Advanced File System and Utilities for Digital UNIX
Version 4.0
AA-QTPZA-TE

Guide to File System Administration

Digital Equipment Corporation
Maynard, Massachusetts

Revision/Update Information: This is a new manual.
Operating System and Version: Digital UNIX Version 4.0 or higher. Future maintenance releases may require higher versions.
Software Version: Advanced File System and Utilities Version 4.0
Date: April 1996

Order Number: AA-QTPZA-TE

Contents

List of Figures

List of Tables

Preface

The Advanced File System (AdvFS) is a file-system option on the Digital UNIX™ (formerly DEC™ OSF/1®) operating system. It provides rapid crash recovery, high availability, and a flexible structure that enables you to manage your file system while it is on line.

The Advanced File System Utilities (AdvFS Utilities) is a layered product that extends the file-system capabilities by including utilities to add volumes, clone *filesets*, defragment *file domains*, stripe files, and balance file domains. The AdvFS Utilities also includes a *graphical user interface* (GUI) to simplify AdvFS management.

This guide describes the AdvFS and AdvFS Utilities products in detail. It provides information on features and functions, and it gives suggestions on how to use these functions. Most functions can be accomplished through either the command line or GUI; command-line procedures are detailed in this manual. The AdvFS GUI provides online help with instructions on completing specific procedures.

Structure of This Document

This administration guide contains the following chapters and appendixes:

Chapter 1 introduces AdvFS, describes the features and benefits, and explains the licensing strategy.

Chapter 2 explains the configuration of AdvFS and how to set up your system.

Chapter 3 explains how to set up and use quotas on AdvFS.

Chapter 4 explains fileset cloning and how to perform online backups with AdvFS filesets.

Chapter 5 provides information and instructions for maximizing the performance of AdvFS.

Chapter 6 provides diagnostic and troubleshooting instructions for specific situations that can be handled at the file-system level.

Appendix A lists all of the commands available for AdvFS functions.

Appendix B provides guidelines for converting from the UNIX File System (UFS) to AdvFS and from AdvFS to UFS.

The Glossary contains definitions of terms unique to AdvFS or that need clarification.

Related Information

In addition to this guide, the AdvFS documentation set includes the following documents:

Advanced File System Utilities Installation Guide

Advanced File System Utilities Reference Manual

Advanced File System Utilities Release Notes

Digital UNIX Software License Management

Digital UNIX Guide to System Administration

Conventions

The following conventions are used in this guide:

Convention	Meaning
UPPERCASE and lowercase	The Digital UNIX system differentiates between lowercase and uppercase characters. Literal strings that appear in text, examples, syntax descriptions, and function descriptions must be entered exactly as shown.
term	This italic type face is used on the first instance of terms defined in the Glossary or text.
variable	This italic typeface indicates system variables.
user input	This bold typeface is used in interactive examples to indicate input entered by the user.
system output	This typeface is used in code examples and other screen displays. In text, this typeface indicates the exact name of a command, option, partition, path name, directory, or file.
%	The percent sign is the default user prompt.
#	A number sign is the default root user prompt.
Ctrl/X	In procedures, a sequence such as Ctrl/X indicates that you must hold down the key labeled Ctrl while you press another key or a pointing device button.

Chapter 1
Introducing AdvFS

The basic design of traditional UNIX file systems has changed little over the years. Traditional file systems require you to take the system down for backup and other maintenance procedures, reconfigure the entire file system just to add space, and run the `fsck` utility for hours after a system crash. These inflexible file systems cost time and money, and drain limited system-management resources.

Systems that rely on a local file system on a server need flexible, highly available file systems. The Advanced File System (AdvFS) and the optional AdvFS Utilities offer a UNIX file system that provides fast reboots, support for multivolume file systems, and many online maintenance, performance, and reconfiguration options. These features let you quickly respond to the changing requirements of a dynamic computing environment.

Overview of the Advanced File System

AdvFS is a log-based file system with an innovative design that provides flexibility, compatibility, data availability, high performance, and simplified system management. AdvFS takes advantage of the 64-bit computing environment and breaks through the constraints of 2-gigabyte file and file-system limits. It is designed to handle files and filesets as large as almost 16 terabytes.

Flexibility

The AdvFS design overcomes problems associated with system capacity and configuration, such as adding or removing disks. System administrators can add and remove storage without unmounting the file system or halting the operating system. As a result, configuration planning is less complicated and more flexible. System administrators can perform file-system backups and maintenance without interrupting users, applications, or data availability.

Compatibility

AdvFS logical structures, quota controls, and backup capabilities are based on existing file-system design. This consistency promotes a smooth transition to the new file system.

From a user's perspective, AdvFS behaves like any other UNIX file system. End users can use the `mkdir` command to create new directories, the `cd` command to change directories, and the `ls` command to list directory contents. Application programmers encounter a familiar UNIX-based programming interface: `open()`, `close()`, `creat()`, and so on. AdvFS is POSIX 1003.1 compliant.

AdvFS replaces or eliminates several standard commands, such as `newfs`, `dump`, `restore`, and `fsck`. All AdvFS commands and utilities are described in Appendix A on page 127.

Data Availability

With AdvFS, rebooting after a system interruption is extremely fast. AdvFS uses write-ahead logging, instead of the `fsck` utility, as a way to check for and repair file-system inconsistencies that can occur when there is a system crash or power failure. The number of uncommitted records in the log, not the amount of data in the file system, dictates the speed of a recovery. As a result, reboots are quick and predictable, taking seconds rather than minutes. User file data can be logged as well, providing better control of data recovery.

To maintain data availability, system administrators can perform backups, file-system reconfiguration, and file-system tuning without taking the system off line.

End users can retrieve their own unintentionally deleted files from predefined trashcan directories or from clone filesets without assistance from system administrators.

High Performance

An extent-based file allocation scheme allows for fewer and larger I/O transfers, increasing sequential read/write throughput and simplifying large data transfers. The file system performs large reads from disk when sequential access is anticipated. It also performs large writes by combining adjacent data into a single disk transfer.

AdvFS supports multivolume file systems, which enables file-level striping. File-level striping improves file transfer rates by spreading I/O among several disks.

Simplified File-System Management

The AdvFS Utilities product provides a graphical user interface (GUI) to simplify system administration. The AdvFS GUI features menus, graphical displays, and comprehensive online help that make it easy to learn and perform AdvFS operations. In addition, the GUI provides summarized system status information, alerts that bring potential trouble to your attention, and scheduling capabilities.

Advanced File-System Features

AdvFS introduces features that are unavailable in traditional file systems such as the BSD™ or the System V file systems. Table 1 lists the main AdvFS features and their benefits:

Table 1 AdvFS Features and Benefits

Feature	Benefit
Rapid crash recovery	Write-ahead logging eliminates the requirement to use the `fsck` utility when recovering from a system failure. Write-ahead logging makes file-system recovery time rapid and independent of file-system size.
Extended capacity	The design supports large-scale storage systems by extending the size of both files and file systems.
Disk spanning	A file or file system can span multiple disks within a shared storage pool.
Unified buffer cache	This cache interacts with the virtual memory system to dynamically adjust the amount of physical memory being used to cache file data.
Online resizing*	The size of the file system can be dynamically changed by adding or removing disk volumes while the system remains in use. This enables both online storage configuration and online file-system maintenance.
Online defragmentation*	System performance can be improved by defragmenting the data on the disk while the system remains in use. Defragmentation makes file data more contiguous on the storage medium.
File-level striping*	File transfer rates can be improved by distributing file data across multiple disk volumes.

(continued)

Table 1 AdvFS Features and Benefits (cont.)

Feature	Benefit
Online backup*	File-system contents can be backed up to media without interrupting the work flow of system users.
File undelete*	File data availability can be improved by allowing system users to recover deleted files.
Graphical user interface*	The GUI simplifies file-system management by organizing AdvFS functions into menu-selected tasks and by graphically depicting file-system status.

*This feature requires the optional AdvFS Utilities license.

File-System Design

Before setting up AdvFS, you need to understand how it differs from traditional UNIX file systems. These differences, although minor with regard to your transition from another file system, play a role in how you plan and maintain this file system.

The UNIX File System (UFS) is a good example of a traditional file system. Typical of the UFS model, each disk (or disk partition) contains one separate file system; you mount the file system into the logical name space using mount points.

The UFS model is rigid. The directory hierarchy layer of the file system is bound tightly to the physical storage layer. When a file system becomes full, this tight binding makes it impossible to move selected files onto another disk without changing the full path names of those files.

The task of dividing a logical directory into directory subtrees and mapping the subtrees onto separate disks requires careful consideration. Even with extensive planning, adjustments to the directory structure are limited with the UFS model.

Unlike UFS, AdvFS consists of two distinct layers, namely the directory hierarchy layer and the physical storage layer. The directory hierarchy layer implements the file-naming scheme and Digital UNIX file-system compliant functions such as creating and opening files, or reading and writing to files. The physical storage layer implements write-ahead logging, caching, file allocation, and physical disk I/O functions.

This decoupled file-system structure enables you to manage the physical storage layer apart from the directory hierarchy layer. This means that you can move files between a defined group of disk volumes without changing file path names. Because the path names remain the same, the action is completely transparent to end users.

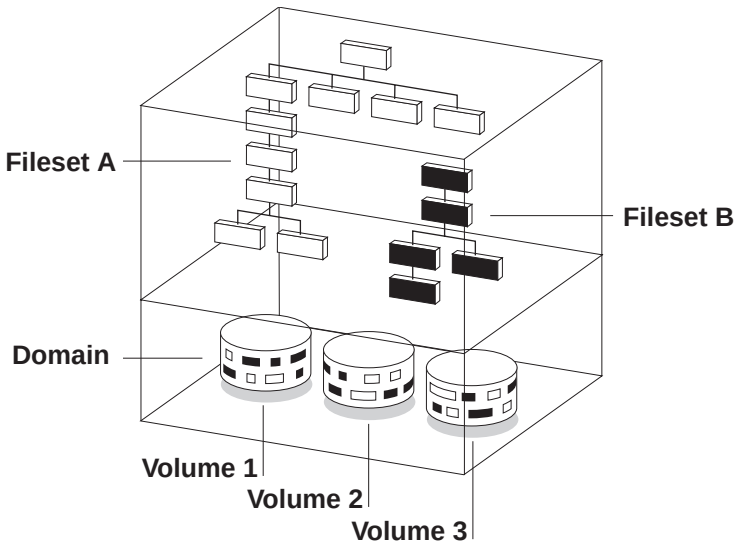
Filesets and File Domains

The two-layer structure is the cornerstone of AdvFS and its design. Supporting the design are two file-system concepts: the fileset and the file domain.

A *fileset* follows the logical structure of a traditional UNIX file system. It is a hierarchy of directory names and file names, and it is what you mount. AdvFS goes beyond UFS, by allowing you to create multiple filesets that share a common pool of storage within a defined set called a file domain.

A *file domain* is a pool of storage to which you can add and remove volumes as storage requirements change. A volume in AdvFS can be a single disk partition, an entire disk, or an aggregate volume provided by a logical volume manager.

Figure 1 AdvFS File System Design



Transaction Log

AdvFS is a log-based file system that employs *write-ahead logging* to ensure the integrity of the file system. Write-ahead logging means that modifications to the metadata (file structure information) are completely written to a transaction log before the actual changes are written to disk. The log is implemented as a circular file buffer so that the contents of the transaction log are written to disk at regular intervals. By committing only completed transactions to disk, the file system is not left in an inconsistent state after an unexpected system crash. By default, only file structures are logged for performance reasons. You can choose to log file data.

AdvFS creates a transaction log for each file domain when the domain is created. The first time the file domain is activated, 4 MB of storage are allocated for the log. It is separate from the user data but can be stored on the same device.

During crash recovery, AdvFS consults the transaction log to confirm file-system transactions. All completed transactions are committed to disk and uncompleted transactions are undone. The number of uncommitted records in the log, not the amount of data in the file system, dictates the speed of recovery. This means that recovery usually takes only a few seconds. Traditional UNIX file systems rely on the `fsck` utility to recover from a system failure. The `fsck` utility can take hours to check and repair a large file system.

Extent-Based Allocation

AdvFS always attempts to write each file to disk as a set of contiguous units called *pages*; a page is 8 kB of disk space. *Contiguous* in this context means storage that is physically adjacent on a disk volume. A set of one or more contiguous pages is called an *extent*. When storage is added to a file in AdvFS, it is grouped in extents as opposed to being allocated a block at a time, as in UFS file systems.

Contiguous placement of the pages means that the I/O mechanism works more efficiently. When a file consists of many small extents, the I/O mechanism must work harder to read or write that file.

File-Storage Allocation

Files are not static; their space requirements change over time. To maintain contiguous file placement without overallocating space on the disk, AdvFS uses a unique file storage allocation scheme. Each time a file is appended, AdvFS adds pages to the file by preallocating one-fourth of the file size, up to 16 pages. If a large write requires more file space, AdvFS attempts to allocate up to 256 contiguous pages. Excess preallocated space is truncated when the file is closed.

For multivolume file domains, new files are allocated sequentially across volumes. Volumes that are more than 86% full (allocated) are not used for new file allocation unless all volumes are more than 86% full. When existing files are appended, storage is allocated on the volume on which the file was initially allocated, until the volume is full.

When a new volume is added to a file domain, it is added to the storage allocation sequence together with the other volumes. Files are allocated to the new volume in turn.

Limitations

Given the dynamic nature of a file system, the file-storage allocation cannot always guarantee contiguous page placement. The following factors affect placement:

- **Excessive disk fragmentation**
  When a disk is fragmented, AdvFS writes data to isolated physical pages, based on availability, instead of writing to contiguous pages.
- **Multiple users**
  When there are many users on a system, file activity increases, which makes files more fragmented on the disk. File fragmentation can reduce the I/O performance of AdvFS. To reduce fragmentation, use the `defragment` utility that is available with the optional AdvFS Utilities product.

Fragments

AdvFS writes files to disk in sets of 8-kB pages. When a file uses only part of the last page (less than 8 kB) a file fragment is created. The fragment, which is from 1 kB to 7 kB in size, is allocated from the *frag file*. The frag file is a special file used for storing the file fragments. It is not visible in the directory hierarchy because it is used only by the system. Using fragments considerably reduces the amount of unused, wasted disk space.

License Registration

The AdvFS and AdvFS Utilities products are separately licensed products. AdvFS is a file-system option on the Digital UNIX operating system. The AdvFS Utilities product provides additional capabilities and is offered as a layered product.

These products have the following separate software-license policies:

- **License Registration for the Advanced File System**
  If you registered the Digital UNIX operating system license product authorization key (PAK), then you can install and use AdvFS base functions. See the *Guide to Installing Digital UNIX* for instructions on installing AdvFS.
- **License Registration for the Advanced File System Utilities**
  Before you can use the file-system utilities, you must register a license product authorization key (PAK) for the Advanced File System Utilities. Contact your Digital representative for additional information.

Chapter 2
Configuring the File System

Configuring AdvFS requires less planning than it does for a traditional file system. With AdvFS you can modify your system configuration at any time without taking down the system. As your system needs change, AdvFS allows you to easily adjust your storage needs up or down to meet your requirements.

To achieve this flexibility, AdvFS implements two new file-system concepts: *filesets* and *file domains*. Filesets and file domains enable a two-layer file-system structure that decouples the physical storage layer of the file system from its directory hierarchy. With this unique architecture, you can manage the logical structure of your file systems independently of the storage volumes that contain them.

File Domains

A file domain is the physical storage layer of the AdvFS file system. It is a defined pool of physical storage that can contain one or more volumes. A volume on AdvFS can be a single disk partition, an entire disk, or an aggregate volume provided by a logical volume manager. Because this storage is managed separately from the directory structure, you can expand and contract the size of the file domain and move files between disks in the file domain without changing file path names.

Creating a File Domain

The first step in setting up an AdvFS file system is creating a file domain. When you create a file domain, you assign an initial volume to the file domain. You can transform a single-volume file domain into a multivolume file domain by adding one or more volumes (with the exception of the root file domain).

A file domain is not a complete file system that you can mount. In order to mount an AdvFS file system, the file domain must contain one or more filesets. You can access files as soon as you mount one or more filesets. See the

Filesets section on page 29 for details.

To create a file domain on the initial volume issue the `mkfdmn` command using the following format, where *volume_name* is the name of a block special device of the initial volume for the domain and *domain_name* is the name of the file domain to be created:

mkfdmn *volume_name domain_name*

Each file domain must have a unique name of up to 31 characters except the slash (/), pound (#), asterisk (*), question mark (?) and space characters. If you want a multivolume file domain, use the `addvol` command to add volumes to the file domain.

Determining the Number of File Domains

The number of file domains you configure on your system depends on your organization's needs. With AdvFS, you can assign all available storage to one file domain or group specific partitions or disks into different file domains.

Establishing multiple file domains allows greater control over your physical resources. You can create file domains to be used by specific projects, groups of users, departments, or any division that makes sense for your organization. For example, you could create file domains for each of your organization's departments, such as engineering, finance, and personnel.

Combining multiple volumes within a single file domain reduces the overall management effort because fewer file domains require less administration. However, a single volume failure within a file domain renders the entire file domain inaccessible. Thus, the more volumes that you have in your file domain the greater the risk that a file domain will fail.

Planning and Configuring File Domains

The following are guidelines to follow when planning and configuring your file domains:

- A *volume* is anything that behaves like a UNIX block device. You can avoid I/O scheduling contention and enhance system performance by dedicating the entire disk (generally partition `c`) to a file domain.
- You can have 100 active file domains per system. A file domain is active when at least one of its filesets is mounted.
- The number of file domains per system depends on the needs of your site and the resources available to the system. Combining multiple volumes within a single file domain simplifies file system management.

- There is one transaction log per file domain that is shared by all filesets in the file domain. When there are many filesets with a large amount of I/O, the transaction log can become a bottleneck because all transactions are completely written to the transaction log before being written to disk (see the Transaction Log section on page 19). Balance the management gains of having multiple filesets in a file domain against the potential performance reduction caused by having all of the log data for all filesets going to one transaction log.
- Although the risk of media failure is slight, a single volume failure within a file domain renders the entire file domain inaccessible. In the case of an unrecoverable media failure, you must recreate the file domain and restore all the files. Thus, the more volumes that you add to your file domain, the greater the risk that a file domain will fail.
- To reduce the risk of file-domain failure, limit the number of volumes per file domain to three or use mirrored volumes created with the Logical Storage Manager (LSM).

To maintain high performance, avoid splitting a disk between two file domains. For example, do not add partition `g` to one file domain and partition `h` of the same disk to another file domain.

The /etc/fdmns Directory

The `/etc/fdmns` directory defines the file domains by providing a subdirectory for each file domain you create on your system. The subdirectories contain a symbolic link to every volume in the file domain. This directory is created and maintained automatically by AdvFS.

Back up the `/etc/fdmns` directory regularly. If the contents of the directory become corrupted or deleted, restore the directory from your most recent backup. In addition, AdvFS provides the `advscan` command that finds the location of AdvFS file domains on volumes or LSM disk groups and can reconstruct the `/etc/fdmns` directory. See the Restoring Data section on page 90 for details on restoring.

Displaying File-Domain Information

If a file domain is active–that is, if at least one fileset is mounted, you can display detailed information about the file domain and the volumes included in it.

Table 2 defines the information displayed by the `showfdmn` command.

Table 2 Definitions of showfdmn Information

Field Title	Definition
`Id`	A unique number, in hexadecimal format, that identifies a file domain.
`Date Created`	The day, month, and time that a file domain was created.
`LogPgs`	The number of 8-kilobyte pages in the transaction log of the specified file domain.
`Domain Name`	The name of the file domain.
`Vol`	The volume number within the file domain. An “L” next to the number indicates that the volume contains the transaction log.
`512-Blks`	The size of the volume in 512-byte blocks.
`1K-Blks`	The size of the volume in 1-kilobyte blocks.
`Free`	The number of blocks in a volume that are available for use.
`% Used`	The percent of volume space that is currently allocated to files or metadata.
`Cmode`	The I/O consolidation mode. The default mode is on.
`Rblks`	The maximum number of 512-byte blocks read from the volume at one time.
`Wblks`	The maximum number of 512-byte blocks written to the volume at one time.
`Vol Name`	The name of the special device file for the volume.

Table 3 defines the additional information included in the `showfdmn` command display for multivolume file domains.

Table 3 Definitions of showfdmn Information for Multivolume Domains

Totals	Definition
`Total Volume Size`	The total size of all volumes in the file domain.
`Total Number of Free Blocks`	The total number of free blocks on all of the volumes in the file domain.
`Total Percent of Volume Space`	The percent of total volume space currently allocated in the file domain.

The following example displays file-domain information for the `domain_1` file domain:

```
# showfdmn domain_1
               Id       Date Created       LogPgs   Domain Name
2bb0c594.00008570   Wed Mar 24 12:33 1995     512      domain_1

Vol   512-Blks     Free  % Used   Cmode  Rblks Wblks    Vol Name
1L      832527    79210     90%      on    128   128   /dev/rz1c
 2      832527     1684     98%      on    128   128   /dev/rz2c

    --------------------------

       1665054    80894     94%
```

Renaming a File Domain

An existing file domain can be assigned a new name. File domains are known to the system by their file-domain identifier, which is a set of numbers that identify the file domain. The file-domain name is an attribute that you assign. When you rename a file domain, the file-domain identifier is not changed.

To rename a file domain, follow these steps:

1. Unmount all the filesets and any related clones.
2. In the `/etc/fdmns` directory, change the old file-domain name to the new one, using the following `mv` command format:

mv /etc/fdmns/*old_domain_name* **/etc/fdmns/***new_domain_name*

3. Edit the `/etc/fstab` file to enter the new domain name and remove the old.
4. Mount the filesets in the renamed file domain.

For example, to rename the `mkting_dmn` file domain as the `advting_dmn` file domain, assuming one fileset, `adv_fs`, is mounted at `/adv_fs`, issue the following commands:

```
# umount mkting_dmn#adv_fs
# mv /etc/fdmns/mkting_dmn /etc/fdmns/advting_dmn
# vi /etc/fstab
```
Change the line: `mkting_dmn#adv_fs` to: `advting_dmn#adv_fs`
```
# mount /adv_fs
```

Removing a File Domain

You can remove a file domain after all filesets in the file domain are unmounted. Removing a file domain destroys the file domain and its filesets. The directory entry in `/etc/fdmns` that defined the file domain is removed and you cannot mount the filesets. Volumes that were assigned to the removed file domains are relabeled as unused and can be reused.

To remove a file domain, unmount all filesets and clone filesets. Then, issue the `rmfdmn` command using the following format, where *domain_name* is the name of the file domain to be removed:

rmfdmn *domain_name*

You will then be prompted to verify the removal. Responding `yes` will complete the removal. A confirmation message will be displayed when the procedure is complete:

`rmfdmn: remove domain` *domain_name*`? [y/n]`**y**

`rmfdmn: domain` *domain_name* `removed`

If you attempt this command when there are mounted filesets, the system displays an error message. AdvFS will not remove an active file domain.

Caution After removing a file domain, you cannot recover any file data from any fileset in the removed file domain.

Filesets

A fileset represents a mountable portion of the directory hierarchy of a file system. Each is a collection of directories and files that form a subtree structure.

Filesets and traditional UNIX file systems are equivalent in many ways. Like traditional file systems, you mount AdvFS filesets. Similarly, filesets are units on which you enable quotas and are units for backing up data.

In contrast with traditional file systems, the directory hierarchy of AdvFS is independent of the storage. Therefore, you can change file placement without affecting the logical structure of the filesets.

AdvFS filesets also support a new feature: *clone filesets*. A clone fileset is a read-only copy of an existing fileset that you create to capture your data at one instant in time. You can back up the contents of the clone fileset while the original fileset remains available to users. See the Cloning for Online Backup section on page 87 for additional information on cloning.

Creating a Fileset

A file domain must contain at least one fileset to be mounted. The single fileset can consume all of the storage available in the file domain. You can also create multiple filesets within a file domain that share the storage pool established for the file domain.

Each fileset can be mounted and unmounted independently of the other filesets in the file domain. You can limit fileset growth within a file domain by assigning fileset quotas.

To create a fileset in a file domain, issue the `mkfset` command using the following format, where *fileset_name* is the name of the fileset to be created:

mkfset *domain_name fileset_name*

Each fileset in the file domain must have a unique name of up to 31 characters. All white space characters and the / # : * ? characters are invalid for fileset names. The same fileset name can be repeated in different file domains but every fileset must have a unique mount point.

Determining the Number of Filesets

The number of filesets you configure on your system depends on your organization's needs. To share storage, you can create multiple filesets in one file domain, but manage the filesets separately. Or, you can set up AdvFS in a standard UNIX file system configuration by creating one fileset per file domain.

Remember that filesets are managed independently of their physical storage. Each fileset can be backed up separately and at different times, or can be assigned quota limits. Consequently, you can group files by their management requirements. For example, you could create a fileset for developer files that will be backed up twice a day and you could create another fileset with quotas imposed to limit the amount of disk space available to the marketing department.

Planning and Configuring Filesets

The following are guidelines to follow when planning and configuring AdvFS filesets:

- You can create an unlimited number of filesets per system; however, the number of mounted filesets is limited to a maximum of 512 minus the number of active file domains. For example, if you have three active file domains, you can mount up to 509 filesets.
- Consider the space limitations that you want to impose. Fileset quotas can be imposed to limit the amount of space a specific fileset can consume. User and group quotas can be imposed in addition to or instead of fileset quotas.
- Consider backup requirements. Files that require different backup schedules can be placed in different filesets. For example, put database tables, which require backup more frequently, in different filesets from the database "engine."
- Consider the availability requirements for data within a fileset. Place data with different requirements in different filesets.

Mounting a Fileset

As with traditional UNIX file systems, AdvFS filesets must be mounted in order to access them. To mount a fileset, use the `mount` command. Identify the file domain, fileset, and mount point using the following format where *mnt_point* is the path to the mount point:

mount *domain_name#fileset_name mnt_point*

To unmount AdvFS filesets, use the standard `umount` command with the following format:

umount *mnt_point*

The /etc/fstab File

AdvFS filesets are added to the `/etc/fstab` file in the same manner that you add any other file system. AdvFS filesets listed in the `/etc/fstab` file are mounted each time you reboot the system.

The fileset entry includes the file domain name, fileset name, mount point, file system type and the mount point options. The user quota and group quota options should be included along with the pass field numbers if quotas are used. An AdvFS `/etc/fstab` entry with user and group quotas enforced should include:

advfs rw,userquota,groupquota 0 2

For example, to automatically mount the `credit_fs` fileset with user and group quotas, add the following line to your `/etc/fstab` file:

```
account_domain#credit_fs /mnt/credit advfs rw,userquota,groupquota 0 2
```

Displaying Fileset Information

Any system user can display detailed information about all filesets and clones in a specified file domain. Root user privilege is required only if the file domain is inactive (filesets unmounted).

Table 4 defines the information displayed by the `showfsets` command.

Table 4 Definitions of showfsets Command Information

Field Title	Definition
`Id`	A fileset identifier is a combination of the file domain identifier and an additional set of numbers that identify the fileset within the file domain.
`Clone Status`	The status of a clone fileset varies, depending on the fileset. Status items can include: `Clone is,` which specifies the name of a clone fileset; `Clone of,` which specifies the name of the parent fileset; or `Revision,` which specifies the number of times you revised a clone fileset.
`Files`	Specifies the number of files in the fileset and the current file usage limit (quota).
`Slim`	The soft limit of the fileset quota. This limit can be exceeded for a period of time specified by the grace period.
`Hlim`	The hard limit of the fileset quota. This limit cannot be exceeded.
`Blocks`	The number of blocks that are in use by a mounted fileset and the current block usage limit (quota). For filesets that are not mounted, zero blocks will display. For an accurate display, the fileset must be mounted.
`Quota Status`	Specifies which quota types (`user`, `group`, or `fileset`) are enabled.

The following is an example of the `showfsets` command display of `big_domain`, which has four filesets:

```
# showfsets big_domain
staff1_fs
       Id           : 2cb9d009.000419f4.1.8001
       Files        :    18554,     SLim=     0,     HLim=    0
       Blocks (512) :   712230,     SLim=     0,     HLim=    0
       Quota Status : user=on  group=on
guest_fs
       Id           : 2cb9d009.000419f4.2.8001
       Files        :     4765,     SLim=     0,     HLim=    0
       Blocks (512) :   388698,     SLim=     0,     HLim=    0
       Quota Status : user=on  group=on
```

```
staff2_fs

       Id           : 2cb9d009.000419f4.1.8001
       Files        :    12987,      SLim=      0,      HLim=    0
       Blocks (512) :   842862,      SLim=      0,      HLim=    0
       Quota Status : user=on  group=on

staff3_fs

       Id           : 2cb9d009.000419f4.3.8001
       Files        :    48202,      SLim=      0,      HLim=    0
       Blocks (512) :  1341436,      SLim=      0,      HLim=    0
       Quota Status : user=on  group=on
```

The following is a `showfsets` command display of `domain_2`, which contains one fileset and one clone fileset:

```
# showfsets domain_2

test_fs

       Id           : 3003f44f.0008ac95.4.8001
       Clone is     : clone_test
       Files        :     7456,      SLim=      0,      HLim=    0
       Blocks (512) :   388698,      SLim=      0,      HLim=    0
       Quota Status : user=on  group=on

Clone_test

       Id           : 3003f44f.0008ac95.5.8001
       Clone of     : test_fs
       Revision     : 2
```

Renaming a Fileset

An existing fileset can be assigned a new name. Filesets are actually known to the system by their fileset identifier, which is a combination of the file-domain identifier and an additional set of numbers that identify the fileset within the file domain. The fileset name is only a fileset attribute that you assign. When you rename a fileset, the fileset identifier is not changed.

Before renaming a fileset, unmount the fileset. You can then rename a fileset by issuing the `renamefset` command using the following format:

renamefset *domain_name old_fileset_name new_fileset_name*

For example, to rename the `mkting` fileset to the `advting` fileset, issue the following command:

```
# renamefset domain_1 mkting_fs advting_fs
```

After renaming a fileset, update the corresponding entries in the `/etc/fstab` file to automatically mount the renamed fileset.

Note Clones of the fileset are not renamed when you rename the fileset.

Removing a Fileset

Filesets can be deleted from a file domain when they are no longer needed. Once filesets are removed, they cannot be remounted. All files in the fileset are destroyed and the data cannot be accessed.

Filesets must be unmounted before they can be removed and cannot have any existing clones. You can then remove a fileset by issuing the `rmfset` command using the following format:

rmfset *domain_name fileset_name*

For example, to remove the `tmp_1` fileset in `domain_1`, issue the following commands:

```
# rmfset domain_1 tmp_1
rmfset: remove fileset fileset_name? [Y/N]y
```

Note If you are removing an entire file domain, you do not need to remove the filesets first. Removing the file domain will destroy all filesets in the process.

Cloning a Fileset

A clone fileset is a read-only snapshot of fileset data structures (metadata). When you clone a fileset, the utility copies only the structure of the original fileset, not the actual data. When a file is modified or removed, the file system copies the original, unchanged data to the clone fileset. Pages that are not modified are not copied to the clone. (This concept is called *copy on write*.) Because the only data contained in the clone fileset is a copy of original data that was modified, the clone fileset usually consumes less disk space than the original fileset.

Clone filesets are useful for the following:

- To perform online backups

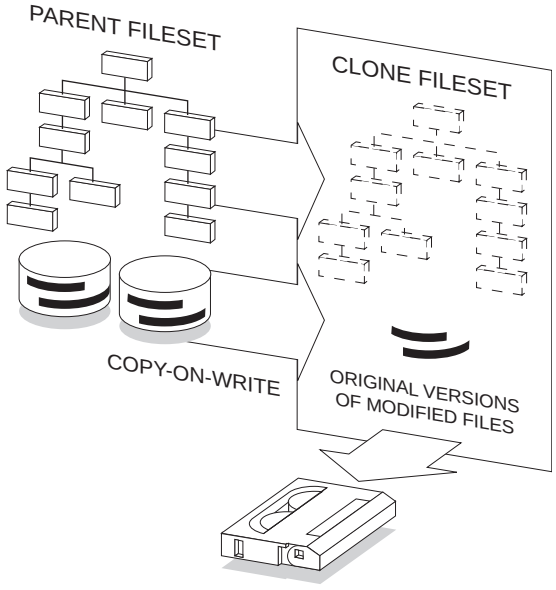
  With traditional UNIX file systems, a standard backup procedure reduces data availability because you must unmount each file system that you intend to back up. In contrast, an AdvFS online backup procedure provides uninterrupted access to your data during the backup. You can perform online backups by cloning filesets and then backing up the clones with the `vdump` command or other supported

backup options. See the Cloning for Online Backup section on page 87 for details on cloning.

- To protect against accidental file deletion or corruption

  You can create a clone fileset daily of each fileset that you plan to access or modify. By leaving the clone fileset on line, you can replace unintentionally deleted or corrupted files without loading backup tapes.

Figure 2 Cloning a Fileset



Creating a Clone Fileset

To create a clone fileset, issue the `clonefset` command using the following format, where the *clone_name* is the name of the clone to be created:

clonefset *domain_name fileset_name clone_name*

Each clone must have a unique name of up to 31 characters. All white space characters are invalid, as are the slash (/), pound (#), colon (:), asterisk (*), and question mark (?). The same clone name can be repeated in different file domains, but every fileset must have a unique mount point.

Mounting a Clone

AdvFS fileset clones must be mounted in order to access them. To mount a clone, use the `mount` command. Identify the clone and mount point using the following format:

mount *domain_name#clone_name mnt_point*

To unmount a clone, use the standard `umount` command using the following format:

umount *mnt_point*

Renaming a Clone

Clone filesets cannot be renamed. To assign a different name to a clone, delete the old clone and create a new clone for the fileset. (Note that this new clone is a snapshot of the original fileset at a later point in time than the deleted clone.)

Removing a Clone

Clones can be deleted from a file domain when they are no longer needed. Once clones are removed, they cannot be remounted. All files in the clone are destroyed and the data cannot be accessed.

Clones must be unmounted before they can be removed. You can then remove a clone by issuing the `rmfset` command using the following format:

rmfset *domain_name clone_name*

For example, to remove the `tmp_clone` clone in `domain_1`, issue the following command:

```
# rmfset domain_1 tmp_clone
rmfset: remove fileset tmp_clone? [Y/N]y
```

Volumes

In AdvFS, a volume is any mechanism that behaves like a UNIX block device: an entire disk, a disk partition, or an aggregate volume provided by a logical storage manager. When initially created, all file domains consist of a single volume. If you have the optional AdvFS Utilities, you can transform a single-volume file domain into a multivolume file domain by adding one or more volumes.

A volume can be assigned to only one file domain. It is associated with its file domain by the *file domain ID* stored in the file domain attributes table of the volume. Each volume in a file domain is assigned a volume index number, starting with 1, when it is initialized. Numbers are reused when volumes are removed and new ones added. When

a volume is removed from a file domain, the file domain ID is cleared in the file domain attributes table.

For AdvFS to function properly, the number of volumes in a file domain with the same file domain ID must remain consistent with the number of volumes identified in the file domain attributes table. Also, the number of links to the volumes in the `/etc/fdmns` file must equal the number of volumes identified in the domain attributes table.

Inconsistencies in these numbers can occur in a number of ways and for a number of reasons. The `advscan` command can be used to attempt to correct these inconsistencies for a file domain if you set the `-f` flag. If you attempt to mount a fileset from an inconsistent file domain, a message similar to the following will be displayed:

```
Volume count mismatch for file domain domains.
domain expects 2 volumes, /etc/fdmns/domain has 1 links.
```

Volume Structure

Each volume in an AdvFS file domain contains the following structures:

- A *bitfile metadata table* (BMT), which is used to store file data structure (metadata), including file attributes, file extent maps, fileset attributes, and the POSIX file statistics.
- A *storage bitmap*, which is used to track free and allocated disk space.
- A *miscellaneous metadata bitfile*, which maps areas of the volume that do not represent AdvFS metadata, such as the disk label and boot blocks.

In addition to these structures, each file domain also has the following structures on one volume in the file domain:

- A *transaction log*, which stores all metadata changes until they are written to disk.
- A *root tag directory*, which defines the location of all filesets in the file domain.

This information is provided only to show how the volume is structured for AdvFS. You cannot change the way AdvFS configures the volume.

Volume Attributes

AdvFS volumes are configured with attributes that determine how data is read, cached, written, and consolidated. When an AdvFS volume is incorporated into a file domain, either by creating the initial file domain or by adding a volume, the default volume attributes are set as follows:

- I/O transfer parameter for both reads and writes is 128 blocks (64kB).

- I/O consolidation mode is on.
- The number of dirty, 512-byte blocks that the file system will cache in memory (per volume in the file domain) is 768 blocks. (*Dirty* means that the data has been written by the application, but the file system has cached it in memory so it has not yet been written to disk.)

You can display or modify the current volume attributes by issuing the `chvol` command using the following format:

chvol *[options] device_name file_domain*

Modifying these default attributes can improve performance in some system configurations. Possible modifications include the following:

- Increasing the transfer parameters, within limits, provides greater disk throughput and leads to a higher performance system. If the disk is fragmented so that the pages of a file are not allocated sequentially, there is no advantage to increasing the I/O size. You can reduce fragmentation by using the `defragment` utility.
- Consolidating a number of I/O transfers into a single, large I/O can improve file-system performance.
- For file-system workloads that are heavily biased toward random writes, increasing the file-system dirty block threshold can improve file write performance.

Adding Volumes

You can create a multivolume file domain to increase the disk capacity of an existing file domain and to perform preventative disk maintenance. Adding volumes to a file domain does not affect the logical structure of the filesets within the file domain. You can add volumes to a file domain without reconfiguring the directory hierarchy layer of the file system.

A newly created file domain consists of one volume, which can be a disk, disk partition, or logical volume. The `addvol` utility enables you to increase the number of volumes within an existing file domain. Use the following format to add a volume to a file domain:

addvol *device_name domain_name*

You can add volumes immediately after creating a file domain, or you can wait until the file domain requires additional space. You can add a volume to an active file domain while its filesets are mounted and in use.

Adding one partition (typically, partition `c`) to a file domain is preferable to adding several partitions (such as `a`, `b`, `g`, `h`). Existing data on the volume you add is

destroyed during the `addvol` procedure. Do not add a volume containing data that you want to keep. While up to 250 volumes per file domain are allowed, limiting the number of volumes to three decreases the risk of disk errors that can cause the entire file domain to become inaccessible.

Overlapping Partitions on Mounted File Systems

You cannot add a volume that would overlap a mounted file system. If you try to add a volume that would cause partitions to overlap with any other volume that is mounted for another file system, a swap area, or a reserved partition (by a database, for example), the system displays an error message and does not permit the `addvol` procedure to complete. The following example shows the error message:

```
# addvol /dev/rz3b big_domain

New filesystem would overlap mounted filesystem(s),
        swap area or reserved partition(s)
Unmount/relabeling required before creating /dev/rrz3b (start 131072 end 393215)
        /dev/rz3b in use as swap (start 131072 end 393215)
addvol: Can't add volume '/dev/rz3b' to file domain 'big_domain'
```

Overlapping Partitions on Unmounted File Systems

You can add a volume with partitions that overlap with an unmounted partition that has a disk label for a file system, a swap area, or a reserved partition. If you attempt to add a volume that would overlap, AdvFS prompts you to confirm the action. If you confirm, the disk label will be overwritten and the new volume added to the AdvFS file domain. The following example shows the confirmation message:

```
# addvol /dev/rz2c domain_1

Partition(s) which overlap /dev/rz2c are marked in use.
If you continue with the operation you can possibly destroy existing data.
CONTINUE? [y/n]y
```

If you want to overwrite partitions that already have a disk label but do not want to be prompted to continue each time, use the `addvol -F` command. Using the `-F` flag allows the volume to be added and overwrites any partition that has a disk label but is not mounted. (AdvFS will not overwrite mounted file systems.)

Removing a Volume

When there is sufficient free space on the remaining volumes, you can remove volumes from a file domain at any time without interrupting users or affecting the logical structure of the filesets in the file domain. When you use the `rmvol` utility, the file system automatically migrates the contents of the selected volume to other volumes in

the file domain. Before you can remove a volume from a file domain, all filesets in the file domain must be mounted. An error will occur if you try to remove a volume from a file domain with unmounted filesets.

To remove a volume, issue the `rmvol` command using the following format:

rmvol *device_name domain_name*

If there is not enough free space on other volumes in the file domain to accept the files offloaded from the departing volume, as many files as possible are moved to available free space on other volumes. Then a message is sent indicating that there is insufficient space. The file domain is not damaged.

You can interrupt the `rmvol` process without damaging your file domain. AdvFS will stop removing files from the volume. Files already removed from the volume will remain in their new location.

Note If a volume does not allow writes after an aborted `rmvol` operation, use the `chvol` command with the `-A` flag to reactivate the volume.

If you remove a volume that contains a stripe segment, the `rmvol` utility moves the segment to another volume that does not already contain a stripe segment of the same file. When a file is striped across all volumes in the file domain, a confirmation is required before removing the volume. If you allow the removal process to continue, more than one stripe segment will be placed on the remaining volumes. See the Creating Striped Files section on page 54 for details on file striping.

Locating AdvFS Partitions

The `advscan` command locates parts of AdvFS file domains on disk partitions and in LSM disk groups. This command is useful:

- When disks have moved to a new system, device numbers have changed, or you have lost track of a file domain location
- For repair, if you delete the /`etc/fdmns` directory, delete a file domain from the `/etc/fdmns` directory, or delete links from a file domain in the `/etc/fdmns` directory

The `advscan` command accepts a list of devices or disk groups and searches all partitions and volumes in each. To determine if a partition is part of an AdvFS file domain, the `advscan` command performs the following functions:

- Reads the first two pages of a partition to determine if it is an AdvFS partition and to find the file-domain information.

- Reads the disk label to sort out overlapping partitions. The sizes of the overlapping partitions are examined and compared to the file-domain information to determine which partitions are in the file domain. These partitions are reported in the output.
- Reads the boot block to determine if the partition is bootable from the AdvFS root.

The a`dvscan` command displays the date the file domain was created, the on-disk structure version, and the last known or current state of the volume.

Options for the `advscan` command allow you to recreate missing file domains from the `/etc/fdmns` directory, missing links, or the entire `/etc/fdmns` directory. Or you can rebuild the `/etc/fdmns` directory manually by supplying the names of the partitions in each file domain. See the `advscan`(8) reference page for details on all options.

To locate AdvFS partitions, issue the `advscan` command using the following format:

advscan *[options] disks*

In the following example, the command scans devices `rz3` and LSM disk group `rootdg` for AdvFS partitions:

```
# advscan rz3 rootdg
Scanning devices /dev/rrz3 /dev/rvol/rootdg
Found domains:
usr_domain
      domain Id   2e09be37.0002eb40
      Created           Thu Jun 23 09:54:15 1994


      domain volumes          2
      /etc/fdmns links        2


      Actual partitions found:
                        rz3g
                        rootdg.vol03
```

Moving Files to Different Devices

Two methods are available for moving files from one device to another. The method you use depends on whether you have the optional AdvFS Utilities product.

If you do not have the optional AdvFS Utilities product, use the following steps to move files to a new file domain created on a different device. Because files that are being modified at the time of the copy may not copy correctly, consider mounting the filesets to be moved as read-only or keeping users from accessing the target filesets before completing the steps.

1. Make a new file domain on the new device.
2. Create filesets in the new file domain for each of the old filesets you will move to the new file domain. Mount the filesets on temporary mount points.
3. Copy all of the files from the old mounted filesets to the new mounted filesets. Unmount the old filesets.
4. Mount the new filesets using the mount points of the old filesets. The directory tree will then be unchanged.
5. Rename the filesets and file domain to be the same as the old ones.

Alternatively, if you have the optional AdvFS Utilities product, use the following steps to move the filesets to a different device. This can be accomplished without interrupting the users and without concern that files are being added, deleted, or modified while the move is occurring.

1. Use the `addvol` command to add a new volume to the existing file domain.
2. Use the `rmvol` command to remove the volume you want to take out of the file domain. All files will be automatically migrated from the old volume and relocated on the remaining volumes.
3. Run the `balance` command to even the distribution of the files among the new and remaining volumes in the file domain. See the Balance Utility section on page 98 for details on balancing file domains.

Setting Up AdvFS

As you begin configuration planning, decide whether you want to set up AdvFS to resemble a traditional UFS configuration. If you have experience with the UFS model, you may want to use that model initially. Once you become more familiar with AdvFS, you can begin to move away from the traditional model.

When planning your configuration, consider setting up the root and `/usr` file systems on AdvFS. Using AdvFS as the root file system enables booting from an AdvFS file domain. By having the `/usr` file system on AdvFS you can significantly reduce the amount of time your system is down after a system failure.

The minimum configuration needed for an active AdvFS file system is one file domain and one mounted fileset.

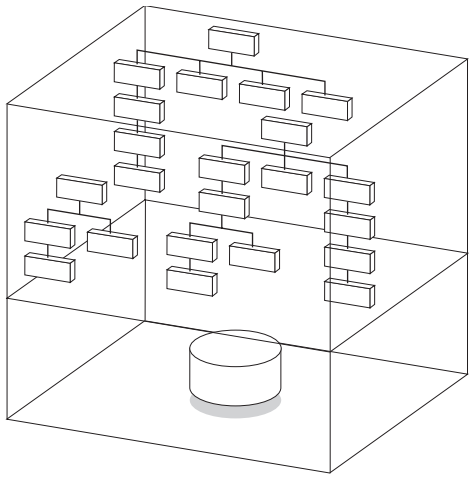
Note The AdvFS root file domain is currently limited to one volume.

Setting Up a Single-Volume File System

The following procedure describes how to set up an active, single-volume file system. Creating a single-volume file domain with a single fileset is equivalent to creating a traditional UFS file system. With AdvFS, you can modify the file system size whenever the storage requirements change.

1. Create a single-volume file domain by using the `mkfdmn` command. Then, you can add more volumes to an existing file domain by using the `addvol` utility.
2. Create one or more filesets by using the `mkfset` command and name each fileset the same as its mount-point directory; for example, if the mount-point directory is `/tmp`, name the fileset `tmp`. (Naming the fileset with the same name as the mount point is recommended but is not required.)
3. Create the mount-point directory by using the `mkdir` command.
4. Mount each fileset by using the `mount` command.

Figure 3 Single-Volume File Domain



The following examples set up active, single-volume file systems with different configurations using AdvFS.

The following example creates a file system with volume `/dev/rz3c` and directory `/staff` available for use with AdvFS. First, the `domain_1` file domain is created. Then, the `staff` fileset is created in `domain_1`. Finally, the `staff` fileset is mounted at the `/staff` mount point.

```
# mkfdmn /dev/rz3c domain_1
# mkfset domain_1 staff_fs
```

```
# mkdir /staff
# mount domain_1#staff_fs /staff
```

The following example creates a single-volume file domain, `domain_2`, and two filesets, `tmp` and `public`, in the file domain. Because the file domain has only one volume, the files in both filesets physically reside on one volume. This is allowed in AdvFS file systems. First, the `domain_2` file domain is created. Then, the `tmp` and `public` filesets are created in `domain_2`. Last, the `tmp` and `public` filesets are mounted on their mount points.

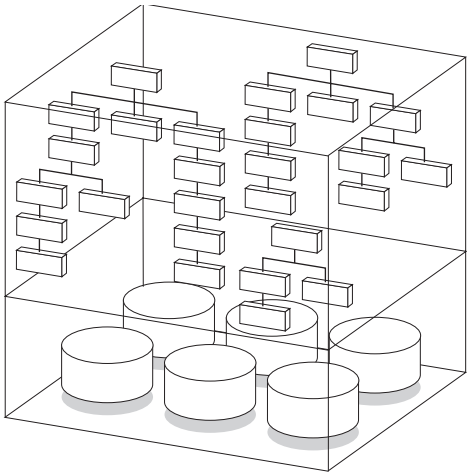
```
# mkfdmn /dev/rz2c domain_2
# mkfset domain_2 tmp_fs
# mkfset domain_2 public_fs
# mkdir /public
# mount domain_2#tmp_fs /tmp
# mount domain_2#public_fs /public
```

Setting Up a Multivolume File Domain

You can set up a multivolume file domain by setting up a single-volume file domain and immediately adding additional volumes. The following procedure describes how to set up an active, multivolume file system:

1. Create a single-volume file domain by using the `mkfdmn` command.
2. Add one or more volumes using the `addvol` command.
3. Create one or more filesets by using the `mkfset` command and name each fileset the same as its mount-point directory; for example, if the mount-point directory is `/tmp`, name the fileset `tmp`.
4. Create the mount-point directory by using the `mkdir` command.
5. Mount each fileset by using the `mount` command.

Figure 4 Multivolume File Domain



The following example creates a multivolume file domain, `domain_1`, with volumes `/dev/rz1c` and `/dev/rz2c`, and then creates and mounts the `public_fs` fileset:

```
# mkfdmn /dev/rz1c domain_1
# addvol /dev/rz2c domain_1
# mkfset domain_1 public_fs
# mkdir /public
# mount domain_1#public_fs /public
```

The following example creates a multivolume file domain using two volumes created with LSM. First the file domain, `domain_2`, is created with two LSM volumes, `/dev/vol/vol01` and `/dev/vol/vol02`, and then the `develop_fs` fileset is created and mounted:

```
# mkfdmn /dev/vol/vol01 domain_2
# addvol /dev/vol/vol02 domain_2
# mkfset domain_2 develop_fs
# mkdir /develop
# mount domain_2#develop_fs /develop
```

You can enlarge a file domain by adding an LSM volume. You first create the volume using LSM, then add the volume to the file domain using the `addvol` command.

The following example creates a 300-MB, mirrored LSM volume and adds that to the `usr_domain` file domain:

```
# volassist -U fsgen make v3_mirr 300m nmirror=2
# addvol /dev/vol/rootdg/v3_mirr usr_domain
```

You can remove an LSM volume from a file domain using the `rmvol` command. You must specify the complete path name of the LSM volume.

The following example removes the `v3_mirr` volume from the `usr_domain` file domain:

```
# rmvol /dev/vol/rootdg/v3_mirr usr_domain
```

Caution Do not use the shrink or grow LSM options with AdvFS file domains. Use the `addvol` and `rmvol` commands to enlarge or reduce the size of file domains.

Setting Up a File Domain with LSM Volumes

You can use volumes created with the Logical Storage Manager (LSM) when creating or adding to an AdvFS file domain. You first create the volume using LSM. Then, create the file domain using the LSM volume, or add the LSM volume to an existing file domain. See the LSM manual for details on creating LSM volumes.

The following example creates a 500-MB mirrored LSM volume, creates a file domain using the LSM volume, creates a new fileset, and then mounts the fileset:

```
# volassist -U fsgen make v2_mirr 500m nmirror=2
# mkfdmn /dev/vol/rootdg/v2_mirr domain1
# mkfset domain1 fset_1
# mount domain1#fset_1 /mnt9
```

Creating a Domain for a Large Number of Files

File systems that contain a very large numbers of files (over 50,000), such as Usenet news servers, require a change in the file-domain configuration. For these file systems, the standard AdvFS metadata extent size allocation may be inadequate, resulting in erroneous “out of disk space” error messages. By altering the extent size allocation when creating the file domain, you can accommodate the large number of files.

In AdvFS, the fileset data structure (metadata) is stored in a file called the bitfile metadata table (BMT). The BMT is equivalent to the UFS inode table except that the UFS inode table is statically allocated. To save space, AdvFS grows the BMT by extents of 128 pages each time additional metadata extents are needed. Handling many small files causes the system to request metadata extents more frequently, causing the metadata to become fragmented. Because the number of extents that describe the BMT is a fixed size resource, the file domain will appear to be out of disk space when in reality the BMT cannot be extended to map new files.

You can reduce the amount of metadata fragmentation in one of two ways: by increasing the number of pages the system attempts to grow the metadata table each time more space is needed or by preallocating all of the space for the metadata table when the file domain is created.

To preallocate all of the metadata table space you expect the file domain to need, use the `mkfdmn` command with the `-p` flag set to specify the number of pages to preallocate. Space that is preallocated for the metadata table cannot be deallocated so do not preallocate more space than you need for the metadata table. Table 5 provides suggested metadata table page estimates for numbers of files. Choose the number of pages from the Metadata Table Size column that corresponds to the number of files you expect the file domain to handle. Then create the file domain using the following format:

mkfdmn -p *number_pages device domain_name*

To configure the file domain to grow by more than 128 pages each time additional metadata extents are needed, create the file domain using the `mkfdmn` command along with the `-x` flag. Specify a number of metadata pages greater than 128. Table 5 provides suggested extent size estimates for numbers of files. Choose the number of pages from the Suggested Extent Size column that corresponds to the number of files you expect the file domain to handle. Then create the file domain using the following format:

mkfdmn -x *number_pages device domain_name*

You can set the extent size to any value, but Table 5 suggests extent sizes based on the number of files you anticipate and shows the resulting size of the metadata table. Table 5 provides *guidelines* for up to 800,000 files.

Table 5 Suggested Extent Sizes

Number of Files	Suggested Extent Size (in Pages)	Metadata Table Size (in Pages)
Less than 50,000	default (128)	3,600
100,000	256	7,200
200,000	512	14,400
300,000	768	21,600
400,000	1024	28,800
800,000	2048	57,600

Note To get the maximum benefit from increasing the number of metadata table extent pages, use the same number of pages when adding a volume with the

`addvol` command as assigned when the file domain was created with the `mkfdmn` command.

Configuring an AdvFS Root File System

There are several advantages to configuring the root file system on AdvFS. You can:

- Restart fast after a crash. You do not run the `fsck` utility after a crash.
- Use one set of tools to manage all local file systems. All features of AdvFS except `addvol` and `rmvol` are available to manage the root file system.
- Use AdvFS with LSM to mirror the root file system. This allows your root file system to survive media failures.

The following restrictions on the AdvFS root file systems are currently enforced:

- The root file domain can only contain one volume. You cannot add volumes to the root file domain.
- The volume must start from the beginning of the physical device (`a` or `c` partitions).
- The root fileset must be the first fileset created in the root file domain.
- You can assign any name to the root file domain and fileset but the same name must be entered in the `/etc/fstab` file.

You can put the root file system on AdvFS during the initial base-system installation or you can convert your existing root file system after installation. Note that when you install AdvFS as root during the initial installation, root will default to the `a` partition.

If you construct your own root file system, you can configure it on the `a` or `c` partition. See the *Installation Guide for Digital UNIX* for instructions on installing AdvFS as the root file system during the initial installation. See the Converting the Root File System section on page 137 for instructions on converting an existing UFS root file system to AdvFS.

Mounting the Root File System in Single-User Mode

The root file system is automatically mounted as read-only when the system is booted in single-user mode. You can change the root fileset mount from `read-only` to `read-write` with the `mount -u` command using the following format:

mount -u /

Use this procedure when you need to make modifications to the root configuration. For example, use it if you need to modify your `/etc/fstab` file.

Changing the Name of the Root File Domain

You can change the name of the root domain (or any domain). The name of a root domain is stored as the directory name in the `/etc/fdmns` directory and in the entry for root in the `/etc/fstab` file. Thus, to change the name of your root domain, use the `mv` command to rename `root_domain` to whatever name you want. Then use an editor to rename the root file domain entry in the `/etc/fstab` file.

Changing the Name of the Root Fileset

Fileset names are stored in the domain and can be changed with the `renamefset` utility. You can only change the fileset name of an unmounted fileset.

Because you cannot unmount the root fileset, you must use an alternative bootable system. In the alternate root file system, create an entry in the `/etc/fdmns` directory for the root file domain containing the root fileset that you want to change. You can then direct the `renamefset` utility to change the name of the root fileset in the non-mounted root domain.

The following example changes the name of the root fileset from `root_fs` to `new_root`. Assume that the root fileset is in the `root_domain` file domain on `/dev/rz2a`.

1. Boot a root file system other than the one you want to change. (It can be UFS.)
2. Make an entry in the `/etc/fdmns` directory of the mounted root for the root file system whose fileset name you want to change. In this example the root domain is named `tmp_root_domain`:

   ```
   # mkdir /etc/fdmns/tmp_root_domain
   ```

3 Change to the new directory and make a symbolic link for `tmp_root_domain`:

```
# cd /etc/fdmns/tmp_root_domain
# ln -s /dev/rz2a
```

4 Use the `renamefset` command to rename the fileset. In this example the fileset is renamed from `root_fs` to `new_root`:

```
# renamefset tmp_root_domain root_fs new_root
```

5 Mount the changed root to update the `/etc/fstab` file:

```
# mount tmp_root_domain#new_root  /mnt
```

6 Edit the `/etc/fstab` entry for `tmp_root_domain`:

```
# cd /mnt/etc
# vi fstab
```

Change the line:

```
root_domain#root_fs        /       advfs rw 0 0
```

to:

```
root_domain#new_root    /       advfs rw 0 0
```

7 Reboot the AdvFS system.

Note Remember, if you change the root file domain and fileset names and forget to change the `/etc/fstab` entries, you will not be able to boot past single-user mode. You will have to fix `/etc/fstab` from single-user mode using the `ed` command or `vi` command in open mode.

Changing the File Domain Size

AdvFS allows you to increase or decrease the amount of storage available to users and applications by adding or removing volumes in a file domain. Both adding and removing volumes can be accomplished while all filesets in the file domain are mounted and without interrupting users or applications. Path names and directories for files do not change, so both tasks are completely transparent to users.

Adding volumes enables you to create a multivolume file domain, increase the disk capacity of an existing file domain, and perform preventative disk maintenance. You can add volumes immediately after creating the file domain, even before creating and mounting filesets. To perform preventative disk maintenance, you can add a new volume to the file domain, migrate your files to the new volume, and then remove the old volume.

You can also remove a volume from a file domain when the storage is no longer needed. When you remove a volume, AdvFS automatically migrates files to remaining volumes with no interruption to users or applications.

After mounting all filesets in the file domain, use the `addvol` command to add volumes or the `rmvol` command to remove volumes using the following format:

addvol *device_name domain_name*

rmvol *device_name domain_name*

Enlarging an Existing File Domain

You can expand a file domain by adding another disk to the file domain or, alternatively, you can replace one of the disks in the file domain with a larger disk.

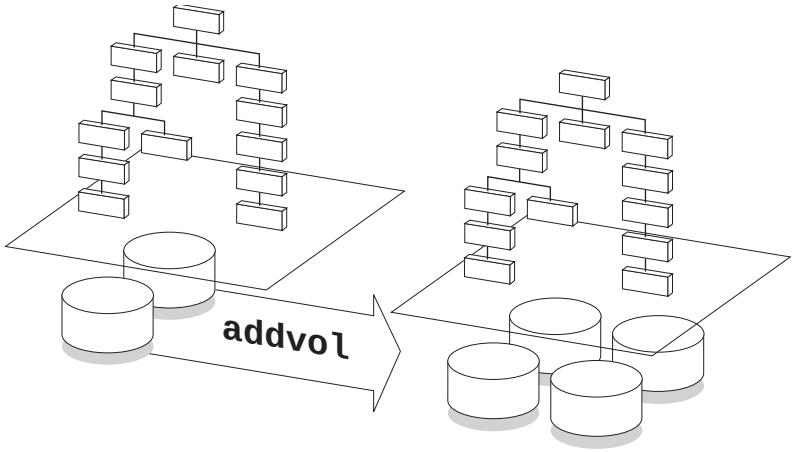
The following example shows how to replace one disk of the `domain_1` file domain with a larger disk. Using the `showfdmn` command, you can display the contents of the file domain and the current disk capacity of each volume:

```
# showfdmn domain_1
               Id           Date Created   LogPgs     Domain Name
2bb0c594.00008570 Wed Mar 24 12:33 1995      512      domain_1

Vol   512-Blks     Free  % Used   Cmode  Rblks Wblks    Vol Name
1L      832527    79210     90%      on    128   128   /dev/rz1c
 2      832527     1684     98%      on    128   128   /dev/rz2c
    -------------------------
       1665054    80894     94%
```

By replacing the `/dev/rz2c` volume with a larger volume, you increase the disk capacity of the file domain. To do this, add the new volume, `/dev/rz3c`, to the file domain and then remove the `/dev/rz2c` volume. Neither adding nor removing volumes affects the directory hierarchy layer; all path names for the users remain the same. Also, the file system can remain active during the disk exchange. Running the `balance` utility after adding and removing volumes evens the file distribution between the volumes. It is not required.

```
# addvol /dev/rz3c domain_1
# rmvol /dev/rz2c domain_1
# balance domain_1
```

Figure 5 Enlarging a File Domain



Reducing the Size of an Existing File Domain

You can reduce the size of a file domain by removing a disk from a file domain or, alternatively, you can replace one of the disks in the file domain with a smaller disk.

The following example shows how to remove one disk of the `domain_2` file domain. Using the `showfdmn` command, you can display the contents of the file domain and the current disk capacity of each volume:

```
# showfdmn domain_2
               Id           Date Created   LogPgs     Domain Name
2bb0c594.00008570 Wed Nov 2  10:23 1995      512      domain_2

Vol   512-Blks     Free  % Used   Cmode  Rblks Wblks    Vol Name
1L      832527   386984     54%      on    128   128   /dev/rz1c
 2      832527   647681     22%      on    128   128   /dev/rz2c
 3      832527   568894     32%      on    128   128   /dev/rz3c

    --------------------------
       2497581 1603559     36%
```

By removing the `/dev/rz2c` volume, you decrease the disk capacity of the file domain. Removing the volume does not affect the directory hierarchy layer; all path names for the users remain the same. Also, the file system can remain active during the disk removal. Running the `balance` utility after removing the volume evens the file distribution between the volumes. It is not required.

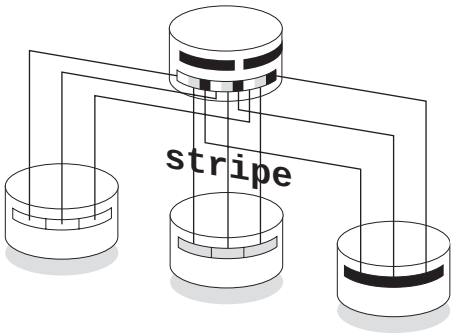
```
# rmvol /dev/rz2c domain_2
# balance domain_1
```

Creating Striped Files

AdvFS *file striping* allows you to increase sequential read/write performance by allocating storage in segments across more than one disk.

File striping, in contrast to disk striping, allows you to stripe individual files according to your needs, without preconfiguring the disks. File striping offers the same performance enhancements for the striped file as disk striping.

Figure 6 Creating Striped Files



Using the `stripe` utility, you can direct a file to distribute segments across specific disks (or volumes) within a file domain. Zero-length files are striped before data is written to the file. As the file is appended, AdvFS determines the number of pages per stripe segment; the segments alternate among the disks in a sequential pattern. For example, the file system allocates the first segment of a two-disk striped file on the first disk and the next segment on the second disk. This completes one sequence, or *stripe*. The next stripe starts on the first disk, and so on. Because AdvFS spreads the I/O of the striped file across the specified disks, the sequential read/write performance of the file increases.

To set up stripe files, first create an empty file. Then, using the `stripe` utility, specify the number of volumes in the file domain over which the file should be striped. Issue the `stripe` command using the following format:

stripe -n *volume_count filename*

For example, the following steps stripe the zero-length (empty) file `abc` over all three volumes in the file domain:

```
# touch abc
# stripe -n 3 abc
```

To stripe an existing file, you must:

1. Create a new, empty file.
2. Stripe the new, empty file.
3. Copy the contents of the old file into the striped file.

The following example stripes file `foo`, which already contains data, across two volumes in the file domain:

```
# touch newfoo
# stripe -n 2 newfoo
# cp foo newfoo
```

Using Trashcans

End users can configure their systems to retain a copy of files they have deleted. *Trashcan directories* can be attached to one or more directories within the same fileset. Once attached, any file deleted from an attached directory is automatically moved to the trashcan directory. The last version of a file deleted from a directory with a trashcan attached can be returned to the original directory with the `mv` command.

Root-user privilege is not required to use this command. However, the following restrictions apply:

- You can restore only the most recently deleted version of a file.
- You can attach more than one directory to the same trashcan directory; however, if you delete files with identical file names from the attached directories, only the most recently deleted file remains in the trashcan directory.

Table 6 lists and defines the commands for setting up and managing trashcans:

Table 6 Trashcan Commands

Command Name	Use	Syntax
`mktrashcan`	Creates the trashcan.	**mktrashcan** *trashcan_name directory*

`shtrashcan`	Shows the contents of the trashcan.	**shtrashcan** *directory*
`rmtrashcan`	Removes the trashcan directory.	**rmtrashcan** *directory*

Note Deleted files in an attached trashcan count against your quota. When you delete files in the trashcan directory, they are unrecoverable.

Using AdvFS with LSM

AdvFS can be used with the Logical Storage Manager (LSM) to provide additional volume-management capabilities. LSM is a disk-management tool that allows you to create mirrored volumes and striped volumes, and to perform other volume-management functions. Use LSM to create and manage the volumes that you use in AdvFS file domains. AdvFS treats LSM volumes just like any other volume such as disks or disk partitions.

In general, use LSM for its volume-management capabilities, such as:

- Volume mirroring (shadowing). Mirroring can improve the read throughput because files can be accessed from either volume depending upon I/O load. Also, mirroring improves availability in case of a disk failure.
- Volume striping, when you want the entire volume striped. (AdvFS stripes individual files.) Striping is useful when large files will be shared and when the transaction log is spread over multiple disks.
- Disk performance monitoring. Detailed information on disk I/O activity is available with LSM disk monitoring.

Use AdvFS to manage file systems and file-level activities:

- Create file domains and filesets.
- Expand and shrink file domains.
- Perform online backups.
- Set quotas on users, groups, and filesets.
- Configure and maintain file systems online.
- Stripe individual files rather than all files on a volume. Do not stripe individual files when using LSM striped volumes.
- Restart the file system fast.

Setting up AdvFS with LSM Volumes

To use LSM with AdvFS, you generally first create the LSM volumes with the desired attributes. LSM options include mirrored volumes, striped volumes, or mirrored and striped volumes. Then, you can create a file domain using the `mkfdmn` command and identifying the LSM volume as the initial volume. To create a multivolume file domain, you can then add an LSM volume to an AdvFS file domain using the `addvol` command just as you would any other volume.

If you already have an AdvFS file domain, you can encapsulate this data into LSM using the encapsulation tools. See the LSM manual for further information.

If mirrored or striped LSM volumes are part of an AdvFS file domain that also includes non-LSM volumes, you do not have control over which files go to the mirrored or striped LSM volumes. To place specific data on mirrored or striped volumes, create an AdvFS file domain that contains only LSM volumes with the attributes that you want. Then, put the files you want mirrored or striped in that file domain.

The `showfdmn` command and the AdvFS GUI include LSM volumes in the file-domain information display. In addition, you can use the `advscan` command to locate AdvFS volumes in LSM disks groups.

Caution Do not use the shrink or grow LSM options with AdvFS file domains. Use the `addvol` and `rmvol` commands to enlarge or reduce the size of file domains.

Examples of Using LSM with AdvFS

The following are examples of how to set up AdvFS file domains with LSM volumes. In each example the volume is created with the desired attributes using LSM. Then an AdvFS file domain is created with the LSM-configured volumes.

The following example creates a 1-gigabyte file domain with two LSM volumes:

```
# volassist make vol01 500m
# volassist make vol02 500m
# mkfdmn /dev/vol/vol01 onegb_domain
# addvol /dev/vol/vol02 onegb_domain
# mkfset onegb_domain onegb_fset1
# mount onegb_domain#onegb_fset1 /fset1
```

The following example creates a file domain with mirrored LSM volumes:

```
# volassist make vol03 500m mirror=yes
# volassist make vol04 500m mirror=yes nmirror=3
# mkfdmn /dev/vol/vol03 safe_domain
# addvol /dev/vol/vol04 safe_domain
# mkfset safe_domain safe_fset1
```

The following example creates a striped LSM volume and then creates a striped AdvFS file domain:

```
# volassist make vol06 600m layout=stripe nstripe=3
# mkfdmn /dev/vol/vol06 striped_domain
```

Chapter 3
Managing Quotas

Enabling quotas allows you to track and control the amount of physical storage that each user, group, or fileset consumes. AdvFS allows you to impose quotas and eliminates the slow reboot activities associated with them on UFS.

The AdvFS quota system is compatible with the Berkeley-style quotas of UFS. However, the AdvFS quota system differs in two ways. AdvFS:

- Differentiates between quota maintenance and quota enforcement. Quota information is always maintained, but enforcement can be activated and deactivated.
- Supports fileset quotas.

Introducing Quotas

You can impose quotas in AdvFS to limit the number of blocks or the number of files that a user, a group of users, or a fileset can use. AdvFS user and group quotas are similar to UFS quotas. You can set a separate quota for each user or each group of users for each fileset.

Fileset quotas limit the amount of disk storage or the number of files by fileset. They restrict the space that a fileset itself can use. Fileset quotas are useful when a file domain contains multiple filesets. Without fileset quotas, any fileset can grab all of the disk space in the file domain.

For example, it is useful to set quotas on filesets that contain home directories such as `/usr/users` because these filesets can grow rapidly. Conversely, setting quota limits on the `/tmp` fileset is not recommended because this fileset is likely to fluctuate in size.

Note You must have root user privilege to set and edit quotas.

Quota Limits

Quotas can have two types of limits: hard and soft. A *hard limit* cannot be exceeded. No more space can be allocated or files created. A *soft limit* permits a period of time during which the limit can be exceeded.

Hard and soft limits can be set or changed by the root user at any time and take effect the next time quotas are activated. Hard and soft limits can be set for users, for groups, and for filesets.

Grace Period

Associated with each soft quota is a grace period. The *grace period* is the amount of time during which the soft limit can be exceeded. When the grace period expires, you cannot create new files or allocate any more disk space. Updating existing files may have unexpected results. To restart the grace period, you must delete files to fall below the soft limit.

The timer for the grace period starts when the user exceeds the soft limit. It is turned off and reset each time usage drops below the soft limit. If you change the grace period after the user has exceeded the soft limit, the old grace period stays in effect until usage drops below the limit.

Note that for each fileset, only one grace period can be set for all users and one grace period can be set for all groups. However, the grace periods set for disk usage and for the number of files do not need to be the same.

When soft limits are imposed, AdvFS sets a default grace period of 7 days. This period can be changed (see the Initiating Quotas and Grace Periods section on page 62). You can specify the grace period in days, hours, minutes, or seconds. You can also:

- Set the grace period to 0 days to impose the default grace period of 7 days.
- Set the grace period to 1 second to allow no grace period.

Quota Files

AdvFS creates quota files to track quotas, grace periods, and fileset usage. The data structure of these files is the same as that used by UFS to hold its quota information and is defined in `/usr/include/ufs/quota.h`. Quota files are maintained within the fileset but, unlike UFS, the user cannot delete or create them. Quota files are present in the fileset even if limits have not been established.

Users and Groups

AdvFS keeps user and group quota information in the root directory of the fileset in the `quota.user` and `quota.group` files. These files are created when the fileset is created and are indexed by user ID and group ID. Each quota file entry contains the following information: hard block limit, soft block limit, block usage, hard file limit, soft file limit, file usage, block grace period, and file grace period.

Quota files are sparse files; that is, there are holes in the file where no user IDs or group IDs fall. If you examine the size of the `quota.user` or `quota.group` file with the `ls` command, your file size may appear enormous. The command shows the space spanned by the whole file. For example:

```
# ls -l quota.user
-rw-r-----  1 root     operator  294912 Jul 20 08:50 quota.user
```

The `du` command shows how many blocks this file actually uses:

```
# du -k quota.user
16      quota.user
```

Filesets

AdvFS keeps fileset soft and hard limits in the structural information associated with the fileset. It contains the same information that the user and group quota files contain: hard block limit, soft block limit, hard file limit, and soft file limit. AdvFS uses the `quota.group` file to maintain the fileset grace period.

Relocating Quota Files

You can relocate the `quota.user` and `quota.group` files to subdirectories of the fileset. You can neither relocate them to other filesets nor delete them. The `/etc/fstab` file entry must be updated to include the path and name of the relocated file(s).

For example,to relocate the `quota.user` file, the following `/etc/fstab` entry will identify `quota.user` as `nquot` and point to its location in the `dir4` subdirectory under the `/mnt` mount point:

```
userquota=/mnt/dir4/nquot
```

If the group quota file is not moved, the complete entry in the `/etc/fstab` file will now be:

```
domain_1#fset /mnt advfs rw,userquota=/mnt/dir4/nquot,groupquota 0 2
```

Initiating Quotas and Grace Periods

The following sections describe how to set and activate:

- User and group quotas and grace periods
- Fileset quotas and grace periods

User and Group Quotas

To establish user and group quotas for a fileset, follow these steps:

1. Modify the `/etc/fstab` file to have the correct mount options.
2. Mount the fileset to which the quotas apply.
3. Use the `edquota` command to set the quota limits.
4. Use the `quotaon` command to enable quota enforcement.

The `/etc/fstab` file must have the quota mount options for the fileset and the fileset must be mounted. The mount point options should include:

advfs rw,userquota,groupquota 0 2

If you specify a group quota, it will apply to all users belonging to that group. Group quotas can take precedence over user quotas. If you specify a user quota that is larger than the group quota, it will have no effect because the group quota will take effect before the user quota is reached.

Use the `edquota` command to set the quotas for users and groups and again to change the grace period. Note that for each fileset, only one grace period can be set for all users and one grace period can be set for all groups. However, you do not have to set the same grace period for the number of blocks and for the number of files.

The `quotaon` and `quotaoff` commands enable and disable enforcement of the quotas you have set.

Follow these general steps to initiate user and group quotas and grace periods:

1. Add quota file mount point options to the `/etc/fstab` file.
2. Issue the `edquota` command with the `-u` flag to set user quotas or the `-g` flag to set group quotas.
3. When the user or group quota information is displayed, modify the values in the limits fields as needed Then, exit the editor, saving the changes.
4. To set user or group grace periods, issue the `edquota` command with the `-ut` flag for user or `-gt` flag for group.
5. When the grace period information is displayed, modify the days as needed. Then, exit the editor, saving the changes.

If quotas have already been activated for a fileset, the new limits become effective immediately. If quotas are not yet activated for a fileset, the new limits become effective as soon as quotas are activated.

Once you have set quotas for a single user, you can use the `edquota` command with the `-p` flag to create prototype quota files. You can then apply the prototype quota to other users you specify. See the Setting Quotas for Multiple Users, Groups, and Filesets section on page 71.

Setting User Quotas

The following example sets quotas for the user `user5`:

1. If they do not already exist, add quota mount-point options to the `/etc/fstab` file. Note that there can be no spaces in the list of options delimited by commas; that is, from `rw` through `groupquota`:

```
domain_1#test1  /test1  advfs rw,userquota,groupquota 0 2
domain_2#test3  /test3  advfs rw,userquota,groupquota 0 2
domain_4#test4  /test4  advfs rw,userquota,groupquota 0 2
```

2. Issue the `edquota` command with the `-u` flag followed by the user name. If you specify more than one user name, the edits will affect all users named. The command creates a temporary file with an ASCII representation of the current quotas assigned to the named users and invokes an editor to allow you to modify the file:

```
# edquota -u user5
Quotas for user user5:
/test1: blocks in use: 0, limits (soft = 0, hard = 0)
        inodes in use: 0, limits (soft = 0, hard = 0)
/test3: blocks in use: 0, limits (soft = 0, hard = 0)
```

```
        inodes in use: 0, limits (soft = 0, hard = 0)
/test4: blocks in use: 0, limits (soft = 0, hard = 0)
        inodes in use: 0, limits (soft = 0, hard = 0)
```

The values for `blocks in use` and `inodes in use` are the current block usage and the number of files for each fileset. You cannot change them. Soft and hard limits of 0 (zero) indicate that no limits have been set.

3 To change user quotas for `user5` for fileset `test3`, edit the file to enter the new limits for disk usage on the `blocks` line and enter the new limits for the number of files on the `inodes` line:

```
/test3:blocks in use: 0, limits(soft=5000, hard=10000)
       inodes in use: 0, limits(soft= 100, hard=  200)
```

4 Exit the editor, saving the changes.

If quotas have already been activated for fileset `test3`, the new limits become effective immediately. If quotas are not yet activated for a fileset, the limits become effective as soon as quotas are activated.

Setting the User Grace Period

When you impose soft limits, you can set one grace period for all fileset users. If you do not specify a grace period, AdvFS defaults to a 7-day grace period. You can choose different grace periods for the number of blocks and for the number of files.

The following example sets the grace period for all users of fileset `test3`:

1 Run the `edquota` command with the user grace period flags. The command creates a temporary file with an ASCII representation of the current grace period and invokes an editor to allow you to modify the file:

```
# edquota -ut
Time units may be: days, hours, minutes, or seconds
Grace period before enforcing soft limits for users:
/test1: block grace period: 0 days,file grace period: 0 days
/test3: block grace period: 0 days,file grace period: 0 days
/test4: block grace period: 0 days,file grace period: 0 days
```

Note that this display may be misleading. Having grace periods of 0 days, as this example shows, does not mean that there is no grace period. It means that the default (7 day) grace period is in effect. If you want to have no grace period, set the grace period to 1 second.

2 To set the user grace period for the number of blocks and for the number of files for `test3`, edit the file to change the existing grace period:

```
Time units may be: days, hours, minutes, or seconds
Grace period before enforcing soft limits for users:
/test1: block grace period: 0 days, file grace period: 0 days
/test3: block grace period: 2 days, file grace period: 3 days
/test4: block grace period: 0 days, file grace period: 0 days
```

3 Exit the editor, saving the changes.

If quotas are on, this grace period becomes effective immediately unless a user has already exceeded the soft limit for `test3`. In that case, the new grace period becomes effective for that user once the user drops below the soft limit.

Setting Group Quotas

The following example sets quotas on `test3` for the group `rsgusers`:

1 If they do not already exist, add quota mount-point options to the `/etc/fstab` file. Note that there can be no spaces in the list of options delimited by commas; that is, from `rw` through `groupquota`:

```
domain_1#test1  /test1  advfs rw,userquota,groupquota 0 2
domain_2#test3  /test3  advfs rw,userquota,groupquota 0 2
domain_4#test4  /test4  advfs rw,userquota,groupquota 0 2
```

2 Issue the `edquota` command with the `-g` flag. If you specify more than one group name, the edits will affect all groups named. The command creates a temporary file with an ASCII representation of the current quotas assigned to the named groups and invokes an editor to allow you to modify the file:

```
# edquota -g rsgusers
Quotas for group rsgusers:
/test1: blocks in use: 0, limits (soft=0, hard=0)
        inodes in use: 0, limits (soft=0, hard=0)
/test3: blocks in use: 0, limits (soft=0, hard=0)
        inodes in use: 0, limits (soft=0, hard=0)
/test4: blocks in use: 0, limits (soft=0, hard=0)
        inodes in use: 0, limits (soft=0, hard=0)
```

The values for `blocks in use` and `inodes in use` are the current block usage and the number of files for each fileset. You cannot change them. Soft and hard limits of 0 (zero) indicate that no limits have been set.

3 To change the group quotas for `test3`, edit the file to enter the new limits for disk usage on the `blocks` line and enter the new limits for the number of files on the `inodes` line:

```
/test3: blocks in use:0,limits(soft=60000, hard=80000)
        inodes in use:0,limits(soft= 6000, hard= 8000)
```

4 Exit the editor, saving the changes.

If quotas have already been activated for fileset `test3`, these limits become effective immediately. If quotas are not yet activated for `test3`, these limits become effective as soon as quotas are activated.

Setting the Group Grace Period

When you impose soft limits, you can set one grace period per fileset for all groups. If you do not specify a grace period, the grace period remains the AdvFS default of 7 days. You can set different grace periods for the number of blocks and for the number of files.

The following example sets the grace period for all groups for fileset `test3`:

1 Run the `edquota` command with the group grace period flags. The command creates a temporary file with an ASCII representation of the current grace period and invokes an editor to allow you to modify the file:

```
# edquota -gt
Time units may be: days, hours, minutes, or seconds
Grace period before enforcing soft limits for groups:
/test1: block grace period: 0 days,file grace period: 0 days
/test3: block grace period: 0 days,file grace period: 0 days
/test4: block grace period: 0 days,file grace period: 0 days
```

Note that this display may be misleading. Having grace periods of 0 days, as this example shows, does not mean that there is no grace period. It means that the default (7 day) grace period is in effect. If you wish to have no grace period, set the grace period to 1 second.

2 To set the group grace period for the number of blocks and for the number of files for `test3`, edit the file to change the existing grace period:

```
Time units may be: days, hours, minutes, or seconds
Grace period before enforcing soft limits for groups:
/test1: block grace period:   0 days, file grace period: 0 days
/test3: block grace period: 12 hours, file grace period: 5 days
/test4: block grace period:   0 days, file grace period: 0 days
```

3 Exit the editor, saving the changes.

If quotas are on, this grace period becomes effective immediately unless a group has already exceeded the soft limit for `test3`. In that case, the new grace period becomes effective for that group once the group drops below the soft limit.

Setting Quota Limits for Special Conditions

You can use the `edquota` command to modify existing quotas for special cases:

- Set the hard limit to 0 days to impose no quota limits. This is the default.
- Set the hard limit to 1 second to permit no disk space allocations.
- Set the soft limit to 1 second and the hard limit to 0 days to permit disk-space allocations on a temporary basis. These limits remain in effect until you unmount the fileset.

Activating Quotas at System Startup

You can automatically start user and group quota enforcement during system initialization by modifying the `/etc/rc.config` flag. Edit the `QUOTA_CONFIG` flag to read:

```
QUOTA_CONFIG=YES
```

This entry causes the `/sbin/init.d` quota script to run the `quotaon` and `quotacheck` commands.

Then, edit the `/etc/fstab` file entry to add `userquota` and `groupquota` to the mount point. Quota enforcement is enabled for the mounted fileset the next time and every time you reboot.

Note If you unmount a fileset when quota enforcement is active, you must explicitly reactivate quota enforcement with the `quotaon` command when you remount the fileset. This must be done even with the `QUOTA_CONFIG=YES` entry.

Activating Quotas Manually

If your system is running and you want to activate new quotas for a mounted fileset, you must run the `quotaon` command.

To establish new user or group quotas do the following:

1. Edit the `/etc/fstab` file entry for your fileset to add `userquota` and `groupquota` to the mount point.
2. Run the `edquota` command to enter the hard and soft limits and to enter the grace period.
3. Run the `quotaon` command to activate the quotas you have chosen.

If your system is set up to initialize quotas (see the Activating Quotas at System Startup section on page 67) you do not need to run the `quotaon` command again unless you have unmounted your fileset. If you have initialized your system without quota enforcement, you must run the `quotaon` command to start enforcement each time you reboot.

The following example activates quotas for the filesets for which quota values were set in the previous sections:

```
# quotaon -av
/test1:  group quotas turned on
/test1:  user quotas turned on
/test3:  group quotas turned on
/test3:  user quotas turned on
/test4:  group quotas turned on
/test4:  user quotas turned on
```

By default, both user and group quotas are affected by the `quotaon` and `quotaoff` commands. You can choose to enable quotas either for users (with the `-u` flag) or for groups (with the `-g` flag). You can also specify the filesets for which user or group quotas will be enforced. See the Deactivating Quotas section on page 81.

Fileset Quotas

Fileset quotas limit the number of files or the amount of disk space a fileset can use. Without fileset quotas imposed, a fileset has access to all of the available disk space in the file domain.

The fileset quotas are set with the `chfsets` command. You can set both soft and hard limits for the number of files and for the number of blocks that a fileset can use. Fileset grace periods are set with the `edquota` command. If you set no grace period, the grace period remains at the AdvFS default grace period of 7 days.

If fileset quotas are set, they are enabled whenever you mount the fileset.

Setting Fileset Quotas

Use the `chfsets` command to define fileset quota values. You can set a soft limit for the number of files (`-F` flag), a hard limit for the number of files (`-f` flag), a soft limit for block usage (`-B` flag), and a hard limit for block usage (`-b` flag). The command displays both the old and new limits.

The following example sets fileset quotas for the `set_1` fileset in `domain_2`. Note that unlike the quota commands, the `showfsets` command displays block usage in 512-byte blocks. If you wish to display kilobyte values, use the `-k` flag.

1 To display existing fileset quotas, use the `showfsets` command:

```
# showfsets domain_2 set_1

set_1
 Id           : 2feff762.00034e3f.1.8001
 Clone is     : set_1_clone
 Files        :       7,  SLim=      0,  HLim=        0
 Blocks (512) :     118,  SLim=      0,  HLim=        0
 Quota Status : user=on  group=on
```

Here `SLim` is the soft limit and `HLim` is the hard limit for the number of files (`Files`) and the current block usage (`Blocks`).

2 Use the `chfsets` command to set the quotas. Note that the arguments for block usage for the `chfsets` command are in units of 1 kilobyte, not 512 bytes as shown by the `showfsets` command display.

```
# chfsets -F 10000 -f 20000 -B 250000 -b 500000 domain_2 set_1

set_1
        Id           : 2feff762.00034e3f.1.8001
        File H Limit : 0 --> 20000
        Block H Limit: 0 --> 500000
        File S Limit : 0 --> 10000
        Block S Limit: 0 --> 250000
```

Here `File H Limit` is the hard limit for the number of files, `Block H Limit` is the hard limit for block usage, `File S Limit` is the soft limit for the number of files, and `Block S Limit` is the soft limit for block usage.

3 To verify the new fileset quotas, run the `showfsets` command again:

```
# showfsets domain_2 set_1
set_1
 Id           : 2feff762.00034e3f.1.8001
 Clone is     : set_1_clone
 Files        :      7,  SLim=  10000,  HLim=    20000
 Blocks (512) :    118,  SLim= 500000,  HLim=  1000000
 Quota Status : user=on  group=on
```

Note The `showfsets` command shows the blocks in units of 512 bytes so they will appear twice as large as the values you input with the `chfsets` command. Use the `showfsets` command with the `-k` flag to display blocks in 1-kilobyte units.

Setting the Fileset Grace Period

Use the `edquota` command to change the grace period for which a fileset can exceed its soft limits. The default AdvFS grace period of 7 days remains in effect until you change it.

You can set only one grace period per fileset, but you can set different values for block usage and number of files. The grace period applies to all users and all groups. If the grace period is reset, the new grace period for the fileset takes effect immediately unless the fileset has already exceeded its soft limits. In that case, the new grace period becomes effective once the fileset drops below the soft limit.

The following example sets the grace period for the `set_1` fileset:

1 Run the `edquota` command with the grace period flags and the fileset name. The command creates a temporary file with an ASCII representation of the current grace period and invokes an editor to allow you to modify the file:

```
# edquota -gt
Time units may be: days, hours, minutes, or seconds
Grace period before enforcing soft limits for set_1:
/test1: block grace period: 0 days,file grace period: 0 days
/test3: block grace period: 0 days,file grace period: 0 days
/test4: block grace period: 0 days,file grace period: 0 days
```

Note that this display may be misleading. Having grace periods of 0 days, as this example shows, does not mean that there is no grace period. It means that the default (7 day) grace period is in effect. If you wish to have no grace period, set the grace period to 1 second.

2. To change the fileset grace period for the number of blocks and for the number of files, edit the file to change the existing grace period:

```
Time units may be: days, hours, minutes, or seconds
Grace period before enforcing soft limits for set_1:
/test1: block grace period: 0 days, file grace period: 0 days
/test3: block grace period: 12 hours, file grace period: 5 days
/test4: block grace period: 0 days, file grace period: 0 days
```

3. Exit the editor, saving the changes.

Activating Fileset Quotas

Running the `chfsets` command activates fileset quotas immediately. No further steps are needed. Fileset quotas are in effect whenever you mount the fileset.

Setting Quotas for Multiple Users, Groups, and Filesets

You can set quotas for more than one user, group, or fileset without accessing and entering values for each one individually. AdvFS allows you to modify quotas for a list of users, groups, or filesets with a single command.

No special procedure is needed to set grace periods for multiple filesets because for each fileset one grace period applies to all users, to all groups, or to all filesets in a file domain.

Multiple Users and Groups

You can use the `edquota` command with the `-p` flag to take an existing set of quotas and establish it as a prototype user or group quota. You can then apply the prototype to one or more users or groups.

For example, you can set up all student accounts to have the same disk usage quota. To do this, you would:

1. Establish one set of quotas with the desired limits for a single student using the `edquota` command.
2. Use the `edquota` command with the `-p` flag to apply the quotas set up for the first user to all students.
3. As students are added at a later date, you can use the `edquota -p` command to apply the prototype quota to new student users.

Prototype User Example

The following example sets up prototype-user quotas that are then used to modify the quotas for other users:

1 Set quotas for one user, `user5`:

```
# edquota -u user5
Quotas for user user5:
/test1:blocks in use:0,limits(soft= 20000,hard= 30000)
       inodes in use:0,limits(soft=   350,hard=   500)
/test3:blocks in use:1,limits(soft= 30000,hard= 40000)
       inodes in use:4,limits(soft=   400,hard=   550)
/test4:blocks in use:2,limits(soft= 10000,hard= 20000)
       inodes in use:1,limits(soft=   150,hard=   200)
/test5:blocks in use:2,limits(soft=100000,hard=150000)
       inodes in use:1,limits(soft=  5000,hard=  7000)
```

2 Create quotas for new users `user7`, `user8`, and `user9`, using the quotas from user `user5` as a prototype:

```
# edquota -p user5 -u user7 user8 user9
```

3 To verify that the quotas were set, run the `edquota` command for `user7`:

```
# edquota -u user7
Quotas for user user7:
/test1:blocks in use:0,limits(soft= 20000,hard= 30000)
       inodes in use:0,limits(soft=   350,hard=   500)
/test3:blocks in use:0,limits(soft= 30000,hard= 40000)
       inodes in use:0,limits(soft=   400,hard=   550)
/test4:blocks in use:0,limits(soft= 10000,hard= 20000)
       inodes in use:0,limits(soft=   150,hard=   200)
/test5:blocks in use:0,limits(soft=100000,hard=150000)
       inodes in use:0,limits(soft=  5000,hard=  7000)
```

Prototype Group Example

The following example sets up prototype group quotas that are then used to modify the quotas for another group:

1 Set quotas for the group, `rsgusers`:

```
# edquota -g rsgusers
Quotas for group rsgusers:
/test1:blocks in use:0,limits(soft=100000,hard=200000)
       inodes in use:0,limits(soft= 10000,hard= 20000)
```

```
/test3:blocks in use:0,limits(soft=300000,hard=400000)
       inodes in use:0,limits(soft= 30000,hard= 40000)
/test4:blocks in use:0,limits(soft=500000,hard=600000)
       inodes in use:0,limits(soft= 50000,hard= 60000)
/test5:blocks in use:0,limits(soft=350000,hard=450000)
       inodes in use:0,limits(soft= 35000,hard= 45000)
```

2 Create quotas for a new group, `rsgstudents`, using the quotas from group `rsgusers` as a prototype:

```
# edquota -p rsgusers -g rsgstudents
```

3 To verify that the quotas were set, run the `edquota` command for `rsgstudents`:

```
# edquota -g rsgstudents
Quotas for group rsgstudents:
/test1:blocks in use:0,limits(soft=100000,hard=200000)
       inodes in use:0,limits(soft= 10000,hard= 20000)
/test3:blocks in use:0,limits(soft=300000,hard=400000)
       inodes in use:0,limits(soft= 30000,hard= 40000)
/test4:blocks in use:0,limits(soft=500000,hard=600000)
       inodes in use:0,limits(soft= 50000,hard= 60000)
/test5:blocks in use:0,limits(soft=350000,hard=450000)
       inodes in use:0,limits(soft= 35000,hard= 45000)
```

Multiple Filesets

You can set quota limits for multiple filesets by listing more than one fileset name when you run the `chfsets` command. See the Setting Fileset Quotas section on page 69.

For example, to change the hard limits for the `data` and `data2` filesets in `test1_domain`, enter the names of both filesets after the `chfsets` command:

```
# chfsets -b 1000 -f 200 test1_domain data data2
data
        Id           : 2fdf591b.000855fa.2.8001
        File H Limit : 11 --> 200
        Block H Limit: 121 --> 1000
data2
        Id           : 2fdf591b.000855fa.3.8001
        File H Limit : 50 --> 200
        Block H Limit: 200 --> 1000
```

See the Setting Fileset Quotas section on page 69 for a description of the output parameters.

Verifying File and Disk Space Usage

You can monitor file and disk space usage even if quotas are not being enforced. If you are enforcing quotas, you can periodically verify your quota setup. The following commands allow you to examine file activity and quota limits for users, groups, and fileset quotas.

Users and Groups

You can display user and group quota information in a number of ways. If you are not the root user, you can display information only for your own files. The root user can display all user and all group quota information for all filesets.

The commands shown in Table 7 are useful for examining disk space and file usage for filesets for which user and group quotas are enforced. See the reference page for each command for details on using the commands.

Table 7 Disk Space Usage Information Commands

Command	Description
`ncheck`	Prints the tag and full path name for each file in the fileset
`quot`	Summarizes fileset ownership
`quota`	Displays disk usage and limits by user or group
`quotacheck`	Checks fileset quota consistency
`repquota`	Summarizes quotas for a fileset

Print the Tag and Full Path Name for Each File

The `ncheck` command lists files by tag (inode) number. By piping the output to the `sort` command, you can use the sorted output as input for the `quot` command to list all files and their owners. Use the following format to generate the listing:

ncheck *domain#fileset* | **sort + on** | **quot -n** *domain#fileset*

Summarize Fileset Ownership

The `quot` command displays block usage and the number of files in the fileset that each user owns. If you do not specify a fileset, the command processes all filesets of type `ro` and `rw` that are listed in the `/etc/fstab` file.

In the following example, the `quot` command is issued with no options to display only blocks:

```
# quot domain_1#set_1
domain_1#set_1:
34128   root
  816   user5
```

In the following example, the `quot` command is issued with the `-f` flag to display both blocks and files:

```
# quot -f domain_1#set_1
domain_1#set_1:
34128     125   root
  816       9   user5
```

Display Disk Usage and Limits

The `quota` command displays the block usage, number of files, and quotas for a user. This command can be run by users to look at their own disk space usage or by the root user to look at system usage.

You can choose to display quota information for all filesets listed in the `/etc/fstab` file or for all mounted filesets. You can restrict your output to those files where usage is over the soft limit.

For each user, this command displays the block usage of the fileset, soft limit (`quota`), hard limit (`limit`), grace period, and number of files used. An asterisk (*) in a column means that that soft quota limit has been exceeded.

The following example shows quota information for the user `user5`:

```
# quota -u user5
Disk quotas for user user5 (uid 446):
Filesystem blocks  quota   limit grace files quota limit  grace
         /     60    100     150           3    10    20
       /usr  11071*  5000   10000 24:40     2    20    40
     /test1    816  20000   30000           9   350   500
     /test2  22032  50000  200000           2  2000  4000
     /test3   2344  10000   15000         370  1000  2000
     /test4  18023* 10000   20000 7days     3   100   150
     /test5  32012* 20000   50000 7days     0  2000  3000
```

The following example shows quota information for the group `rsgusers`:

```
# quota -g rsgusers
Disk quotas for group rsgusers (gid 15):
Filesystem blocks  quota   limit grace files quota limit  grace
         /    118    200     300           2    20    40
       /usr  23184* 10000   20000 7days     2    40    80
     /test1  36136 100000  200000         124 10000 20000
     /test2  44064 200000  400000           4  2000  4000
     /test3   3587  30000   60000         628  3000  5000
     /test4  51071 150000  300000           6  1050  1800
     /test5  61044 100000  200000           3 10000 20000
```

Verify Fileset Quota Consistency

The `quotacheck` command verifies that actual block use and number of files are consistent with the established limits. It examines user and group files, builds a table of current disk usage, and compares this table with that stored in the disk quota file. If any inconsistencies are detected, AdvFS updates both the quota file(s) and the current system copy.

If you do not enable quotas automatically at system startup, it is a good practice to run the `quotacheck` command when quotas are first enabled. To ensure accuracy, run this command when there is no activity on the system.

The `quotacheck` command only checks filesets that have the `userquota` or `groupquota` option specified in the `/etc/fstab` file. By default both user and group quotas are checked, but you can specify either by selecting the `-u` or `-g` flag.

The `quotacheck` command requires that filesets be mounted with quotas enabled. Select the `-v` flag (verbose) to display inconsistencies found and procedures performed during the checking process.

The following example shows a verbose check of the fileset `set_1` that displays no inconsistencies:

```
# quotacheck -v domain_1#set_1
*** Checking user and group quotas for domain_1#set_1 (/test1)
```

The following example checks all filesets that have quotas defined in the `/etc/fstab` file. In this example the `quotacheck` command fixes inconsistencies in `/usr`:

```
# quotacheck -va
*** Checking user and group quotas for /dev/rrz0g (/usr)
*** Checking user and group quotas for domain_1#set_1 (/test1)
/usr: root   fixed: inodes 3057 -> 3022 blocks 100616 -> 123440
/usr: system fixed: inodes 2483 -> 2488 blocks 91721 -> 114568
/usr: adm    fixed: inodes 280 -> 240   blocks 487 -> 464
```

In this display, `inodes` is the number of files and `blocks` is the block usage.

Summarize Quotas by Fileset

The `repquota` command displays the actual disk usage and quotas for the specified filesets. To be included in the summary, the fileset must have a quota entry in the `/etc/fstab` file. By default both user and group quotas are reported, but you can specify either by using the `-u` flag for user or the `-g` flag for group.

For each user or group, the `repquota` command prints the current number of files, the amount of space used, and the quota limits established with the `edquota` command.

The following example summarizes quotas for a single fileset mounted on `/test1`:

```
# repquota -v /test1
*** Report for user quotas on /test1 (domain_1#set_1)
                  Block limits              File limits
User       used    soft    hard  grace  used  soft  hard  grace
root  --  34088       0       0          123     0     0
user5 --    816   20000   30000            9   350   500
```

The following example displays information for all filesets in `/etc/fstab` that have quotas defined. This example contains both UFS and AdvFS files:

```
# repquota -va
*** Report for group quotas on /usr (/dev/rz0g)
                  Block limits              File limits
Group         used  soft  hard  grace   used  soft  hard  grace
system   -- 114568     0     0          2488     0     0
daemon   --    144     0     0             1     0     0
uucp     --    801     0     0             8     0     0
mem      --   1096     0     0            10     0     0
bin      -- 108989     0     0          3219     0     0
mail     --    209     0     0             2     0     0
terminal --     56     0     0             2     0     0
adm      --    464     0     0           240     0     0
operator --    392     0     0             3     0     0
211      --   6937     0     0            33     0     0
*** Report for user quotas on /usr (/dev/rz0g)
                  Block limits              File limits
User          used  soft  hard  grace   used  soft  hard  grace
root     -- 123440     0     0          3022     0     0
bi       -- 102534     0     0          2940     0     0
uucp     --    729     0     0             7     0     0
adm      --      1     0     0             1     0     0
user5    --     15    18    24             1     0     0
kraetsch --   6937     0     0            35     0     0
*** Report for group quotas on /test1 (domain_1#set_1)
                  Block limits              File limits
Group         used  soft  hard  grace   used  soft  hard  grace
system   --  22816     0     0            50     0     0
daemon   --  12088     0     0            82     0     0
*** Report for user quotas on /test1 (domain_1#set_1)
                  Block limits              File limits
User          used  soft  hard  grace   used  soft  hard  grace
root     --  34088     0     0           123     0     0
user5    --    816 20000 30000             9   350   500
*** Report for group quotas on /test3 (domain_2#set_1)
```

```
                  Block limits              File limits
Group        used   soft  hard  grace   used  soft  hard  grace
system   --  1593      0     0            6     0     0
*** Report for user quotas on /test3 (domain_2#set_1)
                  Block limits              File limits
User         used   soft  hard  grace   used  soft  hard  grace
root     --  1593      0     0            6     0     0
```

Filesets

You can also examine how system resources are being used by looking at fileset activity.

Table 8 commands are useful for examining disk space and file usage of filesets:

Table 8 Fileset Disk Usage Information Commands

Command	Description
`df`	Displays disk space used and available disk space for a fileset
`showfdmn`	Displays the attributes of a file domain
`showfsets`	Displays the attributes of filesets in a file domain

Display Used and Available Disk Space

The `df` command displays the disk space used and available for a fileset as follows:

- If a fileset quota has been set, the command displays the amount of space available for the fileset.
- When both soft and hard quota limits are set, the command calculates the disk space available using the soft limit.
- If there is less space in the domain than is allowed by the fileset quota, the command displays the actual space available in the file domain.
- If fileset quotas have not been established, the command displays the domain size; all unused space is available to each fileset.

The following example displays the amount of space available for `fileset_1`:

```
# df /fileset_1
Filesystem        512-blocks  Used Avail  Capacity  Mounted on
test_domain#fileset1    1500  1750     0      117%  /fileset1
```

The `df` command calculates capacity using the soft limit for the amount of space available. Because the usage is over the soft limit, the capacity is determined by the actual space used (1750/1500) and appears as more than 100%.

AdvFS calculates each fileset capacity independently. If the domain has multiple filesets, all unused space is available for each fileset unless the space is limited by fileset quotas. Because the space is counted more than once, the `df` command displays the total capacity as more than 100%. In the following example, the filesets `domain_1#test3` and `domain_1#test4` each can use all of the available disk space from the volumes in `domain_1`:

```
# df
Filesystem   512-blocks     Used     Avail Capacity  Mounted on
domain_1#test3  2000000   390820     98864    80%    /test3
domain_1#test4  2000000   271580     98864    73%    /test4
```

Display File Domain Attributes

Although it does not specifically address filesets, the `showfdmn` command is useful for obtaining statistics to make decisions about filesets and their quotas. The command shows the attributes of a file domain and information about each volume in the domain. For single-volume or multivolume file domains, the command shows the total volume size, the total number of free blocks, and the total percentage of volume space currently allocated.

Display Fileset Attributes

The `showfsets` command with the `-q` flag shows file usage, hard and soft limits, and grace period information for the filesets in the specified domain. It shows the block usage, the block usage limit, the number of files, and the file limit. The correct information will be displayed only if the fileset is mounted.

The following example shows fileset information for the domain `test_domain`:

```
# showfsets -q test_domain
                Block (512) Limits       File Limits
   Fileset  BF  used soft hard grace  used soft hard grace
   fileset1 +-  1750 1500 2000 11:32    35  300  400
```

In this example, the plus sign (+) in the `BF` field means that the soft limit for block usage is exceeded. An asterisk (*) indicates that the hard limit has been reached.

Deactivating Quotas

You can turn off quota enforcement either temporarily or permanently. You can obtain file and disk space usage information regardless of whether you are enforcing quotas.

Users and Groups

The `quotaoff` command turns off quota enforcement until the `quotaon` command is run again either manually or through system initialization that turns quotas on.

The `umount` command turns off quotas before it unmounts a fileset. If you remount the fileset, you must run the `quotaon` command to enforce user and group quotas for the fileset.

If you want to permanently turn quotas off for a user or group, use the `edquota` command to set quota limits to 0 (zero). To prevent quotas from ever being activated for a fileset, run the `quotaoff` command. Then, remove the `userquota` and `groupquota` entries for the fileset in the `/etc/fstab` file.

Filesets

Use the `chfsets` command with the hard and soft limits set to 0 (zero) to deactivate quotas on a fileset.

Chapter 4
Backing Up and Restoring

AdvFS provides extended file-system backup capabilities with the `vdump` and `vrestore` commands. In addition, AdvFS filesets can be safely backed up on line using the AdvFS `clonefset` utility.

While the `dump` command supports UFS exclusively, the `vdump` command can be used to back up not only AdvFS filesets, but also UFS and other standard file systems. The `dump` and `restore` commands function differently from the `vdump` and `vrestore` commands. The `dump` command works at the inode level so it can handle only UFS files. The `vdump` command works at the file level. It scans the directories and uses regular POSIX file-system calls to access directories and files. This processing method allows the `vdump` command to back up different types of file systems.

Caution The tools you use to back up and restore files must be compatible. For example, if you use the `vdump` command to back up a file system, you must use the the `vrestore` command to restore saved files. You cannot use the `vrestore` command to restore files backed up with the `dump` command.

Backing Up Data

The `vdump` command provides features that are not available with the UFS `dump` command. For example, you can back up mounted filesets or individual subdirectories, print the names of files as they are backed up, or compress your data.

There are many options available for the `vdump` command. See the `vdump`(8) reference page for details on all of the flags. To issue a `vdump` command, use the following format:

vdump *options fileset_name*

The vdump command creates a three-part array of fixed-size blocks called a *saveset* as it copies all files that are new or have changed after a certain date to the default storage device or the device that you specify. The first block of the saveset contains the block size and other saveset attributes. The vdump command then makes two passes through the directory hierarchy of the file system being backed up. In the first pass it saves the directories and the file names to the second area of the saveset. In the second pass, it writes the files to the third area of the saveset.

A saveset can span multiple tapes or a tape can contain multiple savesets. Savesets on tapes are delimited by file marks that are written when the saveset is closed by the vdump command.

Unique Features of vdump

The vdump command has a number of functions that the UFS dump command does not have. These extended features are available for all file-system types (not just AdvFS) that you back up with the vdump command. You can:

- Save mounted filesets.
- Choose the subdirectory that you want to back up. You do not need to dump an entire fileset.
- Compress files to minimize the saveset size.
- Specify the number of in-memory buffers. You can maximize throughput by choosing a number compatible with your storage device. Specify the same number when you run the vrestore command.
- Display the current vdump version number.
- Display help information during the dump process.
- Limit your display to error messages. You do not need to display warning messages.
- Display the names of files as they are backed up.
- Configure output with an error-protection system that will allow you to recover data even if there is a read error when you restore.

Because the AdvFS vdump command supports other file-system types, you can use a single backup utility for your entire facility. You can use the unique features provided by the vdump command on all your file systems.

Dumping to Tape

You can place multiple savesets on one tape with the `vdump` command. Set the `-N` flag to specify no rewind or specify a no-rewind device such as `/dev/nrmt0h`. This ensures that the tape does not rewind when the dump finishes.

If a saveset requires more than one tape to complete, you will be prompted to mount more.

Do not combine the output from the `dump` and `vdump` utilities on the same tape. If the `vrestore` command is used to recover files from a tape created by the `dump` utility, the results are unpredictable and can result in data loss.

Dumping Subdirectories

You can selectively back up individual subdirectories of a fileset by specifying the subdirectory with the `-D` flag of the `vdump` command. Without the `-D` flag, if you specify a subdirectoroy instead of a fileset on the command line, the `vdump` command backs up the entire fileset that contains the named subdirectory. If you specify the `-D` flag, backup is always run at level 0.

Compressing Filesets

You can compress filesets as they are backed up. This reduces the amount of storage required for the backup and allows the dump to run faster on slow devices because less data is written. Use the `-C` flag with the `vdump` command to request compression.

Dumping with Error Protection

You can use the `-x` flag with the `vdump` command to place exclusive-or (XOR) blocks on your tape so that the `vrestore` command can recover damaged blocks. The `vdump` command creates these blocks every *n* number of blocks you specify to form a *checksum* block. The valid range of *n* is 2 to 32; the default is 8.

If a block is bad and you have dumped it using the `-x` flag, the `vrestore` command automatically skips the bad block. It reads the rest of the *n*-1 blocks and the checksum block. (The checksum block is ignored unless a previous bad block is found.) The bad block is then recreated from the good blocks and the checksum block.

Dumping with error protection requires saving one extra block for every *n* blocks. It can correct only one block in each series of *n* blocks when the blocks are restored. This means there is a trade-off:

- If you believe tapes are error prone or you require extremely accurate backups and you have many tapes available for backup, set the value of `-x` to 2. This will permit error correction of one in two bad blocks. It will require 50% more tape because after every two dump blocks, a checksum block will be written.
- If you believe that tapes are generally reliable but you want to be able to correct a rare bad block, set the value of `-x` to 32. This will require 3% more tape because an extra block will be added for every 32 blocks written. You could then recover information from any one bad block in the group of 32 dump blocks.

Verifying Backup

You can check your saveset and make sure you have backed up the files you intended. After your backup is complete, run the `vrestore` command with the `-t` flag to display the files you have saved. This will not initiate the restore procedure.

Dumping and Restoring Files Remotely

Although AdvFS does not have a remote dump command equivalent to the UFS `rdump` command, it is possible to dump AdvFS files to remote locations. Use the `vdump` command on the local host. Then access the remote host using the `rsh` command and copy the file to tape with the `dd` command. To perform this procedure, the `/.rhosts` file on the remote system must allow access from root on the local node.

The following example dumps a fileset named `sar` to a tape on node `rachem:`

```
# /sbin/vdump -f - /sar | rsh rachem dd of=/dev/rmt0h
```

To restore the fileset `sar` from the remote tape drive, enter:

```
# rsh rachem -n dd if=/dev/rmt0h obs=60k | /sbin/vrestore -x -f - -D /sar
```

Note that the output block size for `dd` has been set to 60 kilobytes to match the default block size of the `vdump` command. If your tape drive is more efficient using another block size, change the size by using the `-b` flag when you issue the `vdump` command.

When you run the `vrestore` command, you must specify the same block size that you used with your `vdump` command. If the output block size for `dd` does not match the `vdump` block size, the `vrestore` command will exit with the following error message:

```
vrestore: unable to use saveset; invalid or corrupt format
```

Because AdvFS backs up at the file level, you can also do remote dumping and recovery by using the `vdump` command to dump NFS mount points. Mount each disk

on the tape server and use the `vdump` command for each mount point. The `vrestore` command can then be used to restore information. However, with this method you will lose some AdvFS extended file attributes because NFS does not recognize them. For example, if you have a striped file, it will be restored to a single volume.

Cloning for Online Backup

A clone fileset is a read-only snapshot of the data in an existing fileset. You create a clone with the `clonefset` utility (typically daily) to capture the fileset data at a particular time. As you modify the data in your original files, AdvFS saves the data that existed in the original, page by page, into the clone.

Cloning is transparent to the user. Clone filesets are built quickly and have little impact on system performance. To clone a fileset, issue the clonefset command using the following format:

clonefset *domain_name fileset_name clone_name*

After mounting the clone fileset, you can backup the clone fileset with any supported backup tool. Because the clone is a picture of the fileset at a particular time, it is not affected by current system activity. You can back up the clone whenever it is convenient.

Note You can create a clone fileset for any AdvFS fileset including root. You cannot clone UFS file systems. Only one clone can exist per fileset.

How Cloning Works

When you create a clone fileset, only pointers to the file metadata (file structure) are stored in the clone fileset. Data files are not copied. When you modify your data in the original fileset, AdvFS uses the concept of copy-on-write to save the pages that existed when the clone fileset was created. As you update data in a file, the original pages associated with the change are copied to the clone fileset. The original pages are then rewritten with the new data. The clone fileset retains the originals of all data that has changed since the clone was created.

To create a clone fileset, AdvFS:

1. Creates a read-only fileset to be the clone.
2. Copies only the original fileset tag directory to the clone.
3. Creates a link between the two filesets.

4 Sets up bookkeeping to track whether a given page has been updated.

Once a page has been added to the clone, it is marked. If the same page is updated again, the clone does not change. It already contains the information that existed when the clone was created.

AdvFS allocates clone fileset space by pages (8 kilobytes). If you modify one page in a large file, then only one additional page is allocated by the clone. Note that if a file is modified so that pages are appended, these pages will not appear in the clone because they were created after the clone was created.

Unless you modify every page of every file in the original fileset during the life of clone, the clone fileset occupies less disk space than the original fileset.

Note Changing text files with an editor may cause the entire original file to be copied to the clone. Many editors rewrite the entire file regardless of what has changed. When this happens, your clone fileset may grow very large. There is no way for AdvFS to alter this process.

When you delete a file that existed when the clone was created, it remains available (but not visible in the original fileset) for the life of the clone. The file is not copied to the clone, but the actual delete is delayed until the clone is deleted. The version of the file that is retained is the one that existed when the clone was created. Later updates are lost.

The size of the clone fileset depends upon the number of updates that occur during the life of the clone. This is not information that is of general interest because it constantly changes as files are updated. Thus, the `df` command does not accurately reflect the size of the clone fileset.

The following steps explain the process AdvFS uses when you modify data in a file after you create a clone fileset:

1 AdvFS checks to see if this is the first time the data has changed for this file. If so, the original data may need to be saved in the clone. AdvFS determines this by examining the status of the clone's metadata.. There are three possibilities:

- Metadata does not exist.

  The file was created after the clone was created and is not part of the clone fileset. Nothing is done to the clone fileset.

- Metadata exists and the data has been marked as saved.

  This is not the first update to this page of the file. Nothing is done to the clone fileset.

- Metadata exists but this page has not been marked as saved.

This is the first change to this page of the file. Original data is being updated. Metadata for this page of the clone is altered and the page marked as saved. Disk space is allocated in the clone fileset and the original data is copied into the new space.

2 AdvFS updates the original file.

Caution When a file domain runs out of disk space, the file system loses its ability to maintain the consistency of files within clone filesets. The original fileset is usable, but the clone fileset is not accurate. A warning message is displayed on both the user's terminal and the system console.

Using Clones

You can create clones either with the command-line interface or with the AdvFS GUI. A description of the command-line procedure follows.

Clones are created with the `clonefset` command. Specify the file domain and fileset you want to clone and then assign a clone fileset name. After creating a mount point and mounting the clone, you can run the `vdump` utility on the clone at any time.

The following example backs up the `public` fileset on line by creating the `public_clone` fileset and backing it up. The file domain in this example is `domain1`.

```
# clonefset domain1 public public_clone
# mkdir /public_clone
# mount -t advfs domain1#public_clone /public_clone
# vdump -0 -u -C /public_clone
```

To remove the `public_clone` fileset, enter:

```
# umount /public_clone
# rmfset domain1#public_clone
```

Backing Up Databases

If your database has an online backup utility, Digital recommends that you use it for your backup. If it does not, you can back up databases with database down time limited to the short time it takes to create the clone fileset. Backing up a database with a clone fileset is the same as backing up any other fileset. You get the same benefits. See the Cloning for Online Backup section on page 87.

The procedure for backing up a database from a clone is as follows:

1. Shut down the database so that all database buffers are flushed and the filesets have a complete, consistent copy of the database files.
2. Clone the fileset and mount the clone.
3. Reactivate the database.
4. When you want to back up the clone fileset, run a backup procedure such as the `vdump` utility or Digital NetWorker.
5. Unmount and delete the clone.

Caution Do not use anything except the database's own utilities to back up an active database. You can use the `vdump` and `vrestore` commands on a database clone.

Restoring Data

The `vrestore` command reads the blocks from a saveset created with the `vdump` command and processes the records in the block. The `vrestore` command will not work on a saveset created by the UFS `dump` command.

You must have write access to the directory you restore to. Root-user privilege allows you to restore to any directory that has write privileges. See the `vrestore`(8) reference page for details on the `vrestore` command options.

Unique Features of vrestore

The `vrestore` command performs a number of activities that the UFS `restore` command does not. You can:

- Display the current `vrestore` version number.
- List the saveset structure.
- Display error messages only. Information messages will not be shown.
- Specify how the `vrestore` command should proceed if it encounters a file that already exists. You can choose whether the command will always overwrite an existing file, never overwrite an existing file, or query you for each event.

Restoring Files

The `vrestore` command allows you to select specific files and directories to be restored. It can restore data from a file, a pipe, magnetic tapes, or disks.

Before you restore files, you can check if the saveset you are accessing contains the information you wish to recover. You can list the names and sizes of all files in your backup by running the `vrestore` command using the `-t` flag. The restore operation will not be performed. You can also display the files and directories saved by running the `vrestore` command with the -i flag. This interactive option allows you to select individual files or directories to restore from a list.

Restoring data from a clone fileset is the same as restoring data from any other fileset.

Restoring Selected Savesets

To restore from a tape containing multiple savesets, use the `mt` command with the `fsf n` (which means forward space *n* files) option to locate the saveset you want to restore. Then use the `vrestore` command.

You can selectively restore files from your saveset with the `-x` flag of the `vrestore` command. You can also specify a destination path other than the current directory for the restored files.

The following example restores the file named `data_file` from the `/mnt/fdump` saveset. It is restored to the `/mnt` directory.

```
# vrestore -f /mnt/fdump -D /mnt -x data_file
vrestore: Date of the vdump save-set: Tue Jun 13 15:27:36 1995
```

AdvFS and NetWorker

Digital's NetWorker Save and Restore product provides scheduled, online, automated backup. Use NetWorker with AdvFS as a comprehensive backup solution. NetWorker can automatically back up multiple servers in a heterogeneous environment. It has a graphical interface and several scheduling options.

To use the automated backup capabilities of NetWorker, use the AdvFS `clonefset` utility to clone all filesets for backup and mount the clone filesets. (You can create a script to accomplish this task.) Then, set up NetWorker to automatically back up the clone filesets on a convenient schedule.

Chapter 5
Performance Tuning

AdvFS provides a number of ways to tune your file system's performance. You can configure your filesets and domains to optimize performance. You can run utilities that defragment, balance, stripe, and migrate your files. You can adjust how AdvFS performs logging, caching, and storage allocation.

Configuring for Performance

To ensure optimal performance, AdvFS limits the number of active filesets, file domains, and volumes that it supports. The following sections outline AdvFS file-system limits.

Fileset and File Limits

While AdvFS allows an unlimited number of filesets per system, only 512 filesets minus the number of active file domains can be mounted simultaneously.

The number of files per fileset is limited to 2^{32}, if not first limited by disk space or by quotas.

The currently tested maximum file size is 128 gigabytes. Larger file sizes are allowed, but have not been tested. Sparse files can be addressed to 16 terabytes.

File-Domain Limits

AdvFS supports a maximum of 100 active file domains per system. A file domain is active when at least one fileset within that domain is mounted. The maximum file-domain size is 128 terabytes.

Each file domain retains its own transaction log. If the file domain has a large number of volumes or includes large numbers of filesets, the log can create a bottleneck

because of high activity. See the Improving the Transaction Log Performance section on page 105.

It is important to consider the number of filesets in your domain when setting up your system because each active file domain counts against the mounted-fileset limit. Table 9 shows how the mounted fileset count increases as the number of active file domains increases.

Table 9 Fileset Count

Active Domains	Filesets Mounted	Mounted Fileset Count
1	2	3
1	99	100
99	99 (1 per domain)	198

Volume Configuration

You can add up to 250 volumes to an AdvFS file domain. However, without disk mirroring, it is inadvisable to add more than three volumes. If you lose a volume, the entire domain becomes inaccessible. The risk of losing a volume, and thus losing access to your file domain, increases as the number of volumes increases.

There is a small performance advantage to dividing disks on different SCSI chains. However, if you must purchase additional controllers, this can be an expensive way to provide a minimal performance improvement in data access.

The theoretical maximum AdvFS volume size is 1 terabyte.

Utilities for Tuning

AdvFS provides utilities for keeping your file system running as efficiently as possible. These tools improve read/write performance by altering the way files are mapped on the disk. They can be run while the system is on line and are transparent to system users and to applications.

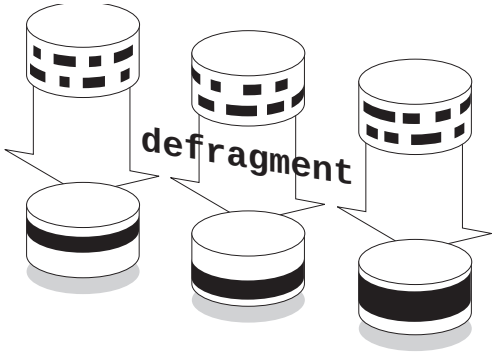
You can improve file read/write performance on your system with utilities that:

- Defragment a file domain.
- Balance a multivolume file domain to even the storage distribution between volumes.
- Stripe files across volumes in a multivolume file domain.
- Migrate files to a different volume within a multivolume file domain.

Defragment Utility

AdvFS attempts to store file data in contiguous blocks on a disk. This collection of contiguous blocks is called a *file extent*. If all data in a file is stored in contiguous blocks, that file has one file extent. However, as files grow, contiguous blocks on the disk may not be available to accommodate the new data. The system must spread the file over discontiguous blocks. As a result, the file is fragmented on the disk and consists of multiple file extents. File fragmentation degrades the read/write performance because many disk addresses must be examined to access a file.

Figure 7 Defragment File Domain



The `defragment` utility reduces the amount of file fragmentation in a file domain by attempting to make the files more contiguous so that the number of file extents is reduced. Defragmentation is an iterative, two-step process that operates on the file domain, as follows:

1. Files are moved out of a region to create an area with contiguous, unallocated space.
2. Fragmented files are written into a region that has more contiguous space so they are less fragmented.

In addition to making files contiguous so that the number of file extents is reduced, defragmenting a file domain often makes the free space on a disk more contiguous so files that are created later will also be less fragmented.

You can improve the efficiency of the defragment process by deleting any unneeded files in the file domain before running the `defragment` utility.

The following restrictions apply to running the `defragment` utility:

1. You must have root user privileges to access the `defragment` command.
2. All filesets in the file domain must be mounted. If you try to defragment an active file domain that includes unmounted filesets, you will get an error message.
3. A minimum free space amount of 1% of the total space or 5 megabytes per volume (whichever is less) must exist in order to run.
4. The `defragment` utility cannot be run while the `addvol`, `rmvol`, `balance`, or `rmfset` command is running in the same file domain.
5. Striped files are not defragmented.

Choosing to Defragment

To determine the amount of file fragmentation that exists in a file domain, use the `defragment` command with the `-v` and `-n` flags. This will show how fragmented the file domain is without starting the `defragment` utility.

Table 10 describes the information displayed by the `defragment` utility when you run the command with the verbose (`-v`) flag.

Table 10 Defragment Utility Output

Heading	Description
`Extents`	Number of extents in the specified domain
`Files w/ extents`	Number of files that have extents
`Avg exts per file w/ exts`	Average number of extents for each file that has one or more extents
`Aggregate I/O perf`	Efficiency of the entire file domain
`Free space fragments`	Number of free space fragments in the domain

You can also use the `showfile` command to check the number of file extents of individual files. The following example shows the attributes and extent information for the `mail` file:

```
# showfile -x mail
       Id Vol PgSz Pages XtntType Segs SegSz Log Perf  File
4198.800d   2   16    27   simple   **    ** off  66%  tutorial
        extentMap: 1
            pageOff    pageCnt    vol    volBlock    blockCnt
                  0          5      2      781552          80
                  5         12      2      785776         192
                 17         10      2      786800         160
               extentCnt: 3
```

Defragment Example

The following example defragments the `accounts_domain` file domain. A time limit of 15 minutes is imposed. Verbose mode is requested to display the fragmentation data at the beginning of each pass made through the file domain and at the end of the defragmentation process.

```
# defragment -v -t 15 accounts_domain
defragment:  Defragmenting domain 'accounts_domain'

Pass 1; Clearing
  Volume 1: area at block    11680 (  103072 blocks): 81% full
  Domain data as of the start of this pass:
    Extents:                  10432
    Files w/extents:           4305
    Avg exts per file w/exts:  2.42
    Aggregate I/O perf:         52%
    Free space fragments:      2743
                    <100K    <1M    <10M    >10M
     Free space:     38%      0%      0%     62%
     Fragments:     2742       0       0       1
Filling
.
.
.
Pass 13; Clearing
  Volume 1: area at block   559744 (   62736 blocks):  0% full
  Volume 2: area at block    76640 (   24624 blocks): 18% full
  Domain data as of the start of this pass:
    Extents:                  4306
    Files w/extents:          4305
    Avg exts per file w/exts:  1.00
    Aggregate I/O perf:        100%
    Free space fragments:        23
```

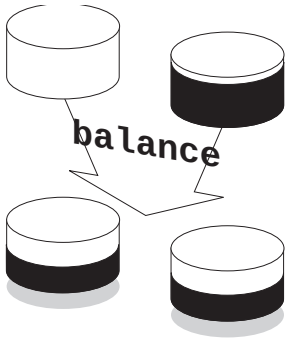
```
                    <100K    <1M    <10M    >10M
     Free space:      0%      9%     27%     64%
     Fragments:        6      10       5       2
Filling

Current domain data:
   Extents:                   4305
   Files w/extents:           4305
   Avg exts per file w/exts:  1.00
   Aggregate I/O perf:        100%
   Free space fragments:        17
                   <100K    <1M    <10M    >10M
    Free space:      0%      6%     29%     65%
    Fragments:        3       8       4       2

Defragment: Defragmented domain 'accounts_domain'
```

Information displayed before each pass and at the conclusion of the defragmentation process indicates the amount of improvement made to the file domain. A decrease in the `Extents` and `Avg exts per file w/extents` values indicates a reduction in file fragmentation. An increase in the `Aggregate I/O perf` value indicates improvement in the overall efficiency of file-extent allocation.

Balance Utility

The `balance` utility distributes the percentage of used space evenly between volumes in a multivolume file domain. This improves performance and evens the distribution of future file allocations.

Figure 8 Balance File Domain



Files are moved from one volume to another until the percentage of used space on each volume in the domain is as equal as possible. (Because the `balance` utility does not

generally split files, file domains with very large files may not balance as evenly as file domains with smaller files.)

The following restrictions apply to running the `balance` utility:

1. You must have root user privileges to access the `balance` command.
2. All filesets in the file domain must be mounted. If you try to balance an active file domain that includes unmounted filesets, you will get an error message.
3. A minimum free space amount of 1% of the total space or 5 megabytes per volume (whichever is less) must exist in order to run.
4. The `balance` utility cannot run while the `addvol`, `rmvol`, `defragment`, or `rmfset` command is running in the same file domain.

Choosing to Balance

To determine if you need to balance your files across volumes, use the `showfdmn` command to display file-domain information. From the `% used` field you can determine if the files are evenly distributed.

In the following example, the `usr_domain` file domain is not balanced. Volume 1 has 85% used space while volume 2 has 0% used space (it has just been added).

```
# showfdmn usr_domain
               Id      Date Created         LogPgs  Domain Name
2dcab512.000d1ed0  Fri May  6 14:22:26 1994    512  usr_domain

  Vol   512-Blks    Free  % Used  Cmode  Rblks  Wblks  Vol Name
   1L     819200  126848     85%     on    128    128 /dev/rz8g
   2      768281  768080      0%     on    128    128 /dev/rz9d
      ----------  -------  ------
         1587481  894928     44%
```

Use the `balance` utility to even file distribution after you have added a volume with the `addvol` command or removed a volume with the `rmvol` command (if there are multiple volumes remaining).

Balance Example

The following example balances the multivolume domain `usr_domain`, examined above:

```
#  balance usr_domain
 balance: Balancing domain 'usr_domain'
 balance: Balanced domain 'usr_domain'

# showfdmn usr_domain
               Id      Date Created         LogPgs  Domain Name
2dcab512.000d1ed0  Fri May  6 14:22:26 1994    512  usr_domain

  Vol   512-Blks    Free  % Used  Cmode  Rblks  Wblks  Vol Name
   1L     819200  459248     44%     on    128    128 /dev/rz8g
   2      768281  433632     44%     on    128    128 /dev/rz9d
      ----------  -------  ------
         1587481  892880     44%
```

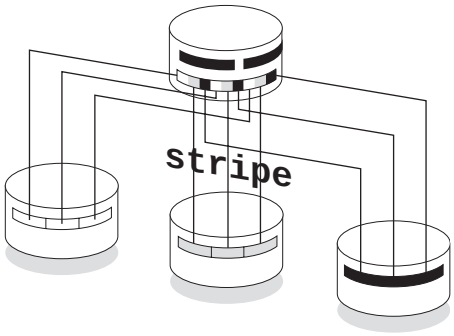
The `balance` utility moved files from volume 1 to volume 2 in order to even the percentage of used space between the two volumes.

Stripe Utility

Striping increases sequential read/write performance by allocating storage in segments across more than one volume. AdvFS allows you to stripe individual files according to your needs. For example, if you stripe files that have heavy input/output requirements, the I/O is spread across the volumes, so that read/write requests to the different disk drives can be overlapped. The AdvFS striping utility does not require you to stripe all your files.

With the `stripe` utility, you direct a file to distribute segments across specific volumes within a file domain. You first create a new zero-length file, stripe it, then write your data to the striped file and delete the original file. You can choose the number of volumes on which to stripe a file.

Figure 9 Stripe Files



As the file is appended, AdvFS determines the number of pages per stripe segment; the segments alternate among the disks in a sequential pattern. For example, the file system allocates the first segment of a two-disk striped file on the first disk and the next segment on the second disk. This completes one sequence, or stripe. The next stripe starts on the first disk, and so on.

You cannot use the `stripe` utility to modify the number of disks that an already striped file crosses or to restripe a file that is already striped. To change the configuration of a striped file, you must repeat the striping process.

Choosing to Stripe

Before you use the `stripe` utility, run the `iostat` utility to determine if disk I/O is the bottleneck operation. The blocks per second and transactions per second should be cross checked with the drive's sustained transfer rate. If the disk access is slow, then striping will improve performance. See the Checking Disk Activity section on page 111.

Stripe Example

The following example creates an empty file, stripes it, copies data into the striped file, then shows the extents of the striped file:

1 Create the empty file `file_1` and stripe it across three volumes in a domain:

```
# touch file_1
# ls -l file_1
-rw-r--r--  1 root   system      0 Aug  1 05:50 file_1
# stripe -n 3 file_1
```

2 Copy the data from the original file to the striped file:

```
# cp orig_file_1 file_1
```

3 Examine the extents of the new striped file:

```
# showfile -x file_1
Id         Vol PgSz  Pages  XtntType  Segs  SegSz  Log  Perf  File
1ed.8053    1    16    115    stripe     3      8  off   75%  file_1
    extentMap: 1
        pageOff    pageCnt    volIndex    volBlock    blockCnt
              0          8           2         160         384
             24          8
             48          8
        extentCnt: 1
    extentMap: 2
        pageOff    pageCnt    volIndex    volBlock    blockCnt
              8          8           3         128         336
             32          8
             56          5
        extentCnt: 1

    extentMap: 3
        pageOff    pageCnt    volIndex    volBlock    blockCnt
             16          8           1        8528         256
             40          8
        extentCnt: 1
  extentCnt: 1
```

Choosing Between AdvFS and LSM Striping

AdvFS implements file striping at an individual file level. Therefore, if only one or a few files are large and have heavy I/O requirements, use the AdvFS `stripe` utility to stripe these files.

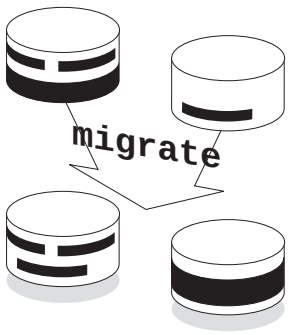
The Logical Storage Manager (LSM) provides volume-level striping. With LSM you preconfigure your volumes for file striping. Then, all files located on the volumes configured for striping will be striped.

Migrate Utility

The `migrate` utility improves file I/O performance by altering the way files are mapped on the disk. You can use the `migrate` utility to move heavily accessed or large files to a different volume in the file domain. You can specify the volume where a

file is to be moved or allow the system to pick the best space in the file domain. You can migrate either an entire file or specific pages to a different volume.

Figure 10 Migrate Files



A file that is migrated will be defragmented in the process if possible. This means that you can use the `migrate` command to defragment selected files.

The following restrictions apply to the `migrate` utility:

- You must have root-user privilege to access this utility.
- You can only perform one migrate operation at a time on the same file.
- When you migrate a striped file, you can only migrate one source volume at a time.
- The `migrate` utility does not evaluate your migration decisions. For example, you can move more than one striped file segment to the same disk, which defeats the purpose of striping the file.

Choosing to Migrate

If you suspect that a fileset or file domain is straining system resources, run the `iostat` utility. If the filesets or file domains are located on devices that appear to be a bottleneck, you can migrate files or pages of files to equalize the load. If a high-performance device is available, you can move an I/O-intensive fileset to the more efficient volume. See the Checking Disk Activity section on page 111.

Choose the `migrate` utility over the `balance` utility when you want to control the files that are moved. The `balance` utility moves files only to optimize distribution. For example, it might move many small files when moving a single larger one would be a better solution for your system.

You can use the `showfile` command with the `-x` flag to check the performance percentage and extent map of a file. A low performance percentage (less than 80%)

indicates that the file is fragmented on the disk. The extent map shows whether the entire file or a portion of the file is fragmented.

The following example displays the extent map of a file called `src`. The file, which resides in a two-volume file domain, has an 18% performance efficiency. The `showfile` command with the `-x` flag lists the extent map for the file `src`:

```
# showfile -x src
       Id Vol PgSz Pages XtntType  Segs  SegSz  Log  Perf  File
2e1b.8004   1   16    11   simple    **     **  off   18%  src
    extentMap: 1
        pageOff    pageCnt     vol    volBlock    blockCnt
              0          1       1      684704          16
              1          1       1      684896          16
              2          1       1      684928          16
              3          1       1      685280          16
              4          1       1      685312          16
              5          1       1      685488          16
              6          1       1      686432          16
              7          1       1      686608          16
              8          1       1      686784          16
              9          1       1      686960          16
             10          1       1      687168          16
        extentCnt: 11
```

The file `src` consists of 11 file extents. This file would be a good candidate to move to another volume to reduce the number of file extents.

Migrate Example

The following example migrates the file `src` examined above. The number of file extents is decreased:

```
# migrate -d 2 src
# showfile -x src
       Id Vol PgSz Pages XtntType Segs SegSz  Log  Perf  File
2e1b.8004   1   16    11   simple   **    **  off  100%  src
    extentMap: 1
            pageOff    pageCnt     vol    volBlock    blockCnt
                  0         11       2        1280         176
            extentCnt: 1
```

The file `src` now resides on volume 2, consists of one file extent, and has a 100% performance efficiency. Note that in the output above, the first data line of the display lists the metadata. The metadata does not migrate to the new volume. It remains in the original location. The `extentMap` portion of the display lists the migrated files.

You can tailor the `migrate` utility to the needs of your system. The following examples illustrate some possibilities.

You can migrate file `abc` and let the system pick a new location in the file domain:

```
# migrate abc
```

You can migrate pages 10 through 99 of file `abc` to volume 2 of the file domain:

```
# migrate -p 10 -n 90 -d 2 abc
```

You can move the pages of a striped file to different volumes within a file domain. For example, if the `abc` file is striped across three volumes (volumes 1, 2, 3) of a 6-volume file domain, you can use the `migrate` utility to move the pages from volume 2 to volume 4. You must specify the page offset and the page count for the pages you want to move in addition to the source volume information. Use the `showfile` command to determine the page count. In the following example, volume 2 contains pages between 8 and 39 of the striped file. To migrate the pages, enter:

```
# migrate -p 8 -n 32 -s 2 -d 4 abc
```

The `abc` file is now striped across volumes 1, 3, and 4.

Improving the Transaction Log Performance

Each file domain has a transaction log that keeps track of fileset activity for all filesets in the file domain. If the log resides on a congested disk or bus, or the file domain contains many filesets, system performance can degrade.

Use the `showfdmn` command to determine the location of the log. The letter L after the volume number indicates the volume on which the log resides. You can monitor performance of this volume with the `iostat` utility. If the volume containing the log appears overloaded, you can:

- Divide the file domain into several smaller domains. Then, each transaction log will handle transactions for fewer filesets.
- Use the `switchlog` command to move the log to a faster or less congested volume.

The `switchlog` command allows you to move the transaction log to any volume in the domain. Refer to the `switchlog`(8) reference page for more information. Moving the log to a faster device can improve performance. Issue the `switchlog` command with the following format:

switchlog *domain_name new_volume_number*

Moving the transaction log may also be useful when you are using LSM and wish to increase reliability by placing your transaction log on a volume that is mirrored.

Chapter 6
Troubleshooting

This chapter examines problems that, while universal for file systems, have unique solutions when your system is configured with AdvFS.

Managing Disk Space

The first step to managing excessive disk space consumption is to request that users delete unnecessary files. There are a number of utilities that look at file usage so that you can monitor storage allocation. You can also limit disk space consumption by imposing quotas on users and groups or on the filesets set up on the system.

Checking Free Space and Disk Usage

You can look at the way space is allocated on a disk by file, fileset, or file domain. This information can help you identify users that stress the system.

Table 11 lists commands that are useful for examining disk space usage:

Table 11 Disk Space Usage Information Commands

Command	Description
`du`	Displays information about block allocation for files.
`df`	Displays disk space usage by fileset.
`showfdmn`	Displays the attributes and block usage for each volume in an active file domain.

Display Block Allocation Information

Use the du command to display information about block allocation for files in specific directories. By specifying the -a flag, the du command displays the number of blocks in use by individual files. Refer to the du(1) reference page for details on all of the options available for this command.

To display disk space usage information for individual files in a directory, use the following du command format:

du -a *directory*

Display Fileset Disk Space Usage

Use the df command to display disk space usage for filesets. Included in the display is the size, in blocks, of the file domain. This is the maximum amount of space that a fileset can occupy. If a fileset quota is set, the quota limit is displayed as the size, in blocks, because this is the maximum amount of space that the fileset can use. The display also includes used space which is the amount of space that a fileset consumes.

To display disk space usage information for all AdvFS filesets on the system, use the following df command format:

df -t advfs

The df command has several other display options. Refer to the df(1) reference page for information on all of the available options.

Display File Domain Disk Space Usage

Use the showfdmn command to display the attributes and block usage for each volume in an active file domain. For multivolume domains, the showfdmn command also displays the total volume size, the total number of free blocks, and the total percentage of volume space currently allocated.

To display disk space usage information for individual file domains, use the following showfdmn command format:

showfdmn *domain_name*

Refer to the showfdmn(8) reference page for information on all of the available options.

To display information about all file domains on a system, run the following commands:

```
# cd /etc/fdmns
# showfdmn *
```

Limiting Disk Space Usage

If your system has been running without any limits on resource usage, you can add quotas to your system to limit the amount of disk space your users can access. AdvFS quotas provide a layer of control beyond that available with UFS. You can limit the number of files or blocks used by a fileset as well as the resources used by individual users and by groups. See the Managing Quotas chapter starting on page 59 for complete information.

You can set two types of quotas: hard limits that cannot be exceeded and soft limits that can be exceeded for a period of time called the grace period. You can turn quota enforcement on and off.

Fileset Quotas

Fileset quotas restrain a fileset from grabbing all of the available space in a file domain. Without them, any fileset can use all of the available space in a file domain. Table 12 lists all of the commands that you use to set up and manage fileset quotas.

Table 12 Fileset Quota Commands

Command	Description
`chfsets`	Changes limits (quotas) for block usage and number of files.
`df`	Displays the limits and actual number of blocks used in a fileset.
`showfdmn`	Displays disk space usage for file domains.
`showfsets`	Displays the number of files and block usage limits for filesets.

User and Group Quotas

User and group quotas limit the amount of space a user or group can allocate for a fileset. Table 13 lists the commands that operate on user and group quotas.

Table 13 User and Group Quotas Commands

Command	Description
`edquota`	Edits quotas and grace periods.
`ncheck`	Displays a list of pairs (tag and path name) for all files in a specified fileset. Use the sorted output as input for the `quot` command.
`quot`	Displays the number of blocks in the named filesets currently owned by each user.
`quota`	Displays disk space usage and limits for users and groups that have quotas enabled.
`quotacheck`	Checks file system quota consistency and corrects it if necessary.
`quotaon`, `quotaoff`	Turn quota enforcement on and off.
`repquota`	Prints a summary of the disk usage and quotas by user, group, or fileset.

Running into Limits

If you are working in an editor and realize that the information you need to save will put you over your quota limit, do not abort the editor or write the file because data may be lost. Instead, remove files to make room for the edited file prior to writing it. You can also write the file to another fileset, such as `tmp`, remove files from the fileset whose quota you exceeded, and then move the file back to that fileset.

AdvFS will impose quota limits in the rare case that a file is less than 8 kilobytes below its quota limit and less than 8 kilobytes are to be added to it. This is because AdvFS is structured to allocate storage in pages of 8 kilobytes each time a file is created or extended. When less storage is needed, the system accesses the `frag` file for the fileset to obtain smaller pieces of storage (1 through 7 kilobytes). However, quota limits are tested *before* the decision is made to allocate a fragment. AdvFS assumes that 8 kilobytes will be added, which would put the file over the quota limit.

Handling Poor Performance

The performance of a disk depends upon the I/O demands upon it. If your file domain is structured so that heavy access is focused on one volume, it is likely that system performance will degrade. Once you have determined the load balance on your system, there are a number of ways to equalize the activity and increase throughput. See the Performance Tuning chapter starting on page 93.

Checking Disk Activity

The first step in determining the cause of poor performance is to examine disk activity. Use the `iostat` utility to display the number of transfers per second (tps) and kilobytes transferred per second (bps). From this you can determine where I/O bottlenecks are occurring. That is, if one device shows sustained large numbers in a column, this device is being hit more than others. Then you can decide what action might increase throughput: moving files, obtaining faster volumes, striping files, etc.

The following example of the output from the `iostat` command displays CPU, terminal, and disk statistics for four disks on a system. The example displays five reports at 1-second intervals:

```
# iostat 1 5
           tty      rz1       rz2       rz3       rz4      cpu
      tin tout bps tps  bps tps  bps tps  bps tps  us ni sy id
        1   52   2   0    1   0   13   1    4   1   8  0  9 83
        1   16   7   1    2   0    5   2    2   0   3  0 10 87
        0    0   0   0    0   0    0   0    0   0   0  0  1 98
        2    2   2   1    0   0   50   6    0   0   9  0  9 82
        1  191   2   1    0   0   47   6    0   0   8  0  9 83
```

Defragmenting Files

As files grow, contiguous space on disk is not available to accommodate new data, so files become fragmented. File fragmentation can reduce system performance because more I/O is required to read or write a file. Run the `defragment` utility using the `-v` and `-n` flags to display fragmentation statistics. Use the following format:

defragment -v -n *domain_name*

From the output you can determine how fragmented your file domain is and if it is a possible cause of poor system performance.

Balancing File Distribution

You can improve system performance if you distribute files evenly over all your volumes. Files that are distributed unevenly can degrade system performance.

You can use the `showfdmn` command to display the percent of used space on each volume in a multivolume file domain. Issue the `showfdmn` command using the following format:

showfdmn *domain_name*

When the percent of used space is uneven among the volumes, you can use the `balance` utility to redistribute the files among the volumes. Issue the balance command using the following format:

balance *domain_name*

When a volume is added to a domain with the `addvol` command, all the files of the file domain remain on the previously existing volume(s) and the new one is empty. Run the `balance` utility to even the file distribution.

Striping Files

AdvFS allows you to choose individual files to stripe across multiple volumes. If your system has very large files with heavy I/O requirements, consider striping these files across volumes so that I/O will be directed to more than one disk.

To stripe a file, you first create a new, zero-length (empty) file, issue the `stripe` command, and then write your file to the striped file. You can choose the number of volumes on which to stripe a file. Issue the `stripe` command using the following format:

stripe -n *volume_count file_name*

Migrating Files

You can use the `migrate` utility to move a heavily accessed file or selected pages of a file to another volume in the file domain. You can move the file to a specific volume or you can let the system choose.

To move an entire file to a specific volume, issue the migrate command using the following format:

migrate -s *source_volume_index* **-d** *destination_volume_index file_name*

Changing System Resources

You can change your file-system size in the following ways:

- Increase the size of a file domain by adding a volume with the `addvol` utility.
- Shrink a file domain by removing a volume with the `rmvol` command.
- Exchange volumes by first adding a new one, moving your files to it, and then removing the old.

These operations can take place on line while all filesets remain mounted. The `addvol` command completes in a few seconds. The `rmvol` command moves the data off the volume to be removed. The time it takes for the command to complete depends upon the amount of data stored on the volume and the load on the system.

Adding Volumes

When your file domain runs out of space, you can add a volume to the domain quickly and without interrupting your users. Use the `addvol` utility to increase the number of volumes within an existing file domain. For optimum performance, each volume you add should consist of the entire disk (typically, partition `c`). Do not add a volume containing any data you want to keep. When you run the `addvol` command, existing data on the added disk is destroyed.

If you are adding volumes because you plan to add a large number of files, see the Creating a Domain for a Large Number of Files section on page 47 before you add the volumes.

Adding volumes to a file domain does not affect the logical structure of the filesets within a file domain. You can add a volume to an active file domain while its filesets are mounted and in use. Run the `balance` utility after a new volume has been added to distribute files to it.

Removing Volumes

When you run the `rmvol` utility, the system automatically migrates the contents of the old volume to another volume in the domain. The logical structure of the filesets in a file domain is unaffected.

If you remove a volume that contains a stripe segment, the `rmvol` utility moves the segment to another volume that does not already contain a stripe segment of the same file. If all remaining volumes contain stripe segments, the system requests confirmation before the segment is moved to a volume that already contains a stripe segment of the file.

You can interrupt the `rmvol` process without damaging your file domain. Files already removed from the volume will remain in their new location. If the volume that has had the files removed does not allow new file allocations after the aborted `rmvol` operation, use the `chvol` command with the `-A` flag to reactivate the volume.

Exchanging Volumes

Without taking your system off line, you can replace a smaller volume with a larger one to provide more disk space, and you can exchange a slower device with a faster one to improve throughput. The procedure is as follows:

1. Add the new volume to the file domain using the `addvol` utility.
2. Remove the old volume with the `rmvol` utility. The system automatically migrates the contents of the old volume to another volume in the domain. The logical structure of the filesets in a file domain is unaffected.
3. If you want a specific file on the new volume, for example if it is heavily accessed and your new volume is fast, use the `migrate` utility to move the file to the new volume. You can also run the `balance` utility to distribute your files when the exchange is complete.

Handling Disk Problems

Back up your data regularly and frequently and watch for signs of impending disk failure. Removing files from a problem disk before it fails can prevent a lot of trouble.

Disk Failure

There is no particular message that will tell you that your disk is about to fail, but some warning messages may indicate potential problems. Run the `uerf` utility to print out the hardware-detected events. This report provides information that may help you identify some hardware-related problems.

Hardware problems cannot be repaired by your file system. If you start seeing unexplained errors for a volume, remove that volume from the file domain as soon as possible. If you can read data from your disk, you can remove the volume with the `rmvol` utility. If you wait and there is a disk failure, your metadata will be inaccessible, and it will be extremely difficult to access your data.

Domain Panic

When a log or metadata write error occurs, AdvFS will initiate a *domain panic*, rather than a system panic, on any non-root file domain. A domain panic prevents further access to the file domain but allows the filesets in the file domain to be unmounted.

When a domain panic occurs, a message is displayed in the following format:

AdvFS Domain Panic; Domain *name* **Id** *domain_Id*

For example:

```
AdvFS Domain Panic; Domain cybase_domain  Id  2dad7c28.0000dfbb
```

After a domain panic, use the `mount` command with the `-p` flag to list all mounted filesets. Then use the `umount` command to unmount all filesets in the file domain specified in the domain panic message. You can then take the necessary steps to correct the hardware problem.

After you have corrected the hardware problem, run the `verify` utility (the file domain structure checker) on the file domain before remounting any filesets. Running this utility will show you whether a log or metadata write error left you with any inconsistent files.

Restoring the File System

You can restore your files with the `restore` or `vrestore` command. You must use the command that corresponds to the dump utility that you used, either the `dump` or `vdump` command. You cannot mix UFS and AdvFS backup utilities.

Restoring the /etc/fdmns Directory

AdvFS must have a current `/etc/fdmns` directory in order to mount filesets. A missing or damaged `/etc/fdmns` directory prevents access to a file domain, but the data within the file domain remains intact. You can restore the `/etc/fdmns` directory from backup or you can recreate it.

If you have a current backup copy of the directory, it is preferable to restore the `/etc/fdmns` directory from backup. Any standard backup facility (`vdump`, `dump`, `tar`, or `cpio`) can back up the `/etc/fdmns` directory. To restore the directory, use the recovery procedure that is compatible with your backup process.

You can reconstruct the `/etc/fdmns` directory manually or with the `advscan` command. The procedure for reconstructing the `/etc/fdmns` directory is similar for both single-volume and multivolume file domains.

If you choose to reconstruct the directory manually, you must know the name of each file domain on your system and its associated volumes.

Reconstructing the /etc/fdmns Directory Using advscan

You can use the `advscan` command to determine which partitions on a disk or Logical Storage Manager (LSM) disk group are part of an AdvFS file domain. Then you can use the command to rebuild all or part of your `/etc/fdmns` domain.

The `advscan` command can:

- List partitions in the order they are found on disk.
- Scan all disks found in any `/etc/fdmns` domain.
- Recreate missing domain directories. The domain name is created from the device name.
- Fix the domain count and links if you specify a domain.
- Operate on LSM disk groups.

For each domain there are three numbers that must match for the AdvFS file system to operate properly:

- The number of physical partitions found by the `advscan` command that have the same domain ID
- The domain volume count (the number stored in the domain attributes table that specifies how many partitions the domain has)
- The number of `/etc/fdmns` links to the partitions, because each partition must be represented by a link

Inconsistencies can occur in these numbers in a number of ways and for a number of reasons.

In general, the `advscan` command treats the domain volume count as more reliable than the number of partitions or `/etc/fdmns` links. Table 14 lists anomalies, possible causes, and suggested corrections. In the table, a letter N represents the value that is expected to be consistent for the number of partitions, domain volume count, and number of links.

Table 14 Fileset Anomalies and Corrections

Number of Partitions	Domain Volume Count	Number of /etc/fdmns Links	Possible Cause	Corrective Action
>N	N	N	`addvol` terminated early	None; domain will mount with N volumes; rerun `addvol`
N	N	<N	`addvol` or `rmvol` terminated early or a link was deleted	Add the partition to `/etc/fdmns`
N	N	>N	Partition missing; either it was deleted or its device name was left out of the `advscan` search	Cannot correct
<N	N	N	Partition missing	Cannot correct
N	>N	N	Disk may be missing	Cannot correct
N	<N	N	Cause unknown	Try setting domain count to N

In the following example there are no missing file domains. The `advscan` command scans devices `rz0` and `rz5` for AdvFS partitions and finds nothing amiss. There are two partitions found (`rz0c` and `rz5c`), the domain volume count reports two, and there are two links entered in the `/etc/fdmns` directory.

```
# advscan rz0 rz5
Scanning disks  rz0 rz5
Found domains:
usr_domain
                Domain Id       2e09be37.0002eb40
                Created         Thu Jun 23 09:54:15 1994
                Domain volumes          2
                /etc/fdmns links        2
                Actual partitions found:
                                        rz0c
                                        rz5c
```

In the following example, directories that define the file domains that include `rz6` were removed from the `/etc/fdmns` directory. This means that the number of `/etc/fdmns` links, the number of partitions, and the domain volume counts are no longer equal.

The `advscan` command scans device `rz6` and recreates the missing file domains as follows:

1. A partition is found containing an AdvFS file domain. The domain volume count reports one, but there is no file-domain directory in the `/etc/fdmns` directory that contains this partition.
2. Another partition is found containing a different AdvFS file domain. The file domain volume count is also one. There is no file-domain directory that contains this partition.
3. No other AdvFS partitions are found. The domain volume counts and the number of partitions found match for the two discovered domains.
4. The `advscan` command creates directories for the two file domains in the `/etc/fdmns` directory.
5. The `advscan` command creates symbolic links for the devices in the `/etc/fdmns` file-domain directories.

The command and output are as follows:

```
# advscan -r rz6
Scanning disks  rz6
Found domains:
*unknown*
                Domain Id       2f2421ba.0008c1c0
                Created         Mon Jan 23 13:38:02 1995

                Domain volumes          1
                /etc/fdmns links        0

                Actual partitions found:
                                        rz6a*
*unknown*
                Domain Id       2f535f8c.000b6860
                Created         Tue Feb 28 09:38:20 1995

               Domain volumes          1
                /etc/fdmns links       0

                Actual partitions found:
                                        rz6b*

Creating /etc/fdmns/domain_rz6a/
        linking rz6a

Creating /etc/fdmns/domain_rz6b/
        linking rz6b
```

Reconstructing the /etc/fdmns Directory Manually

If you accidentally lose all or part of your `/etc/fdmns` directory, and you know which file domains and links are missing, you can reconstruct it manually.

The following example reconstructs the `/etc/fdmns` directory and two file domains where the names of the file domains are known. Each contains a single volume (or special device). Note that the order of creating the links in these examples does not matter. The file domains are:

`domain1` on `/dev/rz1c`
`domain2` on `/dev/rz2c`

To reconstruct the two single-volume file domains, enter:

```
# mkdir /etc/fdmns
# mkdir /etc/fdmns/domain1
# cd /etc/fdmns/domain1
# ln -s /dev/rz1c
# mkdir /etc/fdmns/domain2
# cd /etc/fdmns/domain2
# ln -s /dev/rz2c
```

The following example reconstructs one multivolume file domain. The `domain1` file domain contains the following three volumes:

```
/dev/rz1c
/dev/rz2c
/dev/rz3c
```

To reconstruct the multivolume file domain, issue the following:

```
# mkdir /etc/fdmns
# mkdir /etc/fdmns/domain1
# cd /etc/fdmns/domain1
# ln -s /dev/rz1c
# ln -s /dev/rz2c
# ln -s /dev/rz3c
```

Recovering from Failure of the Root Domain

A catastrophic failure of the disk containing your AdvFS root file domain requires that you recreate your root file domain and then restore the root file domain contents from your backup media.

The following example assumes that you are booting from the CD-ROM device DKA400, which is the installation Stand Alone System (SAS). The tape drive is `tz5`. Typing the `show device` command from the boot prompt shows `A/5/0` for device `TLZ06`. The root is being restored to device `rz1`, which is an `RZ25` disk:

1. Boot your system as stand-alone:

   ```
   b DKA400
   ```

2. Pick option:

   ```
   3) UNIX Shell
   ```

   You will now be at the # prompt in single-user mode.

3. Make the device special files for the tape and the disk:

   ```
   # MAKEDEV rz1
   # MAKEDEV tz5
   ```

4. Make the disk label:

   ```
   # disklabel -rw -t advfs rrz1a rz25
   ```

5. Create the root file domain and fileset. Note that if you have changed the root file domain name or fileset name, use the new names:

   ```
   # mkfdmn -r /dev/rz1a root_domain
   # mkfset root_domain root
   ```

6. Mount the newly created root domain and restore from tape using a restore utility compatible with your dump utility:

   ```
   # mount root_domain#root /mnt
   # cd /mnt
   # vrestore -x -D .
   ```

You can now boot your restored root domain.

Restoring a Multivolume usr Domain

To restore a multivolume /usr file system, the usr_domain file domain must first be reconstructed with all of its volumes before you restore the files. However, creating a multivolume file domain requires the addvol utility, and the addvol command will not run unless the License Management Facility (LMF) database, which resides in the /usr/sbin directory, is available. See the lmf(8) reference page for information.

On some systems the /var directory, where the LMF database resides, and the /usr directory are both located in the usr fileset. So the directory containing the license database must be recovered from usr fileset before the addvol command can be accessed. On some systems the /var directory is in a separate fileset. If this is the case, the addvol command can be recovered first and then can be used to add the volumes.

The following example restores a multivolume file domain where the /var directory and the /usr directory are both in the usr fileset in the usr_domain file domain consisting of the rz1g, rz2c, and rz3c volumes. The procedure assumes that the root file system has already been restored.

1. Mount the root fileset as read/write:

```
# mount -u /
```

2. Create usr_domain using the initial volume:

```
# rm -rf /etc/fdmns/usr_domain
# mkfdmn /dev/rz1g usr_domain
```

3. Create and mount the /usr and /var filesets:

```
# mkfset usr_domain usr
# mount -t advfs usr_domain#usr /usr
```

4. Create a soft link in /usr because that is where the lmf command looks for its database:

```
# ln -s /var /usr/var
```

5. Insert the /usr backup tape:

```
# cd /usr
# vrestore -vi
(/) add sbin/addvol
(/) add sbin/lmf
(/) add var/adm/lmf
(/) extract
(/) quit
```

6. Reset the license database:

```
# /usr/sbin/lmf reset
```

7. Add the extra volumes to `usr_domain`:

```
# /usr/sbin/addvol  /dev/rz2c usr_domain
# /usr/sbin/addvol  /dev/rz3c usr_domain
```

8. Do a full restore of the `/usr` backup:

```
# cd /usr
# vrestore -xv
```

The following example restores a multivolume file domain where the `/usr` and `/var` directories are in separate filesets in the same multivolume domain, `usr_domain`, consisting of `rz1g`, `rz2c`, and `rz3c`. This means that you must mount both the `/var` and the `/usr` backup tapes. The procedure assumes that the root file system has already been restored.

1. Mount the root fileset as read/write:

```
# mount -u /
```

2. Create `usr_domain` using the initial volume:

```
# rm -rf /etc/fdmns/usr_domain
# mkfdmn /dev/rz1g usr_domain
```

3. Create and mount the `/usr` and `/var` filesets:

```
# mkfset usr_domain usr
# mkfset usr_domain var
# mount -t advfs usr_domain#usr /usr
# mount -t advfs usr_domain#var /var
```

4. Insert the `/var` backup tape and restore from it:

```
# cd /var
# vrestore -vi
(/) add adm/lmf
(/) extract
(/) quit
```

5. Insert the `/usr` backup tape:

```
# cd /usr
# vrestore -vi
(/) add sbin/addvol
(/) add sbin/lmf
```

```
(/) extract
(/) quit
```

6 Reset the license database:

```
# /usr/sbin/lmf reset
```

7 Add the extra volumes to `usr_domain`:

```
# /usr/sbin/addvol  /dev/rz2c usr_domain
# /usr/sbin/addvol  /dev/rz3c usr_domain
```

8 Do a full restore of `/usr` backup:

```
# cd /usr
# vrestore -xv
```

9 Insert the `/var` backup tape and do a full restore of `/var` backup:

```
# cd /var
# vrestore -xv
```

Recovering from a System Crash

As each domain is mounted after a crash, it automatically runs recovery code that checks through the transaction log to ensure that any file-system operations that were occurring when the system crashed are either completed or backed out. This ensures that AdvFS metadata is in a consistent state after a crash.

Verifying File System Consistency

If you want to be sure that the metadata is consistent, you can run the `verify` command to verify the file-system structure. This utility checks on-disk structures such as the bitfile metadata table (BMT), the storage bitmaps, the tag directory, and the `frag` file for each fileset. It verifies that the directory structure is correct and that all directory entries reference a valid file and that all files have a directory entry.

Note The `verify` command replaces the `msfsck` command of earlier releases.

If the `verify` command is unable to mount a fileset due to the failure of a file domain, as a last resort run the command with the `-F` flag. This will cause the fileset to be mounted using the `-d` option of the `mount` command, which mounts the fileset without running recovery on the file domain. This will cause your file domain to be inconsistent because the file structure will not have been checked and made consistent.

The following example verifies the `domainx` file domain, which contains the filesets `setx` and `sety`:

```
# verify domainx
+++Domain verification+++
Domain Id 2f03b70a.000f1db0
Checking disks ...
Checking storage allocated on disk /dev/rz10g
Checking storage allocated on disk /dev/rz10a
Checking mcell list ...
Checking mcell position field ...
Checking tag directories ...
+++ Fileset verification +++
+++ Fileset setx +++
Checking frag file headers ...
Checking frag file type lists ...
Scanning directories and files ...
     1100
Scanning tags ...
     1100
Searching for lost files ...
     1100
+++ Fileset sety +++
Checking frag file headers ...
Checking frag file type lists ...
Scanning directories and files ...
     5100
Scanning tags ...
     5100
Searching for lost files ...
     5100
```

Displaying On-Disk Structures

Table 15 lists the on-disk structure dumping utilities that enable you to examine a file domain with suspected metadata corruption. The commands display raw data from the disk in a number of formats.

Table 15 On-Disk Structure Dumping Utilities

Command	Description
`shblk`	Displays unformatted disk blocks
`shfragbf`	Displays `frag` file information

(continued)

Table 15 On-Disk Structure Dumping Utilities (cont.)

Command	Description
`vbmtchain`	Displays mcells that describe metadata for a file
`vbmtpg`	Displays a formatted page of the bitfile metadata table (BMT)
`vfile`	Displays the contents of a file from an unmounted domain
`vfragpg`	Prints a single header page of a `frag` file
`vlogpg`	Displays a formatted page of the log
`vlsnpg`	Displays the logical sequence number (LSN) of a log page
`vtagpg`	Displays a formatted page of the tag directory

Moving an AdvFS Disk to an Undamaged Machine

If a machine has failed, it is possible to move disks containing AdvFS file domains to another computer running AdvFS. As explained in this section, you connect the disk(s) to the new machine and modify the `/etc/fdmns` directory so the new system will recognize the transferred volume(s).

Caution Do not use either the `addvol` command or the `mkfdmn` command to add the volumes to the new machine. Doing so will delete all data on your disk.

If you do not know what partitions your domains were on, you can add the disks on the new machine and run the `advscan` command, which may be able to recreate this information. You can also look at the disk label on the disk to see which partitions in the past have been made into AdvFS partitions. This will not tell you which partitions belong to which file domain.

For example, assume a system has a file domain, `testing_domain`, on two disks, `rz3` and `rz4`. This domain contains two filesets: `sample1_fset` and `sample2_fset`. These filesets are mounted on `/data/sample1` and `/data/sample2`. If the motherboard of the machine fails, you need to move the disks to another system. Assume you also want disks to use different SCSI ID numbers, for example: `rz6` and `rz8`, because `rz3` and `rz4` are already in use on the second computer.

Assume you already know that the file domain that you are moving had partitions `rz3c`, `rz4a`, `rz4b`, and `rz4g`. You would then take the following steps:

1 Shut down the working machine to which you are moving the disks.

2. Connect the disks from the bad machine to the good one. Configure the disk that was `rz3` on the old machine as `rz6` on the new one. Configure the old `rz4` as `rz8`.

3. Reboot. You do not need to reboot to SAS; multiuser mode works because you can complete the following steps while the system is running.

4. You may have to make special device nodes for the two new disks, `rz6` and `rz8`. For example, if `/dev/rz6c` doesn't exist, then as root user do the following:

```
# cd  /dev
# /dev/MAKEDEV rz6
```

   If necessary, do the same for `rz8`.

5. Modify your `/etc/fdmns` directory to include the information from the transferred domains:

```
# mkdir -p /etc/fdmns/testing_domain
# cd /etc/fdmns/testing_domain
# ln -s /dev/rz6c rz6c
# ln -s /dev/rz8a rz8a
# ln -s /dev/rz8b rz8b
# ln -s /dev/rz8g rz8g
# mkdir /data/sample1
# mkdir /data/sample2
```

6. Edit the `/etc/fstab` file and add the fileset mount-point information:

```
testing_domain#sample1_fset /data/sample1
testing_domain#sample2_fset /data/sample2
```

7. Mount the volumes:

```
# mount /data/sample1
# mount /data/sample2
```

   Note that if you run the `mkfdmn` command or the `addvol` command on partition `rz6c`, `rz8a`, `rz8b`, or `rz8g`, or an overlapping partition, you will destroy the data on the disk.

Appendix A
AdvFS Commands

This appendix summarizes the AdvFS commands.

AdvFS Base System Commands

The following tables list and describe each of the AdvFS commands available in the base portion of AdvFS. These commands are included with the Digital UNIX license; they do not require a layered product license. If you installed the reference page subset on Digital UNIX, you can access reference pages for each of these commands by issuing the `man` command.

Table 16 AdvFS Configuration Commands

Command	Description
`chfile`	Changes the attributes of a file
`chvol`	Changes the attributes of a volume
`mkfdmn`	Creates a file domain
`mkfset`	Creates a fileset within a file domain
`renamefset`	Renames an existing fileset
`rmfdmn`	Removes a file domain
`rmfset`	Removes a fileset from a file domain
`switchlog`	Moves the AdvFS log file to a different volume in a file domain

Table 17 AdvFS Information Display Commands

Command	Description
`advfsstat`	Displays file-system statistics
`ncheck`	Prints the tag and full path name for each file in the file system.
`showfdmn`	Displays the attributes of a file domain
`showfile`	Displays the attributes of a file
`showfsets`	Displays the attributes of filesets in a file domain

Table 18 AdvFS Backup Commands

Command	Description
`vdump`	Performs full and incremental fileset backup
`vrestore`	Restores files from backup media

Table 19 AdvFS Check and Repair Commands

Command	Description
`advscan`	Locates AdvFS partitions on disks
`mountlist`	Checks for mounted AdvFS filesets
`tag2name`	Prints the path name of a file given the tag number
`verify`	Checks for and repairs file-system inconsistencies

Table 20 AdvFS Quota Commands

Command	Description
chfsets	Changes the attributes of a fileset
edquota	Edits user and group quotas
quot	Summarizes fileset ownership
quota	Displays disk usage and limits by user or group
quotacheck	Checks file-system quota consistency
quotaoff	Turns quotas off
quotaon	Turns quotas on
repquota	Summarizes quotas for a file system

Table 21 On-Disk Structure Dumping Utilities

Command	Description
shblk	Displays unformatted disk blocks
shfragbf	Displays frag file information
vbmtchain	Displays mcells that describe metadata for a file
vbmtpg	Displays a formatted page of the bitfile metadata table (BMT)
vfile	Displays the contents of a file from an unmounted domain
vfragpg	Prints a single header page of a frag file
vlogpg	Displays a formatted page of the log
vlsnpg	Displays the logical sequence number (LSN) of a page of the log
vtagpg	Displays a formatted page of the tag directory

AdvFS Utilities Commands

Table 22 lists and describes the AdvFS Utilities commands. These commands require the optional AdvFS Utilities product license. If you installed the AdvFS Utilities reference page subset, you can access reference pages for each of these commands by issuing the `man` command.

Table 22 AdvFS Utilities Commands

Command	Description
`addvol`	Adds a volume to an existing file domain
`advfsd`	Starts the AdvFS graphical user interface (GUI) daemon
`balance`	Balances the percentage of used space between volumes
`clonefset`	Creates a read-only copy of a fileset
`defragment`	Makes the files in a file domain more contiguous
`dtadvfs`	Starts the AdvFS GUI
`migrate`	Moves a file to another volume in the file domain
`mktrashcan`	Attaches directories to a trashcan directory, which stores deleted files
`rmtrashcan`	Detaches a specified directory from a trashcan directory
`rmvol`	Removes a volume from an existing file domain
`shtrashcan`	Shows the trashcan directory, if any, that is attached to a specified directory
`stripe`	Interleaves storage allocation of a file across two or more volumes within a file domain

Appendix B
Converting File Systems

This appendix contains procedures to convert a `/usr` file system, the root file system, and a data file system to AdvFS. Also included are instructions for converting your entire system from AdvFS to UFS.

The methods provided here are guidelines; that is, they are suggestions that illustrate the process of conversion. Specific file names, tape drives, and disk partitions depend on your system.

Converting a /usr File System

By converting the `/usr` file system to AdvFS, you can reduce the amount of time your system is down after a system failure. During the initial installation of AdvFS, you can install `/usr` on AdvFS. If you do not, you can use one of the following methods:

- Use a backup tape
- Use an intermediate file
- Use a second disk

Using a Backup Tape

You can convert the `/usr` (UFS) file system to an equivalent AdvFS file system by backing up the existing file system to tape and restoring it to an AdvFS environment.

Requirements

- Root user privilege
- Backup device and media
- Five percent more disk space for the converted file system
- The Advanced File System installed on your system

Assumptions

- Existing UFS configuration:

File system `/usr`

Disk partition `/dev/rz3g`

- New AdvFS configuration:

File system `/usr`

Disk partition `/dev/rz3g`

File domain `usr_domain`

Fileset `usr`

Procedure

Use the following procedure as a guide for converting the file system:

1 Log in as root on the system containing the `/usr` file system.

2 Use the AdvFS `vdump` command to back up the `/usr` file system to `/dev/rmt0h`, the default tape drive. Enter the following sequence of commands:

```
# mt rewind
# cd /usr
# vdump -0 .
```

3 Edit the `/etc/fstab` file.

a. Search for the entry that mounts `/usr` as a UFS file system, such as:

```
/dev/rz3g           /usr        ufs rw 1 2
```

b. Replace the previous line with the following entry, which mounts `/usr` as an AdvFS file system:

```
usr_domain#usr      /usr        advfs  rw 1 0
```

4. Shut down the system by entering the following command:

   ```
   # shutdown -h now
   ```

5. Reboot the system in single-user mode. See the *Digital UNIX Guide to System Administration* for instructions on invoking single-user mode.

6. In single-user mode, mount the root file system as `rw`, create the `usr_domain` file domain, and create the `usr` fileset by entering the following sequence of commands:

   ```
   # mount -u /
   # mkfdmn /dev/rz3g usr_domain
   # mkfset usr_domain usr
   ```

7. Mount the `usr` fileset on the `/usr` directory by entering the following command:

   ```
   # mount -t advfs usr_domain#usr /usr
   ```

8. While there is no activity on the system, restore the `/usr` file system from tape to the `usr` fileset by entering the `vrestore` command:

   ```
   # vrestore -x -D /usr
   ```

9. Boot the system to multiuser mode. Once the system prompt returns, the converted `/usr` file system is ready to use.

Using an Intermediate File

You can convert the `/usr` (UFS) file system to the equivalent AdvFS file system by backing up the existing file system to a file and restoring it to an AdvFS environment.

Requirements

- Root user privilege.
- Disk space for an intermediate file. (The file system containing the intermediate file can be on the same disk or a different disk.)
- Five percent more disk space for the converted file system.
- The Advanced File System installed on your system.

Assumptions

- Existing UFS configuration:

  File system `/usr`

  Disk partition `/dev/rz3g`

  Intermediate file `/tmp/usr_bck`

- New AdvFS configuration:

  File system `/usr`

  Disk partition `/dev/rz3g`

  File domain `usr_domain`

  Fileset `usr`

Procedure

Use the following procedure as a guide for converting the `/usr` file system:

1 Log in as root on the system containing the `/usr` file system.

2 Use the AdvFS `vdump` command to back up the `/usr` file system to `/tmp/usr_bck`, the intermediate file. Enter the following sequence of commands:

```
# cd /usr
# vdump -0f /tmp/usr_bck /usr
```

3 Edit the `/etc/fstab` file.

a. Search for the entry that mounts `/usr` as a UFS file system:

```
/dev/rz3g           /usr        ufs rw 1 2
```

b. Replace the previous line with the following entry, which mounts `/usr` as an AdvFS file system:

```
usr_domain#usr      /usr        advfs  rw 1 0
```

4 Shut down the system by entering the following command:

```
# shutdown -h now
```

5 Reboot the system in single-user mode. See the *Digital UNIX Guide to System Administration* for instructions on invoking single-user mode.

6. In single-user mode, mount the root file system as `rw`, create the `usr_domain` file domain, and create the `usr` fileset by entering the following sequence of commands:

```
# mount -u /
# mkfdmn /dev/rz3g usr_domain
# mkfset usr_domain usr
```

7. Mount the `usr` fileset on the `/usr` directory by entering the following command:

```
# mount -t advfs usr_domain#usr /usr
```

8. While there is no activity on the system, restore the `/usr` file system from the intermediate file to the `usr` fileset by entering the following command:

```
# vrestore -xf /tmp/usr_bck -D /usr
```

9. Boot the system to multiuser mode. Once the system prompt returns, the converted `/usr` file system is ready to use.

Using a Second Disk

You can convert the `/usr` (UFS) file system on one disk to the equivalent `/usr` (AdvFS) file system on a different target disk.

Requirements

- Root user privilege
- A second disk with 5% more disk space for the converted file system
- The Advanced File System installed on your system

Assumptions

- Existing UFS configuration:

  File system `/usr`

  Disk partition `/dev/rz3g`

- New AdvFS configuration:

  File system `/usr`

  Disk partition `/dev/rz2c`

  Mount directory `/usr.advfs`

  File domain `usr_domain`

  Fileset `usr`

Procedure

Use the following procedure as a guide for converting the `/usr` file system:

1. Log in as root on the system containing the `/usr` file system.
2. Create a file domain and fileset by entering the following sequence of commands:

   ```
   # mkfdmn /dev/rz2c usr_domain
   # mkfset usr_domain usr
   ```

3. Create a mount-point directory and mount the new fileset on the directory by entering the following sequence of commands:

   ```
   # mkdir /usr.advfs
   # mount -t advfs usr_domain#usr /usr.advfs
   ```

4. Change to the `/usr` directory:

   ```
   # cd /usr
   ```

5. While there is no activity on the system, copy the contents of the UFS file system to the AdvFS file system by entering the following command:

   ```
   # vdump -0f - -D . | vrestore -xf - -D /usr.advfs
   ```

6. Edit the `/etc/fstab` file.

   a. Search for the entry that mounts `/usr` as a UFS file system, such as:

   ```
   /dev/rz3g           /usr        ufs rw 1 2
   ```

   b. Replace the previous line with the following entry, which mounts `/usr` as an AdvFS file system:

   ```
   usr_domain#usr    /usr        advfs  rw 1 0
   ```

7. Shut down and reboot the system. Once the system prompt returns, the converted `/usr` file system is ready to use.

Converting the Root File System

By converting the root file system to AdvFS you can boot your system from an AdvFS file domain and use AdvFS as the root (/) file system. The AdvFS root file domain must reside on a single disk. During initial installation of Digital UNIX, you can install root on AdvFS. If you do not, you can use the following method.

Note Before you begin the conversion, check the size of the existing UFS root partition. The target AdvFS root file domain can contain only one volume and must be large enough to accommodate the converted root file system.

Requirements

- Root user privilege.
- A second bootable disk. (You must use partition `a` or `c`.)
- The Advanced File System installed on your system.

Assumptions

- Existing UFS configuration:

  File system `root`

  Mount directory `/newroot`

  Disk partition `/dev/rz1a`

- New AdvFS configuration:

  File system `root`

  Mount directory `/newroot`

  Disk partition `/dev/rz2a`

  File domain `root_domain`

  Fileset `root`

Procedure

Use the following procedure as a guide for converting the root file system:

1. Log in as root on the system containing the root file system.
2. Create a file domain and fileset by entering the following commands:

```
# mkfdmn -r -t rz26 /dev/rz2a root_domain
# mkfset root_domain root
```

3. Create a mount-point directory and mount the new fileset on the directory by entering the following commands:

```
# mkdir /newroot
# mount  -t advfs root_domain#root /newroot
```

4. Restore the UFS root file system to the root fileset by entering the following command:

```
# vdump 0f - / | (cd /newroot; vrestore -xf -)
```

5. Make the disk with the root file domain a bootable disk using the following commands:

```
# disklabel -r /dev/rrz2a > /tmp/rz2label
# disklabel -t advfs -r -R /dev/rrz2a /tmp/rz2label rz26
```

6. Edit the `/etc/fstab` file on the AdvFS root fileset to indicate the new root entry.

   a. Search `/newroot/etc/fstab` for the entry that previously mounted root as a UFS file system, such as:

   ```
   /dev/rz1a   /  ufs rw 1 1
   ```

   b. Replace the previous line with the following entry, which mounts root as an AdvFS file system:

   ```
   root_domain#root  /  advfs rw 1 0
   ```

7. After editing is complete, shut down the system using the following command:

```
# shutdown -h now
```

8. Reset the boot default device, `BOOTDEF_DEV`, to point to the disk with the new root file domain. This procedure is hardware-specific. Refer to your hardware manual for instructions.

9. Reboot the system to enable the AdvFS root file system.

The converted root file system is ready to use.

Because the AdvFS root file domain is limited to one disk, you cannot use the `addvol` command to extend the root file domain.

Converting a Data File System

By converting your data file systems to AdvFS, you can eliminate lengthy reboots. Moreover, you can easily modify your file-system configurations to meet changing system requirements.

There are two different methods for converting data file systems from UFS to AdvFS:

- Use a backup tape
- Use a second system

Using a Backup Tape

You can convert a data (UFS) file system to the equivalent data (AdvFS) file system by backing up the existing file system to tape with the `vdump` command and restoring it with the `vrestore` command to an AdvFS environment.

Requirements

- Root user privilege
- Backup device and media
- Five percent more disk space for the converted file system
- The Advanced File System installed on your system

Assumptions

- Existing UFS configuration:

  File system `/staff2`

  Mount directory `/staff2`

  Disk partition `/dev/rz2c`

- New AdvFS configuration:

  File system `/staff2`

  Disk partition `/dev/rz2c`

  File domain `staff_domain`

  Fileset `staff2`

Procedure

Use the following procedure as a guide for converting the /staff2 file system:

1. Log in as root on the system containing the /staff2 file system.
2. Use the AdvFS vdump command to back up the /staff2 file system to /dev/rmt0h, the default tape drive. Enter the following sequence of commands:

```
# mt rewind
# mount /staff2
# cd /staff2
# vdump  -0 .
# unmount /staff2
```

3. Create a file domain and fileset by entering the following sequence of commands. After creating the file domain, you need to confirm that you want to write over the UFS file system.

```
# mkfdmn /dev/rz2c staff_domain
Partition(s) which overlap /dev/rz2c are marked in use.
If you continue with the operation you can possibley destry existing data.
CONTINUE? [y/n]y
# mkfset staff_domain staff2
```

4. Create a mount-point directory and mount the new fileset on the directory by entering the following sequence of commands:

```
# mount  -t advfs staff_domain#staff2 /staff2
```

5. Restore the /staff2 file system from tape to the staff2 fileset by entering the following command:

```
# vrestore  -x  -D /staff2
```

6. Edit the /etc/fstab file.
   a. Search for the entry that previously mounted /staff2 as a UFS file system:

```
/dev/rz2c          /staff2        ufs rw 1 2
```

   b. Replace the previous line with the following entry, which mounts /staff2 as an AdvFS file system:

```
staff_domain#staff2     /staff2      advfs  rw 1 0
```

The converted /staff2 file system is ready to use.

Using a Second System

You can transfer an existing data file system to a new system, then you can convert the file system to AdvFS.

Requirements

- Two systems: one system that supports the `tar` utility and one Digital UNIX operating system
- Root user privilege on the target system
- Five percent more disk space for the converted file system
- The Advanced File System installed on the target system

Assumptions

- Existing UFS configuration:

  File system `/staff4`
- New AdvFS configuration:

  File system `/staff4`

  Disk partition `/dev/rz2c`

  Mount directory `/staff4`

  File domain `staff_domain`

  Fileset `staff4`

Procedure

Use the following procedure as a guide for converting the `staff4` file system:

1. Log in to the system containing the /`staff4` file system and back up the file system using the following command:

   ```
   # tar c /staff4
   ```

2. Log in as root user on the target system.

3. Create a fileset in the `staff_domain` file domain by entering the following command:

   ```
   # mkfset staff_domain staff4
   ```

4. Create a mount-point directory and mount the new fileset on the directory by entering the following sequence of commands:

   ```
   # mkdir /staff4
   # mount  -t advfs staff_domain#staff4 /staff4
   ```

5. Restore the `/staff4` file system from the default tape drive, `/dev/rmt0h`, by entering the following sequence of commands:

   ```
   # mt rewind
   # tar x /staff4
   ```

6. Edit the `/etc/fstab` file. Add the following line, which mounts `/staff4` as an AdvFS file system:

   ```
   staff_domain#staff4     /staff4         advfs  rw
   ```

The `staff_domain` file domain now includes the `staff4` fileset, which is ready to use.

Converting from AdvFS to UFS

Converting your entire system from AdvFS to UFS is a multistep process. You first convert the AdvFS root file system to UFS. Then you convert each AdvFS fileset to a UFS file system.

Converting the Root to UFS

To convert the root file system, you must mount a UFS disk while your AdvFS root fileset is mounted. Use the following procedure as a guideline for converting your file system:

1. Log in as root user.

2. Create a UFS file system:

```
# newfs  /dev/rz2a  rz26
```

3. Create a mount-point directory and mount the UFS file system:

```
# mkdir  /newroot
# mount  -t ufs /dev/rz2a /newroot
```

4. Restore the AdvFS root file system to the `/dev/rz2a` UFS file system:

```
# vdump  -0f -  /  |  (cd /newroot; vrestore -xf -)
```

5. Make the disk containing the UFS file system a bootable disk:

```
# disklabel -r /dev/rrz2a > /tmp/rz21abel
# disklabel -t ufs -r -R /dev/rrz2a /tmp/rz2label rz26
```

6. Edit the `/etc/fstab` file on the UFS file system to refer to the new root entry.
   a. Search `/newroot/etc/fstab` for the entry previously mounted as root for the AdvFS file system:

```
root_domain#root    /  advfs   rw 1 0
```

   b. Replace the previous line with the following entry which mounts root as a UFS file system:

```
/dev/rz2a   /    ufs  rw 1 1
```

7. Shut down the system by entering the following command:

```
# shutdown -h now
```

8. Reset the boot default device, `BOOTDEF_DEV`, to the new root disk. (Refer to your hardware manual for specific information.)
9. Reboot the system to enable the UFS root file system.

Converting a Fileset to UFS

Once the root file system is converted to UFS, you can convert your filesets. The following example assumes your AdvFS file domain contains one volume and only one fileset. If the AdvFS file domain contains multiple filesets, then you must create a separate UFS file system for each fileset.

Caution Be sure you perform a full backup on all AdvFS filesets before you start the conversion.

1. Make a backup of the AdvFS fileset to your backup device:

```
# vdump -0f /dev/rmt0a /staff2
```

2. Unmount the fileset:

```
# umount /staff2
```

3. Delete the fileset:

```
# rmfset staff_domain staff2
```

4. Remove the file domain using the `rmfdmn` command:

```
# rmfdmn staff_domain
```

5. Create the UFS file system for the specified disk type:

```
# newfs /dev/rz2c rz26
```

6. Edit your `/etc/fstab` file.
   a. Search for the entry that mounts `/staff2` as an AdvFS fileset, such as:

      ```
      staff_domain#staff2 /staff2 advfs rw 1 0
      ```

   b. Replace the previous line with the following entry, which mounts `/staff2` as a UFS file system:

      ```
      /dev/rz2c /staff2 ufs rw 1 2
      ```

7. Mount the UFS file system:

```
# mount -t ufs /dev/rz2c /staff2
```

8. Use the `vrestore` command to load the files from the backup into the UFS file system:

```
# vrestore -xvf /dev/rmt0a -D /staff2
```

If your file domain contains multiple volumes, you must verify that the disk space allocated to a fileset will not exceed the limit of the UFS file system disk partition. You may need to create multiple UFS file systems to hold the filesets in the file domain.

Glossary

/etc/fdmns directory

A directory that defines the file domains by providing a subdirectory for each file domain created on the system.

/etc/fstab file

A file that identifies filesets that are to be mounted at system reboot.

balance

To even the distribution of files between volumes of a file domain.

bitfile metadata table

See BMT.

block

A 512-byte unit of disk storage.

BMT

An array of 8-kilobyte pages, each with a header and an array of mcells. A BMT contains all metadata for all files with storage on a volume.

buffer cache

The area of memory that contains the blocks of data waiting to be written to disk.

checksum

See exclusive-or (XOR).

clone fileset

A read-only copy of a fileset that is created to capture fileset data at a particular time. The contents of the clone fileset can be backed up while the original fileset remains available to users.

contiguous

Storage that is physically adjacent on a disk volume.

defragment

To make files and free space in a file domain more contiguous.

dirty

Data that has been written by the application, but the file system has cached it in memory so it has not yet been written to disk.

domain panic

A condition that prevents further access to the file domain but allows the filesets in the file domain to be unmounted. AdvFS initiates a domain panic when a log or metadata write error occurs.

exclusive-or (XOR)

Blocks created during tape backup for error recovery.

extent

One or more contiguous blocks on disk. Contiguous in this context means physically adjacent on a volume.

file domain

A named pool of storage that contains one or more volumes. Each file domain must have at least one fileset.

file domain identifier

A set of numbers that identify the file domain to the system.

file extent

See extent.

fileset

A hierarchy of directory and file names. It represents a mountable portion of the directory hierarchy of the AdvFS file system.

fileset quota

A quota that limits the amount of disk storage that a fileset can consume or the number of files a fileset can contain.

frag file

A file that allocates storage for files that are less than 8-kB in size (one page). Using fragments reduces the amount of wasted disk space.

file fragment

Created when a file uses only part of the last page (less than 8 kB) of file storage allocated or has a total size of less than 8 kB.

grace period

The period of time a quota's soft limit can be exceeded as long as the hard limit is not exceeded.

GUI

A graphical user interface.

hard limit

The quota limit for disk block usage or number of files that cannot be exceeded.

HSM

The Hierarchical Storage Manager (Layered Product). HSM provides large amounts of tertiary storage for systems where speed of access is not important.

inode

An identifier for a UFS file that is like an AdvFS file domain tag.

kernel

The portion of an operating system that controls process scheduling and system resources such as virtual memory and storage devices.

Logical Storage Manager

See LSM.

LSM

The Logical Storage Manager (Layered Product). LSM is a volume management system that shadows volumes and provides volume-level striping.

metadata

File structure information such as file attributes, extent maps, and fileset attributes.

migrate

To move files from one volume to another within a file domain.

miscellaneous metadata bitfile

Maps areas of the volume that does not represent AdvFS metadata, such as the disk label and boot blocks.

NetWorker

The NetWorker Save and Restore Solution. NetWorker provides scheduled, online automated back up.

page

An allocation of 8 kB of contiguous disk space (16 blocks).

product authorization key (PAK)

License to access Digital Equipment Corporation software.

quota file

A file that keeps track of number of files, disk block usage, and grace period per user ID or per group ID. Fileset quota information is stored within the fileset.

root tag directory

Defines the location of all filesets in a file domain. Each file domain has one.

saveset

A collection of blocks created to save AdvFS backup information.

soft limit

The quota value beyond which disk block usage or number of files is allowed only during the grace period.

storage bitmap

Tracks free and allocated disk space. Each volume in a file domain contains one.

stripe

Allocates successive sections of files across successive volumes in a file domain.

transaction log

The log that records changes to metadata before the changes are written to disk. At regular intervals the changes in the transaction log are written to disk. The log is circular and is eventually overwritten.

trashcan

The directory that contains the most recently deleted files from an attached directory. The trashcan directory is set up by each user for user files.

volume

Anything that behaves like a UNIX block device. This can be a disk, disk partition, or logical volume.

write-ahead logging

The process where the modifications to the file-structure information are completely written to a transaction log before the actual changes are written to disk. This process is used to ensure file-system consistency in the event of a system failure.

XOR

See exclusive-or (XOR).

Index

D

E

F

G

H

I

L

M

N

O

P

Q

R

S

T

U

V

Ordering Additional Information

You can order additional user information electronically, by telephone, or by mailing an order. If you need help with your order, call 800-343-4040.

Electronic Orders

To place an order at the Electronic Store, dial 800-DEC-DEMO (800-332-3366) using a 1200- or 2400-bps modem. If you need assistance using the Electronic Store, call 800-DIGITAL (800-344-4825).

Telephone and Direct Mail Orders

Your Location	Call	Contact
Continental USA, Alaska, or Hawaii	800-DIGITAL	Digital Equipment Corporation PO Box CS2008 Nashua, NH 03061
Puerto Rico	809-754-7575	Local Digital subsidiary
Canada	800-267-6215	Digital Equipment of Canada Attn: DECdirect Operation KAO2/2 PO Box 13000 100 Herzberg Road Kanata, Ontario, Canada K2K 2A6
International		Local Digital subsidiary or approved distributor
Internal[1]		SSB Order Processing – NQO/V19 or U.S. Software Supply Business Digital Equipment Corporation 10 Cotton Road Nashua, NH 03063-1260

[1] For internal orders, you must submit on Internal Software Order Form (EN-01740-07).

How Did We Do?

Please take a moment to let us know whether the online and paper documentation that we provided with this product is useful to you. We are particularly interested in comments about clarity, organization, figures, examples (or the lack thereof), the index, page layout, and, of course, accuracy.

When you send a comment, be sure to provide ample background information to help us locate the right source files. Include the product name and version, the document name, and the page number.

Internet address:	doc_comments@zso.dec.com
Fax numbers:	800-DEC-FAXX (800-332-3299) 206-865-8890
Mailing address:	Publications Manager Digital Equipment Corporation 14475 NE 24th Street Bellevue, WA 98007