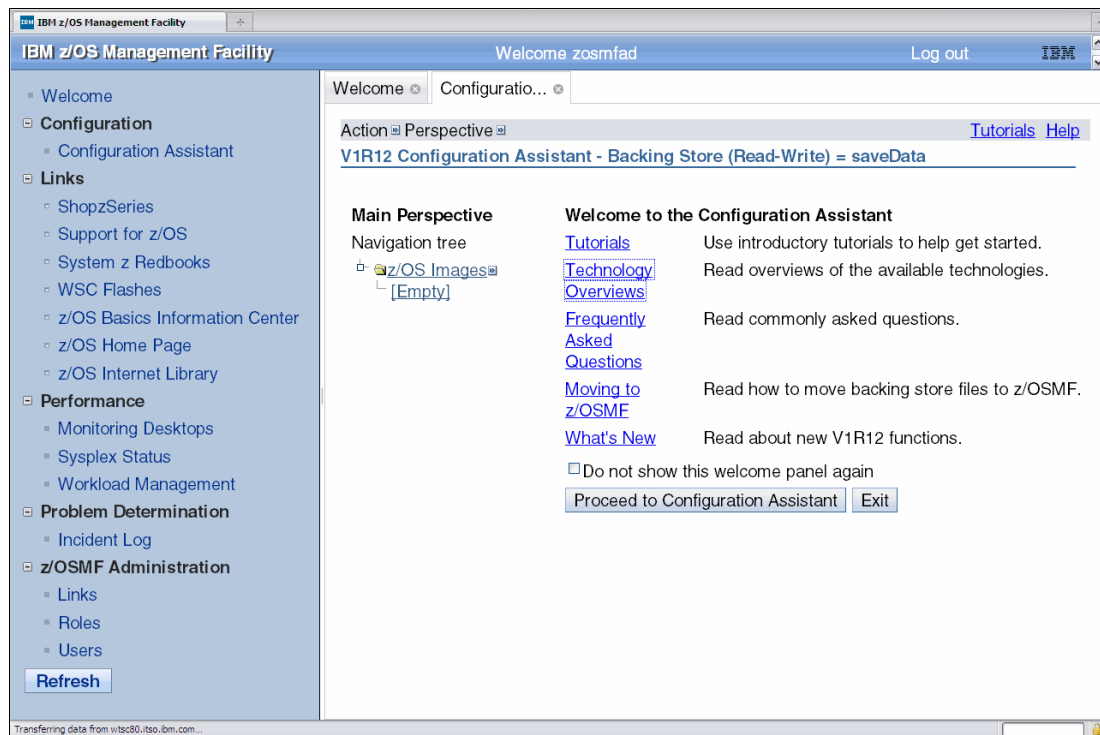


Figure 40-7 Configuration Assistant

## 40.7.4 Links category with a set of links

The Links category with a set of links gives access to resources providing a common launch point for accessing resources beyond z/OSMF. Note the following points:

- ▶ Various links are predefined in the product.
- ▶ Administrators can define additional links to share commonly used resources for their installation.

## 40.7.5 Performance category (V1R12)

In z/OSMF V1R12, a new category known as the Performance category is added to the z/OSMF navigation area, as shown in Figure 40-6 on page 762. The tasks in this category will help you to manage and monitor your installation's performance. In z/OSMF V1R12, the Performance category can contain the following tasks:

<b>Workload Management task</b>	This task manages z/OS Workload Manager (WLM) service definitions and provides guidelines for WLM to use when allocating resources.
<b>Monitoring Desktops task</b>	This task monitors the performance of the z/OS sysplexes and Linux images (System z and Intel) in your installation. Specifically, you can use it to monitor most of the metrics supported by RMF Monitor III, create and save custom views of the metrics, and display real-time performance data as bar charts.
<b>Sysplex Status task</b>	This task assesses the performance of the workloads running on the z/OS sysplexes in your environment. The Sysplex Status task also provides you with a single location where you can define the z/OS sysplexes and Linux images to be monitored in the Monitoring Desktops task.

For resource monitoring of Sysplex Status and Monitoring Desktops, the z/OSMF Resource Monitoring application provides integrated performance monitoring of z/OS sysplexes in the client's environment.

## 40.7.6 Problem Determination category

The Incident Log application will help all system programmers with problem data management tasks. It provides experienced teams with procedural advantages through an incident log summary and detailed views of z/OS dump incidents.

The Incident Log provides a consolidated list of SVC Dump-related problems, along with details and diagnostic data captured with each incident. It also facilitates sending the data for further diagnostics.

## 40.7.7 z/OSMF Administration category

The z/OSMF Administration category contains the product administration tasks; these are typically performed by the z/OSMF administrator at your installation.

- ▶ The Links task allows the administrator to define links for other external web applications and websites that users can launch from z/OSMF. Links can be added to any category in the z/OSMF navigation area.

- ▶ The Roles task allows the administrator to modify z/OSMF roles.
- ▶ The Users task allows the administrator to define new z/OSMF users and to modify or delete existing z/OSMF users.

The z/OSMF authorization services for administrators allows an installation to provide the following new functions:

- ▶ Display a welcome page and deployed z/OSMF applications in a task list
- ▶ Authorization services for the administrator
- ▶ Allow dynamic addition of links to non-z/OSMF resources

## 40.8 Problem determination in z/OS V1R12

The Incident Log has been available since z/OSMF in z/OS V1R11. With that release, you can display lists of incidents and work with incidents like filtering and sorting because they are in a table format. The Incident Log provides a summary view of all problems that are occurring, which makes it easy to find information about all abends that can occur on your sysplex across all components.

The focus on problem determination with z/OSMF makes troubleshooting your system easier and faster. The incident log and z/OS diagnostic data gathering greatly improves tasks related to:

- ▶ Identifying system-detected problems (those related to SVC dumps taken by the system)
- ▶ Collecting diagnostic materials related to a problem and sending materials to IBM or another company's support area
- ▶ Directing the system to perform a system dump for a previously-recognized problem

### **z/OS V1R12 improvements**

With z/OS V1R12, a number of improvements to the Incident Log function will help you to manage problem data more easily. The following areas have been improved:

- ▶ Support for encryption of all incident files, including dumps, to be sent to IBM in parallel through FTP to save time.

Parallel FTP of dumps that breaks dumps into multiple data sets that can be sent through FTP in parallel to reduce transmission time.

- ▶ Ability to specify additional data sets to an incident and add free-form comments to new fields for problem descriptions and FTP destinations.

The creation of diagnostic log snapshots is based on the SYSLOG and LOGREC data sets, and on 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.

### 40.8.1 Using the z/OSMF Incident Log

Using the z/OSMF Incident Log improves problem determination and troubleshooting for system programmers. To use the Incident Log task, you must ensure that a number of z/OS components and facilities are enabled on your system. These functions support the collection of diagnostic data and the creation of diagnostic logs.

The Incident Log task allows you to view additional information about an incident, such as:

- ▶ The symptom string
- ▶ Reason codes
- ▶ A list of collected diagnostic data

An incident can represent a multisystem SVC dump, which consists of a primary dump and multiple secondary dumps taken on other systems in the sysplex. In such cases, the Incident Log task shows all the SVC dumps associated with the incident, with a single set of diagnostic snapshot files.

The Incident Log task allows you to specify additional information that you want to track about an incident, such as who is assigned to resolve the issue, which business applications are impacted, which component is the source of the issue, and which solution has been implemented.

**Important:** In z/OSMF V1R12, the CIM server must be active on your system if you plan to use the Incident Log task or the Workload Management task. You must ensure that the server is active before continuing to the next step. To verify that the CIM server is started, enter the `D A,CFZCIM` command.

## 40.8.2 Transmitting data to IBM

To transmit data to IBM, attach additional files to send with an incident. The Incident Log task allows you to specify additional diagnostic data files to send with an incident. You can attach z/OS UNIX files and cataloged sequential data sets.

### Send and encrypt diagnostic data files

The Incident Log task provides a Send Diagnostic Data wizard that you can use to send diagnostic data to IBM or another FTP destination. You can send the system-supplied and user-supplied diagnostic data files by using one of two methods:

- ▶ Unencrypted standard File Transfer Protocol (FTP)
- ▶ Parallel FTP with the option of encryption. This method is available if the z/OS Problem Documentation Upload Utility is installed and is configured correctly.

## 40.8.3 z/OS Problem Documentation Upload Utility

The z/OS Problem Documentation Upload Utility is a parallel FTP utility that is designed to send documentation in a more efficient manner to IBM FTP sites. The utility 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 Problem Documentation Upload Utility web page:

<ftp://ftp.software.ibm.com/s390/mvs/tools/mtftp/>

If the tool is installed and properly configured, z/OSMF uses its parallel FTP and encryption capabilities. The tool must be installed on both the z/OSMF host system and the FTP destination system. Otherwise, the send will fail or the destination might not be able to decrypt, reassemble, or read the files. For information and documentation regarding the tool, see Appendix C, “z/OS Upload utility” on page 815.

**Important:** Various z/OSMF tasks, such as the Incident Log, use FTP to transmit data. If your network contains a firewall that blocks FTP traffic or does not allow authentication using FTP, you must perform an additional action to allow the traffic to pass.

## 40.8.4 Implementing the Incident Log

An *incident* is a potential system problem that has occurred. Currently, the Incident Log model supports ABENDs and user incidents as incidents. z/OSMF retrieves the incidents from the system and displays them in the Incident Log table shown in Figure 40-8 on page 767.

A maximum of 500 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.

The Incident Log table shows the summary list of incidents from systems in the sysplex. Also displayed are fields related to each incident, such as system, sysplex, z/OS release and component, and two other fields:

- ▶ The problem number and tracking ID.
- ▶ They can be set for individual incidents; most of the other fields are fixed and are really properties of that incident.

Incident Type	Component ID	Component Name	Date and Time (GMT)	Description
User Initiated			Aug 11, 2010 2:14:14 PM	IOSAS & DEVMAN
User Initiated			Aug 11, 2010 2:10:26 PM	IOSAS AND DEVMAN
ABEND S00C4	5752SC1C3	IOS	Jul 23, 2010 12:22:48 AM	COMPON=IOS,COMPID=SC1C3,ISSUER=H AUTOCONFIG HOM RTN
ABEND S00C0	5752SC1C3	IOS	Jul 20, 2010 7:58:52 PM	COMPON=IOS,COMPID=SC1C3,ISSUER=H DISCOVER/AUTOCFG MAIN
ABEND S00C0	5752SC1C3	IOS	Jul 20, 2010 7:58:52 PM	COMPON=IOS,COMPID=SC1C3,ISSUER=H DISCOVER/AUTOCFG SVCS
ABEND S00C0	5752SC1C3	IOS	Jul 20, 2010 7:48:38 PM	COMPON=IOS,COMPID=SC1C3,ISSUER=H DISCOVER/AUTOCFG MAIN
ABEND S00C0	5752SC1C3	IOS	Jul 20, 2010 7:48:38 PM	COMPON=IOS,COMPID=SC1C3,ISSUER=H DISCOVER/AUTOCFG SVCS
ABEND S00C0	5752SC1C3	IOS	Jul 20, 2010 7:41:10 PM	COMPON=IOS,COMPID=SC1C3,ISSUER=H DISCOVER/AUTOCFG MAIN
ABEND S00C0	5752SC1C3	IOS	Jul 20, 2010 7:41:10 PM	COMPON=IOS,COMPID=SC1C3,ISSUER=H DISCOVER/AUTOCFG SVCS
ABEND S00C0	5752SC1C3	IOS	Jul 20, 2010 7:38:29 PM	COMPON=IOS,COMPID=SC1C3,ISSUER=H DISCOVER/AUTOCFG MAIN
ABEND S00C0	5752SC1C3	IOS	Jul 20, 2010 7:38:29 PM	COMPON=IOS,COMPID=SC1C3,ISSUER=H DISCOVER/AUTOCFG SVCS

Figure 40-8 Incident log summary information

Right-clicking the panel shown in Figure 40-8 on page 767 displays the Set Tracking ID drop-down; see Figure 40-9 on page 768.

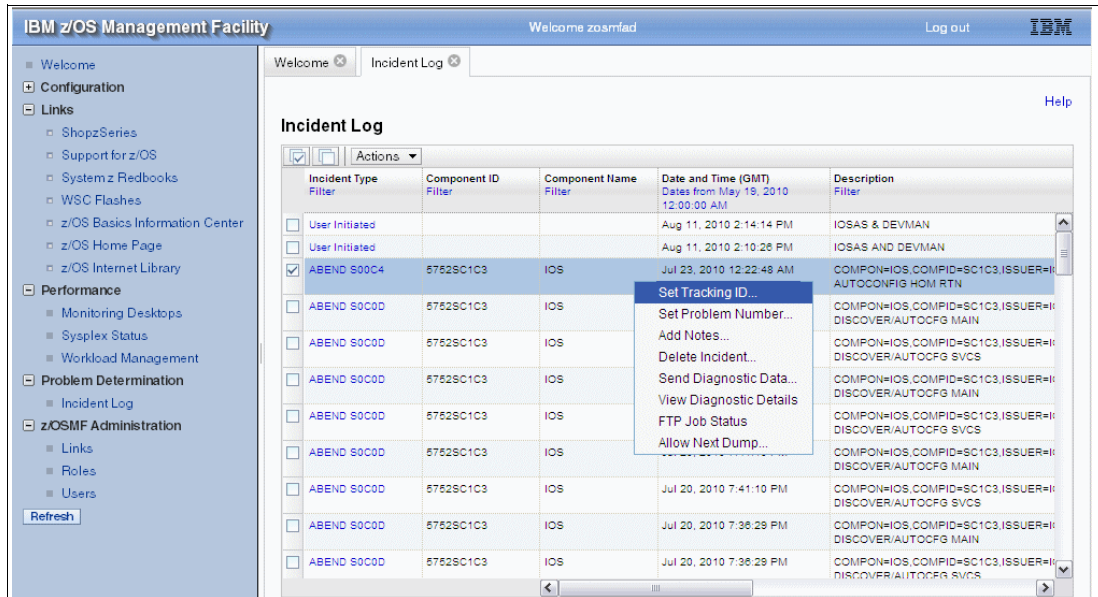


Figure 40-9 Set Tracking ID drop-down

## Setting a problem number

The Incident Log task lets you correlate an incident with a problem number or tracking ID. The problem number allows you to associate an incident with an IBM problem number (such as a PMR or an ETR number) or an ISV problem number. With the tracking ID, you can associate an incident with a problem record in your installation's problem management system. The intent is to give it an external problem number that is meaningful as a tracking number used to refer to a problem reported to a service provider.

**Note:** IBM FTP servers use the PMR or ETR number to link the files sent with the PMR or ETR you created. If the PMR or ETR number is incorrect, the files might be lost because they cannot be linked to a PMR or ETR. z/OSMF does not check the validity of the problem number. It checks only for syntax errors.

You can set a problem number by right-clicking an incident. A box will appear, as shown in Figure 40-10 on page 769, and you can set the problem number there, or from another panel. The point of having a tracking ID is so you can relate it to a particular incident with your internal tracking system.

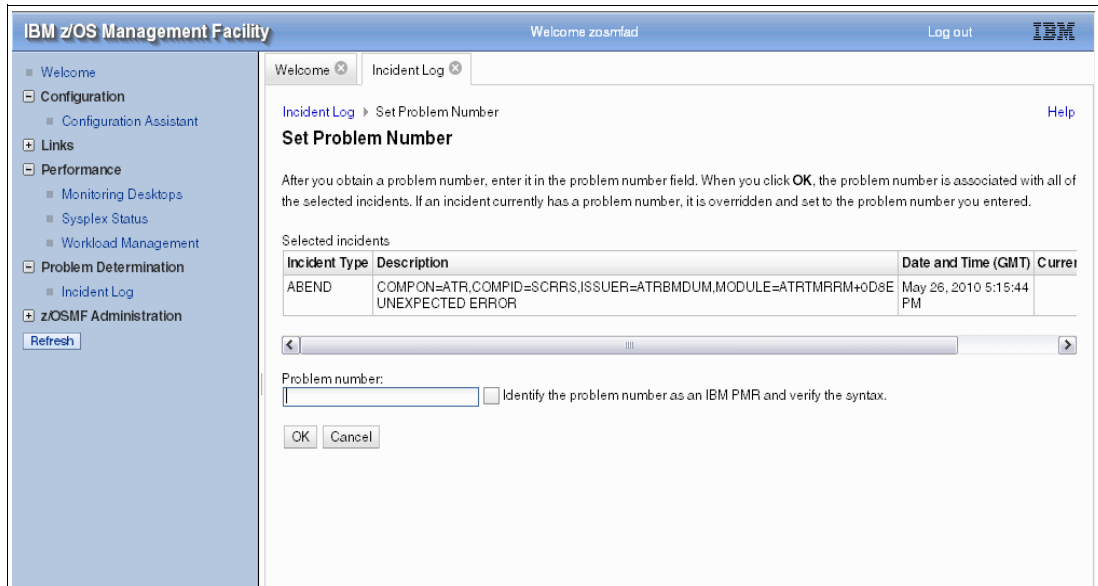


Figure 40-10 Set a problem tracking number

## Managing an incident

You can perform various tasks on an incident. For example, an incident can be deleted. You can send the diagnostic data and view the diagnostic details. You can view the FTP job status, so if you send the diagnostic data, you can view the FTP status for that submitted job.

The Actions drop-down menu from the taskbar at the top, shown in Figure 40-11 on page 769, allows you to perform other tasks, such as working with tables, filtering, sorting, and configuring columns. You can also sort within the table by clicking the columns. You can filter using a single filter or multiple filters for more complex sorts.

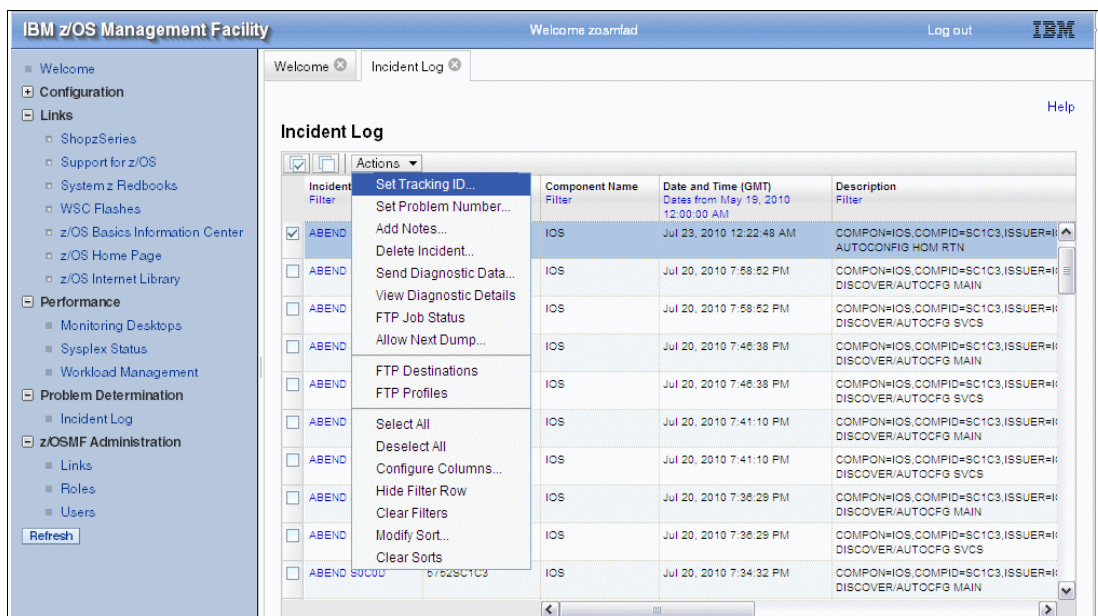


Figure 40-11 Actions drop-down - Incident Log



## Viewing Incident details

The View Diagnostic Details pull-down menu, shown in Figure 40-11 on page 769, displays all available information about the incident including a reformatted version of the summary data in the Incidents table.

You can use the Diagnostic Data tab on the View Diagnostic Details panel to view a list of the system-supplied diagnostic data files and to specify additional files to send with the incident. When an incident occurs, the system typically creates an SVC dump and collects diagnostic log snapshots of the operations log, error log, and error log summary. If a multisystem dump was taken, an incident can have multiple SVC dumps associated with it. For incidents that occurred on systems running z/OS V1R9 or earlier, only the SVC dump (if it is stored in a shared pack) is captured.

From the panel shown in Figure 40-11 on page 769, select **View Diagnostic Details**, and Figure 40-12 on page 770 is displayed. Figure 40-13 is a sample view of diagnostic details taken from the summary information.

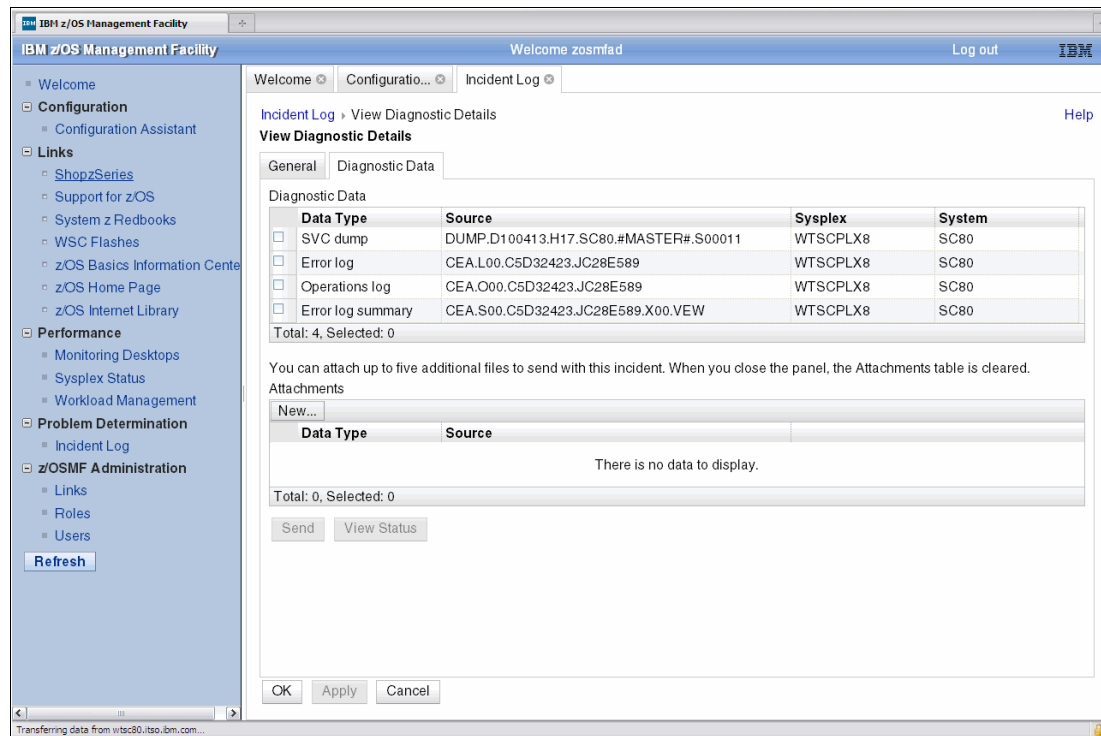


Figure 40-12 Incident Log View Diagnostic Details panel

The Diagnostic Data shows the lists of data sets, including the LOGREC, OPERLOG, and error log that has been captured for this incident. These data sets are shown as they are called in z/OS. You can selectively send diagnostic data if required. When you send diagnostic data from the incident log panel, it sends all of the data associated with the incident. As an option, you can send only subset.

When you select the General button shown in Figure 40-12 on page 770, then Figure 40-13 on page 771 is displayed. Figure 40-13 shows a sample view of diagnostic details taken from the summary information. You see the details about the dump in addition to two editable fields:

- ▶ The problem number
- ▶ The tracking number



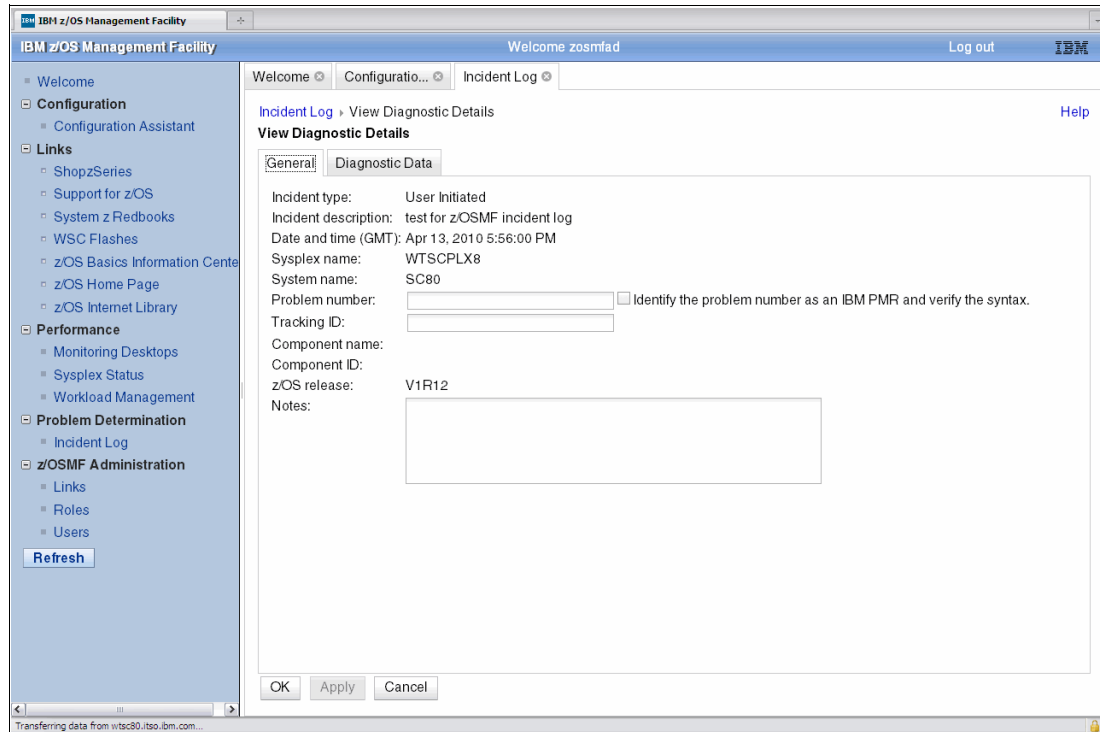


Figure 40-13 Incident log showing incident details

## Sending data to IBM

z/OS V1R12 has added the ability to attach additional diagnostic data to be sent to the FTP destination. The send action is a guided wizard depicted in Figure 40-14 on page 772. It shows the data, allows you to select destination, associate that with a profile, and allows you to edit the JCL that is built, namely the job card information, because each client has various requirements for the job card. After the job card is edited, it will be persistent from that point on until you edit it again. Then the JCL is built and away it goes.

With z/OSMF for z/OS V1R12, this is new support for:

- ▶ The encryption of all incident files, including dumps, to be sent to IBM
- ▶ The parallel FTP of dumps, which gives the capability of breaking dumps into multiple data sets that can be sent through FTP in parallel to reduce overall transmission time

**Note:** The Incident Log task provides a Send Diagnostic Data wizard that you can use to send diagnostic data to IBM or another FTP destination. You can send the system-supplied and user-supplied diagnostic data files using one of two methods: unencrypted standard File Transfer Protocol (FTP) or parallel FTP with the option of encryption. Parallel FTP with the option of encryption is available if the z/OS Problem Documentation Upload Utility is installed and configured correctly.

If the tool is installed and properly configured, z/OSMF makes use of its parallel FTP and encryption capabilities. The tool must be installed on both the z/OSMF host system and the FTP destination system. Otherwise, the send will fail or the destination might not be able to decrypt, reassemble, or read the files.

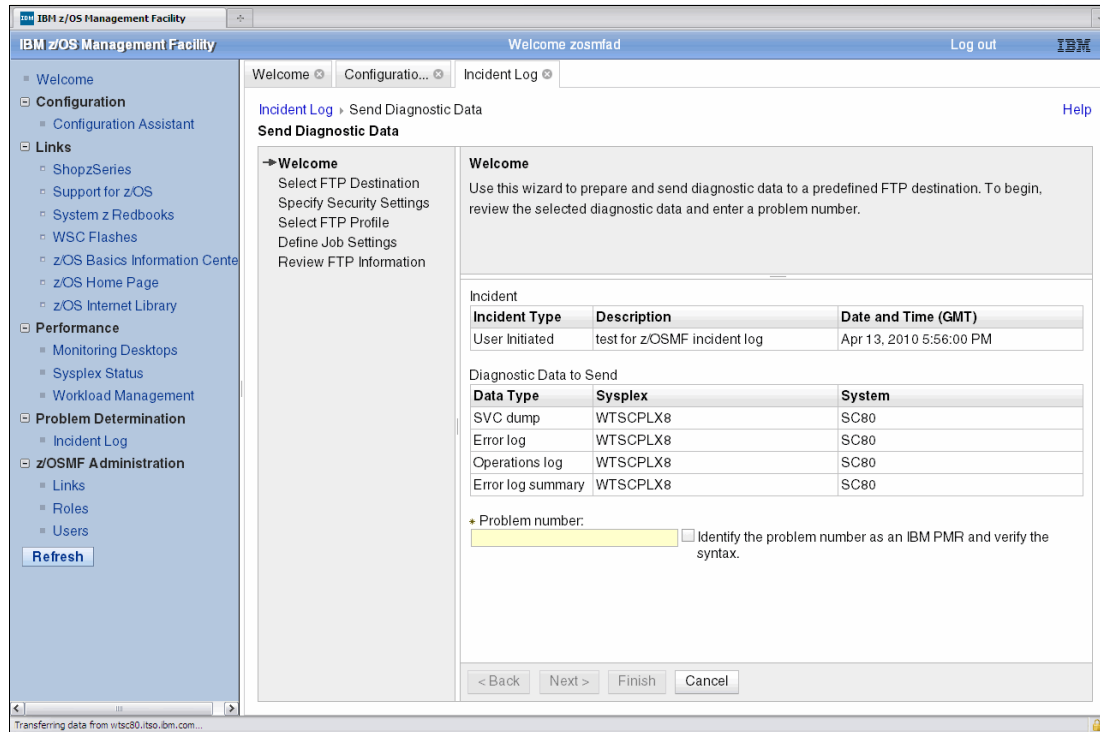


Figure 40-14 Send diagnostic data

## FTP destinations

Figure 40-15 on page 773 shows an example of FTP destinations. IBM has predefined four destinations (the top four shown), such as for MVS and Tivoli. You have the ability to add more destinations to IBM, independent software vendors (ISVs), or elsewhere. You can set up the destination so that it is anonymous, or requires an ID and password. You can also define firewall profiles and associate a firewall with a particular destination.

Users can define destinations where incident diagnostic data can be sent through FTP. They can also define FTP profiles for firewall definition and associate them.

When the send action is initiated, the wizard guides you through the steps to send diagnostic data over to IBM, an ISV, or any other destination. It prepares (terses) the information needed to send the data to IBM (or other target) and initiates the FTP action.

## Managing FTP destinations and profiles

To send diagnostic data in z/OSMF, you must define both the destination to which the data is being sent and the firewall or proxy that the data crosses. The Incident Log task allows you to define the destinations on the FTP Destinations panel and the firewalls or proxies on the FTP Profiles panel.

The Incident Log task provides substitution variables that you can use when specifying the information required by your firewall or proxy. For a list of the substitution variables, see the FTP profile substitution variables help topic provided in the z/OSMF online help. To access the help, click the Help link, which is located in the upper rightmost corner of each panel.

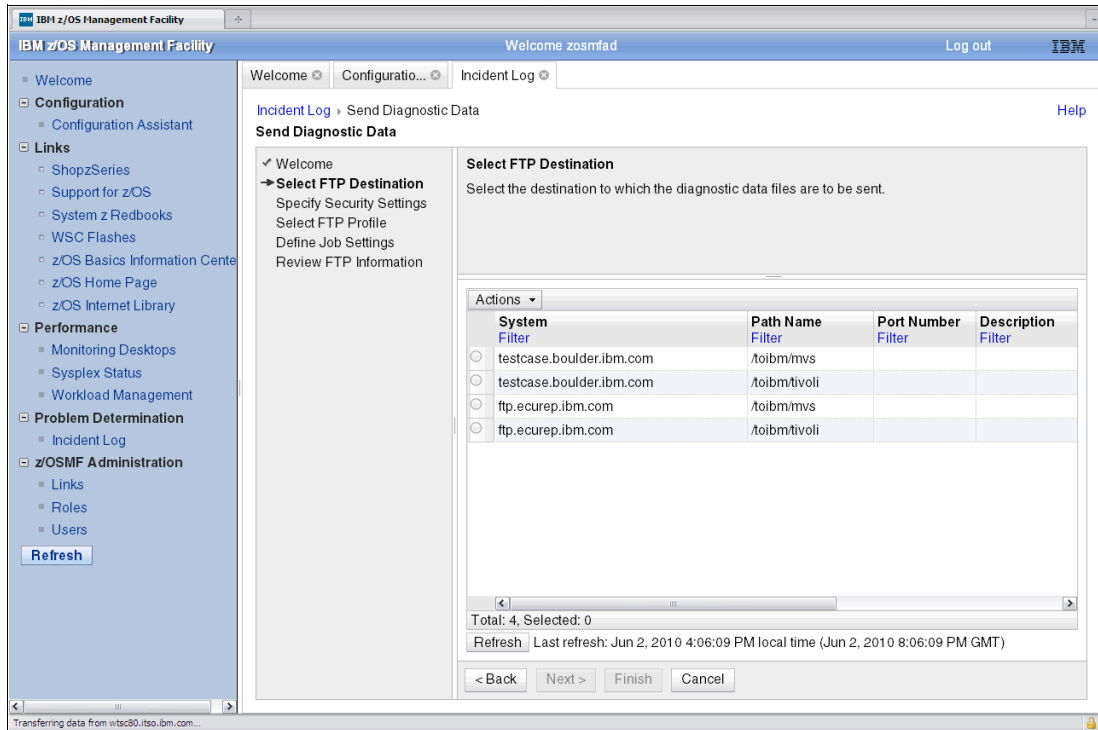


Figure 40-15 Incident Log destinations

## Sending data with encryption

With z/OSMF V1R12, the Incident Log will provide support for encrypted and parallel FTP, as depicted in Figure 40-16 on page 774. If the installation uses the Problem Document Upload Utility and has it set up, you can use it to send diagnostic data to IBM. When defining a FTP destination, indicate then that the utility is to be used, as shown by the button in Figure 40-16 on page 774.

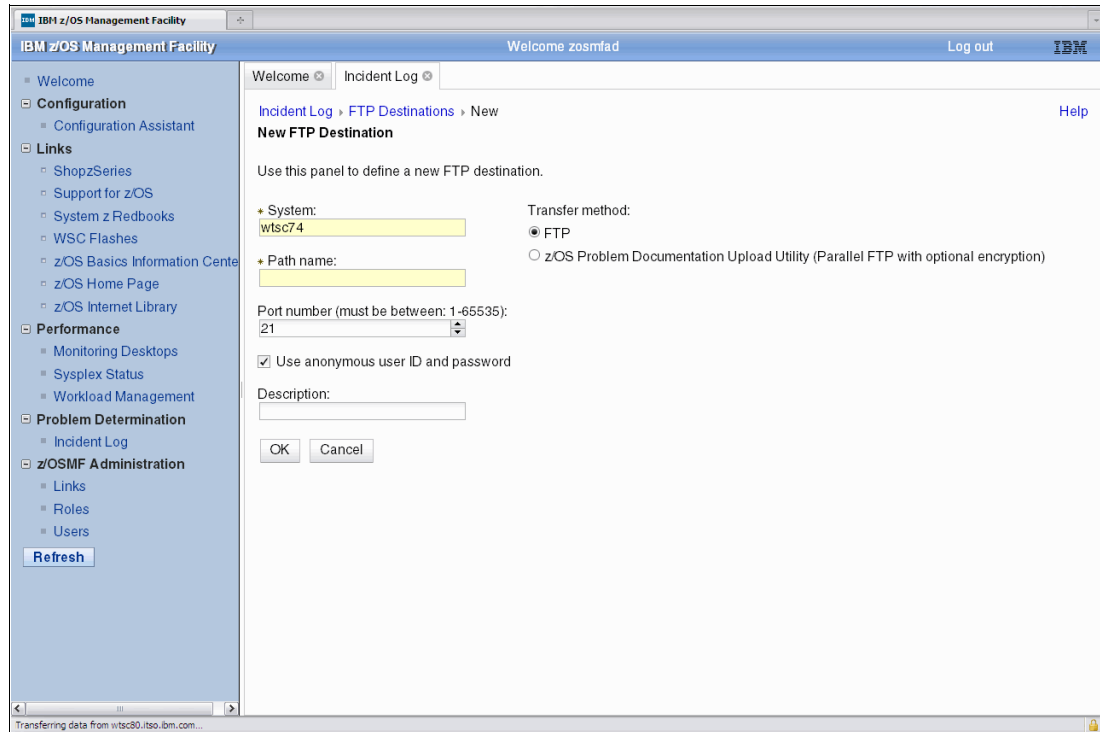


Figure 40-16 Defining a new FTP destination

## 40.9 z/OSMF Configuration Assistant

The IBM Configuration Assistant for z/OS Communications Server, an optional GUI-based tool, can also be 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 dramatically reduce the amount of time required to create network configuration files.

The IBM Configuration Assistant for z/OS Communications Server provides a guided interface for configuring TCP/IP policy-based networking functions. You can use the Configuration Assistant to generate the Policy Agent files.

**Note:** You can use the z/OS Communications Server Configuration Assistant on your workstation and then later migrate your work to the z/OSMF Configuration Assistant environment.

### z/OSMF V1R12 enhancements

The following new functions are available for z/OSMF V1R12:

- ▶ Supporting the configuration of IKE version 2
- ▶ Supporting the configuration of new cryptographic algorithms for IPSec and IKE
- ▶ Supporting the configuration of FIPS 140 cryptographic mode for IKE
- ▶ Supporting the configuration of certificate trust chains and certificate revocation lists

Figure 40-17 on page 775 shows what the Configuration Assistant for the z/OS Communications Server looks like on z/OSMF. As can be seen, it lists all the various

functionalities that the Communications Server can create and activate the corresponding policies when the Configuration Assistant is entered. From this panel you can create the z/OS images and TCP/IP stacks for the systems that you want to configure for any of the supported functionalities. Select the TCP/IP stack that you want to configure and the technology such as IP Sec or AT-TLS. Click **Action** and select **Configure** to begin configuring that functionality.

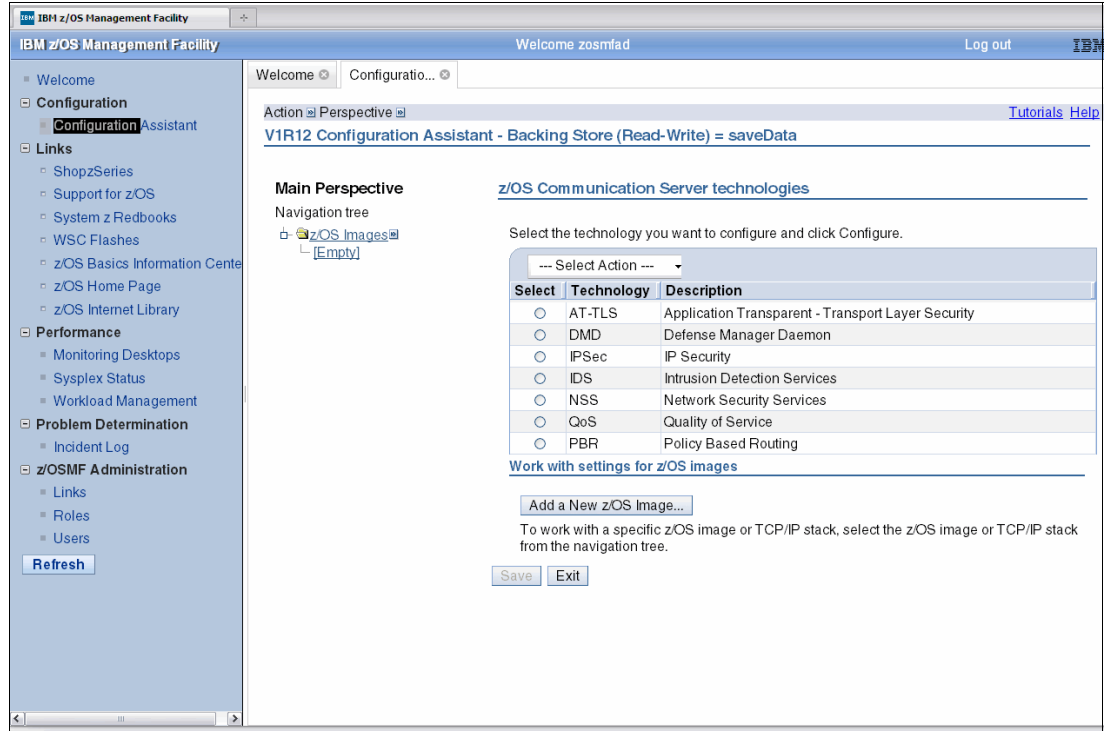


Figure 40-17 Configuration Assistant for z/OS Communications Server

## 40.9.1 Implementing Configuration Assistant

The Configuration Assistant is a stand-alone application that runs on the Windows operating system and requires no network connectivity or setup to begin using it. 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 to implement AT-TLS, IPSec, NSS, PBR, QoS, and IDS. It can dramatically reduce the amount of time required to generate and maintain policy files for these functionalities. The Configuration Assistant is intended to replace manual configuration of the policy functionality, but it can also incorporate policy data directly from the policy agent.

When selecting the plug-in, using the Configuration Assistant task, you can do the following:

- ▶ Configure new policies for the following TCP/IP, policy-based networking disciplines:
  - IP Security, including IKE
  - Network Security Services (NSS)
  - Defense Manager daemon (DMD)
  - Application Transparent TLS (AT-TLS)
  - Intrusion Detection Services (IDS)
  - Policy-based Routing (PBR)
  - Quality of Service (QoS)

- ▶ Import previously defined policies for IP Security, AT-TLS, IDS, and PBR
- ▶ Review Application Setup Tasks containing detailed instructions for getting a supported policy discipline up and running

**Configuring a plug-in:** The `izusetup.sh` script resides in the `/bin` subdirectory of the z/OSMF product file system. By default, this is `/usr/lpp/zosmf/V1R12/bin`. This location is set through the `IZU_CODE_ROOT` variable in your global settings. Indicate which plug-ins are to be added in your override file, which you specify on the `-overridefile` option. In the override file, mark the plug-ins to be added with the character `A`.

If you omit the override file, the script prompts you for the plug-ins to be added. You are prompted only for plug-ins that are not already deployed in your configuration. Respond to the prompt with one or more of the following plug-in identifiers:

- 1 Incident Log plug-in
- 2 Configuration Assistant plug-in
- 3 Workload Management plug-in
- 4 Resource Monitoring plug-in
- N None (no plug-ins)

## Policy-based networking functions

z/OS Communications Server provides an extensive set of publications online to help you configure your z/OS systems for the policy-based networking functions. It is useful to use the Communications Server Configuration Assistant for z/OS because it can save you a significant amount of time.

The Configuration Assistant helps users build their networking policies, and then generates configuration text files for installation. It guides users through setup tasks for the policy-base environment, including generation of configuration files, sample started procedures, and RACF profiles.

The Configuration Assistant guides you through the configuration of the following technologies:

- ▶ IP Security
  - Define policy rules to permit and deny TCP/IP traffic with IP filters and rules to protect your enterprise data with IP Security.
- ▶ AT-TLS
  - Define policy to protect your applications using Transport Layer Security.
- ▶ Intrusion Detection Services
  - Protect your z/OS system from potential misuse of critical system resources.
- ▶ Policy-based routing
  - Define policy to route TCP/IP traffic outbound over selected interfaces.
- ▶ Quality of Service
  - Define policy to ensure levels of service.

In addition of helping you to configure these functions, the Configuration Assistant can help to set up the applications and tasks needed to run these functions. If you have already tackled configuring these functions and have policy configuration files, the Configuration Assistant can read your configuration text files and they can start to be used from then on.

## Using Configuration Assistant

The Configuration Assistant can currently be downloaded to be run as an application on a Microsoft Windows workstation. It has been available since z/OS V1R7, but clients really wanted it to be running on z/OS. The Windows-based Configuration Assistant requires that not only the tool but also the z/OS network policies be downloaded onto the workstation. The policies are updated and maintained on the workstation, and then FTPed back to z/OS to be enabled.

Thus, the Configuration Assistant is being added to z/OSMF so users can have these configuration tasks running on the system and have the policies maintained in the file system on z/OS, and be deployed more easily. The Configuration Assistant itself can guide users through the required tasks from generation of the policies, making sure they are correct, and activate them as needed. Although the current functionality of the Configuration Assistant is not being updated at this time, making it available on z/OSMF satisfies a client need.

If you have been using the Configuration Assistant for z/OS Communications Server on Windows, you can begin using the Configuration Assistant on z/OSMF by following the instructions in the IBM z/OS Management Facility User's Guide under the Configuration Assistant for z/OS Communications Server. If you are familiar with the Windows GUI, the Configuration Assistant on z/OSMF is essentially the same.

**Note:** The Configuration Assistant for the z/OS Communications Server is only available with z/OSMF V1R11 and z/OS V1R12.

z/OS V1R10 users will not see the Communications Server category in z/OSMF. z/OS V1R10 and earlier users can still use the separate web tool.

### 40.9.2 z/OS V1R12 IPSec and IKE support

z/OS Communications Server IPSec and IKE support is added to use z/OS cryptographic modules that are designed to address the Federal Information Processing Standard (FIPS) 140-2 security requirements for cryptographic modules.

FIPS 140 defines a set of security requirements for cryptographic modules to obtain higher degrees of assurance regarding the integrity of those modules. FIPS 140-2 provides four increasing, qualitative levels of security intended to cover a wide range of potential applications and environments.

**Note:** Enabling FIPS 140 provides a higher degree of assurance of the integrity of the cryptographic modules that are used, including ICSF and System SSL. However, enabling FIPS 140 might result in a reduction in performance. FIPS 140 restricts the available set of cryptographic algorithms, and it might require additional setup and configuration.

z/OS V1R12 Communications Server support is configurable so that it will only utilize underlying security modules (System SSL and ICSF) that are operating in FIPS 140 mode. System SSL and ICSF are designed to address the requirements for FIPS 140-2 level 1.

You configure FIPS 140 mode independently for each of the IKED, NSSD, and TCP/IP stack components. FIPS 140 mode must also be configured in ICSF and System SSL. If you use FIPS 140 mode in several components but not others, the resulting system might not be operating in FIPS 140 mode. The components that are configured to use FIPS 140 mode may only use cryptographic services from components that are also operating in FIPS 140 mode, so the FIPS 140 mode mismatch can cause functional problems.

## IPSec RFC4301 compliance

Beginning in z/OS V1R12, all IP security connectivity rules must be compliant with RFC 4301. The ability to set Not compliant for a stack and bypass RFC 4301 checking is no longer available. Compliance with RFC 4301 is enforced.

IP filter policy support for filtering fragments was improved in z/OS V1R10. Prior to z/OS V1R10, Communications Server filtered all IP fragments using a policy of first possible filter match, and filtered IPv6 fragments as protocol IPv6Frag.

Beginning in z/OS V1R10, Communications Server follows rules and restrictions established by RFC 4301 to ensure proper classification of fragments. RFC 4301, “Security Architecture for the Internet Protocol,” specifies the base architecture for IPSec-compliant systems, including restrictions on the routing of fragmented packets.

**Note:** Communications Server does not implement stateful fragment checking. Therefore, restrictions were added as required by RFC 4301 to require that IP security connectivity rules applying to routed traffic not apply to specific ports, types, or codes. In z/OS V1R10 and z/OS V1R11, a setting was provided to optionally configure whether or not the RFC 4301 restrictions are to be applied. Beginning in z/OS V1R12, this setting has been removed. All IP security connectivity rules must support the RFC 4301 rules and restrictions.

RFC 4301 compliance for IPSec filter rules is planned to become mandatory. RFC 4301 “Security Architecture for the Internet Protocol” specifies the base architecture for IPSec-compliant systems, including restrictions on the routing of fragmented packets. Compliance enforcement may require minor changes to IP filters for IP traffic that is routed through z/OS. The Configuration Assistant will be designed to identify any non-compliant IP filters and policy agent will not install an IPSec policy that contains any non-compliant IP filters.

## 40.10 z/OSMF Workload Management task

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. 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. Specifically, you can define, modify, view, copy, import, export, and print WLM service definitions.

In a z/OS environment, work competes for resources, serialized by locks and latches, as follows:

- ▶ Low importance work could hold a resource and high importance work could have to wait for it.
- ▶ Incorrect WLM classification of system work can lead to serious system problems and even outages.

The WLM administrative application provides little support to review and optimize service definitions, as follows:

- ▶ It is difficult to see the relationship of policy elements and to compare them.
- ▶ Recommendations and best practices for the specification and optimization of service definitions are scattered over several manuals and have to be applied without tool support.



- ▶ Users must walk through drill-down and interim panels to create and change policy elements.

## 40.10.1 WLM and z/OSMF

Key functions available in the Workload Management task in z/OSMF include the following:

- ▶ Display a list of service definitions.

The Workload Management task provides a list of the WLM service definitions that have been defined in z/OSMF along with history information (such as when the service definition was installed or modified), messages, and user activity. The list of service definitions is retrieved from the service definition repository, which refers to the directory in the z/OSMF data file system in which the data for the Workload Management task is stored.

- ▶ Work with multiple service definitions.

In the Workload Management task, you can work with multiple service definitions simultaneously. To do so, open the service definitions with which you want to work in its own View, Modify, Copy, or Print Preview tab. You can also define multiple service definitions at the same time by opening several New tabs.

- ▶ Install service definitions.

The Workload Management task provides features that you can use to install a service definition into the WLM couple data set for the z/OSMF host sysplex.

**Note:** If you plan to use the Internet Explorer browser to work with WLM service definitions, ensure that the browser is enabled for automatic prompting for file downloads. This setting prevents the file download blocker from being invoked when you download service definitions to your workstation. Otherwise, if automatic prompting is disabled (the default setting), the download blocker prompts you to accept these file downloads, causing your z/OSMF session to be reloaded and your active tabs to be closed. To avoid this disruption, enable automatic prompting for file downloads. For more information, see the online help for the Workload Management task.

Using the Internet Explorer 7 browser can result in slow responsiveness if you open multiple tabs, work with large WLM service definitions, or use the Monitoring Desktops task for long periods of time. As an alternative, you can use the Mozilla Firefox 3.5 browser or Internet Explorer 8 browser.

## 40.10.2 z/OS V1R12 z/OSMF

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 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.

### WLM and CIM server

During the z/OSMF configuration process, you can select to have z/OSMF create an exec program with sample RACF commands for authorizing z/OSMF tasks to your CIM server. Choose this option if your configuration will include plug-ins that require the use of the CIM server, and you are configuring the CIM server for the first time and have not yet created the authorizations for the CIM server.

## Focus on WLM policy editor and resource monitoring

z/OSMF V1R12 has integrated the workload management application which enables you to manage WLM service definitions and provide guidelines for WLM to use when allocating resources.

### 40.10.3 WLM customization with z/OS V1R12

When using the Workload Management Selection button shown in Figure 40-6 on page 762, the following customization actions can be implemented when using this function:

- ▶ Define a group for accessing Workload Manager (WLM) resources.

This group name is used to allow the Workload Management task to access WLM resources on your system.

The variable name is `IZU_WLM_GROUP_NAME`

**Default:** WLMGRP

- ▶ The WLM group GID or AUTOGID.

Define a WLM group, GID, to use for the WLM group administrator identity. Instead of specifying the GID value, you can enter AUTOGID to have RACF automatically generate a unique ID. For more information about the AUTOGID operand, see *z/OS Security Server RACF Security Administrator's Guide*.

The variable name is `IZU_WLM_GROUP_GID`

**Default:** 9600

**Note:** You can assign a group identifier (GID) to a RACF group by specifying a GID value in the OMVS segment of the RACF group profile, or by using the AUTOGID keyword.

### 40.10.4 WLM primary window

When you select Workload Manager from the primary window shown in Figure 40-6 on page 762, then the window shown in Figure 40-18 on page 781 is displayed. Authorization to access this window is required from a z/OSMF administrator.

Actions that require the Workload Management task to interact with the sysplex are limited to the sysplex in which the z/OSMF host system is a member. Such actions include the following:

- ▶ Installing a service definition
- ▶ Activating a service policy
- ▶ Viewing the sysplex status

If you want to interact with another sysplex, z/OSMF must be installed on a system in that sysplex and you must log into that z/OSMF instance. You can use the service definition import and export functions to copy a service definition from one z/OSMF instance to another z/OSMF instance.

From the window shown in Figure 40-18 on page 781, you can perform the following functions:

**Manage service definitions** Define, modify, view, copy, import, export, print, and install WLM service definitions.

- Manage service policies** For the sysplex, you are able to activate or print one of the service policies that is defined in the installed service definition.
- Manage settings** You can define the code page and backup sequential data set for the sysplex. Specify how long to keep the service definition history. Enable or disable the service definition and service policy consistency checking feature. Display or suppress information messages.
- View status of sysplex** You can view the status of WLM on each system in the sysplex and view details about the installed service definition and the active service policy.

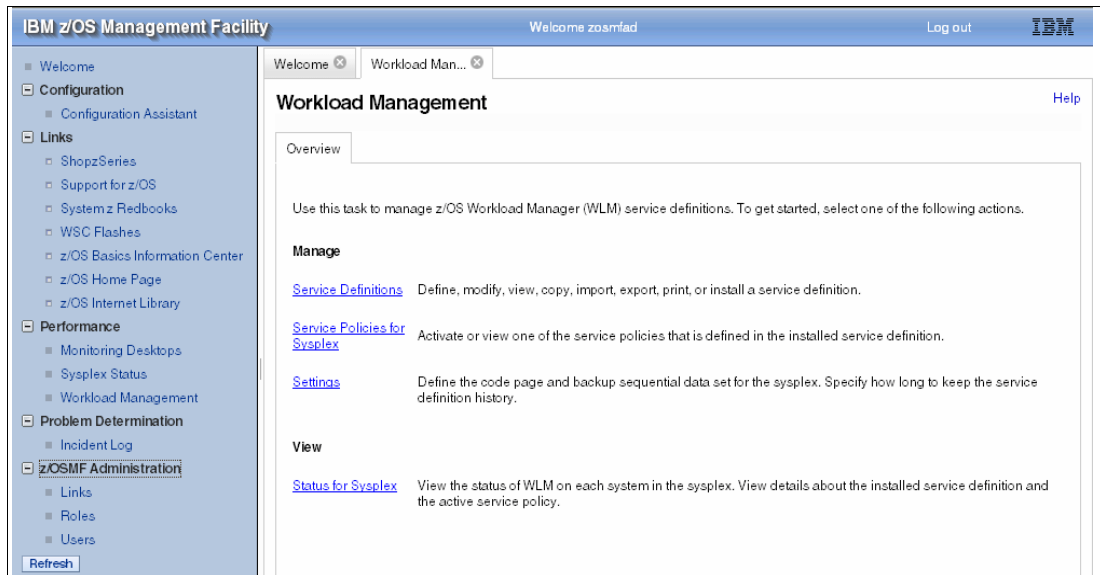


Figure 40-18 Workload Manager primary window

## z/OSMF WLM task processing

The resource monitoring task enables you to monitor the performance of the z/OS sysplexes or Linux (System z or Intel) images in your environment. With the Monitoring Desktops task, you can monitor most of the metrics supported by the Resource Measurement Facility™ (RMF) Monitor III, create and save custom views of the metrics, and display real-time data as bar charts.

The Sysplex 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 Sysplex Status task also provides a single location where you can define the z/OS sysplexes to be monitored in the Monitoring Desktops task.

### 40.10.5 Workload Manager policy editor

The Workload Manager (WLM) policy editor application facilitates the creation and editing of WLM service definitions, installation of WLM service definitions, activation of WLM service policies, and monitoring of the WLM status of a sysplex and the systems in a sysplex.

With z/OSMF, this allows you to manage WLM service definitions and 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.

With z/OS V1R12, you can perform problem data management tasks through the Incident Log, which helps to centralize problem data for your system and simplifies the process of sending diagnostic data to IBM or another destination.

### **Extract the installed service definition**

The Workload Management task automatically extracts the service definition that is installed in the WLM couple data set for the z/OSMF host sysplex, and stores it in the service definition repository so that you can view it, modify it, or activate one of its service policies.

z/OSMF integrates its repository to store service definitions to implement the following functions:

- ▶ Import and export service definitions in XML format
- ▶ Print service definitions
- ▶ Create, edit, and review service definitions in tabular format
- ▶ Direct navigation between policy elements during editing or viewing of service definitions
- ▶ Best practice checking for service definitions
- ▶ Support the installation of service definitions and the activation of service policies
- ▶ Display WLM status of systems in the sysplex

### **z/OSMF workload management implementation**

This implementation allows you to use the following functions:

- ▶ Simplify creating and editing WLM service definitions by displaying the elements of a service definition in tabular form. This allows an installation to:
  - Create or edit service definition elements directly in tables
  - Create and edit WLM service definitions supported by best practice checks
  - Direct navigation between policy elements during editing or viewing service definitions
  - Serialize editing of the active service definition
- ▶ Simplify handling through an integrated repository for WLM service definitions.

The WLM service definitions are stored in a repository integrated in the z/OSMF file system. The WLM service definitions can be exported to the local workstation or a host data set, or imported from a file on the local workstation or a host data set. The WLM service definitions can be printed using the print menu of the web browser.

- ▶ Create the installation of WLM service definitions and activation of WLM service policies.
- ▶ Monitor the WLM status of a sysplex and the systems in a sysplex, because the WLM status report is automatically updated if the WLM status on the systems changes.
- ▶ Open multiple windows to enable users to perform tasks simultaneously.

This simplifies migration because policy elements can be copied from one service definition to another. This also simplifies operation because users can start to edit a service definition, interrupt the editing to activate a service policy, and then continue with the editing without losing the context.

## Accessing the policy editor

WLM policy editor functionality, as shown in Figure 40-19 on page 783, is integrated into z/OSMF V1R12. This allows the creation and editing of WLM service definitions, the installation of WLM service definitions, the activation of WLM service policies, and the monitoring of the WLM status of a sysplex and the systems in a sysplex.

The WLM policy editor is available on the z/OS Management Facility, with all the same function as in the web download tool and with many new features. You have direct access to the WLM couple data set to install or extract service definitions. There is no need to FTP the WLM policy files. You can activate service policies and monitor the WLM status in the sysplex. The z/OSMF repository is used to store service definitions and is a starting point for edit, print, and install operations.

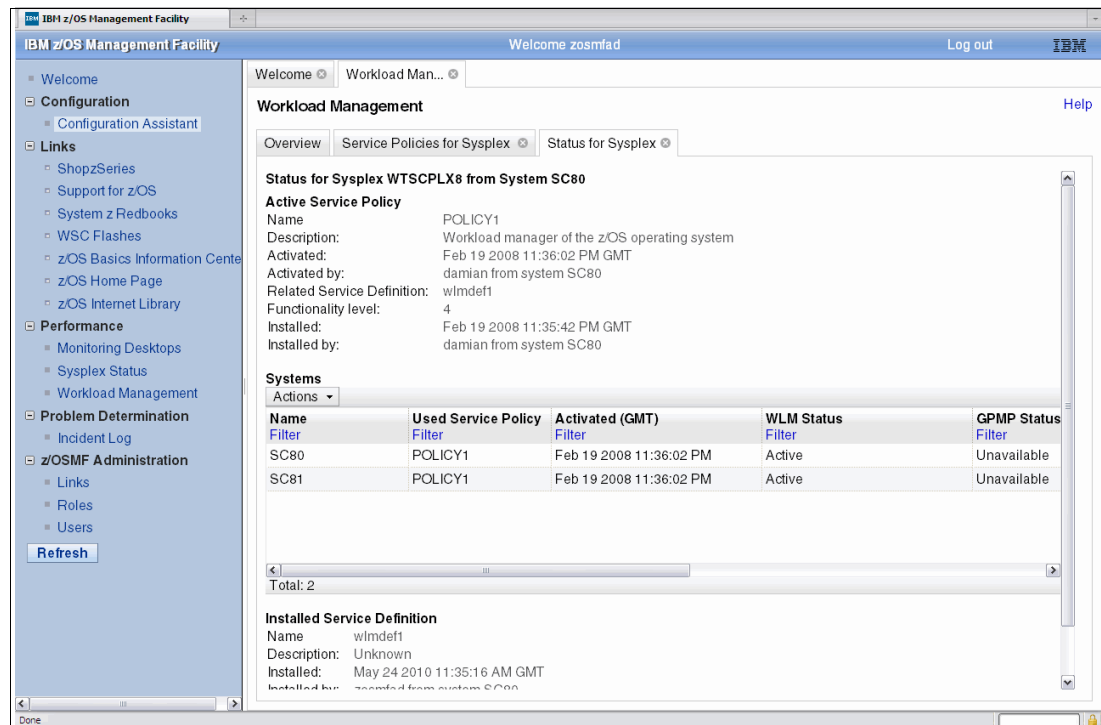


Figure 40-19 WLM policy editor display

## Policy editor additional functions

The WLM policy editor simplifies the creation, modification, and review of service definitions. The following capabilities can be invoked:

- ▶ Policy elements are presented in tables.
- ▶ Tables can be filtered and sorted.
- ▶ There is direct editing of policy elements within tables.
- ▶ Best practice hints are displayed automatically when specifying policy elements.
- ▶ Several service definitions can be opened simultaneously.
- ▶ You can cut, copy, and paste policy elements between service definitions.

## 40.11 Resource Monitoring application plug-in

The z/OSMF Resource Monitoring application provides integrated performance monitoring in client environments. It supports z/OS, z/OS sysplexes and Linux images (System z and Intel) in your installation. It requires the RMF z/OS data server (DDS) on each sysplex that is being monitored, and the Linux data gatherer (rmfpms) to be running on the Linux image that is being monitored.

There are two z/OSMF tasks:

- ▶ The Monitoring Desktops task

The Monitoring Desktops task in z/OSMF provides a browser-based user interface that you can use to monitor the performance of the z/OS sysplexes or Linux (System z or Intel) images in your environment. With the Monitoring Desktops task, you can monitor most of the metrics supported by the Resource Measurement Facility (RMF) Monitor III, create and save custom views of the metrics, and display real-time data as bar charts.

For z/OS sysplexes, the Monitoring Desktops task takes its input from a single data server on one system in the sysplex. That data server gathers data from the RMF Monitor III data gatherer on each image in the sysplex. This function is called the Distributed Data Server (DDS). To monitor several sysplexes, ensure that each sysplex has an active DDS.

For Linux images or guests, the Monitoring Desktops task takes its input from the RMF Linux data gatherer (rmfpms) running on the image that is being monitored. To monitor several Linux systems, ensure that each system has an active data gatherer. More information is provided in the online help topic for Linux data gatherer.

- ▶ The Sysplex Status task

The Sysplex Status 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.

**Important:** If you plan to use the tasks from the Resource Monitoring plug-in, then enable the optional priced feature, Resource Measurement Facility (RMF), on one of the systems in your enterprise. For information about enabling features, see *z/OS Planning for Installation*, GA22-7504.

### 40.11.1 Monitoring Desktops task

The Monitoring Desktops task in z/OSMF V1R12 includes the following key functions:

- |                                     |   |
|-------------------------------------|---|
| <b>Create monitoring desktops</b>   | In the Resource Monitoring task, you can create monitoring desktops or custom views that you can use to monitor the performance of the z/OS sysplexes and Linux images in your environment.   |
| <b>Save monitoring desktops</b>     | In the Resource Monitoring task, you can save monitoring desktops. Doing so allows you to reuse the monitoring desktop or template so that you can easily view performance data for your monitored z/OS sysplexes and Linux images from the same angle. |
| <b>Multiple monitoring desktops</b> | In the Resource Monitoring task, you can work with multiple monitoring desktops simultaneously. To do so, open the desktops that you want to work with in a new tab   |

in the z/OSMF work area or in a new browser tab or window.

### Monitor multiple resources

In the Resource Monitoring task, you can collect data for multiple resources simultaneously. To do so, associate the metrics in a desktop with other resources.

### Create desktops

You can create desktops that are not associated with a specific sysplex. This will streamline the number of desktops that you must create because you can simply create one desktop and use it for all the sysplexes in your installation.

### Monitor performance over time

The Monitoring Desktops task provides controls that you can use to browse through the samples that have been collected for the metric groups contained in a monitoring desktop. Up to 100,000 samples are collected for a desktop. To browse the samples, use the slider and the backward and forward arrows provided in each metric group.

## Monitoring Desktops task panel

As shown in Figure 40-20 on page 785, the Monitoring Desktops panel has the title and a help link on the top and a tab pane with one open tab (Desktops). In the tab, there is a table with one column (Name). The table contains eight desktops (Common Storage Activity, Coupling Facility Activity, General Activity, and so on). On the bottom of the table, there is a Refresh button and a time stamp showing when the table was refreshed.

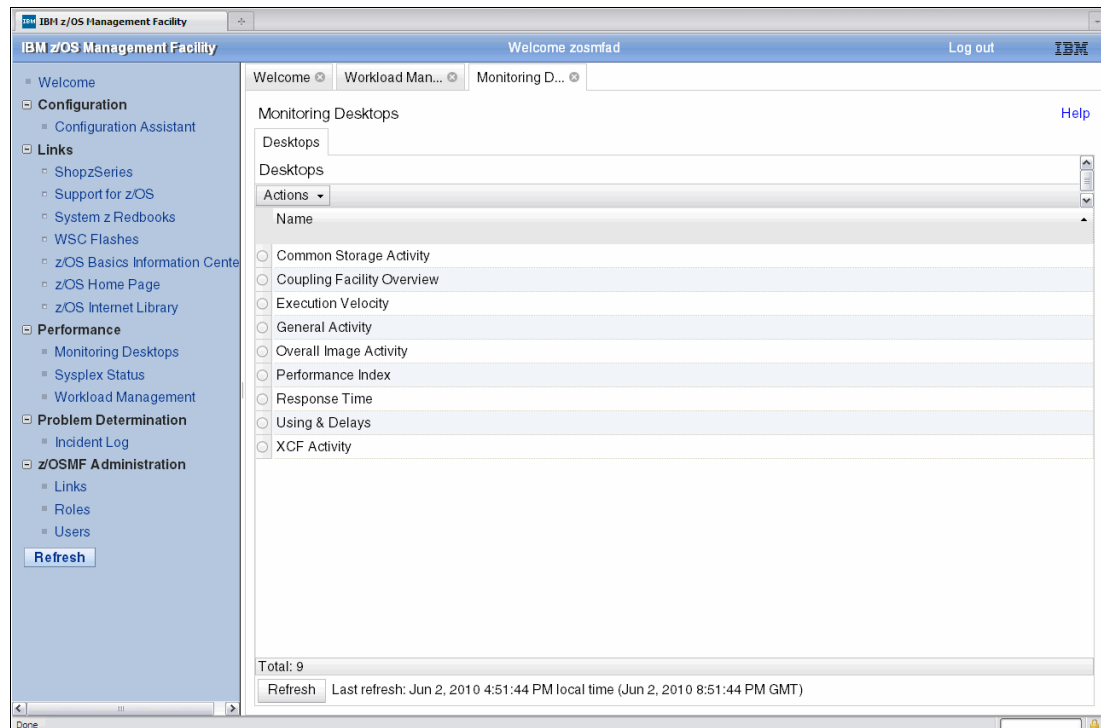


Figure 40-20 Resource monitoring from the Monitoring Desktops task

**Important:** Before you can start using the Monitoring Desktops task, you must define the z/OS sysplexes or Linux images to be monitored in the Sysplex Status task. To display the Sysplex Status task, expand the Performance category in the navigation area and select Sysplex Status.

## 40.11.2 System Status task

Figure 40-21 on page 787 shows the Sysplex Status panel. On the top, the panel title and a brief description of the panel functionality is displayed. In the middle of the panel, a table containing the resources is shown. The table in our system has two entries. All of them are in the connectivity state `Connected` and they represent two sysplexes.

**Note:** The Sysplex 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 Sysplex Status task also provides a single location where you can define the z/OS sysplexes to be monitored in the Monitoring Desktops task.

In the performance index status column, every resource has a separate state (green, yellow, or red). The last two columns show information about the WL.

**Note:** This information shown in Figure 40-21 on page 787 is obtained from the RMF DDS.

On the bottom of the table, there is a Refresh button and a time stamp showing when the table was refreshed.

### Resources in Sysplex Status panel

The sysplexes and any Linux images that are defined to z/OSMF are listed in the Resources table. By default, z/OSMF supplies the LOCALPLEX resource entry and specifies an asterisk (\*) for its host name. z/OSMF uses this resource entry to detect and connect to the RMF Distributed Data Server (DDS) that is running in the sysplex in which the z/OSMF host system is a member.

The Resources table can contain up to 100 resource entries. The resource entries are not shared across z/OSMF users; therefore, only you can view or modify the entries.

**Installation consideration:** To authenticate with a DDS, z/OSMF supports the creation and transmission of a PassTicket that uses the z/OSMF user ID. PassTickets are created on behalf of the WebSphere Application Server servant user ID; therefore, in the security management product for your installation (for example, RACF), the WebSphere Application Server servant user ID must be authorized to generate PassTickets.

If z/OSMF cannot generate a PassTicket on behalf of the servant user ID, or if it cannot authenticate with the DDS using your z/OSMF user ID, then it cannot retrieve any data for the sysplex.

For more information about configuring the DDS, see the topic about system prerequisites for the Monitoring Desktops and Sysplex Status tasks in *IBM z/OS Management Facility Configuration Guide*, SA38-0652.



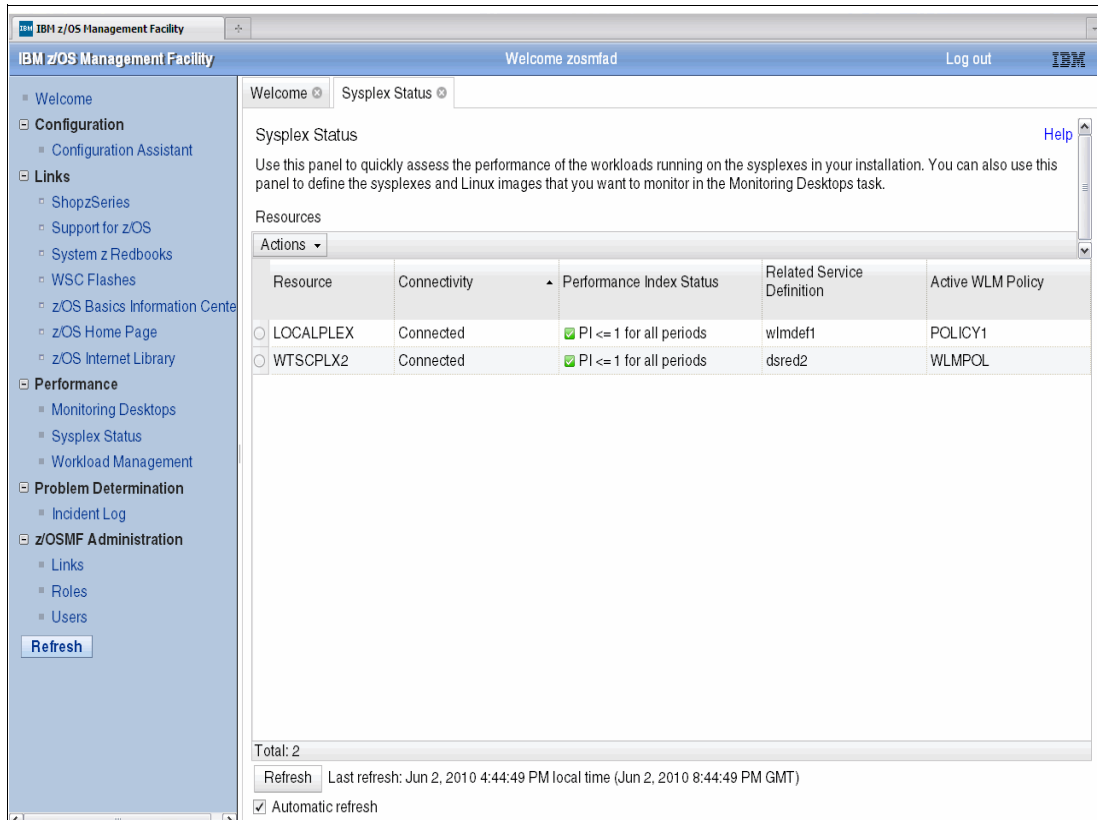


Figure 40-21 Resource monitoring - sysplex status

### Select an option from the Monitoring Desktops panel

Figure 40-22 on page 788 shows the “Common Storage Activity” that is selected from Figure 40-20 on page 785 of z/OSMF Monitoring Desktops panel. Inside the tab there is a title area (showing “Common Storage Activity (Running)”) and a toolbar with the buttons Start (disabled), Pause, Save (disabled), Actions (dropdown).

The rest of the panel is divided vertically into two metric groups “CSA & ECSA (Systems)” and “SQA & ESQA (Systems)”. Each metric group shows a bar chart containing blue and red bars with numbers at the right.

Both charts have SC80, SCL81 labels on the y axis. Corresponding to each label, there are two bars (a blue bar and a green bar). On the bottom of the chart, is a legend that assigns a metric to each color (metrics are corresponding to the metric group title). Below the legend is a slider and a time interval.

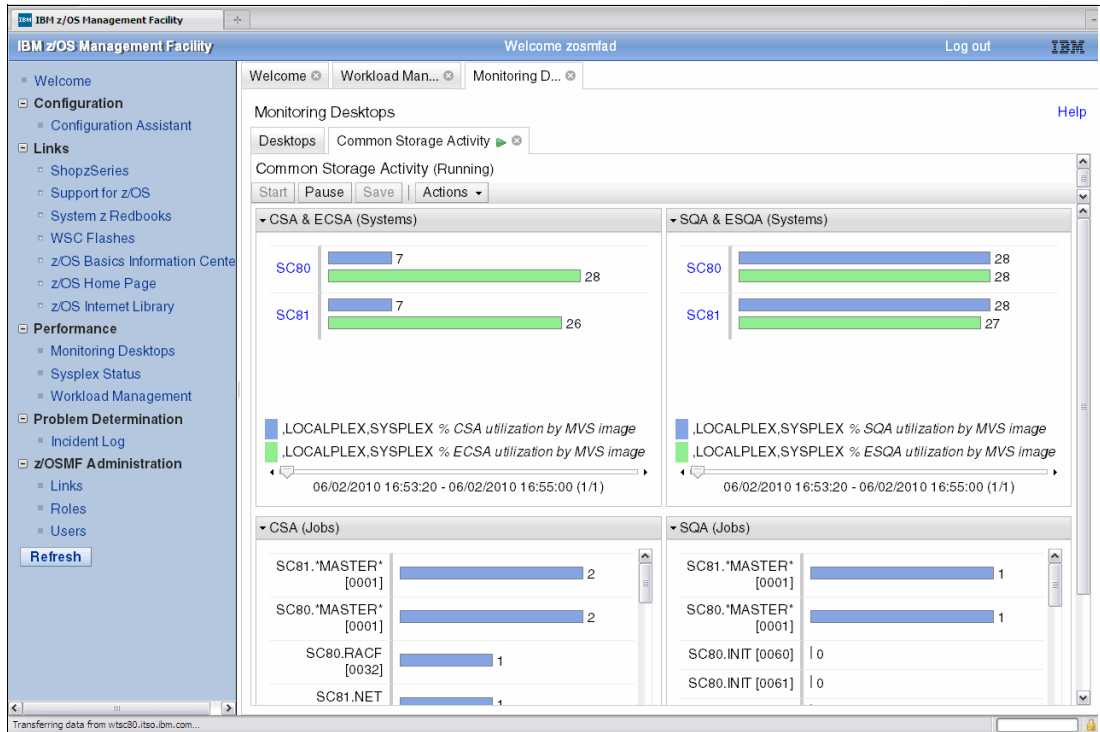


Figure 40-22 Common storage activity

Figure 40-23 on page 788 shows the “Add Metric” dialog. At the top of the panel are three required fields, namely “Add to metric group” (combo box), Selected Resource, and Selected Metric.

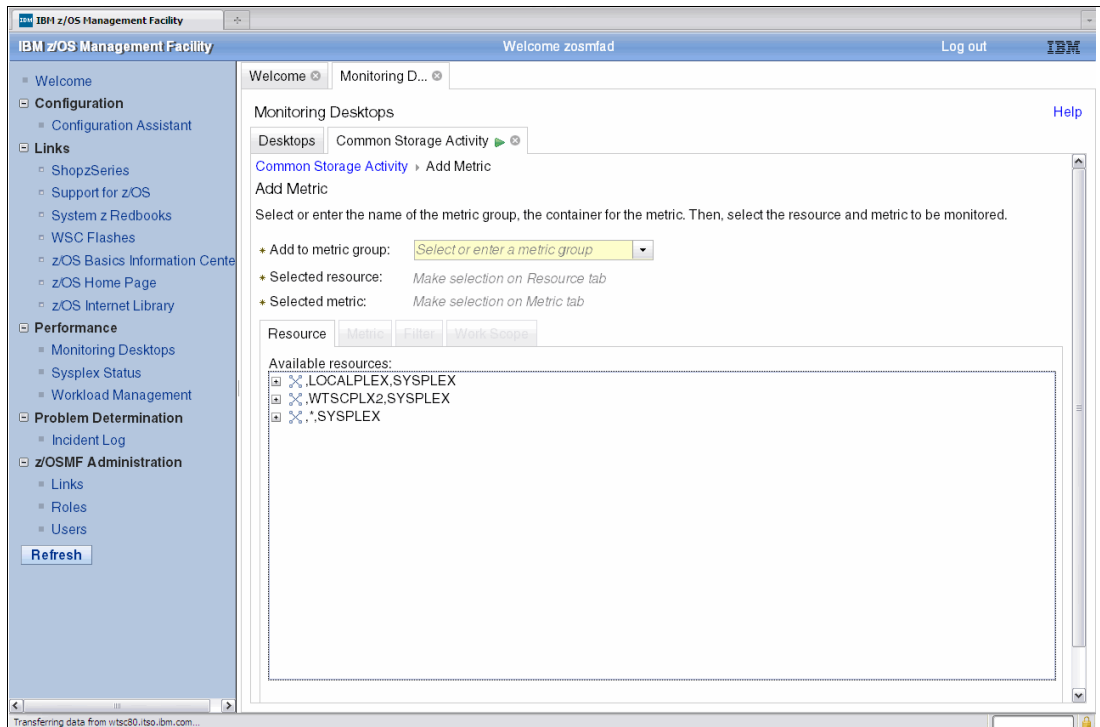


Figure 40-23 Add metric panel

## 40.12 z/OSMF administration

The primary purpose of the tasks provided for z/OSMF administration are authorization functions for the administrator to authorize additional users. The administrator has the ability to add users (those with valid z/OS user IDs) and to assign roles to them so these users can get access to separate tasks. In V1R12, predefined roles are set: when users are assigned a role, then they cannot perform any other tasks.

**Note:** In z/OSMF, a *role* represents the ability to perform one or more tasks. These tasks include both general administrative actions for z/OSMF and task-specific actions for the z/OS system to be managed. Your installation can modify the tasks for a role through the Roles task.

The product z/OSMF Administration tasks, shown in Figure 40-22 on page 788, are typically performed by the z/OSMF administrator at your installation, as follows:

- |                   |  |
|-------------------|--|
| <b>Links task</b> | This task allows the administrator to define links for other external web applications and websites that users can launch from z/OSMF. Links can be added to any category in the z/OSMF navigation area. |
| <b>Roles task</b> | This task allows the administrator to modify z/OSMF roles.   |
| <b>Users task</b> | This task allows the administrator to define new z/OSMF users and to modify or delete existing z/OSMF users.   |

### 40.12.1 z/OSMF administrator defining users and roles

The z/OSMF administrator must authorize the user to z/OSMF and assign a role for the user to start working with z/OSMF tasks.

- ▶ The user must have a valid user ID on the z/OS system.
- ▶ The security administrator must authorize the user to the required z/OS stack for the z/OSMF tasks. Scripts are provided to perform the complete authorization.

**Note:** Installing a new release of z/OSMF will preserve any installation-specific links you have defined. If you fall back from a z/OSMF V1R12 system to a z/OSMF V1R11 system, your link definitions are retained but are displayed in the Links category, regardless of which categories were specified for the links.

#### Defining users

To add users, the z/OSMF administrator needs to:

- ▶ Set up the RACF access using the sample scripts.
- ▶ Assign users to either the Administrator role or the User role.

When new users are added to z/OSMF, they must be assigned a role.

#### Roles task

To work with roles for z/OSMF, your user ID must be assigned to a z/OSMF role that is permitted to the Roles task. By default, only the z/OSMF Administrator role can work with roles. To view and work with roles in z/OSMF, expand the z/OSMF Administration category in the navigation area and select **Roles**, as shown in Figure 40-24 on page 790. This will begin a sequence of steps for defining which tasks are assigned to a z/OSMF role. For assistance with this task, see the online help for the Roles task.

With z/OSMF V1R12, there are only two roles: administrator and user. You can define what a user can see and set what a user can do. The four roles that z/OSMF supports are listed here and also shown in Figure 40-24.

- Guest**                      User is not logged into z/OSMF.
- Authenticated guest**    User is logged into z/OSMF but has no role assigned.
- Administrator**            User can access all tasks in z/OSMF.
- User**                        User can access all tasks except Administrator tasks.

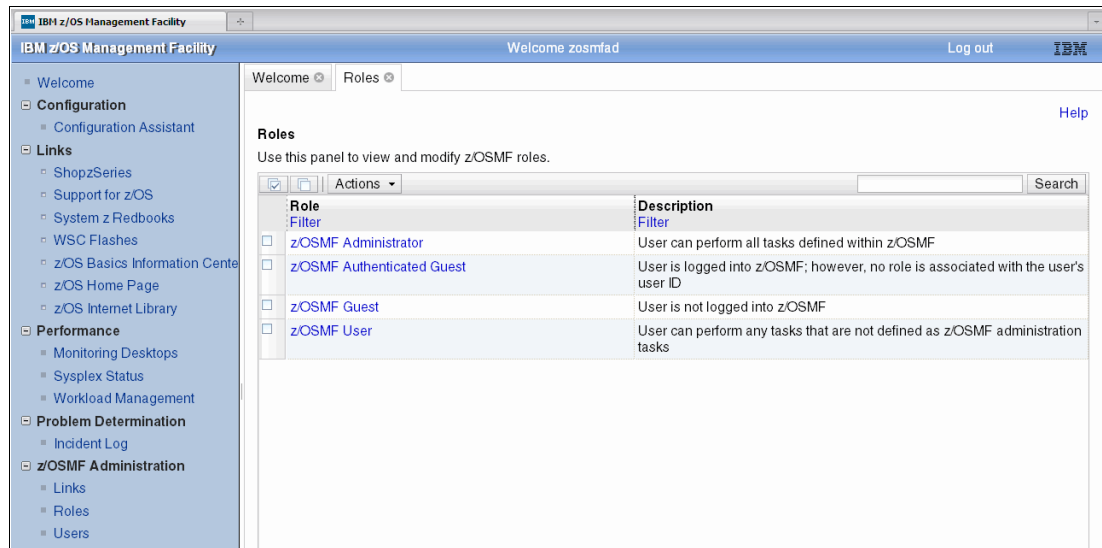


Figure 40-24 z/OSMF administration: defining a role

## Users task

On the bottom of the panel shown in Figure 40-24 an additional box, Users, lists all the users who are assigned that particular role. This is how an administrator manages users and their roles. The Role task shows all users assigned to a specific role.

## Links task

z/OSMF allows the administrator to dynamically add links to non-z/OSMF resources, for example, to ISV products and commonly used installation websites. In z/OSMF V1R12 links can be added to any category.

Figure 40-24 on page 790 shows the left side of each z/OSMF panel that contains the predefined links provided by IBM, and any new links added by the z/OSMF administrator.

Note the following points:

- ▶ The links are available to all users of z/OSMF.
- ▶ The administrator can define which roles have access to each of the defined links.
- ▶ The IBM predefined links are accessible to all users, including guests, by default.
- ▶ A new (V1R12) interface enables you to add non-z/OSMF launch points and links to the left hand side navigation tree.
- ▶ Links can be added using the links GUI or by a program through a shell script.

Figure 40-25 on page 791 shows the Links task selection. The Links panel allows you to customize the z/OSMF navigation area with links to external resources, as needed by the

z/OSMF users at your installation. From this panel, you can manage links (add, remove, or modify them) and allow users to access the links.

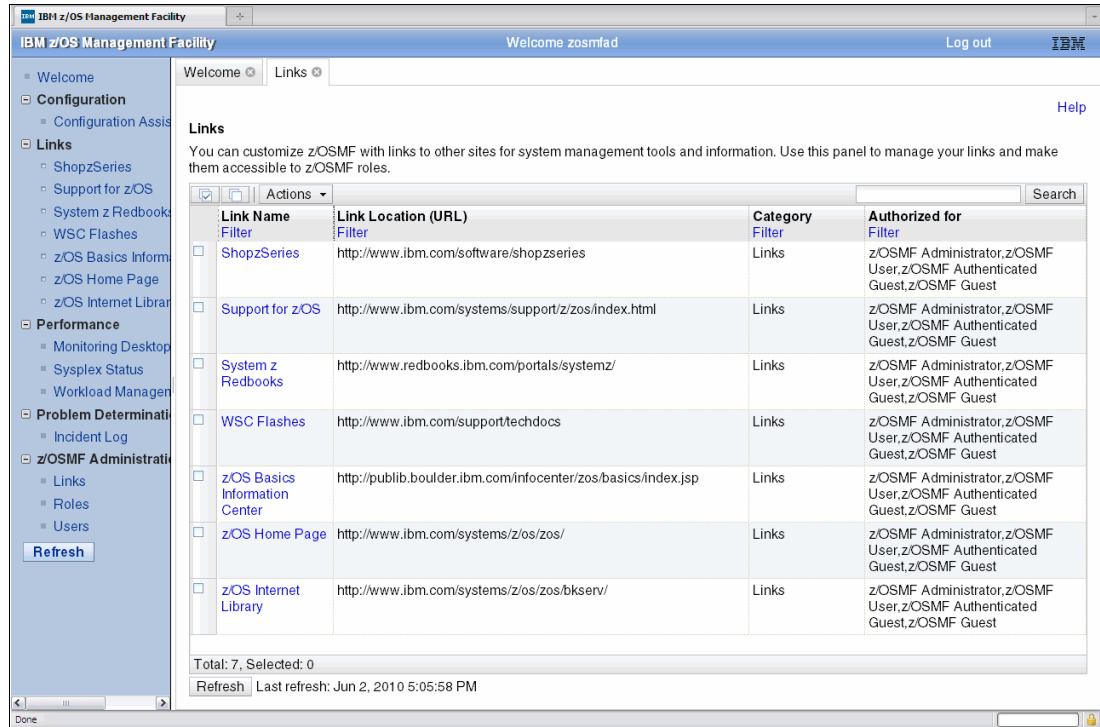


Figure 40-25 Defined links

## Defining a new link

When you click the Actions button shown in Figure 40-25, the window shown Figure 40-26 on page 792 is displayed. To define a new link to z/OSMF, use the New action provided on the Links panel and then specify a name, a location (a URL), and a category for the link. You can optionally assign the link to one or more z/OSMF roles.

**Note:** Your installation can customize z/OSMF with links to external sites for system management tools and information. The process of defining a link to z/OSMF includes the following actions:

- ▶ Specifying the link name and its location (a URL)
- ▶ Selecting a z/OSMF category for the link
- ▶ Assigning the link to one or more z/OSMF roles

Depending on the user's browser settings, the link opens as either a new browser window or a new browser tab. To define links for z/OSMF, your user ID must be assigned to a z/OSMF role that is permitted to define links. By default, only the z/OSMF Administrator role can define links.

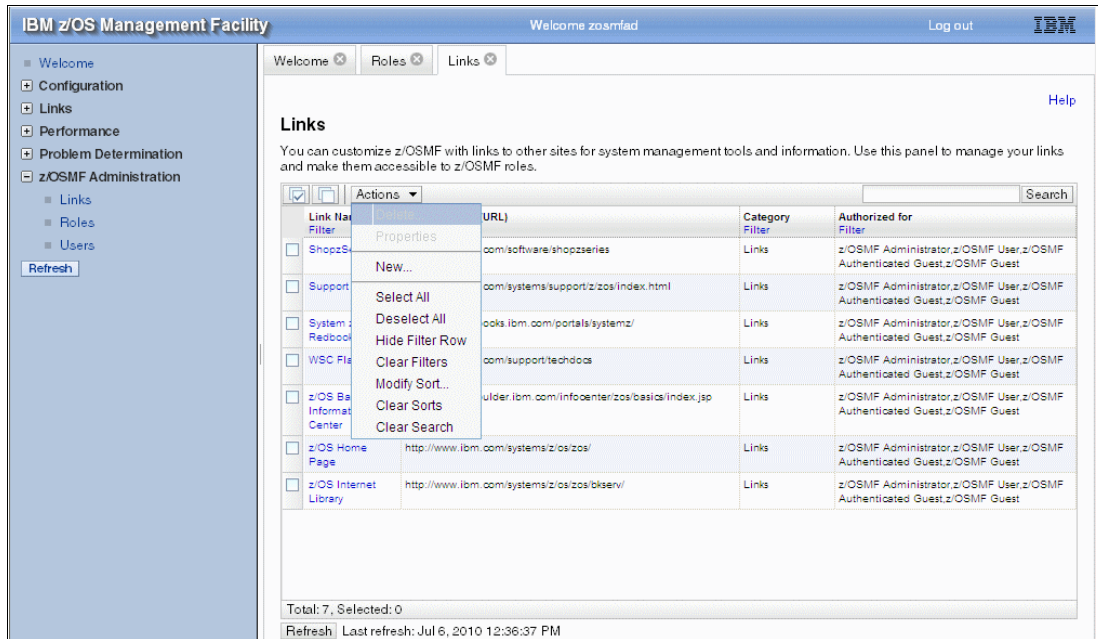


Figure 40-26 Select Actions and then New to define a new link

After you select New, the screen shown on Figure 40-27 on page 792 is displayed showing the possible properties that can be set for a link. You can use this New Link and Properties for link panel to define new links and modify existing links.

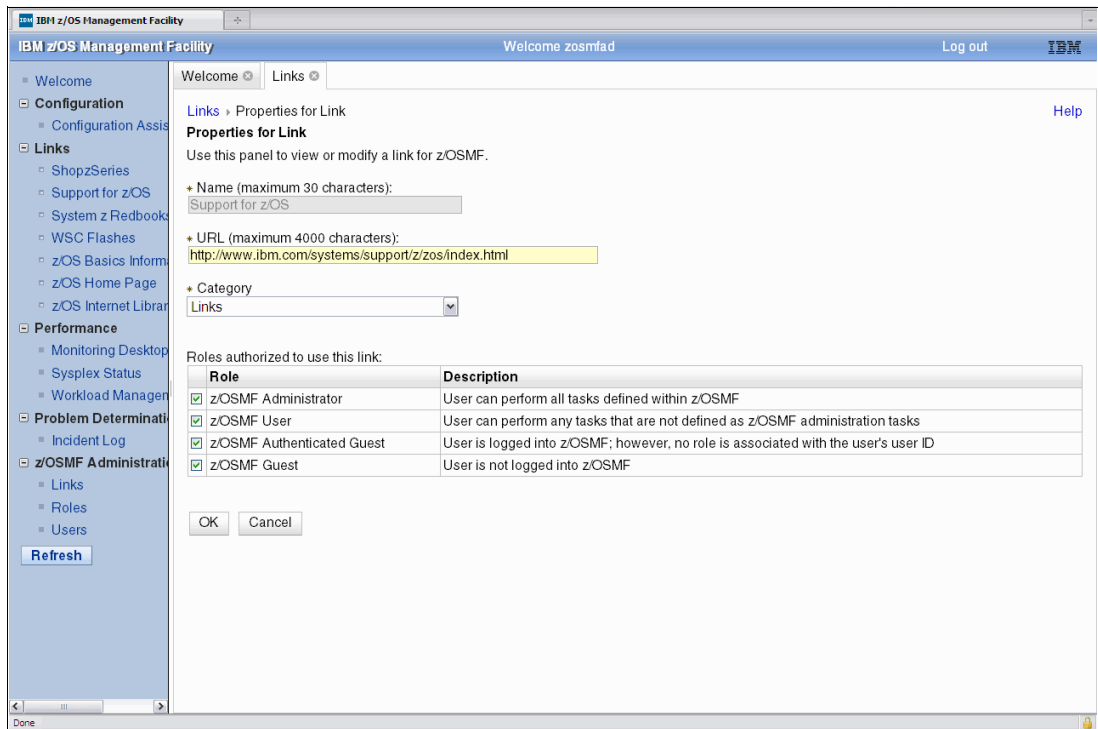


Figure 40-27 Properties of a link

### Adding a new link

Figure 40-28 on page 793 shows how to add a link. The main body of the panel shows where to add the name and the fully qualified URL for the link, and which roles can view or access

that link. Notice that several roles are shown: the administrator, the user, the guest, and an authenticated guest.

In many instances you want information and links to be made available to all users and guests,. However, there can be instances where you are linking to internal documentation and you do not want to share that access with just any guest. This is covered by the definitions of an administrator, user, and guests can access.

- ▶ Authenticated guests have valid user IDs, but do not have roles assigned yet.
- ▶ The administrator has the option to limit access to links to authenticated guests. You can use this for roles where teams share information with each other.

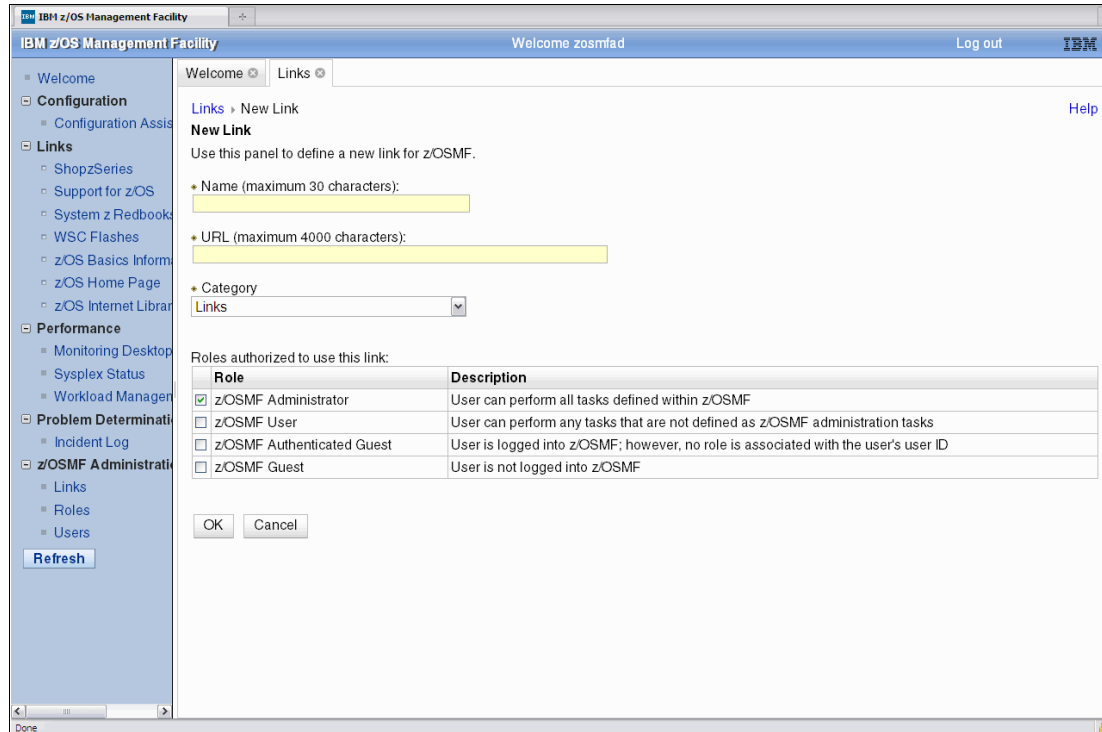


Figure 40-28 Defining a new link

## Adding a link with a new programming interface

The following new script interface allows users to add a link to z/OSMF from a program:

```
izusetup.sh -file <config file name> -addlink <link file name>
```

where:

- ▶ <link file name> is the name of the input file that contains the link properties of the link to be added to z/OSMF.
- ▶ LinkName=SampleLink
 

The LinkName input can be any string value that is 1 to 30 characters in length. If the LinkName value does not meet this requirement or the property is missing from the input file, then message IZUG604E will be issued and the link will not be added to z/OSMF.
- ▶ LinkURL=http://www.samplelink.com
 

The LinkURL input can be 1 to 4000 characters and must not violate URL syntax according to the checking that is performed by the constructor for the URL class. If the LinkURL value does not meet this requirement or the property is missing from the input file, then message IZUG603E will be issued and the link will not be added to z/OSMF.

- ▶ `LinkNavigationCategory=3`  
The valid `LinkNavigationCategory` values are: Configuration 4; Links 3; Performance 9; Problem Determination 2; z/OSMF Administration 1. If an invalid category is specified or the property is missing from the input file, then message IZUG502E will be issued and the link will not be added to z/OSMF.
- ▶ `LinkAuthorizedRoles=z/OSMF administrator, z/OSMF user, z/OSMF authenticated guest, z/OSMF guest`  
The `LinkAuthorizedRoles` value needs to be a comma (,) separated list of valid z/OSMF roles, or it can be empty or not provided. The list entries must *exactly* match the role names provided with z/OSMF. Any entries which do not match will be ignored and message IZUG503W will be issued for each role that does not match a supported role. Be aware that the link will always be added but, if all list entries are invalid, then there will be no role assignments for the link.

**Note:** The following script options are new for z/OSMF V1R12:

- ▶ A new option `-add` is added to the `izusetup.sh` script. This option allows you to add plug-ins to an already configured instance of z/OSMF. Specify the `-add` option with the existing options `-config`, `-prime`, and `-finish` to limit the scope of these script operations to new plug-ins only.
- ▶ A new option `-addlink` is added to the `izusetup.sh` script. This option provides an alternative way to add a link to the z/OSMF navigation area. In most cases, though, it is best practice to use the Links task to add a link.
- ▶ A new option `-service` is added to the `izusetup.sh` script. This option allows you to apply service to your z/OSMF configuration.
- ▶ The configuration variable `IZU_CIM_CONFIGURE` is added. For a newly configured CIM server, this option allows you to request z/OSMF to create a RACF commands template to assist your installation's security administrator with the setup. By default, this option is N (no).

### Adding a link consideration

When adding or defining a new link, the file might need to be UTF-8 and thus tagged to allow Japanese characters. A sample link file is shipped with the product and will also be included in the publications. The sample will also include commentary to assist users in providing input. The script can be invoked while the WebSphere Application Server OEM product is running or stopped. Also consider the following:

- ▶ z/OSMF does not provide function to modify or delete an existing link. Users must use the existing Links task GUI for those actions.
- ▶ If any other errors occur while processing the input file (that is, I/O exceptions and so on), then message IZUG504E will be issued and the link will not be added to z/OSMF.
- ▶ If there are multiple errors, only the first error encountered will be reported.
- ▶ If a link by the same name already exists in z/OSMF, then message IZUG505E will be issued and the link will not be added to z/OSMF.

## 40.12.2 Checking the client-side environment

The following URL is documented in the user's guide for checking the client-side environment. You download a checker from this URL to your station. The checker collects the client (your station) setup data and indicates any issues for the specific browser being used.

`https://hostname:port/zosmf/IzuUICommon/environment.jsp`



Figure 40-29 on page 795 shows the results of the z/OSMF environment checker after running in a station with a back-level browser.

Environment Option	Settings as of 2010-06-02T21:16:29.370Z	Requirements
JavaScript	JavaScript enabled	Enable JavaScript
Cookies	Cookies enabled	At a minimum, enable cookies for the z/OSMF server site
Pop-up Windows	Pop-up windows enabled	At a minimum, allow pop-up windows from the z/OSMF server site
Frames	Frames enabled	Enable frames
Screen Resolution	1280 by 1024	Minimum screen resolution of 1024 by 768
Browser Content Dimensions	1280 by 821	Minimum browser content dimensions of 800 by 600
Browser Name and Version Browser User-Agent value	Internet Explorer 6.0 Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322; .NET CLR 3.0.04506.30; .NET CLR 2.0.50727; .NET CLR 3.0.4506.2152; .NET CLR 3.5.30729)	Mozilla Firefox Version 3.0 (minimum service level 3.0.6) or Version 3.5 Microsoft Internet Explorer Version 7 or Version 8
Operating System	Microsoft Windows XP	Microsoft Windows XP, Vista, and Windows 7
Add-ons	No problem add-ons detected	Firebug may cause a performance impact to z/OSMF
Plug-ins	Java Adobe Acrobat Reader version 7 Flash Player 10 Shockwave Player 10 Windows Media Player	Some plug-ins may cause a performance impact to z/OSMF
z/OSMF Login ID	guest	An unauthenticated user will be "guest"
z/OSMF Version	Version Number: 1 Release Number: 11 Build Number: driver31	z/OSMF version

Figure 40-29 Environment checker warning: problem with the browser level

Figure 40-30 on page 795 shows the results of z/OSMF environment checker after running in a station with an adequate browser level.

Environment Option	Settings as of 2010-05-21T12:06:06.747Z	Requirements
JavaScript	JavaScript enabled	Enable JavaScript
Cookies	Cookies enabled	At a minimum, enable cookies for the z/OSMF server site
Pop-up Windows	Pop-up windows enabled	At a minimum, allow pop-up windows from the z/OSMF server site
Frames	Frames enabled	Enable frames
Screen Resolution	1280 by 1024	Minimum screen resolution of 1024 by 768
Browser Content Dimensions	986 by 526	Minimum browser content dimensions of 800 by 600
Browser Name and Version Browser User-Agent value	Firefox 3.6.2 (.NET CLR 3.5.30729) Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.2.2) Gecko/20100316 Firefox/3.6.2 (.NET CLR 3.5.30729)	Mozilla Firefox Version 3.0 (minimum service level 3.0.6) or Version 3.5 Microsoft Internet Explorer Version 7 or Version 8
Operating System	Microsoft Windows XP	Microsoft Windows XP, Vista, and Windows 7
Add-ons	No problem add-ons detected	Firebug may cause a performance impact to z/OSMF
Plug-ins	Microsoft Office 2003 Mozilla Default Plug-in Adobe Acrobat Shockwave Flash Windows Presentation Foundation Windows Media Player Plug-in Dynamic Link Library Microsoft® DRM Microsoft® DRM	Some plug-ins may cause a performance impact to z/OSMF
z/OSMF Login ID	guest	An unauthenticated user will be "guest"
z/OSMF Version	Version Number: 1 Release Number: 11 Build Number: driver31	z/OSMF version

Figure 40-30 Environment checker with an adequate browser level

## 40.13 z/OSMF installation considerations

Note the following additional details regarding z/OSMF:

- ▶ One instance of z/OSMF can manage only one local system or sysplex, based on the scope of the task to be performed. For example, problem determination collects information across all systems in a sysplex so you need one instance of z/OSMF at any time.
- ▶ Multiple users can log into the same instance of z/OSMF from separate workstations or browsers. Although there is no hardcoded limit, z/OSMF is expected to support up to 15 concurrent users.
- ▶ If you have multiple sysplexes, you will want to manage all those sysplexes. So, although you are required to have one z/OSMF instance for each sysplex, you can manage all your sysplexes by opening multiple z/OSMF windows on your browser on the workstation and logging on to all the z/OSMF instances.
- ▶ At any time only one active instance of z/OSMF is supported within a sysplex. Additional instances can be created (for example for test, service update, or backup), but they cannot actively manage the systems at the same time or using the same data repository. This is enforced by z/OS through global enqueue.

## 40.14 z/OSMF dependencies

There are no hardware dependencies. As for the software, z/OSMF V1R12 requires z/OS V1R12. The WebSphere Application Server OEM is to be at a minimum level 7.0.0.9 with APAR PM10520.

## 40.15 Migration and coexistence

The z/OSMF V1R12 version is only supported on a z/OS V1R12 system. Consider the following options.

**Note:** z/OSMF V1R11 is supported on z/OS V1R10 with maintenance, and on z/OS V1R11 and z/OS V1R12.

- ▶ If you have a z/OSMF V1R11 system and want to migrate to z/OSMF V1R12 on a z/OS V1R12 system, there is script support for doing so that requires z/OSMF V1R11 to be at the PTF UK52956/APAR PK97274 level prior to migration.
- ▶ You can use `izumigrate.sh` to migrate the V1R11 configuration file to the V1R12 format (see *IBM z/OS Management Facility Configuration Guide*, SA38-0652). This script is new with V1R12.

```
izumigrate.sh -file /etc/zosmf/izuconfig1.cfg
```

The script will create a backup copy of the configuration file. After migrating the configuration file to the V1R12 format, run the setup steps through `izusetup.sh` to deploy and enable the V1R12 level of applications. Then use the `-add` option to add the new applications in z/OSMF V1R12.

## Coexistence

Coexistence applies to lower-level systems that coexist (share resources) with the latest z/OS systems. If you require the capability to fall back from z/OSMF V1R12 to a lower-level system (z/OSMF V1R11 on a z/OS V1R10 or later system), and yet retain the use of the data repository from z/OSMF V1R12, then APAR PM09519 must be on the system with z/OSMF V1R11.

CEA APAR OA32285 (on the lower-level z/OS systems V1R10 and V1R11) is also required. If this is not installed, then under certain circumstances various data sets will not be deleted when the rest of the incident is deleted.

## 40.16 Installation considerations

The overall installation process consists of the following steps.

- ▶ Install the software (code) using ServerPac or SMP/E.
- ▶ Configure WebSphere Application Server OEM Edition. Configuration for WebSphere Application Server OEM Edition and z/OSMF has three basic phases:
  - Set up the configuration files.
  - Create the security definitions.
  - Build executables (run time files).
- ▶ Configure z/OS prerequisites (if necessary).
- ▶ Configure z/OSMF.
- ▶ Start WebSphere Application Server OEM Edition.
- ▶ Login to z/OSMF.

Most phases are driven through the use of z/OS UNIX configuration scripts. Phase 1 can be run interactively (interview style) or silently (fastpath mode).

Note that all setup instructions are provided for the RACF security product only. z/OSMF will run with other security products with equivalent instructions.

### 40.16.1 SMP/E installation scenario

z/OSMF can be ordered in several ways.

#### **z/OSMF V1R12 ordered in a z/OS ServerPac**

Ordering z/OSMF V1R12 by using a z/OS Server Pac provides you with default customization through post-install customization. It also provides a Full System Replace installation path and software upgrade jobs and documentation, but changes based on your existing environment could be necessary.

See the WebSphere Application Server OEM Edition configuration guide and the z/OSMF user's guide for guidance. The product configuration scripts are available to set up if the defaults are not viable.

#### **z/OSMF V1R12 ordered in a CBPDO**

If you order z/OSMF V1R12 in a CBPDO, then z/OSMF can be installed on z/OS V1R12. You can use the Program directory to get started. See WebSphere Application Server OEM

Edition Configuration Guide and *IBM z/OS Management Facility Configuration Guide*, SA38-0652, for guidance. There are product configuration scripts to use for the setup.

## 40.16.2 Configuration process overview

Both WebSphere Application Server OEM Edition and z/OSMF have three basic phases for configuration, as illustrated in Figure 40-31:

- ▶ Set up the configuration files.
- ▶ Create the security definitions.
- ▶ Build the executables (run time files).

Most phases are driven through the use of z/OS UNIX configuration scripts.

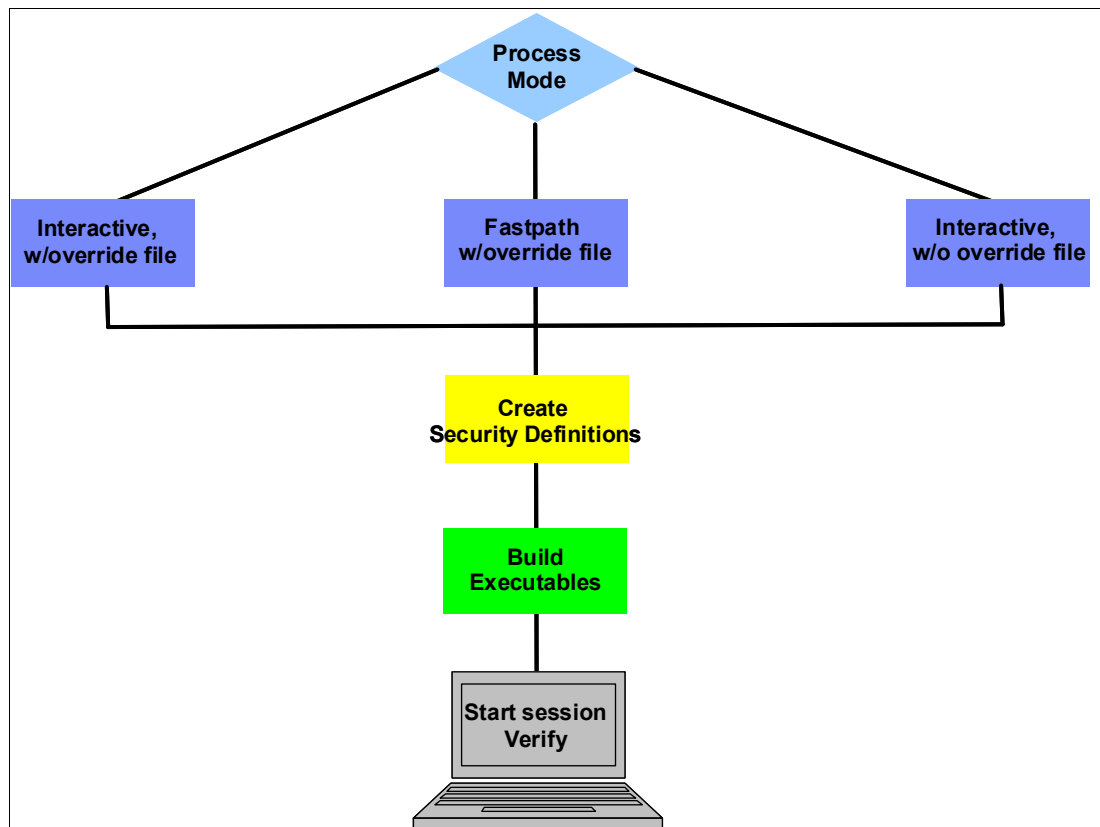


Figure 40-31 Configuration process overview

### Phase 1 scripts

As illustrated in the figure, you can run the scripts in three separate ways:

- ▶ Install interactively with an override file.

You can think of it as a PARMLIB member with only the values that you want to change. The script prompts you for all values as it displays the values from your override file as defaults. If the values are not found in the override file, they are taken from the specified configuration file. In response to each prompt, you must either press Enter to accept your installation-specific value, or type a new value. Note that when you use this mode, you need to preset the configuration session with your installation-specific values.

This method allows you to avoid having to enter your values interactively in response to script prompts. Instead, you simply need to review each value displayed by the script and then press Enter to accept it.

- ▶ Use the FASTPATH mode (minimal prompts).

Using this mode requires an override file or a configured configuration file. The script runs to completion without any interactive prompting. The WebSphere Application Server OEM Edition has minimal prompting. Values are used as supplied in the specified override file. Any values not found in the override file are taken from the configuration file. If a value is not found in either location, then the script ends with an error message indicating the first value that was not found.

When you use this mode you can supply your data in a stand-alone file. There is no need to review the values interactively. However, you need to have verified that all of your configuration data is supplied through the configuration file, or the optional override file, or a combination of both files. Then, you need to rerun the configuration process to update an erroneous value in an existing configuration file, and do not want to repeat the prompts.

- ▶ You can interactively install without an override file (or a configured configuration file). The script prompts you for all values, displaying the values from the configuration file as defaults. In response to each prompt, you must either press Enter to use the configuration file value, or type your installation-specific value.

Use this mode when you have determined that most of the IBM-supplied defaults are appropriate for your installation, and you prefer to supply the few needed modifications interactively in response to script prompts.

Note, however, that certain values have no IBM defaults, and these always require your input.

### 40.16.3 WebSphere Application Server OEM Edition configuration process

The WebSphere Application Server OEM configuration process occurs in three stages, for both the WebSphere Application Server OEM and z/OSMF applications, in the following order:

- ▶ Stage 1 configuration- there are three modes
  - Interactive mode (with an override file) - “Advanced”
  - Fastpath mode
  - Interactive mode (without an override file) - “Typical”
- ▶ Stage 2 security setup
  - Submit security customization jobs, BBOSBRAK, BBOSBRAM, and BBOCBRAK
- ▶ Stage 3 server instance creation
  - Invoke WASOEM.sh in create mode

**Note:** Most phases are driven through the use of z/OS UNIX configuration scripts. Phase 1 can be run interactively (interview style) or silently (fastpath mode).

### 40.16.4 Configure z/OS for full Incident Log functionality

The z/OSMF Incident Log uses existing best practices for data management for problem determination.

Consider the following implementation options:

- ▶ Sysplex dump directory (required).
- ▶ Use of System Logger for SYSLOG (OPERLOG) and LOGREC.  
The z/OSMF V1R12 Incident log also supports the creation of diagnostic log snapshots based on the SYSLOG and LOGREC data sets, and the OPERLOG and LOGREC sysplex log streams.
- ▶ Dump analysis and elimination (DAE) is active and its symptom data set is available.
- ▶ Automatic dump data set allocation.
- ▶ AMATERSE program is enabled to run.
- ▶ CEA and System REXX components are available.
- ▶ CIM server setup.

### 40.16.5 Authorizing additional users

With V1R12, there is a new script allowing you to authorize new users. This new script *izuadduser.sh* has the following syntax:

```
izuadduser.sh -file /etc/zosmf/izuconfig1.cfg -userid USERID
```

This script creates a REXX exec *izuadduser\_USERID.rexx* with RACF commands for authorizing an existing z/OS user ID to z/OSMF and its configured plug-ins. The content of the exec is automatically tailored for the plug-ins you select when configuring z/OSMF.

**Note:** This script replaces the scripts *izuaddcoreuser.sh* and *izuaddloguser.sh* from the previous release.

### z/OSMF overall configuration description

Table 40-2 shows the steps needed to configure z/OSMF.

The first column describes the steps for configuring z/OSMF. The second column lists the commands used to invoke the script to perform the action. The third column identifies the user required to perform the action.

Table 40-1 z/OSMF configuration steps

Action to perform	Script invocation	Performed by
Step 1: Create the initial configuration	<code>izusetup.sh -file /etc/zosmf/izuconfig1.cfg -config [...other options...]</code>	Superuser
Step 2: Run the security commands	<code>/etc/zosmf/izuconfig1.cfg.rexx</code>	Security administrator
Step 3: Verify the RACF security setup	<code>izusetup.sh -file /etc/zosmf/izuconfig1.cfg -verify racf</code>	Security administrator
Step 4: Prime the z/OSMF data file system	<code>izusetup.sh -file /etc/zosmf/izuconfig1.cfg -prime</code>	Superuser
Step 5: Complete the setup	<code>izusetup.sh -file /etc/zosmf/izuconfig1.cfg -finish</code>	z/OSMF administrator

Action to perform	Script invocation	Performed by
Step 6: Access the z/OSMF Welcome task	At the end of the z/OSMF configuration process, you can verify the success of your configuration changes by opening your browser to the z/OSMF Welcome task.	Any authorized z/OSMF user

## 40.17 Preparing your workstation for z/OSMF

To work with z/OSMF, your workstation requires any of the following browsers:

- ▶ Microsoft Windows XP 32-bit
- ▶ Windows 7 Professional 32-bit
- ▶ Windows Vista Business Edition 32-bit

No other versions of Windows or any other operating systems are supported at this time. The z/OSMF interface is optimized for a window resolution of 1024 by 768 pixels or higher. Therefore, it is best practice to set your window resolution to that size. Otherwise, you can experience clipping of content.

To access z/OSMF on the z/OS system, your workstation requires one of the following web browsers.

- ▶ Mozilla Firefox Version 3.5 or Version 3.0 (with a minimum service level of 3.0.15)
- ▶ Microsoft Internet Explorer Version 7 or Version 8

Table 40-2 lists the supported browsers and workstation platforms.

*Table 40-2 Supported web browsers and workstation platforms*

Browser	Windows XP 32-bit	Windows Vista Business Edition 32-bit	Windows 7 Professional 32-bit
Firefox 3.0.15	Yes	Yes	No
Firefox 3.5	Yes	Yes	Yes
Internet Explorer 7	Yes	Yes	No
Internet Explorer 8	Yes	Yes	Yes

### 40.17.1 Browser considerations

To work with z/OSMF, your browser must be enabled for JavaScript. z/OSMF uses session cookies to track which users are logged in from a specific browser.

On your workstation, if you want to have multiple users logged in from a single location, or if you want to log into multiple servers at the same host name, you might need to either launch another browser instance (as with Internet Explorer), or configure another browser profile (as with Firefox).

For information about creating Firefox profiles, see the Mozilla website:

<http://www.mozilla.com>

If you plan to use the Internet Explorer browser to work with WLM service definitions, ensure that the browser is enabled for automatic prompting for file downloads. This setting prevents the file download blocker from being invoked when you download service definitions to your workstation. Otherwise, if automatic prompting is disabled (the default setting), the download blocker prompts you to accept these file downloads, causing your z/OSMF session to be reloaded and your active tabs to be closed. To avoid this disruption, enable automatic prompting for file downloads. For more information, see the online help for the Workload Management task.

### Internet Explorer 7 browser responsiveness

Using the Internet Explorer 7 browser can result in slow responsiveness if you open multiple tabs, work with large WLM service definitions, or use the Monitoring Desktops task for long periods of time. As an alternative, you can use the Mozilla Firefox 3.5 browser or Internet Explorer 8 browser.

**Note:** Various users run Microsoft Windows Internet Explorer Version 8 with the Compatibility View feature enabled, which allows websites to appear as they do when viewed with Internet Explorer Version 7. In such cases, the environment checker identifies the browser as Internet Explorer V7 for the environment option Browser Version and Name. This behavior is normal.

## 40.18 Summary

The IBM z/OS Management Facility is a new program product (5655-S28) that is provided at no additional cost for z/OS clients. It provides support for a modern, web browser based-management solution for z/OS.

z/OSMF delivers solutions in a task-oriented user interface. The functions include:

- ▶ Configuration Assistant for z/OS Communication Server

The Configuration Assistant for z/OS Communication Server provides simplified configuration and setup of TCP/IP policy-based networking functions.

- ▶ Incident Log

The Incident Log provides a consolidated list of SVC Dump-related problems, along with details and diagnostic data captured with each incident. It also facilitates sending the data for further diagnostics.

- ▶ Links

Links to resources provide common launch point for accessing resources beyond z/OSMF.

- ▶ Resource Monitoring (new for V1R12)

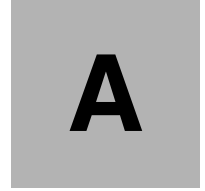
- ▶ WLM Policy Editor (new for V1R12)

WLM Policy Editor facilitates the creation and editing of WLM service definitions, the installation of WLM service definitions, and the activation of WLM service policies.

- ▶ z/OSMF Administration

z/OSMF Administration provides authorization services for administrators, including adding users, defining roles, and dynamically adding links to non-z/OSMF resources.





## **SDSF Java API example**

This appendix provides a complete example of how to use the SDSF Java API.

## SDSF Java API complete example

Example A-1 shows how to use the SDSF Java API to search for jobs in the ST panel according to their completion code. For each job found with an ABEND code, the program then uses the JZOS Java toolkit to browse the job's JESMSGLOG and print the ABEND symptom dump, described by message IEA995I.

Example A-1 shows how to run the example program and sample output.

*Example: A-1 SDSF Java API complete example*

---

```
import com.ibm.zos.sdsf.core.*;
import com.ibm.jzos.*;
import java.io.IOException;
import java.util.List;

public class GetAbendedJobsFromSDSF {

    public static void main(String[] args) {

        // verify arguments
        if (args.length > 0) {
            System.out.println(
                ">> Searching ABENDED jobs in prefix " + args[0] + "...");
        }
        else {
            System.out.println("Usage: java GetAbendedJobsFromSDSF <job-prefix>");
            System.exit(4);
        }

        // get the list of jobs matching job-prefix
        ISFRequestSettings settings = new ISFRequestSettings();
        settings.addISFPrefix(args[0] + "*");

        ISFStatusRunner runner = new ISFStatusRunner(settings);
        List<ISFStatus> stJobList = null;

        try {
            stJobList = runner.exec();
        }
        catch (ISFException e) {
            System.out.println("ST runner error");
            System.exit(8);
        }

        // scan the returned list of jobs
        for (ISFStatus stJob: stJobList) {
            String jobRc = stJob.getValue("retcode");

            if (jobRc.startsWith("ABEND")) {

                // print job details
                System.out.println(
                    ">> " + stJob.getValue("retcode") + " detected for job " +
```

```

        stJob.getValue("jname") +
        " (" + stJob.getValue("jobid") + ")" +
        " owned by " + stJob.getValue("ownerid") + ". Details:"
    );

    // print the IEA995I message from job JESMSGLG
    stJob.browseAllocate();
    try {
        printJESMSGLG(runner.getRequestResults());
    }
    catch (IOException ioe) {
        System.out.println("IO error reading JESMSGLG.");
        System.exit(8);
    }
}
}

private static void printJESMSGLG(ISFRequestResults allocResults)
    throws IOException {

    ZFile          zFile      = null;
    boolean        keepPrint = false;
    int            printPos  = 0;
    ISFAllocationEntry jesmsglg = allocResults.getAllocationList().get(0);

    try {
        zFile = new ZFile(
            "//DD:" + jesmsglg.getDDName(), "rb,type=record,noseek"
        );

        byte[] recBuf = new byte[zFile.getLrecl()];
        int nRead;
        String encoding = ZUtil.getDefaultPlatformEncoding();
        while ((nRead = zFile.read(recBuf)) >= 0) {
            String record = new String(recBuf, 0, nRead, encoding);

            printPos = record.indexOf("IEA995I");
            if (printPos > -1) {
                keepPrint = true;
                while (((nRead = zFile.read(recBuf)) >= 0) && keepPrint) {
                    String symdump = new String(recBuf, 0, nRead, encoding);

                    if (symdump.indexOf("END OF SYMPTOM DUMP") > -1)
                        keepPrint = false;

                    System.out.println(symdump.substring(printPos));
                }
            }
        }
    }
    finally {
        zFile.close();
    }
}
}

```

```
}  
}
```

---

In the next figure we show how to run the example Java program. We run the program to search for jobs in prefix PELEG\*. In this case, the program found one job, PELEG806, which ended with an S806 ABEND code.

```
PELEG @ SC74:/u/peleg>java GetAbendedJobsFromSDSF PELEG  
>> Searching ABENDED jobs in prefix PELEG...  
>> ABEND S806 detected for job PELEG806 (JOB07315) owned by PELEG. Details:  
SYSTEM COMPLETION CODE=806 REASON CODE=00000004  
TIME=11.41.52 SEQ=00208 CPU=0000 ASID=0050  
PSW AT TIME OF ERROR 070C1000 813159B6 ILC 2 INTC 0D  
NO ACTIVE MODULE FOUND  
NAME=UNKNOWN  
DATA AT PSW 013159B0 - 8400181E 0A0D18FB 180C181D  
AR/GR 0: 91DE55D6/00001F00 1: 00000000/84806000  
2: 00000000/00000000 3: 00000000/00000000  
4: 00000000/00000000 5: 00000000/007FF0F8  
6: 00000000/000000FF 7: 00000000/00000000  
8: 00000000/007CE150 9: 00000000/01315EB8  
A: 00000000/00000000 B: 00000000/00000000  
C: 00000000/00000000 D: 00000000/007CE150  
E: 00000000/84806000 F: 00000000/00000004  
END OF SYMPTOM DUMP
```

Figure A-1 Running the SDSF Java API example



# B

## **WTOR messages on Auto-Reply**

In this appendix we provide a complete list of all WTOR messages included on AUTOR00 parmlib member.

## B.1 Auto-Reply AUTOR00 parmlib member

Example B-1 shows the AUTOR00 parmlib member that is distributed in SYS1.IBM.PARMLIB.

*Example: B-1 AUTOR00 parmlib member*

```
***** Top of Data *****
/*****/ 00010001
/* */ 00020001
/* Auto-Reply Policy Specifications */ 00030001
/* */ 00040001
/* PROPRIETARY STATEMENT= */ 00050001
/** Proprietary Statement *****/ 00060001
/* */ 00070001
/* LICENSED MATERIALS - PROPERTY OF IBM */ 00080001
/* 5694-A01 COPYRIGHT IBM CORP. 2010 */ 00090001
/* */ 00100001
/* STATUS= HBB7770 */ 00110001
/* */ 00120001
/** End of Proprietary Statement *****/ 00130001
/* */ 00140001
/* Function: Provide auto-reply policy definitions for common WTORs */ 00150001
/* */ 00160001
/* Notes : The message descriptions are just comments. In some */ 00170001
/* cases, the WTORs refer to other messages and those */ 00180001
/* other messages are included here for documentation. */ 00190001
/* */ 00200001
/* For JES2 messages, the wildcard ? is used as the first */ 00210001
/* character since the first character is installation */ 00220001
/* dependent. */ 00230001
/* */ 00240001
/* The rule delay time is a suggested value. Actual @P3A*/ 00250001
/* values are determined by the WTOR owners. @P3A*/ 00260001
/* */ 00270001
/* Rules used to determine if a WTOR should be considered for */ 00280001
/* auto-reply processing: */ 00290001
/* */ 00300001
/* 1 System Detected Problem */ 00310001
/* */ 00320001
/* If the system detected an error during execution of an */ 00330001
/* operator initiated action, for example if an operator */ 00340001
/* reply could cause corruption, an auto-reply is needed. */ 00350001
/* */ 00360001
/* Reply: Cancel the action */ 00370001
/* Rationale: Resources could be held up; respond quickly. */ 00380001
/* Worst case is that the user has to reinitiate */ 00390001
/* the action. */ 00400001
/* Delay time: 60 seconds (reject quickly and allow user */ 00410001
/* to reinitiate) */ 00420001
/* */ 00430001
/* */ 00440001
/* 2 System Detected Recovery Issue */ 00450001
/* */ 00460001
/* Tough love on sick but not dead situations. */ 00470001
```

```

/* */ 00480001
/* Reply: Terminate */ 00490001
/* Rationale: Want to quickly address before situation */ 00500001
/* further deteriorates. */ 00510001
/* Delay time: 60 seconds */ 00520001
/* */ 00530001
/* */ 00540001
/* 3 System Detected Dynamic Changes */ 00550001
/* */ 00560001
/* For dynamic changes that were made, choose the latest */ 00570001
/* system active configuration when there is a recognized */ 00580001
/* discrepancy. */ 00590001
/* */ 00600001
/* Reply: Option that reflects latest configuration */ 00610001
/* Rationale: Dynamic changes were intended, but not */ 00620001
/* hardened. Return to intended state (dynamic) */ 00630001
/* Delay Time: 30 seconds */ 00640001
/* */ 00650001
/* */ 00660001
/* 4 Confirmation WTORs */ 00670001
/* */ 00680001
/* If a generic confirmation message, reply negative. */ 00690001
/* */ 00700001
/* Reply: Negative confirmation (e.g., NO, CANCEL ...) */ 00710001
/* Rationale: If the operator has not responded immediately */ 00720001
/* to the message, assume there is some confusion */ 00730001
/* and do not automatically assume the original */ 00740001
/* command was correctly entered. Allow him to */ 00750001
/* re-enter the command. */ 00760001
/* Delay time: 60 seconds */ 00770001
/* */ 00780001
/* */ 00790001
/* 5 Continue with IPL */ 00800001
/* */ 00810001
/* If there is a condition that is preventing the system */ 00820001
/* from IPLing, reply to allow the system to continue to */ 00830001
/* IPL. */ 00840001
/* */ 00850001
/* Reply: Option to allow IPL to continue. (e.g., GO, */ 00860001
/* CONTINUE ...) */ 00870001
/* Rationale: If the condition is preventing the system from */ 00880001
/* ipling and the system needs to be up to */ 00890001
/* correct the condition, reply to allow the */ 00900001
/* system to continue the IPL. */ 00910001
/* Delay time: 60 seconds */ 00920001
/* */ 00930001
/* */ 00940001
/* 6 Component/Product Recommended Values */ 00950001
/* */ 00960001
/* Component level expert specified values. */ 00970001
/* */ 00980001
/* Reply: Specified by component level expert */ 00990001
/* Rationale: Component Level expert specifications may */ 01000001
/* over-ride auto-reply rules. */ 01010001
/* Delay time: Specified by component level expert */ 01020001

```

```

/* */ 01030001
/* */ 01040001
/* Change Activity: */ 01050001
/* $LO=AUTOR HBB7770 081225 PDSS: Auto-Reply support */ 01060001
/* $P1=ME15924 HBB7770 080421 PDED: Fix message id type */ 01070001
/* $P2=ME16939 HBB7770 090831 PDSS: Fix typo in prolog */ 01080001
/* $P3=ME17157 HBB7770 090917 PDSS: Update XCF delay values */ 01090001
/* $L1=ME17489 HBB7770 091029 PDSS: Add RMM messages */ 01100001
/* */ 01110001
/*****/ 01120001
/* $HASP070 SPECIFY RECOVERY OPTIONS ('RECOVER' OR 'TERMINATE' OR */ 01130001
/* 'SNAP' AND, OPTIONALLY, ',NODUMP') */ 01140001
/* */ 01150001
/* Rule: 2 */ 01160001
/* */ 01170001
Msgid(?HASP070) Delay(30S) Reply(TERMINATE) 01180001
/*****/ 01190001
/* $HASP294 WAITING FOR RESERVE (VOL volser). REPLY 'CANCEL' TO */ 01200001
/* END WAIT */ 01210001
/* */ 01220001
/* Rule: 1 */ 01230001
/* */ 01240001
Msgid(?HASP294) Delay(30S) Reply(CANCEL) 01250001
/*****/ 01260001
/* $HASP360 jobname REQUESTS ACCESS TO JESNEWS (Y OR N) */ 01270001
/* */ 01280001
/* Rule: 4 */ 01290001
/* */ 01300001
Msgid(?HASP360) Delay(60S) Reply(N) 01310001
/*****/ 01320001
/* $HASP457 FORWARDED DATA SET NAME FOUND. SHOULD JES2 FORWARD? */ 01330001
/* ('Y' OR 'N') */ 01340001
/* */ 01350001
/* Rule: 3 */ 01360001
/* */ 01370001
Msgid(?HASP457) Delay(60S) Reply(Y) 01380001
/*****/ 01390001
/* $HASP811 REPLY Y TO CONTINUE OR N TO TERMINATE START PROCESSING */ 01400001
/* */ 01410001
/* Rule: 5 */ 01420001
/* */ 01430001
Msgid(?HASP811) Delay(30S) Reply(Y) 01440001
/*****/ 01450001
/* ANTU2220D "READY FOR FLASHCOPY. REPLY 'I' TO INITIATE, 'C' TO */ 01460001
/* CANCEL" */ 01470001
/* */ 01480001
/* Rule: 4 */ 01490001
/* */ 01500001
Msgid(ANTU2220D) Delay(60S) Reply(C) 01510001
/*****/ 01520001
/* ANT8925A device_number TERMINATE STORAGE CONTROL SESSION */ 01530001
/* session_number? REPLY 'Y' OR 'N' */ 01540001
/* */ 01550001
/* Rule: 4 */ 01560001
/* */ 01570001

```



```

Msgid(ANTX8925A) Delay(60S) Reply(N) 01580001
/*****/ 01590001
/* ANTX8926A device_number RECOVER STORAGE CONTROL SESSION */ 01600001
/* session_number? REPLY 'Y' OR 'N' */ 01610001
/* */ 01620001
/* Rule: 4 */ 01630001
/* */ 01640001
Msgid(ANTX8926A) Delay(60S) Reply(N) 01650001
/*****/ 01660001
/* ANTX8942A REMOVE device_number FROM STORAGE CONTROL SESSION */ 01670001
/* session_number? REPLY 'Y' OR 'N' */ 01680001
/* */ 01690001
/* Rule: 4 */ 01700001
/* */ 01710001
Msgid(ANTX8942A) Delay(60S) Reply(N) 01720001
/*****/ 01730001
/* ANTX8943A TERMINATE ALL type SDM SESSIONS? REPLY 'Y' OR 'N' */ 01740001
/* */ 01750001
/* Rule: 4 */ 01760001
/* */ 01770001
Msgid(ANTX8943A) Delay(60S) Reply(N) 01780001
/*****/ 01790001
/* ANTX8944A TERMINATE STORAGE CONTROL SESSION session_id ON STORAGE*/ 01800001
/* CONTROL ssid? REPLY 'Y' OR 'N' */ 01810001
/* */ 01820001
/* Rule: 4 */ 01830001
/* */ 01840001
Msgid(ANTX8944A) Delay(60S) Reply(N) 01850001
/*****/ 01860001
/* ANTX8973A device_number SUSPEND STORAGE CONTROL SESSION */ 01870001
/* session_number? REPLY 'Y' OR 'N' */ 01880001
/* */ 01890001
/* Rule: 4 */ 01900001
/* */ 01910001
Msgid(ANTX8973A) Delay(60S) Reply(N) 01920001
/*****/ 01930001
/* ANTX8978A EXECUTE CREFRESH FORCE? REPLY 'Y' OR 'N' */ 01940001
/* */ 01950001
/* Rule: 4 */ 01960001
/* */ 01970001
Msgid(ANTX8978A) Delay(60S) Reply(N) 01980001
/*****/ 01990001

```

---

## B.2 List of Auto-Reply messages

Example B-2 shows a list of the auto-reply messages that exist in the AUTOR00 parmlib member.

*Example: B-2 List of auto-reply messages*

---

```

$HASP070 SPECIFY RECOVERY OPTIONS ('RECOVER' OR 'TERMINATE' OR 'SNAP' AND,
OPTIONALLY, ',NO DUMP')
$HASP294 WAITING FOR RESERVE (VOL volser). REPLY 'CANCEL' TO END WAIT
$HASP360 jobname REQUESTS ACCESS TO JESNEWS (Y OR N)

```

\$HASP457 FORWARDED DATA SET NAME FOUND. SHOULD JES2 FORWARD? ('Y' OR 'N')  
 \$HASP811 REPLY Y TO CONTINUE OR N TO TERMINATE START PROCESSING  
 ANTU2220D "READY FOR FLASHCOPY. REPLY 'I' TO INITIATE, 'C' TO CANCEL"  
 ANTX8925A device\_number TERMINATE STORAGE CONTROL SESSION session\_number? REPLY 'Y' OR 'N'  
 ANTX8926A device\_number RECOVER STORAGE CONTROL SESSION session\_number? REPLY 'Y' OR 'N'  
 ANTX8942A REMOVE device\_number FROM STORAGE CONTROL SESSION session\_number? REPLY 'Y' OR 'N'  
 ANTX8943A TERMINATE ALL type SDM SESSIONS? REPLY 'Y' OR 'N'  
 ANTX8944A TERMINATE STORAGE CONTROL SESSION session\_id ON STORAGE CONTROL ssid? REPLY 'Y'  
 OR 'N'  
 ANTX8973A device\_number SUSPEND STORAGE CONTROL SESSION session\_number? REPLY 'Y' OR 'N'  
 ANTX8978A EXECUTE CREFRESH FORCE? REPLY 'Y' OR 'N'  
 ANTX8981A SUSPEND ALL XRC SESSIONS? REPLY 'Y' OR 'N'  
 ARC0310A CAN TAPE volser BE MOUNTED ON DEVICE devno? REPLY Y OR N  
 ARC0311A SYSTEM TIMER INOPERABLE - CAN volser BE MOUNTED? REPLY Y OR N  
 ARC0314A CAN THE nvol VOLUME(S) ABOVE BE MOUNTED FOR {RECYCLE | RECOVER | RESTORE}? REPLY Y  
 OR N  
 ARC0346A OPEN HAS NOT COMPLETED FOR TAPE volser MOUNTED IN DEVICE ddd. REPLY Y TO START  
 ADDITIONAL minutes MINUTES  
 ARC0380A RECALL WAITING FOR VOLUME volser IN USE BY HOST procid, FUNCTION function. REPLY  
 WAIT, CANCEL, OR MOUNT  
 ARC0387A RECOVER OF DATA SET dsname TIMED OUT WAITING FOR TAPE VOLUME volser TO BECOME  
 AVAILABLE. SHOULD THE DATA SET RECOVER REQUEST CONTINUE TO WAIT? REPLY Y OR N  
 ARC0505D {PRIMARY SPACE MANAGEMENT | SECONDARY SPACE MANAGEMENT | INTERVAL MIGRATION |  
 AUTOMATIC BACKUP | AUTOMATIC DUMP} ABOUT TO START, REPLY 'Y' TO START OR 'N' TO SKIP IT  
 ARC0803A WARNING: AUDIT OF CATALOG MAY DEGRADE PERFORMANCE, REPLY 'Y' TO START AUDIT OR 'N'  
 TO CANCEL AUDIT COMMAND  
 ARC0825D RECYCLE TAPE LIST CREATED, DSN=dsname. DO YOU WISH TO CONTINUE? REPLY 'N' TO STOP  
 RECYCLE OR 'Y' WHEN READY TO MOUNT TAPES.  
 ARC0962A ALL VOLUMES NOT CONTAINED IN THE SAME TAPE LIBRARY OR STORAGE GROUP. ENTER 'C' TO  
 CANCEL OR MAKE CORRECTION AND ENTER 'R' TO RETRY  
 ARC6254A ABACKUP CANNOT ALLOCATE TAPE VOLUME volser BECAUSE ANOTHER DFSMSHSM FUNCTION HAS  
 IT IN USE. RETRY? REPLY Y OR N  
 BPXI078D STOP OF NLSname\_type REQUESTED, REPLY 'Y' TO PROCEED. ANY OTHER REPLY WILL CANCEL  
 THIS STOP.  
 BPXI083D RESPAWNABLE PROCESS job\_name ENDED. REPLY R TO RESTART THE PROCESS. ANYTHING ELSE  
 TO END THE PROCESS.  
 BPXM055D THIS SYSTEM WILL BE DISABLED AS A FILESYSTEM OWNER. REPLY 'Y' TO CONTINUE OR ANY  
 OTHER KEY TO EXIT.  
 BPXM061D REPLY "Y" TO PROCEED WITH ACTIVATION. ANY OTHER REPLY ENDS THE COMMAND.  
 BPXM063D REPLY "Y" TO PROCEED WITH DEACTIVATION. ANY OTHER REPLY ENDS THE COMMAND.  
 BPXM120D F BPX0INIT,FILESYS=funcname SHOULD BE USED WITH CAUTION. REPLY 'Y' TO CONTINUE.  
 ANY OTHER REPLY TERMINATES.  
 CBR9810D Reply 'QUIT' to terminate or 'GO' to proceed with recovery.  
 CNZ0012D REPLY 'CANCEL' TO CANCEL COMMAND cmdtext.  
 CNZ9009D CONTINUE WITH MIGRATION? REPLY N TO ABORT OR Y TO CONTINUE  
 CP04205I CPC name: Enter '1' to keep waiting for pending activation or '2' to accept  
 current capacity setting  
 CP04206I CPC name: Enter '1' to keep waiting for pending deactivation or '2' to accept  
 current capacity setting  
 EDG0103D DFSMSrmm SUBSYSTEM INTERFACE IS INACTIVE - ENTER "IGNORE", "CANCEL" OR "RETRY"  
 EDG1107D REQUESTS WAIT TO BE PROCESSED - REPLY "STOP", "QUIESCE", "RESTART", OR "M=xx"  
 EDG1200D I/O ERROR ON CONTROL DATA SET WHEN PROCESSING MESSAGE msg\_number, REPLY EITHER  
 "RETRY" OR "CANCEL"  
 EDG1203D INVENTORY MANAGEMENT PREVENTED PROCESSING OF MESSAGE msg\_number, REPLY EITHER  
 "RETRY" OR "CANCEL"  
 EDG2103D PERMANENT JOURNAL ERROR - REPLY "R" TO RETRY, "I" TO IGNORE, "D" TO DISABLE OR  
 "L" TO LOCK  
 EDG2106D JOURNAL AND CONTROL DATASET DO NOT MATCH - REPLY "C" TO CANCEL, "D" TO DISABLE OR  
 "L" TO LOCK  
 EDG3213D ANOTHER GETVOLUME CURRENTLY IN PROGRESS - ENTER "RETRY", "CANCEL", OR "IGNORE"

EDG4000D JOURNAL FILE IS LOCKED DURING action FOR volser BY jobname, stepname, ddname;  
ENTER "RETRY" OR "CANCEL"

EDG4001D DFSMSrmm I/O ERROR IN action FOR volser BY jobname, stepname, ddname; ENTER  
"RETRY" OR "CANCEL"

EDG4010D BACKUP IN PROGRESS DURING action FOR volser BY jobname, stepname, ddname; ENTER  
"RETRY" OR "CANCEL"

EDG8008D DFSMSrmm I/O ERROR DURING task function REQUEST FOR volser - ENTER "RETRY" OR  
"CANCEL"

EDG8010D BACKUP IN PROGRESS DURING task function REQUEST FOR volser - ENTER "RETRY" OR  
"CANCEL"

EDG8011D DFSMSrmm SUBSYSTEM IS NOT ACTIVE DURING task function FOR volser - ENTER "RETRY"  
OR "CANCEL"

EDG8013D DFSMSrmm JOURNAL FILE IS LOCKED DURING task function REQUEST FOR volser - ENTER  
"RETRY" OR "CANCEL"

EDG8102D DFSMSrmm SUBSYSTEM NOT ACTIVE DURING function PROCESSING FOR volser - ENTER  
"RETRY", "IGNORE", OR "CANCEL"

EDG8108D DFSMSrmm I/O ERROR DURING function PROCESSING FOR volser - ENTER "RETRY" OR  
"CANCEL"

EDG8110D BACKUP IN PROGRESS DURING function PROCESSING FOR VOLUME volser - ENTER  
"RETRY" OR "CANCEL"

EDG8113D DFSMSrmm JOURNAL FILE IS LOCKED DURING function function PROCESSING FOR VOLUME  
volser - ENTER "RETRY" OR "CANCEL"

EDG8121D ENTER volume req\_volser INTO LIBRARY lib\_name AND REPLY "RETRY", OTHERWISE REPLY  
"CANCEL" OR "CONTINUE"

EDG8122D ENTER volume req\_volser INTO LIBRARY lib\_name AND REPLY "RETRY", OTHERWISE REPLY  
"CANCEL"

EDG8123D ENTER volume req\_volser EXPORTED IN STACKED VOLUME stack\_volser LOCATION  
loc\_name SHELF shelf\_number HOME LOCATION home - IMPORT VOLUME TO LIBRARY lib\_name AND  
REPLY "RETRY", OTHERWISE REPLY "CANCEL"

ERB306D sid : REPLY WITH OPTIONS OR GO

IAT2855 JES3/VTAM OPEN ACB FAILURE, SPECIFY "RETRY" TO ATTEMPT OPEN AGAIN OR "TERM" TO  
TERMINATE SNARJP

IAT3155 SPOOL DATA INTEGRITY CHECKING IS ACTIVE; DO YOU WANT TO TURN IT OFF? (OFF OR  
CONTINUE)

ICH15041A VALID RESPONSES ARE 'CONTINUE' OR 'CANCEL'

IEA029D {IEASVC|ALLOC|SCHED} PARMLIB MEMBER HAS AN UNBALANCED COMMENT. REPLY YES TO  
CONTINUE IPL OR NO TO RESPECIFY {SVC|ALLOC|SCHED} PARM

IEA893A NOT READY. REPLY U WHEN DEVICES ARE READY, OR NO IF NOT MOUNTING. dev,dev,...

IEE799D VARY CONSOLE DELAYED - REPLY RETRY OR CANCEL

IEE800D CONFIRM VARY FORCE FOR {nnnnnnn|dev[, (dev,...)]} - REPLY NO OR YES

IEF739D CONFIGURATION CHANGE DELAYED DUE TO EXCESSIVE WAIT ON PREVIOUS EDT - REPLY  
'WAIT' OR 'TERM'.

ISG017D CONFIRM PURGE REQUEST FOR SYSTEM sysname - REPLY NO OR YES

ISG027D CONFIRM RESTART-RING FOR SYSTEM sysname - REPLY NO OR YES

ISG082D CONFIRM REBUILD-RING FOR SYSTEM sysname - REPLY NO OR YES

ISG101D CONFIRM PURGE FOR ACTIVE SYSTEM sysname - REPLY NO OR YES

ISG117D CONFIRM REACTIVATE SHOULD BE COMPLETED - REPLY NO OR YES

ISG186D GRS CTC dev WAS TARGET OF VARY OFFLINE, FORCE. REPLY KEEP TO HAVE GRS RETAIN THE CTC  
OR FREE TO REMOVE THE CTC FROM GRS.

ISG220D REPLY C TO CANCEL RNL CHANGE COMMAND, OR S FOR SUMMARY OFRNL CHANGE PROGRESS.

ISG366D CONFIRM SETGRS REQUEST ON SYSTEM system-name. REPLY {ENQMAXA|ENQMAXU}=value TO  
CONFIRM OR C TO CANCEL.

ISG366D CONFIRM REQUEST TO MIGRATE THE CNS TO system-name. REPLY CNS=system-name TO CONFIRM  
OR C TO CANCEL.

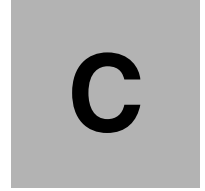
ISG880D WARNING GRSRNL=EXCLUDE IS IN USE. REPLYING FORCE WILL RESULT IN THE USE OF  
SPECIFIED RNSL. REPLY C TO CANCEL.

IXC222D REPLY U TO USE RESOLVED DATA SETS OR R TO RESPECIFY COUPLEXX

IXC289D REPLY U TO USE THE DATA SETS LAST USED FOR typename OR C TO USE THE COUPLE DATA  
SETS SPECIFIED IN COUPLExx IXC394A ARM ELEMENT IN USE. REPLY Y TO CONFIRM THAT elementname

SHOULD BE DEREGISTERED OR N TO CANCELIXC403D sysname STARTED INITIALIZATION AT hh:mm:ss.  
REPLY W TO WAIT FOR sysname OR I TO COMPLETE INITIALIZATION.

---



## **z/OS Upload utility**

The IBM z/OS Problem Documentation Upload utility is a parallel File Transfer Protocol (FTP) utility that is 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 shorter transmission time for very large data sets (such as standalone dumps). You can optionally encrypt the data sets.

## C.1 z/OS Problem Documentation Upload utility

There are two work files for each parallel FTP session (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 starts. This process continues between the A and the B files, until everything in the input file is sent.

You can have up to 20 parallel FTP sessions running simultaneously. The work data sets are dynamically allocated and can range in size from 1 MB to 9,999 MB. You can experiment to see what works best in your environment, but here are some guidelines:

- ▶ Start with three or four parallel FTP sessions. Too many parallel FTP sessions can saturate the network link.
- ▶ Use medium size work data sets.

If the work data sets are very small in relationship 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 a lot of delay by starting and stopping the FTP sessions for each file.

If the work data sets are very large in relationship 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, which means the copy time is not overlapping with data that needs to be sent through FTP. For example, if you were sending 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.

The parallel FTP program 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 TRSMAN to compress the input data set before using the parallel FTP program to send it to the IBM FTP site. In addition, 192-bit triple Data Encryption Standard (DES) can be requested by using the CIPHER\_KEY keyword. Without the keyword, the data is just compressed. With 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) and 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 at:

<http://www.ibm.com/systems/z/os/zos/bkserv/r10pdf/#zarchpops>

## C.2 z/OS Problem Documentation Upload utility JCL

When coding JCL for this utility, you might want to place your user ID and password into a separate concatenated 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.

This section lists the following JCL examples:

- ▶ “Example: Simple FTP connection” in Figure C-1 on page 817.

- ▶ “Example: FTP connection using a proxy server” in Figure C-2 on page 818.
- ▶ “Example: FTP connection using a proxy server with proxy user ID” in Figure C-3 on page 819.
- ▶ “Example: Using a proxy server with the FTPCMDS DD statement” in Figure C-4 on page 819.
- ▶ “Example: Using a proxy server with the FTPCMDS DD statement and a port specification on the TARGET\_SYS parameter” in Figure C-5 on page 820.
- ▶ “Example: Forcing PASSIVE mode using FTPCMDS inline DD statement” in Figure C-6 on page 820.
- ▶ “Example: Using a surrogate userid and NETRC data set” in Figure C-7 on page 821.

```

//STEPLIB DD DISP=SHR,DSN=SHANNON.MTFTP.LOAD
//SYSUDUMP DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DISP=SHR,DSN=H44IPCS.WESSAMP.TRKS055K
//SYSIN DD *
USERID=anonymous
PASSWORD=anonymous
TARGET_SYS=testcase.boulder.ibm.com
TARGET_DSN=wessamp.bigfile
WORK_DSN=wes.ftpout
CC_FTP=03
WORK_DSN_SIZE=500
DIRECTORY=/toibm/mvs/
PMR=12345.123.123
//

```

Figure C-1 Simple FTP connection//FTP EXEC PGM=MTFTPS

The JCL statements for the z/OS Problem Documentation Upload utility are as follows:

- |                 |  |
|-----------------|--|
| <b>SYSPRINT</b> | The data set can be either SYSOUT or a sequential data set. The data set must be RECFM=FB, LRECL=134.  |
| <b>SYSUT1</b>   | The sequential data set to transfer to IBM. The record formats currently supported are fixed, fixed blocked, variable, and variable blocked. For example, the dump data set that you are sending to IBM. The input has to be either a sequential data set, a member of a PDS, or a member of a PDSE.   |
| <b>SYSIN</b>    | A sequential data set that uses the following control statements. The data set must be RECFM=FB, LRECL=80. The parallel FTP utility is managed through the following JCL control statements with these guidelines: <ul style="list-style-type: none"> <li>▶ Use an asterisk (*) in the first column to indicate comments.</li> <li>▶ Keywords must start in column one.</li> <li>▶ Use control statements that are in the form VERB=OPERAND.</li> <li>▶ Mixed case verbs and operands are allowed.</li> <li>▶ The operand starts in the column after the equal sign and goes to the first blank column except TARGET_SYS, DIRECTORY, CIPHER_KEY, ACCOUNT, USERID, and PASSWORD, which can contain blanks.</li> </ul> |

- ▶ Anything after the first blank is ignored except for any operands that can contain blanks. In those cases, do not use blanks from column one to the end of the operand.

**TARGET\_SYS** The name of the TCP/IP system to transfer the files to using FTP. One through 64 characters, dotted decimal format is allowed, no default value, can contain blanks, and it must be specified.

If using a proxy server, this should be the name of the proxy server.

You can include additional FTP command parameters on the TARGET\_SYS parameter by using the z/OS UNIX specifications as shown in the topic “FTP command -- Entering the FTP environment” in *z/OS Communications Server: IP User's Guide and Commands*, SC31-8780. For example, to trace output (-d) and use a specific ftpdata\_filename (-f//"WES.MYFTP.DATA"):
 

```
TARGET_SYS=-d -f//"WES.MYFTP.DATA" testcase.boulder.ibm.com
```

Use the -p parameter to specify an alternate IP stack.

**USERID** The user ID on the target system that is used to send the files. One through 64 characters, no default value, does not have to be specified, and can contain imbedded blanks. If USERID and PASSWORD are not supplied and NETRCLEVEL=2, the values from the NETRC data set is used for the FTP sessions.

If using a proxy server, this can be the full login to the remote system in the format `userid@remote.system.name`.

## Traversing the firewall

You can use the examples shown in Figure C-2 and Figure C-3 on page 819 as a starting point to traverse a firewall or proxy server. There are very 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, 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.

```
//FTP EXEC PGM=MTFTPS
//STEPLIB DD DISP=SHR,DSN=SHANNON.MTFTP.LOAD
//SYSUDUMP DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DISP=SHR,DSN=H44IPCS.WESSAMP.TRKS055K
//SYSIN DD *
USERID=anonymous@testcase.boulder.ibm.com
PASSWORD=proxyid@proxypw
TARGET_SYS=your.proxy.server.name
TARGET_DSN=wessamp.bigfile
WORK_DSN=wes.ftpout
CC_FTP=03 WORK_DSN_SIZE=500
DIRECTORY=/toibm/mvs/
PMR=12345.123.123
//
```

Figure C-2 FTP connection using a proxy server



```

//FTP EXEC PGM=MTFTPS
//STEPLIB DD DISP=SHR,DSN=SHANNON.MTFTP.LOAD
//SYSUDUMP DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DISP=SHR,DSN=H44IPCS.WESSAMP.TRKS055K
//SYSIN DD *
USERID=anonymous@proxyid@testcase.boulder.ibm.com
PASSWORD=proxyid@proxypw
TARGET_SYS=proxy.server.name
TARGET_DSN=wessamp.bigfile
WORK_DSN=wes.ftpout
CC_FTP=03
WORK_DSN_SIZE=500
DIRECTORY=/toibm/mvs/
PMR=99999.000.000
//

```

Figure C-3 FTP connection using a proxy server with proxy user ID

```

//FTP EXEC PGM=MTFTPS
//STEPLIB DD DISP=SHR,DSN=SHANNON.MTFTP.LOAD
//SYSUDUMP DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DISP=SHR,DSN=H44IPCS.WESSAMP.TRKS055K
//FTPCMDS DD DISP=SHR,DSN=WES.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=99999.000.000
//
*****
* The WES.FTPCMDS.DATA data set contains *
* user anonymous pw userid@company.com *
*****

```

Figure C-4 Using a proxy server with the FTPCMDS DD statement

```

//FTP EXEC PGM=MTFTPS
//STEPLIB DD DISP=SHR,DSN=SHANNON.MTFTP.LOAD
//SYSUDUMP DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DISP=SHR,DSN=H44IPCS.WESSAMP.TRKS055K
//FTPCMDS DD *
user anonymous pw userid@company.com
//SYSIN DD *
USERID=proxyid@testcase.boulder.ibm.com
PASSWORD=proxypw
TARGET_SYS=proxy.server.name
TARGET_DSN=SVCD
WORK_DSN=HLQ.FTPOUT
CC_FTP=03
WORK_DSN_SIZE=500
DIRECTORY=/toibm/mvs/
PMR=99999.000.000
CIPHER_KEY=PMR99999sad

```

Figure C-5 A proxy server with FTPCMDS statement and port specification TARGET\_SYS parameter

```

//FTP EXEC PGM=MTFTPS
//STEPLIB DD DISP=SHR,DSN=SHANNON.MTFTP.LOAD
//SYSUDUMP DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DISP=SHR,DSN=H44IPCS.WESSAMP.TRKS055K
//FTPCMDS DD *
locsite fwfriendly
//SYSIN DD *
USERID=proxyid@testcase.boulder.ibm.com
PASSWORD=proxypw
TARGET_SYS=proxy.server.name
TARGET_DSN=SVCD
WORK_DSN=HLQ.FTPOUT
CC_FTP=03
WORK_DSN_SIZE=500
DIRECTORY=/toibm/mvs/
PMR=99999.000.000
CIPHER_KEY=PMR99999sad
//

```

Figure C-6 Forcing PASSIVE mode using FTPCMDS inline DD statement

Figure C-7 on page 821 shows the sample 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.

**Note:** You can find information about the use of the NETRC data set in *z/OS Communications Server: IP User's Guide and Commands*, SC31-8780.

You can find information about the use of the FTP.DATA data set in *z/OS Communications Server: IP Configuration Reference*, SC31-8776.

```

//FTP EXEC PGM=MTFTPS
//SYSUDUMP DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DISP=SHR,DSN=H44IPCS.WESSAMP.GB01
//SYSIN DD *
TARGET_SYS=-f"//'WES.MYFTP.DATA'" testcase.boulder.ibm.com
PMR=99999.999.000
DIRECTORY=/toibm/mvs/
TARGET_DSN=wes.gb01tst
work_dsn=wes.ftpout
cc_ftp=03 WORK_DSN_SIZE=500
//
For this instance, 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 C-7 Using a surrogate userid and NETRC data set





## **BCPii Metal C-example**

This appendix contains an example written in C in order to call the BCP internal interface, along with the JCL to compile, assemble and linkedit this proposed code.

It also gives an example of a System REXX which allows to call such an example from the console as if it is a z/OS command.

This should offer the opportunity to insert such new z/OS commands into classical automation tools used for managing the exploitation of the zSeries systems.

## D.1 C code

Example D-1 shows the source of the C program proposed as an example for programming the BCP internal interface.

*Example D-1 C-program calling BCPii*

```
*-----*
*
*   MODULE NAME= HWI001
*
*   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 elementY18";
    char null_char;
} CPC_elements;
typedef struct IMAGE_elements {
    char elementY9";
    char null_char;
} IMAGE_elements;
/* ----- */
/* Function prototypes */
/* ----- */
```

```

void print_diagarea(HWI_DIAGAREA_TYPE DA);
void CPC_AnswerArea_into_Array(char Y~,CPC_elements Y~,int);
void Image_AnswerArea_into_Array(char Y~,IMAGE_elements Y~,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_targetY17~;
    char IMAGE_targetY8~;
    char *CPCconnecttypevaluePtr, *IMAGEconnecttypevaluePtr;
    char answerareaY9000~;
    char *answerarea_ptr;
    char HWI_MMODEL_valueY20~;
    char HWI_OSTYPE_valueY20~;
    char * query_ptr;
    HWI_DIAGAREA_TYPE diagarea;
    HWI_CONNTOKEN_TYPE CPCinconnecttoken, CPCoutconnecttoken;
    HWI_CONNTOKEN_TYPE IMAGEoutconnecttoken;
    HWI_QUERYPARM_TYPE queryparmY1~;
    CPC_elements List_of_CPCsY500~;
    IMAGE_elements List_of_ImagesY100~;
    struct WTO_PARM { /* mapped by IEZWPL macro. */
        unsigned short len; /* total length of structure */
        unsigned short code; /* routing code flag */
        char textY80~;
    } 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, 0x00, sizeof(List_of_CPCs));
answerarealen = sizeof(List_of_CPCs);
answerarea_ptr = &answerarea;
/* ----- */
/* 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_CPCs[i].element);
    CPCconnecttypevaluePtr = &CPC_target;
    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_CPCs[i].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;
        queryparm.AttributeIdentifier = HWI_MMODEL;
        queryparm.AttributeValue_Ptr = &HWI_MMODEL_value;
        queryparm.AttributeValueLen = sizeof(HWI_MMODEL_value);
        queryparm.AttributeValueLenReturned = -1;
        query_ptr = (char *)&queryparm;
        /* ----- */
        /* 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 = &answerarea;
/* ----- */
/* 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, 0x00,
            sizeof(IMAGEoutconnecttoken));
        strcpy(IMAGE_target,List_of_Images[k].element);
        IMAGEconnecttypevaluePtr = &IMAGE_target[0];
        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_Images[k].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;
queryparmY0^.AttributeIdentifier = HWI_OSTYPE;
queryparmY0^.AttributeValue_Ptr = &HWI_OSTYPE_valueY0;
queryparmY0^.AttributeValueLen=sizeof(HWI_OSTYPE_value);
queryparmY0^.AttributeValueLenReturned = -1;
query_ptr = (char *)&queryparmY0;
/* ----- */
/* 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 textÝ80";
    } 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 text[80];
} 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, &CPCCConnectToken);
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, CPCCConnectToken, &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(CPCCConnectToken);
}
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 * CPCCConnectTokenPtr) /* Output */
{
    HWI_CONNTOKEN_TYPE CPCinconnecttoken;
    char CPC_target[17];
    char * CPCCConnectTypeValuePtr;
    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_targetY0";
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_targetY9";
char * ImageConnectTypeValuePtr;
HWI_DIAGAREA_TYPE diagarea;
memset(Image_target,' ',sizeof(Image_target));
strncpy(Image_target,"LPC",strlen("LPC"));
ImageConnectTypeValuePtr = &Image_targetY0";
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 text[80];
    } 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 text[100];
    } 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 answerareaŸ",
                               CPC_elements ListŸ",
                               int number)
{
    int i,j;
    i = 0; j = 0;
    while( i < number)
    {
        ListŸi".elementŸ0" = answerareaŸj";
        ListŸi".elementŸ1" = answerareaŸj+1";
        ListŸi".elementŸ2" = answerareaŸj+2";
        ListŸi".elementŸ3" = answerareaŸj+3";
        ListŸi".elementŸ4" = answerareaŸj+4";
        ListŸi".elementŸ5" = answerareaŸj+5";
        ListŸi".elementŸ6" = answerareaŸj+6";
        ListŸi".elementŸ7" = answerareaŸj+7";
        ListŸi".elementŸ8" = answerareaŸj+8";
        ListŸi".elementŸ9" = answerareaŸj+9";
        ListŸi".elementŸ10" = answerareaŸj+10";
        ListŸi".elementŸ11" = answerareaŸj+11";
        ListŸi".elementŸ12" = answerareaŸj+12";
    }
}

```

```

        ListYi".elementY13" = answerareaYj+13";
        ListYi".elementY14" = answerareaYj+14";
        ListYi".elementY15" = answerareaYj+15";
        ListYi".elementY16" = answerareaYj+16";
        ListYi".elementY17" = answerareaYj+17";
        ListYi".null_char = '\0';
        i++;
        j=j+18;
    }
}
/* ----- */
/* Save the returned list of images to the answer area.      */
/* ----- */
void Image_AnswerArea_into_Array(char answerareaY",
                                IMAGE_elements ListY",
                                int number)
{
    int i,j;
    i = 0; j = 0;
    while( i < number)
    {
        ListYi".elementY0" = answerareaYj";
        ListYi".elementY1" = answerareaYj+1";
        ListYi".elementY2" = answerareaYj+2";
        ListYi".elementY3" = answerareaYj+3";
        ListYi".elementY4" = answerareaYj+4";
        ListYi".elementY5" = answerareaYj+5";
        ListYi".elementY6" = answerareaYj+6";
        ListYi".elementY7" = answerareaYj+7";
        ListYi".elementY8" = answerareaYj+8";
        ListYi".null_char = '\0';
        i++;
        j=j+9;
    }
}

```

---

## D.2 JCL to compile, assemble and link-edit

Example D-2 shows the JCL for compiling, assembling and linking the C-code provided in Example D-1 on page 824.

*Example D-2 JCL to compile and linkedit*

---

```

//COMPC2 JOB (999,POK),CONWAY,MSGCLASS=H,REGION=OM,
// NOTIFY=&SYSUID
//JOBPARM S=*,L=9999
/*-----
/* C COMPILE PROCEDURE FOR METAL C
/*-----
//EDCCB PROC INFILE=, < INPUT ... REQUIRED
// CREGSIZ='OM', < 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 DSN=&LIBPRFX..SCEERUN2,DISP=SHR
//          DD DSN=&LIBPRFX..SCEERUN,DISP=SHR
//          DD DSN=&LNGPRFX..SCNCMP,DISP=SHR
//SYSIN   DD DSN=&INFILE,DISP=SHR
//SYSLIB  DD DSN=SYS1.SIEAHRV.H,DISP=SHR
//          DD PATH='/usr/include/metal/',
//          PATHOPTS=(OWRONLY)
//OPTFILE DD DSN=&OPTFILE,DISP=SHR
//SYSLIN  DD DSN=&OUTFILE,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSOUT  DD SYSOUT=*
//SYSPRT  DD SYSOUT=*
//          PEND
/*-----
/*  CALL OF INSTREAM PROCEDURE EDCCB
/*-----
//CL      EXEC EDCCB,
//          OPTFILE=LAFITTE.CNTL(COPTS),
//          INFILE=LAFITTE.CNTL(HWIQ01),
//          OUTFILE=LAFITTE.SOURCE.ASM(HWIQ01)
/*-----
/*  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(HWIQ01),DISP=SHR
/*-----
/*  LINK METAL MODULE
/*-----
//L.CSSLIB DD DSN=SYS1.CSSLIB,DISP=SHR
//L.SYSLMOD DD DSN=LAFITTE.APF.LOAD(HWIQ01),DISP=SHR
//L.SYSLIN DD DISP=(OLD,DELETE),DSN=&&OBJ
//          DD *
//          INCLUDE CSSLIB(HWISSET)
//          INCLUDE CSSLIB(HWIEVENT)
//          INCLUDE CSSLIB(HWILIST)
//          INCLUDE CSSLIB(HWIDISC)
//          INCLUDE CSSLIB(HWICONN)
//          INCLUDE CSSLIB(HWICMD)
//          SETCODE AC(1)
//          NAME HWIQ01(R)
/*
//
//

```

---

## D.2.1 C compiler options

Example D-3 shows the options specified in step CL for compiling the C code as shown in Example D-2 on page 842.

### *Example D-3 Compiler options*

---

```
SERVICE(BUILD DATE: 10/18/2010)
      METAL
      NOSEARCH
      search(/usr/include/metal/)
```

---

## D.3 BCPii example as a System REXX

Example D-4 shows an example of a REXX procedure calling the BCPii-calling program.

### *Example D-4 REXX procedure calling the 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 D-5 on page 844, when inserted in the SYS1.ARX, can then be called as a new z/OS command as shown in Example D-5.

### *Example D-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=0000400000000000C705BA95383ED8A6  
869 Query zHybrid  
SC74 2010347 09:18:30.68 TSU02712 IEA631I OPERATOR LAFITTE NOW INACTIVE, SYST

---



# 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

For information about ordering these publications, see “How to get Redbooks” on page 850. Note that some of the documents referenced here may be available in softcopy only.

- ▶ *z/OS Distributed File Service zSeries File System Implementation z/OS V1R11*, SG24-6580
- ▶ *z/OS Version 1 Release 11 Implementation*, SG24-7729
- ▶ *z/OS Version 1 Release 10 Implementation*, SG24-7605
- ▶ *z/OS Version 1 Release 9 Implementation*, SG24-7427
- ▶ *z/OS Version 1 Release 8 Implementation*, SG24-7265
- ▶ *z/OS Version 1 Release 7 Implementation*, SG24-6755
- ▶ *DFSMS V1.10 and EAV Technical Guide*, SG24-7617
- ▶ *DFSMS Release 10 Technical Update*, SG24-6120
- ▶ *z/OS V1R8 DFSMS Technical Update*, SG24-7435
- ▶ *z/OS V1R7 DFSMS Technical Update*, SG24-7225
- ▶ *Using SDSF in a JES3 Environment*, REDP-4531
- ▶ *z/OS Planned Outage Avoidance Checklist*, SG24-7328

## Other publications

These publications are also relevant as further information sources:

- ▶ *z/OS Migration*, GA22-7499
- ▶ *z/OS Planning for Installation*, GA22-7504
- ▶ *z/OS Introduction and Release Guide*, GA22-7502
- ▶ *z/OS Program Directory*, GI10-0670
- ▶ *z/OS License Program Specifications*, GA22-7503
- ▶ *SMP/E V3R5.0 for z/OS Commands*, SA22-7771
- ▶ *z/OS MVS Planning: Operation*, SA22-7601
- ▶ *z/OS MVS Initialization and Tuning Reference*, SA22-7592
- ▶ *z/OS MVS Authorized Assembler Services Guide*, SA22-7608
- ▶ *z/OS Problem Management*, G325-2564
- ▶ *z/OS MVS Programming: Assembler Services Reference, Volume 2 (IAR-XCT)*, SA22-7607

- ▶ *z/OS MVS System Commands*, SA22-7627
- ▶ *z/OS MVS Installation Exits*, SA22-7593
- ▶ *z/OS MVS System Messages, Volume 8 (IEF-IGD)*, SA22-7638
- ▶ *z/OS MVS JCL Reference*, SA22-7597
- ▶ *z/OS SDSF Operation and Customization*, SA22-7670
- ▶ *z/OS MVS Initialization and Tuning Reference*, SA22-7592
- ▶ *z/OS DFSMS Using the New Functions*, SC26-7473
- ▶ *z/OS UNIX System Services Planning*, GA22-7800
- ▶ *z/OS UNIX System Services Command Reference*, SA22-7802
- ▶ *z/OS UNIX System Services Programming: Assembler Callable Services Reference*, SA22-7803
- ▶ *z/OS Distributed File Service zSeries File System Administration*, SC24-5989
- ▶ *z/OS UNIX System Services File System Interface Reference*, SA22-7808
- ▶ *z/OS Security Server RACF Command Language Reference*, SA22-7687
- ▶ *ServerPac: Installing Your Order* (no order number; custom-built to your order)
- ▶ *ServerPac: Using the Installation Dialog*, SA22-7815
- ▶ *System z10 Processor Resource/Systems Manager Planning Guide*, SB10-7153
- ▶ *z/OS MVS System Commands*, SA22-7627
- ▶ *z/OS MVS System Messages, Volume 9*, SA22-7639
- ▶ *z/OS MVS Setting Up a Sysplex*, SA22-7625
- ▶ *z/OS MVS Capacity Provisioning User's Guide*, SC33-8299
- ▶ *z/OS MVS Programming: Sysplex Services Guide*, SA22-7617
- ▶ *z/OS MVS Diagnosis: Reference*, GA22-7588
- ▶ *z/OS MVS Planning: Workload Management*, SA22-7602
- ▶ *z/OS DFSMSdfp Diagnosis*, GY27-7618
- ▶ *z/OS DFSMSdfp Storage Administration*, SC26-7402
- ▶ *z/OS DFSMSdfp Advanced Services*, SC26-7400
- ▶ *z/OS DFSMSshm Storage Administration*, SC35-0421
- ▶ *z/OS DFSMSshm Implementation and Customization Guide*, SC35-0418
- ▶ *z/OS DFSMSrmm Implementation and Customization Guide*, SC26-7405
- ▶ *z/OS DFSMSrmm Reporting*, SC26-7406
- ▶ *z/OS DFSMS Using Data Sets*, SC26-7410
- ▶ *z/OS DFSMS Using the New Functions*, SC26-7473
- ▶ *z/OS DFSMS Access Method Services for Catalogs*, SC26-7394
- ▶ *z/OS DFSORT Installation and Customization*, SC26-7524
- ▶ *IBM z/OS Management Facility Configuration Guide*, SA38-0652
- ▶ *z/OS MVS Diagnosis: Tools and Service Aids*, GA22-7589
- ▶ *z/OS IBM Health Checker for z/OS: User's Guide*, SA22-7994
- ▶ *z/OS MVS Planning: Workload Management*, SA22-7602

- ▶ *z/OS Communications Server: IPv6 Network and Application Design Guide*, SC31-8885
- ▶ *z/OS Communications Server IP Configuration Guide*, SC31-8775
- ▶ *z/OS TSO/E Customization*, SA22-7783
- ▶ *z/OS XL C/C++ Run-Time Library Reference*, SA22-7821
- ▶ *z/OS Infoprint Server Customization*, S544-5744
- ▶ *z/OS Infoprint Server Operation and Administration*, S544-5745
- ▶ *z/OS XML System Services User's Guide and Reference*, SA23-1350
- ▶ *z/OS Security Server RACF Command Language Reference*, SA22-7687
- ▶ *Processor Resource/Systems Manager Planning Guide*, IBM System z10, SB10-7153
- ▶ *z/OS Metal C Programming Guide and Reference*, SA23-2225
- ▶ *z/Architecture Principles of Operation*, SA22-7832

## Online resources

These Web sites are also relevant as further information sources:

- ▶ The `tsocmd` command Tools and Toys README documentation available at the following site:  
<ftp://ftp.software.ibm.com/s390/zos/tools/tsocmd/tsocmd.readme.txt>
- ▶ Working Draft, Standard for Programming Language C++  
<http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2009/n2857.pdf>
- ▶ Variadic templates  
<http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2007/n2242.pdf>
- ▶ C++0x long long  
<http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2005/n1811.pdf>
- ▶ IBM Virtualization Engine TS7700 Series Bulk Volume Information Retrieval Function User's Guide Version 1.5 at:  
<http://www-03.ibm.com/support/techdocs/atmsastr.nsf/WebIndex/WP101094>
- ▶ Infoprint Port Monitor can be downloaded directly to your Windows system from this Web site:  
<http://www.ibm.com/support/docview.wss?rs=727&uid=psd1P1000597>
- ▶ This tool displays whether z/OS UNIX directory caching is active, and you can get the tool from:  
<ftp://ftp.software.ibm.com/s390/zos/tools/wjsip/wjsipndc.txt>
- ▶ The `wjsfsmon` utility is mentioned in APAR OA29619 and can help you to determine which systems are accessing your zFS read-write file systems and whether your zFS read-write file systems are accessed from multiple systems:  
<http://www.ibm.com/servers/eserver/zseries/zos/unix/tools>  
<http://www-03.ibm.com/systems/z/os/zos/features/unix/bpxalty2.html>  
<ftp://ftp.software.ibm.com/s390/zos/tools/wjsfsmon/wjsfsmon.txt>
- ▶ The `INITSIZE` and `SIZE` specification for structures should be determined by the `CfSizer` (Coupling Facility Structure Sizer) tool:  
<http://www.ibm.com/systems/support/z/cfsizer/>

- ▶ The recommended exclusion list for the migration tracker program is at the following Web site:

<http://www-03.ibm.com/servers/eserver/zseries/zos/downloads/>

## How to get Redbooks

You can search for, view, or download Redbooks, Redpapers, Technotes, draft publications and Additional materials, as well as order hardcopy Redbooks publications, at this Web site:

[ibm.com/redbooks](http://ibm.com/redbooks)

## Help from IBM

IBM Support and downloads

[ibm.com/support](http://ibm.com/support)

IBM Global Services

[ibm.com/services](http://ibm.com/services)



# Index

## Symbols

\_\_IBMCPP\_TR1\_\_  
  C++ compiler option 242  
.MSGCOLR statement 111  
  MPFLSTxx parmlib member 110  
\$D SPOOL,UNITDATA command 622  
\$HASP443 message 626  
\$S SPOOL command 625  
\$T SPOOLDEF command 620, 624  
\$T SPOOLDEF,CYL\_MANAGED=ALLOWED command 622  
\$T SPOOLDEF,TGSPACE=(MAX=nnnn) command 624

## Numerics

3390 Model A 387

## A

ACS routine  
  SHAREOPATIONS(3,3) 91  
ADDSUMM keyword 202  
AF\_INET6 277  
AF\_INET6 NETWORK statement 277  
affinity nodes 73  
ALLOCxx parmlib member  
  MEMDSENQMGMT keyword 254  
  parameter, TEMPDSFORMAT 257  
  SYSTEM TEMPDSFORMAT 257  
AMASPZAP service aid 188  
AMD059D message  
  SADMP prompt 203  
AMDSADDD  
  REXX utility 186  
AMDSADDD utility 186  
ANFUXRSP response notification exit  
  Inforprint Server 211  
aopmig 213  
APAR OA12774  
  RMF HiperDispatch 75  
APAR OA27495  
  zAAP on zIIP support 352  
APAR OA29619 298, 307, 312, 319, 849  
  zFS support 298  
APAR OA29712 298  
  requires APAR OA29619 298  
APAR OA29786 312  
APAR OA30548  
  log stream data set size 93  
  SystemLogger 94  
APAR OA30928  
  GNMP support 592  
APAR OA31112 316, 318  
APAR OA32093  
  Inforprint Server health check 218

APARs for HiperDispatch 74  
ARCPRPDO  
  DFSMSrmm PDA trace records 170  
ARECOVER command 405  
asymmetric cryptographic process 462  
asymmetric encryption 466  
ATRBCARR macro 736  
automatic data set grouping  
  extended mode 210  
AUTOR00 parmlib member 99, 103, 108  
  auto-reply policy 99  
  DFSMSrmm support 163  
  sample parmlib member 105  
auto-reply notification messages 106  
auto-reply policy 98  
auto-reply rules 103  
AUTORxx parmlib member 55, 99, 106  
  syntax rules 100

## B

BACKVOL command 153  
BADTRACK initialization statement  
  EAV considerations 635  
base addressing space 391–392  
basic catalog structure (BCS) 130  
Basic HyperSwap 414, 417, 419  
  XCF communication 425  
BCPii address space 436  
  startup and shutdown 453  
BCPii diagnostics tools 455  
BCPii overview 435  
BCPii protocol 731  
BEGINPARALLEL keyword 745  
BEGINPARALLEL statement  
  z/OS Communications Server 745  
BreakPointValue (BPV)  
  dump data sets 185  
BUFUSEWARN 647  
BUFUSEWARN option 647  
BVIR volume map 172

## C

C/C++ applications 244  
C++ compiler errors 242  
C99 function 242  
C99 Standard 522  
C99 standard  
  restrict keyword 510  
CA reclaim 138, 141  
CA space 137  
Capacity Provisioning Manager 727  
  performance index (PI) 728  
capture UCB for spool  
  JES2 627

- CBLOC keyword 370
- CBLOC paramete 371
- CCCAWMT 77
- CCCCcccH notation 391
- CEA address space
  - BCPii use 453
- CEE.SCEESAMP data set 246
- CEE.SCEEUMAP
  - LE library 59
- CEEPRMxx parmlib member 58, 246, 248
  - run-time options 245
  - using OVR NONOVR attribute 58
- CEX3 coprocessor
  - secure keys 472
- CF structure
  - placement considerations 564
- CFLEVEL requirements 563
- CFRM policy
  - cluster structures 564
- CFSTRHANGTIME interval
  - structure hang recovery 680
- CFSTRHANGTIME keyword
  - SFM policy 679
- CFSTRHANGTIME parameter
  - SFM policy 684, 689
- CFSTRHANGTIME value 683
- CIM 753
- CK panel
  - SDSF health check 610
- cluster structures
  - CFRM policy 564
- CNIDTRxx parmlib member
  - migration assistance tracker 52
- coexistence scenarios
  - EAV systems sharing data sets 409
- COLLECTINACTIVE parameter
  - PFA support 500
- Common Information Model 753
- Common Storage Usage Prediction Repo 497
- Communications Server
  - IPSec and IKE support 777
- Configuration Assistant
  - z/OSMF 763
- Configuration Assistant task
  - z/OSMF 763, 774
- CONSOLxx parmlib member
  - WTO console buffers 114
- CONTROL Q command 114
- CPC Reset Profile 565
- CPF macro 113
- CPMF architecture 536
- CPU Measurement Sampling Facility 540
- cryptographic coprocessor group 539
- cryptographically generated address (CGA) 275
- CSVDYNEX macro 378
- CSVRTLS macro 51
- CSVRTLS services
  - removed from z/OS 51
- CTnGRSxx parmlib member
  - CTRACE options 659

- OPTIONS parameter 658
- CUNUNlxx parmlib member 718
- Customized Offerings Driver 46
- cylinder-managed space
  - EAV volume 390

## D

- D IOS,DCM command 570
- D M=CHP(xx) command 569
- D M=SWITCH 576
- D M=SWITCH command 576
- D RRS command 737
- D RRS,URSTATUS command 741
- D WLM command 587
- D WLM,SYSTEMS command 592
- D XCF,G command
  - display system critical member 666
- Data Encryption Standard (DES) 462
- data sharing
  - relationship to IRD 549
- DATACLAS
  - SHAREOPTIONS(3,3) 91
- DCM 552
- DCOLLECT command 175
- DEFAULTLSNAME 647
- DEFINE EA catalogs 128
- DEFINE RECATALOG command 130
- DES wrapping-key 474
- detected system failures 480
- DEVSUPxx parmlib member
  - NON\_VSAM\_XTIOT 224
  - NON\_VSAM\_XTIOT=YES 170
- DFRMM 170
- DFSMSShm
  - EAV support 404
- DFSMSShm Journal 405
- DFSMSrmm
  - EAV eligible 168
  - specifies EATTR=OPT 169
- DFSMSrmm CLIST Processing dialog 161
- DFSMSrmm Command Menu 160
- DFSMSrmm journal
  - EAS eligible 169
- DIAGSIM 178
- DIAGxx parmlib member
  - CBLOC keyword 370
  - CBLOC parameter 370–372
- directory caching 318
- discretionary workload 357
- DISPLAY 268
- DISPLAY AUTOR command
  - auto-reply support 108
- DISPLAY DUMP,OPTIONS command 206
- DISPLAY GRS command
  - latch contention 269
- DISPLAY GRS commands
  - identify latch number 268
- DISPLAY HS command 417
- DISPLAY HS,STATUS command 425
- DISPLAY IPLINFO command

- zAAP on zIIP support 353
- DISPLAY M=DEV command 575
- DISPLAY MPF command 112
- DISPLAY OPDATA,TRACKING command
  - migration assistance tracker 52
- Display OSAINFO 68
- DISPLAY PROG command 378
- DISPLAY RRS command 738
- DISPLAY RTLS command 51
- DISPLAY SYMBOLS command 369
- DISPLAY XCF,COUPLE command 684
- Distributed Management Task Force 753
- dlsym() function 213
- DMTF 753
- DocumentInfo data structure 212
- Dojo technology 756
- driving system 44
- DS8000 4.0 version 387
- DSCBs 397
- DSNTYPE=EXTREQ
  - DASD dump data sets 185
- DSSXMMODE parameter 152
- dump option
  - DEFERTND=(YES/NO) 205
- duplicate temporary data set name 257
- DVD delivery
  - z/OS ordering 44
- Dynamic Channel-path Management 552
  - activating for the first time 569
  - benefits 553, 555
  - improved resource utilization 553
  - reduced channel requirements 555
  - WLM role 556

## E

- EADSCB=OK 399–400
- EADSCB=OK keyword 399
  - extended attribute DSCBs 399
- EAS 391
- EATTR options
  - dump data set 185
- EATTR value 400
- EATTR=NO 185
- EATTR=OPT 185
- EATTR=OPT JCL parameter
  - JES2 EAV sppol 623
- EAV volume 386
- EAVFTED REXX exec
  - via TSO 362
- EDGJACTP sample JCL 165
- EDGJCEXP sample job 171
- elliptic curves 464
- ENF 68
  - BCPii support 455
- Enhanced PSP Tool (EPSPT) 49
- ESPIE exit 365
- ESPIE exits
  - RTM percolation 364
- ESTAEX exit
  - SPIE override parameter 365

- EWLM subsystem type 53
- EXIT1 and EXIT2 statements
  - CSVLLAxx parmlib member 376
- extended addressable (EA) catalogs 128
- extended addressing space 391, 393
- extended-format sequential data sets 395

## F

- F DFRMM,QUERY ACTIVE command 159
- f hisproc command 540
- F hisproc,BEGIN command 541–542
- F hisproc,END command 542
- F PFA,DISPLAY,CHECKS,DETAIL command 493
- F PFA,UPDATE command 482
- F WLM,GPMP,START command 592
- F ZFS,QUERY,LEVEL command 309
- Fast Response Cache Accelerator support 298, 316
- feature test macro \_TR1\_C99 242
- filesys 308
- FIPS 140 777
- FIPS 140 cryptographic mode for IPsec 763
- FLDSTATS options 642
- FLDSTATS parameter 642
- FLDSTATS(xxxx) keyword 643
- FLOOD and FLOODPOL parameters
  - message flood 644
- FLOOD parameter 644
- FLOODPOL parameter 645
- FLOWA 658
- FOMF03011 message 283
- fork() 64-bit copy processing 118
- fork() processing 117
- format-3 DSCB 397
- format-4 DSCB 398
- format-9 DSCB 397
- FREE command
  - allocated PDS 192
- FREEVOL command 405
- FRR command
  - RESUME(YES/NO) 153
- FRR recovery exits
  - SDWA time of error 366
- FRR SDWA
  - above the 16M line 365
- FRRECOV COPYPOOL FROMDUMP command 155
- FUNCTION keyword
  - IXCJOIN macro 670
- function shipping 301

## G

- GDPS environment
  - peer-to-peer remote copy (PPRC) 699
- GDPS HyperSwap API services 417
- genxlt source 245
- GPMP 69
- Guest Platform Management Provider (GPMP) 69, 585

## H

- Hardware instrumentation services (HIS) 536
- HASP414 message 626
- HASPINDEX data set 604
- health check exception
  - PFA 480
- Health Check History (CKH) panel 610
- health check parameter
  - CPUSLEFTB4NEEDHD 84
- Health Checker
  - BPX.SUPERUSER 270
- HIGHOFFLOAD parameter 94
- HiperDispatch 435
  - overview 72
  - WLM considerations 75
  - z196 processor 583
- HiperDispatch = YES 78
- HiperDispatch messages 74
- HiperDispatch=NO 76, 80
- HiperDispatch=YES 77
- HIS address space 540
- history data
  - SDSF panels 608
- HVEMCA address space
  - GPMP 596
- HWIBCPH address space 443
- HWISTART procedure 436
- HyperSwap API address space 414
- HyperSwap disablement 424
- HyperSwap management address space 414
- HyperSwap resiliency 426
- HyperSwap session 417
- Hypervisor 435
- HZSPRMxx parmlib member
  - USS\_HFS\_DETECTED check 272

## I

- I/O autoconfiguration 698
- IBM DS8000
  - multicylinder units 391
- IBM Health Checker for z/OS
  - PFA checks 479–480
- IBM zEnterprise 196 server 65
- ICSF HCR7770 468
- IDAVDT proclib member 143
- IDAVDT00 parmlib member 143
- IDAVDTxx parmlib member 142
  - VSAM record management trace 142
- IEAARR 366
- IEADMCxx parmlib member
  - DEFERTND option 205
- IEAOPTxx parmlib member
  - HiperDispatch parameters 77
  - HiperDispatch=YES 74
  - TIMESLICES parameter 358
  - zAAP on zIIP support 355
- IEASYMxx parmlib member 369
- IEASYMxx parmlib member 321
- IEASYSxx parmlib member 98

- AUTOR= parameter 99
- auto-reply parameter 98
- large page support 125
- zAAP on zIIP support 353
- IEASYSy parmlib member
  - auto-reply 100
- IEATEDS macro 361
- IEATEDS service 627
- IEAVFTED REXX exec 361
- IEAVMXIT exit 640
- IEAVTED REXX exec 363
- IEAVTFTED REXX exec
  - under IPCS 362
- IEE094D message
  - DEFERTND prompt 205
- IEE256I message
  - zAAP on zIIP support 354
- IEFDDSRV assembler macro 255
- IEFDDSRV macro 256
- IEFSSNxx parmlib member
  - BEGINPARALLEL keyword 744–745
- IFAHONORPRIORITY=NO 78
- IFAHONORPRIORITY=YES 76
- IFASMF DL dump program 642
- IFASMF DL program
  - SMF log streams 228
- IFASMF DP dump program 642
- IGDSMSxx parmlib member 147
  - CA\_RECLAIM keyword 138
- IGDSMSxx parmlib memberVOLSELMSG keyword 147
- IKE Version 2 763
- IKJ55072I message 222
- IKJTSOxx parmlib member
  - IEASYMUP member 323
- Incident Log 765, 767
  - z/OSMF 765
- InfiniBand (CIB) channels 702
- Infoprint Central on z/OS V1R12
  - fallback considerations 218
- Infoprint Port Monitor for Windows 218–219
- Infoprint Server LPD
  - large file support 212
- Infoprint Server Printer Inventory files 217
- INFOPRINT\_V2DB\_CHECK
  - Infoprint Server health check 218
- ini 502
- ini file 503
- INITSTACK parameter 736
- INMN network 69, 585
- Intelligent Resource Director
  - components 550
  - introduction 549
- intra node management network (INMN) 584
- intranode management network (INMN) 582
- IOEFSPRM DD member 307
- IOEPRMxx parmlib member 307
- IOSTmmm module 568
  - refreshing 568
- IP PrintWay extended mode 210
- IP SETDEF statemen 362

- ISFPRMxx parmlib member 613, 617
  - SDSF auto-reply support 110
- ISGENQ service 656
- ISO/IEC TR 19768 242
- IXCJOIN macro 669
  - FUNCTION keyword 670
- IXG2711 messageSystem Logger 89
- IXG282I message 91
- IXG283I message
  - SHAREOPTIONS correction 91
- IXGQBUF version 4
  - new fields 93
- IXLCONN TERMLEVEL keyword 680
- IXLSTR.structure\_name profile 565

## J

- January 19, 2038 251
- Java 1.4
  - PFA requirement 480
- Java 6.0 SR1
  - large page support 124
- JCT utility
  - IATUTJCT 635
- JES2 TSO/E user logon 222

## L

- large page coalesce 125
- large page support 123
- latch identity 269
- LFAREA parameter
  - large page support 125
- LISTCONTROL STATUS subcommand 159
- LOCALSYSAREA
  - REQUEST=GETSTOR 119
- LOCALSYSAREA=NOIYES
  - IARV64 macro 119
- logstream structure
  - health check 609
- LPAR-related data
  - WLM REQLPDAT service 584
- LSNAME 647

## M

- Managed System Infrastructure for Setup (msys for Set-up) 44, 64
- MAXDUMPRECOVERTASKS parameter 154
- MAXDUMPRECOVERTASKS(nn) 154
- MCU 390
- MCUs 390
- mean time to recovery 744
- mean time to wait 237
- MEMDSENQMGMT feature 254
- memory mapping (mmap) 274
- MEMSTALLTIME
  - impaired member 673
  - SFM policy 663, 665
- MEMSTALLTIME parameter 53
- message AMD059D

- operator prompt for ADDSUMM 203
- message arrival rate check 488
- Message Flood Automation 640
  - IEAVMXIT exit 640
- message IOSHM0303 425
- message IOSHM0315I 425
- Metal C 524
- Metro Mirror
  - with HyperSwap session 420
- Metro Mirror (PPRC) 415
- mmap()
  - support for NFS client 328
- Monitor III CPC Capacity report 232
- MOSIZE installation default 181
- MOWRK parameter 181
- MOWRK=YES 181
- MPFLSTxx parmlib member
  - .MSGCOLR statement 111
  - message presentation 110
- MSA-X3 476
- MSGPRT=NONE 178
- MSI dynamic exit routine 746
- msys for Setup 44
- MTTR 360, 744
- MTTR trace 360
- MTTtr addressing scheme
  - JES2 spool 620
- multicylinder unit
  - EAV volume 390
- multicylinder units 390
- multiple TSO/E logons
  - JES2 support 222, 626
- my fault errors 367

## N

- NAMEMANGLING(ANSI) 524
- Namespace Association 520
- New cryptographic algorithms for IPSec and IKE 763
- NFS SMF record 42 329
- nfsstat 334
- nfsstat shell program 334
- NOBUFFS 647
- non-IBM control units 568
- NORWSHARE 309, 311
- norwshare 307, 310
- norwshare file system 311
- not my fault errors 367

## O

- OMEGAMON z/OS Management Console 750
- OpenSSH
  - an XPLINK application 346
  - support for key rings 343
- OpenSSH V1R2 336
- OSA-Express3 68
- OSA-Express-3 in QDIO mode 67

## P

- p hisproc 542
- PARTIALOK keyword 153
- PARTREL macro 136
- PAV-alias UCB 387
- PDA trace records
  - DFSMSrmm creation 170
- PFA 480
- PFA install script
  - AIRSHREP.sh 502
- pfuser 502, 504
- Postprocessor CPU Activity report 232
  - statistics on single work units 231
- Postprocessor Enterprise Disk Systems (ESS) report 239
- PPRC paths 417
- Predictive Failure Analysis (PFA) 480
- Problem Documentation Upload Utility 766
- PROGxx parmlib member 354, 376
  - LNKLST UPDATE 379
- PROJECTCPU keyword 357
- PROPERTY statement
  - SDSF properties 609
- public key 462
- public RSA host key 344

## R

- RCF\_LCCA\_ABOVE\_16M 370
- RECORD file format
  - z/OS UNIX files 278
- Redbooks Web site 850
  - Contact us xxiii
- Related 847
- REPORT MISSINGFIX command 49
- RESMGR servic 366
- REXX exec
  - AMDSADDD 184
- REXX utility
  - AMDSADDD 186
- RFC 4301 compliance 778
- RFC 5014 14, 32
- RFC-5014 275
- RMF Postprocessor JCL 231
- RMF STORF report
  - storage frames for HyperSwap 427
- RMM CLIST data set 169
- RMM LISTCONTROL STATUS
  - TSO subcommand 159
- Roles task
  - z/OSMF administration 789
- RSM data copy service
  - SDUMP usage 121
- RTM enhancements 364
- RUN\_ON\_MOUNT=NO parameter
  - HFS health check 271
- Runtime Diagnostics 482, 485
- Run-Time Library Services (RTL) 51
- RWSHARE 309, 311
- rwshare 307, 311

## S

- S99DASUP bit
  - flag S99FLAG2 256
- S99DASUP flag
  - suppress type 30 records 254
- SADMP data sets
  - cylinder managed space 184
- SADMP dump data set utility 184
- SAF key ring 344
- SAPI JES2 \$TRACE 628
- SDATA option
  - GRSQ 656
- SDM journal data sets 408
- SDSF Health Checker display 608
- SDSF Java API 598
- SDSF SR panel
  - auto-reply support 110
- SDUMP improvements 121
- SEARCH command 616
- SET CKLIM command 609, 611
- SET command
  - tracker exclusion list 52
- SET RTLS command 51
- SET SMS=xx command 138
- SET UNI system command 716
- SETALLOC command
  - SYSTEM TEMPDSFORMAT setting 257
  - TEMPDSFORMAT option 259
- SETAUTOR command 110
  - auto-reply processing 107
- SETCEE command 58, 248
- SETCON command
  - migration assistance tracker 52
- SETDEF subcommand 189
  - output to s PDS 190
- SETHS command
  - planned HyperSwap 417
- SETHS RESUMEIO command 419
- SETHS UNBLOCK command 425
- SETIOS DCM,REFRESH command 568
- SETIOS DCM=OFF command 571
- SETPROG command
  - LNKLST UPDATE 379
- SETPROG EXIT command 376
- SETPROG EXIT,ADD command 378
- SETSMF FLOOD(ONIOFF) command 644
- SETSMS CA\_RECLAIM(NONE) command 138
- SETSYS USECYLINDERMANAGEDSPACE(Y) command 405
- SETXCF START,POLICY,TYPE=CFRM command 693
- sftp command 346
- SHA-1 algorithm
  - for keys of all sizes 470
- SHAREOPTIONS(3,3)
  - Sysem Logger 90
- showattr command 329
- SMB server 318
  - APAR OA31112 298
- SMF arrival rate check 487, 489
- SMF data set recording 641

- SMF flood statistics 642
- SMF flooding statistics 642
- SMF NOBUFFS parameter 648
- SMF record type 42 subtype 26 328
- SMF record type 90 dubtype 33
  - auto-reply support 110
- SMF records type 113 535
- SMF Type 113 record data 545
- SMF Type 30 records
  - S99DASUP flag 254
- SMF30DAS 256
- SMF70 subtype 1
  - CPU data section SMF70WTD 237
- SMFEWTM macro 641
- SMFINTVAL parameter 545
- SMFPRMxx parmlib member
  - FLOOD(ONIOFF) 644
  - FLOODPOL parameter 645
- SMP/E V3R5 49
- SO/IEC 9899
  - 1999 Standard C (C99) namespace 242
- source address selection 276
- source address selection support 275
- SPIEOVERRIDE=YES 365
- SSI 83
  - JES2 support 628
- SSI function code 83 612
- STATECHANGE parameter 543
- subchannel set
  - third set with z/OS V1R12 66
- subchannel sets
  - 3 on IBM zEnterprise 196 (z196) servers 65
- supervised learning support 487
- superzap
  - EAV support 188
- Superzap control statements 188
- SYS1.IBM.PARMLIB(AUTOR00) 103
- SYS1.MANxx data sets 642
- SYS1.PARMLIB(SMFPRMxx) 329
- SYS1.SCUNIMG(CUNIDHC2) 716
- SYSBCPII CTRACE component
  - BCP11 cuts trace records 455
- SYSIKJUA enqueue 222
- sysplex=filesys 306–307, 309
- sysplex-aware 302
- system area
  - new 64-bit storage area 119
- System Managed Rebuild
  - used by WLM 563
- System Managed Rebuild support 563
- System REXX 752
- System SSL
  - ECC-related data structures 467
  - running in FIPS mode 468
- SYSTRACE subcommand 194

## T

- TCBCMPC field 368
- TCBEndingAbnormally flag 368
- TCBENDNG bit 367

- TEMPDSFORMAT 257
- TEMPDSFORMAT(INCLUDELABEL) 258
- TEMPDSFORMAT(UNIQUE) 258
- temporary data set name 258
- TERMLEVEL keyword 680
- Timed Event Data report 361
- TPC-R new options 419
- TPC-R support
  - HyperSwap 417
- track-managed space 390
- translation lookaside buffer (TLB) 123
- TS7700 Virtualization Engine 171
- tsocmd utility 292

## U

- UCB nocapture 168
- ucmap source 245
- Unicode 715
- Unicode conversion image 718
- UR family 737

## V

- variadic templates 516
- VARY CU command 713
- VARY SWITCH command 573, 576
- VARY SWITCH,OFFLINE command 573
- VERBX IEAVTSFS command 121
- vertical lows
  - HiperDispatch processing 73
- virtual file system interface (VFS) 279
- VRSRETAIN report 167
- VSAM partial release 325
- VSAM SHAREOPTIONS attribute
  - System Logger 92
- VVDS VVR 136

## W

- WAITINIO service 122
- WBEM 754
- WebSphere Application Server OEM Edition 756
- WebSphere MQ
  - JES2 sysplex support 612
- wjfsmon utility 319
- WLM cluster structures 564
- WLM LPAR CPU Management
  - CF structure 563
- work units (WEBs)
  - RMF CPU Activity report 231
- workload balancing
  - relationship to IRD 549
- wrapping key set 474
- wrapping keys 476

## X

- XCF member impairment 664
- XCF member status monitoring 663
- XCF system status monitoring 662
- XES hang detect function 678

XPXSRPTS  
 RMF Postprocessor report ddname 228  
XTIOT option 224  
XTIOT support 21  
 Binder support 6  
 DFSMS BAM 20  
 DFSORT 20  
 RACF 35  
 TSO/E 15  
 uncaptured UCBs 40

## Z

z/OS Management Facility (z/OSMF) 750  
z/OS Problem Documentation Upload Utility 766  
z/OS SMB server 316  
z/OS UNIX directory caching 318  
z/OSMF  
 Configuration Assistant 763  
z/OSMF Incident Log 765  
z196 65  
z196 (2817-780) 66  
zAAP on zIIP  
 restrictions 353  
ZAAPAWMT 77  
zAAPAWMT  
 HiperDispatch 79  
ZAAPAWMT keyword 357  
ZAAPZIIP support 352  
zBX 708  
zDAC 65  
zEnterprise 196 servers  
 CFCC Level 17 67  
zEnterprise BladeCenter Extension 582  
zEnterprise Unified Resource Manager 68, 585  
zFS APAR OA29619 298  
zFS V1R9 302  
zfsadm config command 309  
zfsadm configquery -syslevel command 310  
zfsadm lsaggr command 315  
zHPF protocols 66  
ZIIPAWMT 77  
 HiperDispatch 79  
ZIIPAWMT keyword 357  
ZOSMIGV1R12\_INFOPRINT\_INVSIZE  
 Infoprint Server health check 218





# **z/OS Version 1 Release 12 Implementation**

(1.5" spine)  
1.5" x 1.998"  
789 <-> 1051 pages







# z/OS Version 1 Release 12 Implementation



**z/OS base, DFSMS,  
HiperDispatch,  
z/OSMF, XCF, GRS,  
SMF, PSA, XML**

**LE, XML, z/OS UNIX,  
zFS, EAV, HyperSwap,  
RRS, TSO/E, Unicode**

**BCPii, PFA, JES2,  
JES3, SDSF,  
Auto-reply, RSM**

This IBM Redbooks publication describes changes in installation and migration when migrating from a current z/OS V1R10 and z/OS V1R11 to z/OS V1R12. Also described are tasks to prepare for the installation of z/OS V1R12, including ensuring that driving system and target system requirements are met, and coexistence requirements are satisfied. New migration actions are introduced in z/OS V1R12. This book focuses on identifying some of the new migration actions that must be performed for selected elements when migrating to z/OS V1R12.

This book describes the following enhancements:

- ▶ z/OS V1R12 installation, HiperDispatch, System Logger, Auto-reply to WTORs, Real Storage Manager (RSM)
- ▶ DFSMS, DFSORT, Services aids, z/OS Infoprint Server, TSO/E, RMF, Language Environment, BCP allocation
- ▶ XML System Services, z/OS UNIX System Services, BCP supervisor, Extended Address Volumes
- ▶ HyperSwap. BCPii, (de)ciphering, Predictive Failure Analysis, C language, Hardware instrumentation services
- ▶ FICON dynamic channel-path management, Workload Manager, SDSF, JES2, JES3, SMF, GRS, XCF, HCD
- ▶ Unicode, Capacity provisioning, RRS, Parallel subsystems initialization
- ▶ z/OS Management Facility (z/OSMF)

## **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](http://ibm.com/redbooks)**

SG24-7853-00

ISBN 0738434981