Table of Contents

Chapter 1. Summary of Changes 1

Chapter 2. Supported AFS Systems 7
2.1. New AFS Systems 7
2.2. Enhanced AFS Systems 7
2.3. Obsolete AFS Systems 7
2.4. Supported AFS Systems 8

Chapter 3. Upgrading to AFS 3.4 11
3.1. Warnings About Operating System Upgrades 12
3.2. Prerequisites for Upgrading 12
3.3. Notes on Upgrading from AFS 3.3 to AFS 3.4 13
3.3.1. Notes for AFS Cells Using HP-UX 9.0 and OSF/1 2.0 14
3.4. Procedures for Upgrading from AFS 3.3 to AFS 3.4 14
3.4.1. Installing Binary Files 15
3.4.2. Upgrading Database Servers 16
3.4.3. Upgrading File Servers 17
3.4.4. Upgrading the Cache Managers on AFS Clients 17
3.4.5. Cleaning Up After Upgrading to AFS 3.4 18
3.5. Procedures for Upgrading from AFS 3.2 to AFS 3.4 19
3.5.1. Installing Binary Files 19
3.5.2. Upgrading Servers 19
3.5.3. Upgrading the Cache Managers on AFS Clients 21
3.6. Procedures for Downgrading from AFS 3.4 to AFS 3.3 22
3.6.1. Installing Binary Files 22
3.6.2. Downgrading Servers in a Cell with Multi-Homed File Servers 23
3.6.3. Downgrading Servers in a Cell Without Multi-Homed File Servers 24
3.6.4. Downgrading the Cache Managers on AFS Clients 25

Chapter 4. Authentication 27
4.1. Kerberos Support for the kaserver Process 27
4.2. Changes to the AFS login Program 27
4.3. Changes to the AIX login Program 27
4.3.1. Support for # and ! Entries in the /etc/passwd File 28
4.3.2. Support for Alternative Authentication Programs with AIX 4.1 28

4.4. Changes to the Alpha OSF/1 login Program 29
4.5. Changes to the Solaris login Program 30
4.6. Support for Secondary Authentication on AIX 3.2 30

Chapter 5. The Backup System 31
5.1. New Backup Configuration File 31
5.1.1. MOUNT Parameter 32
5.1.2. UNMOUNT Parameter 33
5.1.3. ASK Parameter 33
5.1.4. AUTOQUERY Parameter 34
5.1.5. NAME_CHECK Parameter 34
5.1.6. BUFFERSIZE Parameter 35
5.1.7. FILE Parameter 35
5.2. Improved Error Messages and Error Handling 36
5.3. Permanent Tape Names 36
5.4. The Tape Coordinator Can Scan Any Tape 37
5.5. Modification to Backup Prompting 37
5.6. The backup Command 37
5.6.1. The -localauth Flag and -cell Argument (New) 37
5.6.2. The backup interactive Command (New) 38
5.6.3. The backup volsetrestore Command (New) 39
5.6.4. The backup labeltape Command 45
5.6.5. The backup readlabel Command 45

Chapter 6. The bos Commands 47
6.1. The bos addkey Command 47
6.2. The bos status Command 48

Chapter 7. The fs Commands 49
7.1. Setting and Getting Cache Manager Server Preferences 49
7.1.1. The fs setserverprefs Command 51
7.1.2. The fs getserverprefs Command 56
7.2. The fs storebehind Command (New) 58
7.3. The fs checkservers Command 60
7.4. The fs exportafs Command 61
7.5. The -cell Argument on fs Commands 63
7.6. The fs copyacl, fs listacl, and fs setacl Commands 63
7.7. The fs newcell Command 63

Chapter 8. The fstrace Commands (New) 65
8.1. Setting the State of an Event Set 66
8.2. Changing the Size of Trace Logs 67
8.3. Dumping the Contents of Trace Logs 67
8.4. Listing Information about Trace Logs 69
8.5. Listing Information about Event Sets 70
8.6. Clearing Trace Logs 71
8.7. Getting Help for Command Usage 71

Chapter 9. The kas Commands 73
9.1. The kas Command Ticket Lifetime 73
9.2. The kas examine Command 73
9.3. The kas setfields Command 73

Chapter 10. The package Command 75
10.1. The package Command Allows Relative Pathnames 75
10.2. Changes to minor device number Argument 75
10.3. Changes to owner name Argument 76
10.4. Changes to group name Argument 76

Chapter 11. The uss Commands 77
11.1. The uss bulk Command 77
11.2. The uss add Command 78

Chapter 12. The vos Commands 79
12.1. The vos restore Command 79
12.2. The vos backup Command 81
12.3. The vos create Command 81
12.4. The vos release Command 81
12.5. The vos rename Command 82
12.6. The vos syncserv Command 82
12.7. Restoring from a Named Pipe 82
12.8. The vos changeaddr Command 82

Chapter 13. Miscellaneous AFS Commands 85
13.1. The afsd Command 85
13.1.1. New Cache Manager Daemon 86
13.1.2. AFS Compares Cache Size to Partition Size 86
13.1.3. Correct Interpretation of White Space in the cacheinfo File 86
13.1.4. The -waitclose Flag Has No Effect on the afsd Command 87
13.2. The butc Command 87
13.2.1. New -localauth Flag 87
13.2.2. Change to the -debuglevel Argument 88
13.3. The fileserver Command 88
13.4. The klog Command 89
13.5. The knfs Command 89
13.6. The pagsh Command 90
13.7. The salvager Command 90
13.8. The scout Command 90
13.9. The upclient Command 91
13.10. The vldb_convert Command 91
13.11. The vlserver Command 91
13.11.1. Change in Values for -p Argument 91
13.11.2. New Log File for the vlserver Process 92
13.12. The volserver Command 93
13.12.1. Change to the -log Flag 93
13.12.2. New -p Argument 93
13.13. The dlog Command 94
13.14. The dpass Command 94
13.15. The up Command 94
13.16. The xstat Utility 95
13.17. The -help Flag 95

Chapter 14. Additional Functional Changes 97
14.1. Multi-Homed File Servers 97
14.2. Support for Unlinking Open Files 98
14.3. Converting OSF/1 Data on DEC Alpha Machines 99
14.4. The fileserver Process Checks for FORCESALVAGE Flag 100
14.5. AFS Supports 8-Bit Characters 101
14.6. AFS Supports Partitions Larger Than 2 GB 101
14.7. New CellServDB Error Message 101

14.8. Cache Manager May Show Volume Name 102
14.9. New Rights for the system:administrators Group 102
14.10. Improved Database Access During Elections 102
14.11. Increase in Server Partitions 102
14.12. Additional AIX 3.2 Support 103
14.13. Changes to the NFS/AFS Translator 103
14.14. Improved Networking Support 103
14.15. Modification to fsync() 103
14.16. Version Strings in Binaries 104
14.17. File Locking Operations 104

Chapter 15. Bug Fixes 105

Chapter 16. Documentation Corrections 111

Index 113

1. Summary of Changes

These release notes describe the changes and new features included in the AFS® 3.4 release. These changes are not documented in the *AFS System Administrator's Guide*, *AFS Command Reference Manual*, or *AFS User's Guide*, although in some cases you are referred to the AFS documentation set for more information. This information is included in the sections titled "**AFS 3.4 Changes**."

Note that this document also contains AFS 3.3 release note information, which has not been incorporated into the AFS documentation set. This information is included in the sections titled "**AFS 3.3 Changes**."

AFS 3.4 supports multi-homed file servers for the first time; refer to the section on multi-homed file servers in Chapter 14 for more information. The Backup System includes many changes to enhance performance and produce clearer status and error messages; refer to the information on the AFS Backup System in Chapter 5. Many changes are introduced in AFS 3.4. Several changes have been incorporated to fix bugs and improve performance of the Cache Manager, database servers, file servers, and the NFS/AFS Translator.

The AFS 3.4 release includes the following interface and functional changes:

Chapter 2 — **Supported AFS Systems**

AFS 3.4 provides support for the following systems:

- AIX 3.2 and 4.1 on an IBM RS/6000
- Alpha OSF/1 2.0 and 3.0 on a DEC Alpha
- HP-UX 9.0, 9.01, and 9.03 on a Hewlett-Packard 9000 Series 700
- HP-UX 9.0, 9.01, and 9.03 on a Hewlett-Packard 9000 Series 800
- HP-UX 9.0, 9.01, and 9.03 on a Hewlett-Packard 9000 Series 800 MP
- IRIX 5.2 and 5.3 on a Silicon Graphics IRIS Indigo or Challenge
- Solaris 2.3 and 2.4 on a Sun 4 (except SPARCstations)
- Solaris 2.3 and 2.4 on a Sun SPARCstation (except Sun 4)
- Solaris 2.3 and 2.4 on a Sun 600 Series MP SPARCstation
- SunOS 4.1.1, 4.1.2, and 4.1.3 on a Sun 4 (except SPARCstations)
- SunOS 4.1.1, 4.1.2, and 4.1.3 on a Sun SPARCstation (except Sun 4)
- SunOS 4.1.2 and 4.1.3 on a Sun 600 Series MP SPARCstation

— Ultrix 4.3, 4.3a, and 4.4 on a DECstation 2100, 3100, or 5000 (single processor only)

Chapter 3 — Upgrading to AFS 3.4

Provides procedures and instructions for

- Upgrading your AFS cell from AFS 3.3 to AFS 3.4
- Upgrading your AFS cell from AFS 3.2 to AFS 3.4
- Downgrading your AFS cell from AFS 3.4 to AFS 3.3

Chapter 4 — Authentication

Changes to AFS authentication and login programs, including

- Kerberos support through UDP ports 88 and 750
- Failed login message in *syslog* file after sixth attempt with the AFS **login** program
- Support for the **#** and **!** character entries in the */etc/passwd* file with the AIX **login** program
- An alternative authentication program, **afs_dynamic_auth**, with the AIX 4.1 **login** program
- Several changes to the Alpha OSF/1 **login** program, including support for the **x** character on password entries
- Support for the */etc/default/login* file and variables in the Solaris **login** program

Chapter 5 — The Backup System

Several major and minor changes to the Backup System, including

- A new backup configuration file to provide jukebox and stacker tape support
- A new **backup interactive** command for placing you into interactive mode for all commands in the **backup** command suite
- A new **backup volsetrestore** command for restoring a chosen set of volumes to their original location or taking a list of volumes from a file and restoring them to a server and partition
- New **-localauth** and **-cell** options added on all **backup** commands in non-interactive mode
- Improved error messages and error handling
- Scanning of any tape by the **butc** command
- Support for permanent names on tapes with the **backup labeltape** and **backup readlabel** command

Chapter 6 — **The bos Commands**	A change to the **bos addkey** command that prevents the misuse of an existing key version number and prompts you to enter the key a second time for verification.
Chapter 7 — **The fs Commands**	Several major and minor changes to the **fs** command suite, including • A new **fs storebehind** command that sets asynchronous writes to file servers on a per-cache manager basis • The addition of VL server preferences for the **fs getserverprefs** and **fs setserverprefs** commands • A new argument structure for the **fs exportafs** command so that command arguments now only support values of *on* and *off*
Chapter 8 — **The fstrace Commands**	A new command suite, **fstrace**, consisting of seven commands that are used by the system administrator to diagnose problems within the AFS kernel. The new **fstrace** command suite includes • The **fstrace setset** command sets an event set's state. • The **fstrace setlog** command changes the size of trace logs. • The **fstrace dump** command dumps the contents of the trace logs. • The **fstrace lslog** command lists information about trace logs. • The **fstrace lsset** command lists information about event sets. • The **fstrace clear** command clears trace logs.
Chapter 9 — **The kas Commands**	Several changes to the **kas** command suite, including • An increase in the ticket length associated with the ticket received from **kas** commands • An update to the **kas examine** command to show the value of the **kas setfields -reuse** argument • Updates to the **kas setfields** command to prevent unauthorized users from gaining access to an account
Chapter 10 — **The package Command**	Several changes to the **package** command and configuration lines, including • Support for relative pathnames on the "**L**" configuration line • Support for hexadecimal, octal, and decimal numbers on the *minor device* argument • An enhancement to locate the owner id in the local file system when the owner name specified by the *owner name* argument cannot be found • An enhancement to locate the group id in the local file system when the group name specified by the *group name* argument cannot be found

Chapter 11 — **The uss Commands**

A new option, **-pipe**, has been added to the **uss bulk** command to assist you in running batch jobs without displaying the password prompt.

Chapter 12 — **The vos Commands**

Several major and minor changes to the **vos** command suite, including

- A new **-maxquota** argument for the **vos create** command to allow you to create the volume and set the quota in the same step
- An enhancement to the **vos release** command to update up to half of the ReadOnly copies simultaneously and to account for files deleted from the ReadWrite volume by not replicating them
- Updates to the **vos restore** command to check if a specified volume exists on the specified partition and a new **-overwrite** argument for situations where standard input (stdin) is redirected from a pipe or a file specified by the **-file** argument
- An update to the **vos syncserv** command to check all servers and continue checking remaining servers if any of the servers are not contacted

Chapter 13 — **Miscellaneous AFS Commands**

Several major and minor changes to miscellaneous AFS commands, including

- A new Cache Manager "cache-truncate" daemon for flushing the AFS cache
- Improvements to the **afsd** command, which now compares the cache size to the partition size upon starting the **afsd** command to ensure that the cache size is not greater than 95% of the partition size and interprets any extra white space found while reading the */usr/vice/etc/cacheinfo* file correctly without failing
- Updates to the **butc** command to support the **-localauth** argument, which assigns tokens that never expire, and two new legal values for the **-debuglevel** argument
- Two new arguments for the **salvager** command
- A new Volume Location Database (VLDB) conversion program for the **vldb_convert** command
- A new log file for the **vlserver** process
- A new **-p** argument for the **volserver** command

Chapter 14 — **Additional Functional Changes**	Several functional changes have been made, including • Support for multi-home file servers • Support for unlinking open files • Program for converting OSF/1 data on DEC Alpha machines • Support for UFS partitions larger than 2 GB • Support for 256 partitions per server • Support for 8-bit international characters • Improved database access during Ubik elections • Detection of a *FORCESALVAGE* flag by the **fileserver** process • Additional support for AIX 3.2
Chapter 15 — **Bug Fixes**	Several comments describing bugs fixed in AFS 3.4 are included in this chapter.
Chapter 16 — **Documentation Corrections**	Several comments describing documentation corrections are included in this chapter.

2. Supported AFS Systems

AFS 3.4 supports a number of new systems while dropping support for some obsolete systems. A complete list of the supported systems appears in a table at the end of this chapter.

2.1. New AFS Systems

The following supported systems are new for AFS 3.4:

- DEC Alpha running OSF/1 2.0
- DEC Alpha running OSF/1 3.0
- IBM RS-6000 running AIX 4.1
- Silicon Graphics IRIS Indigo or Challenge running IRIX 5.2
- Silicon Graphics IRIS Indigo or Challenge running IRIX 5.3
- Sun 4 (except SPARCstations) running Solaris 2.4
- Sun SPARCstations running Solaris 2.4
- Sun 600 Series MP SPARCstations running Solaris 2.4

2.2. Enhanced AFS Systems

The following supported system is enhanced for AFS 3.4:

- Hewlett-Packard 9000 Series 800 MP (multi-processor) running HP-UX 9.0

2.3. Obsolete AFS Systems

The following systems that were supported in AFS 3.3 or AFS 3.3a are not supported in AFS 3.4:

- Hewlett-Packard 9000 Series 300/400 running HP-UX 9.0
- NeXT (68030 or 68040 systems) running NeXT OS Release 3.0
- IBM-RT/PC running AIX 2.2.1
- IBM-RT/PC running AOS Release 4
- Silicon Graphics IRIS Indigo or Challenge running IRIX 5.0.1 (client only)
- Silicon Graphics IRIS Indigo or Challenge running IRIX 5.1 or 5.1.1
- Sun 3 (68020 systems) running SunOS 4.1.1, 4.1.2, or 4.1.3
- Sun 3 (68030 systems) running SunOS 4.1.1, 4.1.2, or 4.1.3

- Sun 4 (except SPARCstations) running SunOS 5.2
- Sun SPARCstations running Solaris 2.2
- Sun 600 Series MP SPARCstations running Solaris 2.2
- VAX systems running Ultrix 4.3

2.4. Supported AFS Systems

Table 2-1 lists all of the systems supported in AFS 3.4. As in previous versions of AFS, you must use the system names shown in the table if you wish to use the *@sys* variable in pathnames (as discussed in Chapter 2 of the *AFS System Administrator's Guide*).

Table 2-1. Supported AFS Systems

System Name	Machines	Operating Systems
Digital Machines		
alpha_osf20	DEC Alpha	Alpha OSF/1 2.0
alpha_osf30	DEC Alpha	Alpha OSF/1 3.0
pmax_ul43	DECstation 2100, 3100, or 5000 (single processor only)	Ultrix 4.3
pmax_ul43a	DECstation 2100, 3100, or 5000 (single processor only)	Ultrix 4.3a or 4.4
Hewlett-Packard Machines		
hp700_ux90	Hewlett-Packard 9000 Series 700	HP-UX 9.0, 9.01, or 9.03
hp800_ux90	Hewlett-Packard 9000 Series 800	HP-UX 9.0
hp800_ux90	Hewlett-Packard 9000 Series 800 MP	HP-UX 9.0
IBM Machines		
rs_aix32	IBM RS/6000	AIX 3.2
rs_aix41	IBM RS/6000	AIX 4.1
Silicon Graphics Machines		
sgi_52	Silicon Graphics IRIS Indigo or Challenge	IRIX 5.2
sgi_53	Silicon Graphics IRIS Indigo or Challenge	IRIX 5.3
Sun Machines		
sun4_411	Sun 4 (except SPARCstations)	SunOS 4.1.1, 4.1.2, or 4.1.3
sun4c_411	Sun SPARCstations	SunOS 4.1.1, 4.1.2, or 4.1.3
sun4m_412	Sun 600 Series MP SPARCstations	SunOS 4.1.2 or 4.1.3

Table 2-1. (Continued) Supported AFS Systems

sun4_53	Sun 4 (except SPARCstations)	Solaris 2.3
sun4_54	Sun 4 (except SPARCstations)	Solaris 2.4
sun4c_53	Sun SPARCstations	Solaris 2.3
sun4c_54	Sun SPARCstations	Solaris 2.4
sun4m_53	Sun 600 Series MP SPARCstations	Solaris 2.3
sun4m_54	Sun 600 Series MP SPARCstations	Solaris 2.4

3. Upgrading to AFS 3.4

This chapter provides procedures for upgrading your cell to AFS 3.4. Please read all of the instructions before beginning the procedures. If you are installing AFS for the first time, skip this chapter and refer to the *AFS Installation Guide* for detailed instructions on how to install AFS.

The following sections provide information for upgrading your cell from AFS 3.3 to AFS 3.4:

- Warnings About Operating System Upgrades
- Prerequisites for Upgrading
- Notes on Upgrading from AFS 3.3 to AFS 3.4
- Procedures for Upgrading from AFS 3.3 to AFS 3.4
    - — Installing Binary Files
    - — Upgrading Database Servers
    - — Upgrading File Servers
    - — Upgrading the Cache Managers on AFS Clients
    - — Cleaning Up After Upgrading to AFS 3.4

If your cell is running AFS 3.2 software currently, Section 3.5 contains information for upgrading your cell from AFS 3.2 to AFS 3.4. Section 3.5 includes the following subsections:

- Installing Binary Files
- Upgrading Servers
- Upgrading the Cache Managers on AFS Clients

If you need to downgrade your cell to AFS 3.3 for any reason, Section 3.6 contains information for downgrading your cell from AFS 3.4 to AFS 3.3. Section 3.6 includes the following subsections:

- Installing Binary Files
- Downgrading Servers in a Cell with Multi-Homed File Servers
- Downgrading Servers in a Cell without Multi-Homed File Servers
- Downgrading the Cache Managers on an AFS Client

3.1. Warnings About Operating System Upgrades

As Chapter 2 of the AFS 3.4 Release Notes makes clear, many sites that plan to upgrade to AFS 3.4 from previous versions of AFS must also upgrade the operating system on several system types (for example, DEC machines to Ultrix 4.3, 4.3a, or 4.4; Hewlett-Packard machines to HP-UX 9.0, 9.01, or 9.03; and SGI machines to IRIX 5.2 or 5.3). If you need to upgrade an AFS machine to a new operating system version, you must take several actions before upgrading the operating system to preserve AFS functionality. These actions include the following:

- You *must* unmount the *vicep***x** partitions on all file server machines to prevent the vendor-supplied **fsck** program from running on them when you reboot the machine during installation of the new operating system. (You should make sure that the partitions are not mounted during reboot until the AFS-modified **vfsck** program has replaced the standard vendor-supplied **fsck** program. If you do not prevent the vendor-supplied **fsck** program from starting before the **vfsck** program, AFS data will be lost.)
- You *must* protect the AFS-modified versions of the **ftpd**, **inetd**, **login**, **rcp**, **rlogind**, **rsh**, and **vfsck** commands if you are installing the AFS 3.4 version of the current system type. For example, if you are upgrading an AFS 3.3 version of the *hp700_ux90* system type to an AFS 3.4 version of the *hp700_ux90* system type, protect the AFS-modified versions of these UNIX commands. After you have successfully installed the operating system, remember to move the AFS-modified commands back to the directories from which they are accessed during normal use.

  If you are installing a new AFS system type, replace the UNIX commands supplied with the native operating system with their modified counterparts supplied with AFS 3.4.

3.2. Prerequisites for Upgrading

Ensure the following before beginning any upgrade operations:

- You have access to the binary files for the AFS 3.4 release (on tape or over the network).
- You have a copy of the *AFS Installation Guide*.
- The partition that houses the */usr* directory on each server machine has 18 megabytes of disk space on which to store the AFS server binaries.
- You can log in to all server and client machines with “root” permissions.
- You are listed in the */usr/afs/etc/UserList* file or you can authenticate as a member of the group *system:administrators*.
- You can identify the network addresses of the database server machines (or you know which database server machine has the lowest network address).

3.3. Notes on Upgrading from AFS 3.3 to AFS 3.4

AFS 3.4 provides support for multi-homed file servers, which are file servers that have multiple network addresses. A multi-homed file server can respond to a client RPC via a different network address than the one initially addressed by the client. By providing multiple paths through which a client's Cache Manager can communicate with it, a multi-homed file server can increase the availability of computing resources and improve performance.

AFS 3.4 supports up to 16 addresses per multi-homed file server machine (8 interface boards). File servers register their network addresses with the Volume Location Database (VLDB) at startup. In AFS 3.3 and earlier versions, file server machines were identified in the VLDB by a single network address. In AFS 3.4, file server machines are represented in the VLDB by a unique host identifier, the */usr/afs/local/sysid* file, which is created by the **fileserver** process during startup. This file server host identifier contains information about all known network addresses for that file server machine. These network addresses are updated whenever the **fileserver** process is restarted.

Note: You cannot install database servers (that is, servers for the Authentication, Protection, Volume Location, and Backup databases) on multi-homed machines. Be aware that if you have file servers and database servers running currently on the same machine with AFS 3.3, and you wish to make the machine multi-homed and upgrade to AFS 3.4, you *must* reconfigure these machines and move the database server functionality to another machine.

Note: AFS 3.4 does not support multi-homed clients or database servers.

Note the following requirements if you plan to use one or more multi-homed machines in your AFS cell:

- Reconfigure any server machines that are currently file servers *and* database servers. You cannot operate multi-homed file servers on machines that also run database servers (database servers cannot be multi-homed).
- Ensure that there are no volume commands or operations in progress when upgrading the database servers to AFS 3.4.
- Install AFS 3.4 database server binaries and restart the database servers *before* upgrading and restarting any file servers with AFS 3.4 file server binaries.
- Restart all database servers simultaneously when upgrading to AFS 3.4, starting with the database server at the lowest network address.

AFS 3.4 Volume Location (VL) servers can interoperate with AFS 3.3 file servers and Cache Managers. AFS 3.4 VL servers cannot interoperate in the same cell with AFS 3.3 and earlier VL servers; when you begin upgrading the database servers in your cell to AFS 3.4, you *must* upgrade them all to AFS 3.4 simultaneously. Some AFS 3.3a and earlier versions of the **vos** command suite do not work with AFS 3.4 VL servers.

You do not need to bring the AFS cell down before upgrading database servers, file servers, or clients to AFS 3.4. You can restart the AFS 3.4 file servers and AFS clients at your convenience after restarting the database servers.

If there are no multi-homed machines, the order in which you install kernel extensions, the database server binaries, and the file server binary is insignificant. You can restart the AFS 3.4 database server, file server, and AFS client binaries at your convenience.

If you plan to use one or more multi-homed machines, AFS 3.4 requires a new VLDB format; therefore, AFS 3.4 VL servers and AFS 3.3 VL servers cannot interoperate in the same cell. Previous versions of AFS required a manual VLDB conversion using the **vldb_convert** command. In AFS 3.4, the **vlserver** process converts the VLDB from a version 3 VLDB format to a version 4 VLDB format automatically when the **vlserver** process detects a multi-homed file server. However, downgrading the VL servers from AFS 3.4 to AFS 3.3 requires a manual conversion of the VLDB using the **vldb_convert** command.

For more information about multi-homed file servers, see Section 14.1.

Note: In AFS 3.4, the **vos changeaddr** command is no longer necessary to change the network addresses of file server machine ports. These ports are registered automatically with the VL server when you restart the file server.

3.3.1. Notes for AFS Cells Using HP-UX 9.0 and OSF/1 2.0

If you plan to upgrade a Hewlett-Packard 9000 Series 800 machine running HP-UX version 9.0 or a DEC Alpha machine running OSF/1 version 2.0 to AFS 3.4, you *must* upgrade all AFS binaries to the 3.4 version. This includes all AFS file server, database server, and client binaries, as well as the kernel extensions. You must also recompile any programs which were compiled against previous versions of AFS.

The **syscall** slot number for AFS has been changed in AFS 3.4 so that AFS and DFS can coexist on the same machine. Previously, this **syscall** slot number conflicted with DFS.

3.4. Procedures for Upgrading from AFS 3.3 to AFS 3.4

The following subsections contain instructions for upgrading from AFS 3.3 to AFS 3.4. These upgrade instructions require you to have “root” permissions. If you have not already, you should read the instructions in the preceding sections as they contain information that should be understood prior to performing the upgrade.

The following steps summarize the procedure to upgrade the machines in your cell to AFS 3.4:

Step 1: Install the binary files for each system type on the binary distribution machines.

Step 2: Upgrade the database servers.

Step 3: Upgrade the file servers.

Step 4: Upgrade the Cache Managers on AFS clients.

Recommendation: Transarc recommends that you complete the steps in Sections 3.4.1 and 3.4.2 in one session.

Note: In the examples using the **bos install** and **bos restart** commands in the following subsections, use the **-cell**, **-localauth**, and **-noauth** arguments with these commands as appropriate.

Note: The instructions assume that you are using multi-homed file servers when you upgrade to AFS 3.4. If you do not have multi-homed file servers, you can upgrade the binaries in any order. However, all database servers should be upgraded at the same time.

Note: If a database server is running on the machine on which you plan to upgrade the file server and the machine is multi-homed, make certain to stop the database server processes using the **bos stop** command. The **bos stop** command changes the status of the specified process (in this case, the database server processes) in the *BosConfig* file to `NotRun`.

3.4.1. Installing Binary Files

Note: If you upgrade the binary distribution machines, be aware that the changes are propagated to all machines of the same system type in the cell. If you do not want to upgrade all of your machines to AFS 3.4, shut down the **upserver** process on binary distribution machines and install the binaries on selected machines only.

You must install the AFS 3.4 binary files by copying them from the distribution tape to the binary distribution machine or by installing the AFS 3.4 binary files over the network directly to the binary distribution machine in your local AFS cell. You can install the new AFS 3.4 binaries to the same location of the existing binaries using the **bos install** command. When you reboot these machines or restart the binaries using the **bos restart** command, the AFS 3.4 binary starts and moves the AFS 3.3 binary to a *.bak* file. The AFS 3.4 binary also moves any AFS binary designated as a *.bak* file to a *.old* file.

The steps required to upgrade a cell depend on the cell's current configuration. Refer to the section of the *AFS Installation Guide* entitled "Setting Up Volumes to House AFS Binaries" (in particular, to its subsection entitled "Loading AFS Binaries into a Volume and Creating a Link to the Local Disk") for detailed steps in copying AFS binary files.

The following instructions assume that all file servers and database servers are to be upgraded to full AFS 3.4 server functionality. Consider the following before upgrading your cell's servers:

- You must upgrade all database servers to use the AFS 3.4 VL server.
- You do not need to upgrade your clients when you upgrade your database servers and file servers, but rather you can upgrade them at your convenience.
- On a per-machine basis, the conversion procedure for each server takes only the amount of time it takes to restart the server process. In general, the time required to complete the upgrade depends on the number of servers in your cell.

For each different system type that you wish to upgrade:

Step 1: Change directories using the UNIX **cd** command to your local cell's distribution location or Transarc's binary distribution location. The following example shows the **cd** command to change to Transarc's binary distribution location:

```
# cd /afs/transarc.com/product/afs/3.4/$systype/root.server/usr/afs/bin
```

where *$systype* specifies the type of operating system binaries that you wish to install.

Step 2: Install the contents of this directory using the **bos install** command in the */usr/afs/bin* directory on the binary distribution machine for each system type, as follows:

```
# bos install -server binary distribution machine name -file * -dir /usr/afs/bin
```

where *binary distribution machine name* is the name of the machine on which you wish to install the AFS binaries for each system type.

Step 3: Wait five minutes for the binaries to be distributed to the server machines via the **upserver** process or repeat the **bos install** command in Step 2 for each server machine of the same system type. If you have machines that are not regularly updated from the binary distribution machine, you should also use the **bos install** command to upgrade the binaries on those machines.

3.4.2. Upgrading Database Servers

AFS 3.4 features a new version of the VLDB that supports multi-homed file servers; see Chapter 14 for additional information on this feature. When the VL server detects a multi-homed file server, the **vlserver** process converts VLDBs from version 3 VLDB format to version 4 VLDB format automatically. If there are no multi-homed file servers in your cell, the VLDB remains in version 3 VLDB format.

Note: If there will be one or more multi-homed file servers, install and restart the database servers *before* upgrading and restarting any file servers.

Step 1: On the database server machine with the lowest network address, copy the *vldb.DB0* (database) file to a different directory, as follows:

```
# cp /usr/afs/db/vldb.DB0 pathname
```

where *pathname* is the name of the directory to which the database file is to be copied.

Step 2: After ensuring that the binaries are installed on each of the server machines, restart the database servers using the **bos restart** command, beginning with the database server at the lowest network address, as follows:

```
# bos restart -server database server machine name -bosserver -all
```

where *database server machine name* is the name of the machine on which you wish to restart the database servers.

Step 3: Install the AFS 3.4 kernel extensions for use either by a dynamic loader or in building a new kernel as described in the appropriate sections of the *AFS Installation Guide* and *AFS System Administrator's Guide*.

When you restart the database servers as directed in the previous steps, Ubik elects the database server with the lowest network address as the synchronization site for the Ubik databases.

3.4.3. Upgrading File Servers

For each file server that you wish to restart as an AFS 3.4 file server, do the following:

Step 1: After ensuring that the new **fileserver** binaries are installed on the file server machines, restart the file servers at your convenience using the **bos restart** command, as follows:

```
# bos restart -server file server machine name -bosserver -all
```

where *file server machine name* is the name of the machine on which you wish to restart the file server.

Step 2: Install the AFS 3.4 kernel extensions for use either by a dynamic loader or in building a new kernel as described in the appropriate sections of the *AFS Installation Guide* and *AFS System Administrator's Guide*.

Note: The */usr/afs/local/sysid* file is a unique, machine-specific identifier file used in AFS 3.4. Upon starting the **fs** process, the file server uses the */usr/afs/local/sysid* file to register its network address or addresses with the VL servers. Be careful not to copy a */usr/afs/local/sysid* file to any other machine.

3.4.4. Upgrading the Cache Managers on AFS Clients

The following instructions assume an AFS client is to be upgraded to full AFS 3.4 functionality. Omit these steps if the AFS client will continue to use AFS 3.3 or AFS 3.3a software.

Recommendation: If you are using multi-homed file servers with AFS 3.4 software, Transarc recommends upgrading AFS Cache Managers to AFS 3.4 at your convenience.

Step 1: Install the AFS 3.4 kernel extensions for use either by a dynamic loader or in building a new kernel as described in the appropriate sections of the *AFS Installation Guide* and *AFS System Administrator's Guide*.

Note: You must use only AFS 3.4 command binaries with AFS 3.4 Cache Managers. Make sure that all AFS binary locations such as */usr/vice/etc* or */usr/afsws* are upgraded to AFS 3.4 command binaries.

Step 2: If the AFS client is to run an AFS 3.4 Cache Manager, replace the **afsd** binary file with the new version of the file. Copy the **afsd** binary file from the directory in AFS in which you installed it to */usr/vice/etc/afsd* on the machine.

```
# cp <installed pathname> /usr/vice/etc/afsd
```

Refer to the section of the *AFS Installation Guide* entitled "Creating */afs* and Starting the Cache Manager" (from the chapter entitled "Installing Additional Client Machines"), to Chapter 13 of the *AFS System Administrator's Guide*, and to Chapter 13 of these release notes for information about using the **afsd** command to configure the Cache Manager.

Step 3: Reboot the client machine.

3.4.5. Cleaning Up After Upgrading to AFS 3.4

When you are satisfied with the state of your cell, or if you need to reclaim local disk space, use the following command:

```
# bos prune -server binary distribution machine name -bak -old
```

where *binary distribution machine name* is the name of the machine on which you wish to prune old and backup versions of AFS binaries.

You do not need to retain the old versions of the server binaries in the */usr/afs/bin* directory, because you could easily install them with the **bos install** command in the event that you wish to downgrade one or more servers.

3.5. Procedures for Upgrading from AFS 3.2 to AFS 3.4

The following subsections contain instructions for upgrading from AFS 3.2 to AFS 3.4. If you have not already, you should read the instructions in Sections 3.1 through 3.3 as they contain information that should be understood prior to performing the upgrade.

3.5.1. Installing Binary Files

You must install the AFS 3.4 binary files by copying them from the distribution tape or over the network as described in Section 3.4.1.

The steps you perform to upgrade any cell depend on the cell's current configuration. Refer to the section of the *AFS Installation Guide* entitled "Setting Up Volumes to House AFS Binaries" and to its subsection entitled "Loading AFS Binaries into a Volume and Creating a Link to the Local Disk" for information about the steps you need to perform to copy the AFS binary files.

3.5.2. Upgrading Servers

The following instructions assume that all file servers and database servers are to be upgraded to full AFS 3.4 server functionality. The instructions indicate steps that can be omitted in certain cases. Consider the following before upgrading your cell's servers:

- You must convert all database servers to use the AFS 3.4 VL server and VLDB.
- You do not need to upgrade your Cache Managers when you upgrade your servers.
- Because your entire cell is unavailable during the upgrade procedure, you should perform the conversion during periods of low system usage or regular network maintenance (such as overnight or over a weekend). On a per-machine basis, the conversion procedure takes only the amount of time it takes to reboot a machine. However, VLDB conversion adds a few extra minutes to the procedure. In general, the time required to complete the upgrade depends on the number of servers in your cell.

Step 1: On each file server machine, issue the following **bos shutdown** command to shut down the File Server, the Volume Server, and the Salvager:

```
# bos shutdown -server file server machine name -instance fs
```

where *file server machine name* is the name of the file server machine on which the server processes are to be shut down, and **fs** specifies that the **fs** process (the File Server, the Volume Server, and the Salvager) is to be shut down.

Step 2: On each database server machine, issue the following **bos shutdown** command to shut down the VL Server:

```
# bos shutdown -server database server machine name -instance vlserver kaserver
ptserver buserver upclient upserver
```

where *database server machine name* is the name of the database server machine on which the server process is to be shut down, and **vlserver kaserver ptserver buserver upclient upserver** specifies the processes that are to be shut down.

Step 3: On the database server machine with the lowest network address, copy the *vldb.DB0* (database) file to a different directory. This is strongly recommended because the conversion utility concludes by removing the old version of the VLDB. The size of this backup copy of the database is not considered in the amount of disk space required to perform the conversion. If you copy the database to a directory in the file system in which it currently resides, make sure the file system has 18 megabytes of disk space to accommodate the conversion process. Copy the file as follows:

```
# cp /usr/afs/db/vldb.DB0 pathname
```

where *pathname* is the name of the directory to which the database file is to be copied.

Step 4: On the database server machine with the lowest network address, issue the **vldb_convert** command to convert the database to AFS 3.3 format. (The binary for this command should be in the */etc* subdirectory of the temporary storage area of the local disk.) The command takes no more than a few minutes to complete the conversion.

```
# vldb_convert -to 3 -from 2
```

Note: You can verify the success of the conversion by running the **vldb_convert** command with the **-showversion** flag.

Step 5: On each server machine, install the AFS 3.4 kernel extensions either for use by a dynamic loader or in building a new kernel. As you perform the steps, copy the necessary files from their location on the local disk in which you installed them. Refer to the section of the *AFS Installation Guide* entitled "Installing an Additional File Server Machine" for information about loading kernel extensions. Perform only the following steps from that section:

- **Step 1**
- **Step 2**
- **Step 3** except for rebooting of the machine (you reboot all server machines in the following steps)
- **Step 5**

Note: Information on using Solaris and SGI kernel extensions can be found in the 3.4 version of the *AFS Installation Guide* or in the appropriate README file located in the */afs/transarc.com/product/afs/3.4/<sysname>* directory.

Step 6: Reboot the database server machine with the lowest network address.

Step 7: Reboot each remaining database server machine.

When you reboot the database server machines, Ubik elects the database server machine with the lowest network address as the synchronization site for the VLDB.

Note: If you have one or more multi-homed file server machines in your cell, the VLDB requires a conversion from version 3 VLDB format to version 4 VLDB format. This VLDB conversion is handled by the **vlserver** process and is automatic upon VL servers detecting multi-homed file servers. If you do not have multi-homed file servers in your cell, the VLDB remains in version 3 VLDB format.

Step 8: Reboot each file server machine at your convenience.

3.5.3. Upgrading the Cache Managers on AFS Clients

The following instructions assume a client machine is to be upgraded to full AFS 3.4 functionality. Omit this section if the client will continue to use AFS 3.2 software; Transarc does not recommend running AFS 3.2 clients with AFS 3.4 database servers and file servers.

Step 1: Install the AFS 3.4 kernel extensions for use either by a dynamic loader or in building a new kernel. As you perform the steps, copy the necessary files from the directory in AFS in which you installed them. Refer to the *AFS Installation Guide* for instructions for loading kernel extensions; perform only the steps in the following sections of the chapter entitled "Installing Additional Client Machines":

— "Loading Client Files onto the Local Disk"
— "Incorporating AFS Modifications into the Kernel"

Note: Information on using Solaris and SGI kernel extensions can be found in the 3.4 version of the *AFS Installation Guide* or in the appropriate README file located in the */afs/transarc.com/product/afs/3.4/<sysname>* directory.

Step 2: If the client machine is to run the AFS 3.4 client software, replace the **afsd** binary file with the new version of the file.

a. Copy the **afsd** binary file from the directory in AFS in which you installed it to the */usr/vice/etc/afsd* location.

b. Refer to the section of the *AFS Installation Guide* entitled "Creating /afs and Starting the Cache Manager" (from the chapter entitled "Installing Additional Client Machines"), to Chapter 13 of the *AFS System Administrator's Guide*, and to Chapter 13 of these release notes for information about using the **afsd** command to configure the Cache Manager.

Step 3: Reboot the client machine.

3.6. Procedures for Downgrading from AFS 3.4 to AFS 3.3

The following subsections contain instructions for downgrading from AFS 3.4 to AFS 3.3. If you have not already, you should read the instructions in Sections 3.1 through 3.3 as they contain information that should be understood prior to performing the downgrade.

The following subsections contain instructions for upgrading from AFS 3.2 to AFS 3.4. If you have not already,

The following instructions assume that all file server and database server machines are to be downgraded to full AFS 3.3 server functionality. The instructions indicate steps that can be omitted in certain cases. Consider the following before downgrading your cell's server machines:

- You must convert all database server machines to use the AFS 3.3 VL server and VLDB.
- You do not need to downgrade your client machines when you downgrade your server machines.
- Because your entire cell is unavailable during the downgrade procedure, you should perform the conversion during periods of low system usage or regular network maintenance (such as overnight or over a weekend). On a per-machine basis, the conversion procedure takes only the amount of time it takes to reboot a machine. However, conversion of the VLDB adds a few extra minutes to the procedure. In general, the time required to complete the downgrade depends on the number of server machines in your cell.

Before proceeding with the procedures for downgrading your cell to AFS 3.3, run the **vldb_convert** command with the **-showversion** flag to determine the version of the VLDB in your AFS cell, as follows:

```
# vldb_convert -showversion
```

- If your cell contains a version 4 VLDB, follow the downgrading instructions contained in Section 3.6.2.
- If your cell contains a version 3 VLDB, follow the downgrading instructions contained in Section 3.6.3.

Note: It is possible for you to have a version 4 VLDB with no multi-homed file servers currently in your cell. If you had multi-homed file servers in your cell previously, the **vlserver** process updated the VLDB to version 4; the VLDB remains at version 4 unless you convert it manually to a different version.

3.6.1. Installing Binary Files

You must install the AFS 3.3 binary files by copying them from the distribution tape or over the network into AFS.

The steps you perform to upgrade any cell depend on the cell's current configuration. Refer to the section of the *AFS Installation Guide* entitled "Setting Up Volumes to House AFS Binaries" and to its subsection entitled "Loading AFS Binaries into a Volume and Creating a Link to the Local Disk" for information about the steps you need to perform to copy the AFS binary files.

3.6.2. Downgrading Servers in a Cell with Multi-Homed File Servers

If there are multi-homed file servers, perform the following steps to downgrade servers from AFS 3.4 to AFS 3.3:

Step 1: On each file server machine, issue the following **bos shutdown** command to shut down the File Server, the Volume Server, and the Salvager:

 # bos shutdown -server *file server machine name* **-instance fs**

where *file server machine name* is the name of the file server machine on which the server processes are to be shut down, and **fs** specifies that the **fs** process (the File Server, the Volume Server, and the Salvager) is to be shut down.

Step 2: On each database server machine, issue the following **bos shutdown** command to shut down the VL Server:

 # bos shutdown -server *machine* **-instance vlserver kaserver ptserver buserver upclient upserver**

where *database server machine name* is the name of the database server machine on which the server process is to be shut down, and **vlserver kaserver ptserver buserver upclient upserver** specifies the processes that are to be shut down.

Step 3: On the database server machine with the lowest network address, copy the *vldb.DB0* (database) file to a different directory. This is strongly recommended because the conversion utility concludes by removing the old version of the VLDB. The size of this backup copy of the database is not considered in the amount of disk space required to perform the conversion. If you copy the database to a directory in the file system in which it currently resides, make sure the file system has 18 megabytes of disk space to accommodate the conversion process, as follows:

cp /usr/afs/db/vldb.DB0 *pathname*

where *pathname* is the name of the directory to which the database file is to be copied.

Step 4: On the database server machine with the lowest network address, issue the **vldb_convert** command to convert the database to AFS 3.3 format. (The binary for this command should be in the */etc* subdirectory of the temporary storage area of the local disk.) The command takes no more than a few minutes to complete the conversion.

vldb_convert -to 3 -from 4

Note: You can verify the success of the conversion by running the **vldb_convert** command with the **-showversion** flag.

Step 5: On each server machine, install the AFS 3.4 kernel extensions either for use by a dynamic loader or in building a new kernel. As you perform the steps, copy the necessary files from their location on the local disk in which you installed them. Refer to the section of the *AFS Installation Guide* entitled “Installing an Additional

File Server Machine" for information about loading kernel extensions. Perform only the following steps from that section:

— **Step 1**
— **Step 2**
— **Step 3** except for rebooting of the machine (you reboot all server machines in the following steps)
— **Step 5**

Note: Information on using Solaris and SGI kernel extensions can be found in the 3.3 version of the *AFS Installation Guide* or in the appropriate README file located in the */afs/transarc.com/product/afs/3.3/<sysname>* directory.

Step 6: Reboot the database server machine with the lowest network address.

Step 7: Reboot each remaining database server machine.

When you reboot the database server machines, Ubik elects the database server machine with the lowest network address as the synchronization site for the VLDB.

Step 8: Reboot each file server machine at your convenience.

3.6.3. Downgrading Servers in a Cell Without Multi-Homed File Servers

If there are no multi-homed file servers, perform the following steps to downgrade servers from AFS 3.4 to AFS 3.3:

Step 1: On each file server machine, issue the following **bos shutdown** command to shut down the File Server, the Volume Server, and the Salvager:

 # bos shutdown -server *file server machine name* **-instance fs**

where *file server machine name* is the name of the file server machine on which the server processes are to be shut down, and **fs** specifies that the **fs** process (the File Server, the Volume Server, and the Salvager) is to be shut down.

Step 2: On each database server machine, issue the following **bos shutdown** command to shut down the VL Server:

 # bos shutdown -server *database server machine name* **-instance vlserver kaserver ptserver buserver upclient upserver**

where *database server machine name* is the name of the database server machine on which the server process is to be shut down, and **vlserver kaserver ptserver buserver upclient upserver** specifies the processes that are to be shut down.

Step 3: On each server machine, install the AFS 3.3 kernel extensions either for use by a dynamic loader or by building a new kernel. As you perform the steps, copy the necessary files from their location on the local disk in which you installed them. Refer to the section of the *AFS Installation Guide* entitled "Installing an Additional

File Server Machine" for information about loading kernel extensions. Perform only the following steps from that section:

— **Step 1**
— **Step 2**
— **Step 3** except for rebooting of the machine (you reboot all server machines in the following steps)
— **Step 5**

Note: Information on using Solaris and SGI kernel extensions can be found in the 3.3 version of the *AFS Installation Guide* or in the appropriate README file located in the */afs/transarc.com/product/afs/3.3/<sysname>* directory.

Step 4: Reboot the database server machine with the lowest network address.

Step 5: Reboot each remaining database server machine.

When you reboot the database server machines, Ubik elects the database server machine with the lowest network address as the synchronization site for the VLDB.

Step 6: Reboot each file server machine at your convenience.

3.6.4. Downgrading the Cache Managers on AFS Clients

The following instructions assume an AFS client is to be downgraded to full AFS 3.3 functionality. Omit this section if the client will continue to use AFS 3.4 software.

Step 1: Install the AFS 3.3 kernel extensions for use either by a dynamic loader or in building a new kernel. As you perform the steps, copy the necessary files from the directory in AFS in which you installed them. Refer to the *AFS Installation Guide* for instructions for loading kernel extensions; perform only the steps in the following sections of the chapter entitled "Installing Additional Client Machines":

— "Loading Client Files onto the Local Disk"
— "Incorporating AFS Modifications into the Kernel"

Note: Information on using Solaris and SGI kernel extensions can be found in the 3.3 version of the *AFS Installation Guide* or in the appropriate README file located in the */afs/transarc.com/product/afs/3.3/<sysname>* directory.

Step 2: If the client machine is to run the AFS 3.3 client software, replace the **afsd** binary file with the new version of the file.

a. Copy the **afsd** binary file from the directory in AFS in which you installed it to */usr/vice/etc/afsd*.

b. Refer to the section of the *AFS Installation Guide* entitled "Creating /afs and Starting the Cache Manager" (from the chapter entitled "Installing Additional Client Machines"), to Chapter 13 of the *AFS System Administrator's Guide*,

and to Chapter 13 of these release notes for information about using the **afsd** command to configure the Cache Manager.

Step 3: Reboot the client machine.

4. Authentication

This chapter describes changes to AFS authentication and login programs for version 3.4. AFS 3.4 contains changes to the following:

- Kerberos support for the **kaserver** process
- AFS **login** program
- AIX **login** program
- Alpha OSF/1 **login** program
- Solaris **login** program

These changes are marked with the heading "**AFS 3.4 Changes**."

This chapter also contains changes from the AFS 3.3 release that have not been incorporated into the full AFS documentation set. These changes are marked with the heading "**AFS 3.3 Changes**."

4.1. Kerberos Support for the kaserver Process

AFS 3.4 Changes

AFS 3.4 supports authentication through either the AFS **kaserver** Authentication Database or through MIT's Kerberos. The **kaserver** Authentication Server listens on UDP port 88, as well as UDP port 750, to support Kerberos authentication.

4.2. Changes to the AFS login Program

AFS 3.4 Changes

In AFS 3.4, the **login** program logs a failed login message in the UNIX system log (*syslog* file) after the sixth failed login attempt rather than the tenth.

4.3. Changes to the AIX login Program

AFS 3.4 Changes

AFS 3.4 contains the following changes to the AIX login program:

- Support for the **#** and **!** character entries in the */etc/passwd* file
- Support for alternative authentication programs with AIX 4.1. Transarc supplies the **afs_dynamic_auth** alternative authentication program with AFS 3.4 for use with AIX 4.1.

4.3.1. Support for # and ! Entries in the /etc/passwd File

AFS 3.4 supports the pound sign (**#**) character in the local */etc/passwd* file on AIX machines. The **#** character indicates that the **login** program goes directly to the AFS Authentication Database to check authentication and skips AIX local authentication and AIX secondary authentication. It is recommended that you include the standard AIX exclamation point (**!**) character as an entry in the */etc/passwd* file. The **!** character entry indicates that the **login** program checks for any AIX secondary authentication.

4.3.2. Support for Alternative Authentication Programs with AIX 4.1

In AFS 3.4, you can establish an alternative authentication program from the AIX 4.1 operating system to AFS. In order to take advantage of this authentication program provided with AFS 3.4, you must make the following configuration changes to AIX 4.1 on the local client machine. Ensure that you installed the **afs_dynamic_auth** program in the */usr/vice/etc* directory on the local client machine.

1. The **registry** variable defines in which domain users are administered. In the */etc/security/user* file on the local client machine running the AIX 4.1 operating system, set the **registry** variable to *DCE*:

   ```
   registry = DCE
   ```

   Note: You must use *DCE* for the **registry** variable. *AFS* is not a valid **registry** variable in AIX 4.1.

   Note: In the */etc/security/user* file on the local client machine, set the **registry** variable of the user "root" to *files*, that is,

   ```
   root:
         registry = files
   ```

   The value *files* designates that user "root" can authenticate using the local password files on the local machine only.

2. In the */etc/security/user* file on the local client machine running AIX 4.1:

   - If the machine is an AFS client only, set the **SYSTEM** variable to

     ```
     SYSTEM = "AFS OR (AFS[UNAVAIL] AND compat[SUCCESS])"
     ```

   - If the machine is both an AFS client and a DCE client, set the **SYSTEM** variable to

     ```
     SYSTEM = "DCE OR DCE[UNAVAIL] OR AFS OR AFS[UNAVAIL] AND
     compat[SUCCESS]"
     ```

3. In the */etc/security/login.cfg* file on the local client machine running AIX 4.1, identify the *DCE* authentication method with the following:

```
DCE:
        program = /usr/vice/etc/afs_dynamic_auth
```

   where **retry** specifies the maximum number of times that a user can consecutively fail to enter the correct password, **timeout** specifies the number of seconds that a user is given to enter their password, and **retry_delay** specifies the number of minutes that a user is locked out of their account if they exceed the maximum number of failed authentication attempts specified by the **retry** variable.

4. In the */etc/security/login.cfg* file on the local client machine running AIX 4.1, identify the *AFS* token with the following:

```
AFS:
        program = /usr/vice/etc/afs_dynamic_auth
        retry = 3
        timeout = 30
        retry_delay = 10
```

   where **retry** specifies the maximum number of times that a user can consecutively fail to enter the correct password, **timeout** specifies the number of seconds that a user is given to enter their password, and **retry_delay** specifies the number of minutes that a user is locked out of their account if they exceed the maximum number of failed authentication attempts specified by the **retry** variable.

4.4. Changes to the Alpha OSF/1 login Program

AFS 3.4 Changes

Four changes have been made to the DEC Alpha OSF/1 **login** program. Descriptions of these changes follow.

1. After you enter your password, the **login** program checks for authentication on the local machine. If you are authenticated on the local machine, the **login** program checks for authentication in AFS. If this attempt to authenticate to AFS fails, the **login** program returns a prompt:

```
Enter AFS password:
```

   If this second AFS authentication attempt fails, you are logged into the local file system without AFS authentication, because you successfully authenticated to the local machine. The same occurs for an empty password entry in the */etc/passwd* file.

2. If an **x** exists on a **passwd** entry in the local */etc/passwd* file, the **login** program gives the following prompt and does not check local authentication.

```
Enter AFS password:
```

If your attempt to authenticate to AFS fails, you are not allowed to access the local file system.

3. For special local users like “root,” “daemon,” and “bin,” the **login** program never prompts for the AFS password or calls an AFS authentication routine for these users.
4. You have five attempts to enter a correct password. If you fail after five login attempts, the session aborts.

Process authentication group (PAG) support is identical to other systems.

4.5. Changes to the Solaris login Program

AFS 3.4 Changes

For Solaris environments, AFS supports the existence of an */etc/default/login* file. In this file, you can set the following variables:

- **CONSOLE** - Allows only “root” to login from the specified terminal or console.
- **ALTSHELL** - Specifies an alternative shell (for “root” only).
- **PASSREQ** - Specifies that a password is required before permitting users access to the local system.
- **TIMEZONE** - Specifies a time zone to overwrite the default time zone.
- **HZ** - Overwrites the default setting of the timer-related **HZ** value.
- **PATH** - Uses the path specified by this variable instead of the default.
- **SUPATH** - Uses the path specified by this variable for “root” instead of the default.
- **ULIMIT** - Sets a limit on the file size of a process controlled by the user.
- **TIMEOUT** - Specifies a default interval of time to wait for the response of a correct password before closing the connections.
- **UMASK** - Specifies the default file creation mask to use for login session (the default value is 022).

Note: AFS 3.4 does not support the **SLEEPTIME** and **IDLEWEEKS** variables.

4.6. Support for Secondary Authentication on AIX 3.2

AFS 3.3 Changes

The 3.3 version of the AFS **login** program supports secondary authentication on the AIX 3.2 operating system. In addition, the AFS **login** program now checks for local UNIX authentication. If the AFS password is the same as the local UNIX password, then it prompts for only one password; if not, it prompts for both passwords. If the UNIX password entry (in the */etc/security/passwd* file) is an asterisk (*), the local authentication is ignored.

5. The Backup System

This chapter describes changes to the Backup System, the Tape Coordinator, and the **backup** command suite for AFS 3.4. In particular, AFS 3.4 contains two new commands, **backup volsetrestore** and **backup interactive**, and the following enhancements:

- A new backup configuration file, */usr/afs/backup/CFG_<tape_device>*
- Improved error messages and error handling
- Permanent tape names
- Improved tape scanning via the Tape Coordinator
- Improved Backup System prompting
- The **backup labeltape** command (enhanced)
- The **backup readlabel** command (enhanced)
- The **-cell** argument and **-localauth** flag on all **backup** commands

These changes are marked with the heading "**AFS 3.4 Changes**."

5.1. New Backup Configuration File

AFS 3.4 Changes

AFS 3.4 supports a new user-defined configuration file for use with the Backup System. Upon startup of AFS, the **butc** command reads the */usr/afs/backup/CFG_<tape_device>* configuration file and configures itself according to the parameters defined in the file. You can configure the Tape Coordinator to call executable routines, to handle changing tapes within a tape stacker or jukebox device, by setting the **MOUNT** and **UNMOUNT** parameters in the *CFG_<tape_device>* file. You can also support tape stacker and jukebox devices by configuring the *CFG_<tape_device>* file to avoid prompting for a tape in the Tape Coordinator window.

The following parameters can be set in the *CFG_<tape_device>* configuration file:

- The **MOUNT** parameter
- The **UNMOUNT** parameter
- The **ASK** parameter
- The **AUTOQUERY** parameter
- The **NAME_CHECK** parameter
- The **BUFFERSIZE** parameter

- The **FILE** parameter

These parameters are described in the following subsections.

5.1.1. MOUNT Parameter

The **MOUNT** parameter names a file. If you want the Backup System to support a tape stacker or jukebox device, you can write an executable routine and put it in this file to perform the tape mount operations for the device. The format for this parameter in the *CFG_<tape_device>* configuration file is

MOUNT <*filename*>

where <*filename*> is the name of the file which contains the executable routine.

Prior to opening the tape device, the Tape Coordinator checks for the **MOUNT** parameter in the *CFG_<tape_device>* configuration file. When the Tape Coordinator locates the **MOUNT** parameter, it executes the executable routine contained in the file specified by the **MOUNT** parameter instead of prompting the operator to mount the tape. This executable routine places a tape into the tape device. The following information is passed from the Tape Coordinator to the executable routine upon request:

- The tape device path (as specified in the */usr/afs/backup/tapeconfig* file)
- The tape operation will be one of the following:
  - The **backup appenddump** command
  - The **backup dump** command
  - The **backup labeltape** command
  - The **backup readlabel** command
  - The **backup restore** command
  - The **backup restoredb** command
  - The **backup savedb** command
  - The **backup scantape** command
- The number of times this tape has been requested (if an error occurs when opening the tape device, this value is incremented by 1 and the routine specified by the **MOUNT** parameter is called again)
- The tape name (value of *none* if not asking for a specific name)
- The tape ID (value of *none* if not asking for a specific ID)

The default action if no **MOUNT** parameter is supplied is to prompt the operator for the name of a tape device.

If an exit code of `0` is returned by the routine specified by the **MOUNT** parameter, the Tape Coordinator operation continues. If an exit code of `1` is returned by the routine specified by the **MOUNT** parameter, the Tape Coordinator operation aborts. If any other exit code is returned by the routine, the Tape Coordinator prompts the operator for the correct tape. If an operator wishes to abort the **MOUNT** operation, the operator can kill the job using the **backup kill** command:

(backup) kill -id <*job ID or dump set name*>

5.1.2. UNMOUNT Parameter

The **UNMOUNT** parameter names a file. If you want the Backup System to support a tape stacker or jukebox device, you can write an executable routine in this file to perform the tape unmount operations for the device. The format for this parameter in the *CFG_*<*tape_device*> configuration file is

UNMOUNT <*filename*>

where <*filename*> is the name of the file which contains the executable routine for use with a tape stacker or jukebox device.

After closing a tape device, the Tape Coordinator executes the routine in the file specified by the **UNMOUNT** parameter (even if the close operation fails). The routine specified by the **UNMOUNT** parameter removes a tape from the tape device. The following information is passed to the executable file from the Tape Coordinator:

- The tape device path (as specified in the */usr/afs/backup/tapeconfig* file)
- The tape operation (**unmount**)

If the **UNMOUNT** parameter is not supplied, the default action is to take no action.

5.1.3. ASK Parameter

The **ASK** parameter determines whether all questions, except the request to mount the tape, are to be asked of the tape operator via the Tape Coordinator window or whether to assume default answers. The format for this parameter in the *CFG_*<*tape_device*> configuration file is

ASK { **YES** | **NO** }

where

- **YES** enables operator prompts in any error cases.
- **NO** disables operator prompts in any error cases but uses default responses.

The error cases for the **ASK** parameter are

- A restore operation failed to restore a volume. The **YES** value asks if the operator wishes to continue the restore operation. The **NO** value will continue the restore operation.
- A dump operation failed to dump a volume. The **YES** value asks if the volume should be retried, omitted, or if the dump should be aborted. The **NO** value will omit the volume from the dump.
- The **backup scantape** command cannot determine if there is a next tape in the dump set. The **YES** value asks if there are more tapes to be dumped. The **NO** value assumes there are more tapes.
- The **backup labeltape** command is attempting to label a non-expired tape. The **YES** value asks if the labeltape operation should proceed or not. The **NO** value will not label the tape.

YES is the default value for the **ASK** parameter.

5.1.4. AUTOQUERY Parameter

The **AUTOQUERY** parameter can disable the initial prompt for tape insertion given by the Tape Coordinator when executing backup commands involving a tape device. The format for this parameter in the *CFG_<tape_device>* configuration file is

AUTOQUERY { **YES** | **NO** }

where

- **YES** prompts the operator for the first tape of a dump set.
- **NO** does not prompt the operator for the first tape of a dump set, but assumes it is already mounted in the tape device. A **NO** value for the **AUTOQUERY** parameter is similar to the **-noautoquery** flag on the **butc** command.

YES is the default value for the **AUTOQUERY** parameter.

5.1.5. NAME_CHECK Parameter

The **NAME_CHECK** parameter can be configured to prevent the Backup System from checking tape names. The format for this parameter in the *CFG_<tape_device>* configuration file is

NAME_CHECK { **YES** | **NO** }

where

- **YES** specifies that when dumping a volume, the Tape Coordinator verifies that the tape name is NULL or the same tape name as the specified dump.
- **NO** specifies that tape name checking is disabled. Any non-expired tape is acceptable.

YES is the default value for the **NAME_CHECK** parameter.

5.1.6. BUFFERSIZE Parameter

The **BUFFERSIZE** parameter allows the allocation of memory to increase the performance of dump and restore operations with the Backup System. The format for this parameter in the *CFG_<tape_device>* configuration file is

BUFFERSIZE <*size*>

where *size* specifies the memory allocation (in KB) for backup dump and restore operations.

For backup dump operations, volumes are read into the memory buffer and then out to tape, in contrast to the normal operation of going from disk to the tape drive at a slower rate. This allows faster transfers of volume data from a file server to the Tape Coordinator machine and faster transfers (streaming of the tape drive) from memory to the tape drive. A buffer size of 1 tape block (16 KB) is the default for the parameter for a backup dump operation.

For backup restore operations, volumes are read into the memory buffer and then out to the File Server. This allows faster transfers of volume data from a File Server to the Tape Coordinator machine and faster transfers (streaming of the tape drive) from memory to the tape drive. A buffer size of 2 tape blocks (32 KB) is the default for the parameter for a backup restore operation.

5.1.7. FILE Parameter

The **FILE** parameter can specify whether backup dump and restore operations are written to a tape device or a file. The format for this parameter in the *CFG_<tape_device>* configuration file is

FILE { **YES** | **NO** }

where

- **YES** specifies that backup dump and restore operations are written to a file. The pathname specified in the */usr/afs/backup/tapeconfig* file is the pathname of the file to which the operation is to be written.
- **NO** specifies that backup dump and restore operations are performed to a tape device.

Note the following requirements if you specify a value of **YES** (backup dump or restore to a file):

- The file must already exist. If the file does not already exist, the Tape Coordinator creates the file.
- If the Tape Coordinator needs another file to continue an operation, the Tape Coordinator prompts for the next tape but continues to use the pathname in the */usr/afs/backup/tapeconfig* file. The file that the Tape Coordinator uses (in the */usr/afs/backup/tapeconfig* file) should be a link to the actual file. As new files are needed, the link is updated to another file.
- Configuring a tape device to **YES** or a file to **NO** does not work.

Blocks are written in 16 KB sizes. The database records position not as filemark position (as tapes do), but as 16 KB position.

Note: Do not specify **YES** as the **FILE** configuration parameter and dump data to a tape device. You cannot restore from a tape device if this is done.

5.2. Improved Error Messages and Error Handling

AFS 3.4 Changes

AFS 3.4 contains several enhancements to error messages, log messages, and error handling. These include the following:

- More Backup System error messages were added containing greater detail.
- Backup System and Tape Coordinator status and error messages are more consistent.
- All Tape Coordinator (**butc**) status and error messages are sent to the *TL_<tape_device>* log file.
- Only error messages are sent to the *TE_<tape_device>* log file.
- Error messages appearing in the *TE_<tape_device>* and the *TL_<tape_device>* log files are more consistent.
- Error and status messages, depending on their level, are sent to the Tape Coordinator window.
- Error and warning messages are differentiated in the *TE_<tape_device>* log file. Error messages are specified as failures; warning messages are not specified as failures.

5.3. Permanent Tape Names

AFS 3.4 Changes

The **backup labeltape** command allows users to label tapes explicitly with a permanent name in AFS 3.4. The Backup System remembers this permanent name across dumps in the Backup System. If a user supplies a permanent name to a tape with the **backup labeltape** command using the **-name** argument, the Backup System remembers the permanent name as the tape is re-used or recycled. The Backup System remembers the permanent name until the user explicitly changes it using the **backup labeltape** command with the **-name** argument. It is recommended that permanent tape names be unique since that is the tape name which is recorded in the Backup Database and which is requested on backup restore operations. The permanent name is listed in the `name` field in the output resulting from the **backup readlabel** command.

If the user does not explicitly name a tape with a permanent name, AFS assigns a non-permanent name to the tape as it did previously. The Backup System produces this non-permanent name by concatenating the volume set and dump level with a tape sequence (for example, *guests.monthly.3*). This name is not permanent and changes whenever the tape label is rewritten by a **backup** command (for example, when using the **backup dump**, **backup labeltape**, and **backup savedb** commands). The AFS-assigned non-permanent name is listed in the `AFS name` field in the output resulting from the **backup readlabel** command.

5.4. The Tape Coordinator Can Scan Any Tape

AFS 3.4 Changes

In AFS 3.4, the Tape Coordinator allows a scan to begin on any tape of a dumpset. In previous versions of AFS, the Tape Coordinator had to start a scan on the first tape of a dump set.

The limitations to a tape scan are:

- If the first volume on the scanned tape is a continuation from the dump set's previous tape, the volume is not added to the Backup Database.
- If the first volume on the scanned tape is part of an appended dump, that volume is not added to the Backup Database; other volumes following the first volume are also not added to the Backup Database. Each volume after the first volume is not added to the Backup Database until a new appended dump is read. The **backup scantape** command must read the dump label of a tape before the Tape Coordinator can add the volume to the Backup Database.

5.5. Modification to Backup Prompting

AFS 3.4 Changes

Previously, the Backup System prompted the user for the name of the tape by sounding a bell and sending a message to the screen. The system reprompted the user for the information every 15 seconds. The repeated messages caused the user's information to scroll off the screen. In AFS 3.4, the system initially prompts for the tape name by sounding a bell and sending a message to the screen but reprompts every 15 seconds only by sounding a bell. This modification keeps the user's information from scrolling off the screen.

5.6. The backup Command

This section describes changes to individual commands in the **backup** command suite for AFS 3.4.

5.6.1. The -localauth Flag and -cell Argument (New)

AFS 3.4 Changes

All commands in the **backup** command suite started in non-interactive mode now support the **-localauth** flag and the **-cell** argument.

The **-localauth** flag assigns the **backup** command a token that never expires and needs to be run from a file server machine as "root." This flag instructs the **backup** command interpreter running on the local file server machine to construct a server ticket using the server encryption key with the highest key version number in the */usr/afs/etc/KeyFile* file on the local file server machine. The **backup** command presents the ticket to the Volume and/or Volume Location Server to use in mutual authentication. This flag is only useful for commands issued on file server machines, since client workstations do not have a */usr/afs/etc/KeyFile* file. It is intended

for **cron**-type processes or jobs included in the machine's */usr/afs/local/BosConfig* file. An example might be a command that automatically runs the **backup dump** command on certain volumes for archival backups. See the chapter in the *AFS System Administrator's Guide* about backing up the system. The flag can also be used if the issuer is unable to authenticate but is logged into the local UNIX file system as "root."

The **-cell** argument specifies the cell in which the Backup System and volumes are to be manipulated. The issuer can abbreviate *cell name* to the shortest form that distinguishes it from the other cells listed in the */usr/vice/etc/CellServDB* file on the client machine on which the command is issued. By default, commands are executed in the local cell, as defined:

- First, by the value of the environment variable **AFSCELL**.
- Second, in the */usr/vice/etc/ThisCell* file on the client machine on which the command is issued.

5.6.2. The backup interactive Command (New)

AFS 3.4 Changes

The new **backup interactive** command starts the backup operation in interactive mode. The **backup interactive** command is the equivalent of entering only **backup** at the system prompt in AFS 3.3. The **backup interactive** command enables the use of the **-localauth** flag and the **-cell** argument and was added to maintain compatibility with the operation of the **backup** command suite in non-interactive mode.

The syntax of the new **backup interactive** command follows:

backup [**interactive**] [**-localauth**] [**-cell** <*cell name*>] [**-help**]

To activate the **-localauth** flag and the **-cell** argument for use with commands in interactive mode, specify them with the **backup interactive** command at the system prompt.

Arguments:

-localauth Assigns the **backup interactive** command a token that never expires and needs to be run from a file server machine as "root." This flag instructs the **backup interactive** command interpreter running on the local file server machine to construct a server ticket using the server encryption key with the highest key version number in the */usr/afs/etc/KeyFile* file on the local file server machine. The **backup interactive** command presents the ticket to the Volume and/or VL Server to use in mutual authentication. This flag is only useful for commands issued on file server machines, since client workstations do not have a */usr/afs/etc/KeyFile* file. It is intended for **cron**-type processes or jobs included in the machine's */usr/afs/local/BosConfig* file. An example might be a command that automatically runs the **backup dump** command on certain volumes for archival backups. See the chapter in the *AFS System Administrator's Guide* about backing up the system. The flag can also be used if the issuer is unable to authenticate but is logged into the local UNIX file system as "root."

-cell Specifies the cell in which the Backup System and volumes are to be manipulated. The issuer can abbreviate *cell name* to the shortest form that distinguishes it from the other cells listed in the */usr/vice/etc/CellServDB* file on the client machine on which the command is issued. By default, commands are executed in the local cell, as defined:

- First, by the value of the environment variable **AFSCELL**.
- Second, in the */usr/vice/etc/ThisCell* file on the client machine on which the command is issued.

5.6.3. The backup volsetrestore Command (New)

AFS 3.4 Changes

A new command, **backup volsetrestore**, has been added to the **backup** command suite. The **backup volsetrestore** command restores all volumes in a volume set or restores one or more individual volumes. The command is useful for recovering from catastrophic losses of data, such as the loss of all volumes on multiple partitions of a file server machine or the loss of multiple partitions from multiple file server machines. The **backup volsetrestore** command can restore specialized collections of volumes, as well as restore different volumes to different sites. Conversely, the **backup volrestore** command restores one or more volumes to a single site, and the **backup diskrestore** command restores all volumes that reside on a single partition to the same partition.

The syntax of the **backup volsetrestore** command follows:

backup volsetrestore [**-name** <*volume set name*>] [**-file** <*file name*>]
[**-portoffset** <*TC port offset*>] [**-n**] [**-localauth**] [**-cell** <*cell name*>] [**-help**]

Arguments:

-name Names a volume set to be restored. The command restores all of the volumes in each of the entries in the specified volume set. You must use either this argument or the **-file** argument. Refer to the "Using the **-name** Argument" section in this description for more information on using this argument.

-file Specifies the full pathname of a file from which the command is to read the name of each volume to be restored and the site (file server machine and partition) to which the volume is to be restored. Specify each volume and site on a different line, using the following format:

machine partition volume

You must use either this argument or the **-name** argument. Refer to the "Using the **-file** Argument" section for information about using this argument.

-portoffset Specifies the port offset number of the tape drive to be used in the operation. If different types of tape drives were used to create the tapes that contain the dumpsets from which data is to be restored, use the **backup volrestore** command. Port 0 is the default.

Note: If multiple volumes are to be restored, the port offset order must be the same for all volumes. That is, all full dumps must be done on the same port offset, all first-level incrementals on the same port offset, etc.

-n Designates that no action is taken. This flag lists all of the volumes to be restored (both the full and incremental dumps of volumes), the location to which the volumes are to be restored, and on which tape the volume is to be found without actually performing the operation. Output from this command can be used as input with the **-file** argument. Include the other arguments as you would to execute the command. You can use this flag with the **-name** argument to write a list of volumes to a file, which you can then modify for use with the **-file** argument. See the "Output" section for information about using the **-n** flag.

-localauth Assigns the **backup** command a token that never expires and needs to be run from a file server machine as "root." This flag instructs the **backup** command interpreter running on the local file server machine to construct a server ticket using the server encryption key with the highest key version number in the */usr/afs/etc/KeyFile* file on the local file server machine. The **backup** command presents the ticket to the Volume and/or Volume Location Server to use in mutual authentication. This flag is only useful for commands issued on file server machines, since client workstations do not have a */usr/afs/etc/KeyFile* file. It is intended for **cron**-type processes or jobs included in the machine's */usr/afs/local/BosConfig* file. An example might be a command that automatically runs the **backup dump** command on certain volumes for archival backups. See the chapter in the *AFS System Administrator's Guide* about backing up the system. The flag can also be used if the issuer is unable to authenticate but is logged into the local UNIX file system as "root."

-cell Specifies the cell in which the Backup System and volumes are to be manipulated. The issuer can abbreviate *cell name* to the shortest form that distinguishes it from the other cells listed in the */usr/vice/etc/CellServDB* file on the client machine on which the command is issued. By default, commands are executed in the local cell, as defined:

- First, by the value of the environment variable **AFSCELL**.
- Second, in the */usr/vice/etc/ThisCell* file on the client machine on which the command is issued.

-help Prints the online help for this command. Do not provide any other arguments or flags with this one.

Description:

The **backup volsetrestore** command restores the contents of specified volumes from tape to the file system. The command performs a full restore of each volume, restoring data from the last full dump and all subsequent incremental dumps (if any) of each volume. Use the **-name** argument or the **-file** argument to indicate the volumes to be restored.

- The **-name** argument lets you restore all of the volumes included in the volume entries in a specified volume set. The command reads the Volume Location Database (VLDB) to determine the volumes to be restored and restores them to the site listed in the VLDB.
- The **-file** argument lets you restore individual volumes that have entries in a specified file. The command restores each volume to the site you specify in the file.

The **-n** flag instructs the command to produce a list of the volumes it would restore without actually restoring any volumes. The command also provides information about the tapes that contain dumps of the volumes. You can use the **-n** flag with the **-file** argument to determine the tapes required to restore the indicated volumes. You can also use the **-n** flag with the **-name** argument to construct a list of volumes that would be restored with a specified volume set; you can then modify the list of volumes as necessary to produce a file for use with the **-file** argument. You could create a file for the **backup volsetrestore** command if you want to restore volumes in a volume set to a different location, restore only a subset of the volume set, or change the order of volume restores within the volume set.

Using the -name Argument:

Use the **-name** argument of the **backup volsetrestore** command to restore the volumes included in a specified volume set. The command reads the VLDB to determine all volumes that satisfy fields of the entries in the volume set. It then looks in the Backup Database to determine the tapes that contain the last full dump and all subsequent incremental dumps of each volume. It restores each volume included in an entry in the volume set to the site listed in the VLDB, overwriting any existing version of the volume.

You can specify the name of an existing volume set, or you can define a new volume set and add entries that correspond to the volumes that need to be restored. It can be useful to define a new volume set when you are starting new file servers and want to create a new volume set for backing up these file servers. For example, suppose you need to restore all volumes that reside on the file server machines named *fs1.abc.com* and *fs2.abc.com*. You can use the **backup addvolset** command to create a new volume set. You can then use the **backup addvolentry** command to add the following entries to the new volume set:

```
fs1.abc.com .* .*
fs2.abc.com .* .*
```

These entries indicate all volumes on all partitions on the machines named *fs1.abc.com* and *fs2.abc.com*. Once the new volume set is defined, you can issue the **backup volsetrestore** command, specifying the name of the volume set with the **-name** argument.

In volume sets created for use with the **backup volsetrestore** command, define entries that match the ReadWrite versions of volumes. The Backup System then searches the Backup Database for a dump of the ReadWrite or Backup volume. If you define a ReadOnly or Backup volume, the Backup System will restore only that volume name (if the ReadWrite volume exists). Also, the volume set expansion may miss volumes that may have been dumped.

Using the -file Argument:

Use the **-file** argument of the **backup volsetrestore** command to restore each volume that has an entry in a specified file. The command examines the Backup Database to determine the tapes that contain the last full dump and all subsequent incremental dumps of each specified volume. It restores each volume to the site indicated in the specified file. The command does not consult the VLDB.

An entry for a volume in a file to be used with the command must have the following format:

machine partition volume [comments...]

The entry provides the following information:

machine Names the file server machine to which the volume is to be restored.

partition Names the partition to which the volume is to be restored.

volume Names the volume to be restored. In general, you should provide the name of the ReadWrite version of the volume.

comments... All remaining text. The command treats any other text provided with the entry for the volume as a comment and ignores it. This text is optional.

Do not use wildcards (for example, **.***) in an entry. Also, do not include a newline character in an entry for a volume; each entry must appear on a single line of the file. Include only a single entry for each volume in the file. The command uses only the first entry for a given volume; it ignores all subsequent entries for the volume.

Note that if you restore a volume to a site other than the site which is indicated in the VLDB and if the volume resides in the location specified in the VLDB, the existing version of the volume is removed when the volume is restored and the volume's entry in the VLDB is updated accordingly. If you restore a volume to the site at which it currently exists, the command overwrites the existing version of the volume.

Output:

If you omit the **-n** flag, the **backup volsetrestore** command returns the unique task ID number associated with the restore operation. The task ID number is displayed in the command window directly following the command line and in the Tape Coordinator's monitoring window if the **butc** command is issued with debug level 1. The task ID number is not the same as the job ID number, which is visible with the **(backup) jobs** command if the **backup volsetrestore** command is issued in interactive mode. The task ID number is a temporary number assigned to the task

by the Tape Coordinator, while the job ID number is a permanent number assigned to the job by the Backup System, which is a number that can be referenced. Note that the task ID and job ID numbers are not assigned to the operation until the command actually begins to restore volumes.

If you include the **-n** flag, the command displays the number of volumes that would be restored, followed by a separate line of information about each volume to be restored (its full and incremental dumps). For each volume, the command provides the following output:

machine partition volume_dumped # `as` *volume_restored; tape_name;* `pos` *position number; date*

The output provides the following information:

machine The host name of the file server machine to which the volume would be restored (for example, *fs1.abc.com*).

partition The name of the partition to which the volume would be restored (for example, */vicepa*).

volume_dumped

The name of the volume that was dumped (for example, *user.frost*). The command displays the name of the Backup version of the volume (for example, *user.frost.backup*) if that version was dumped.

volume_restored

The name with which the volume would be restored (for example, *user.frost*). The command always displays the name and volume ID of the ReadWrite version of the volume.

tape_name The name of the tape that contains the volume (for example, *user.full.3*).

position number

The position of the volume with respect to other volumes on the tape that contains the dump set (for example, *31* specifies that there are 30 volumes preceding the current one on the tape).

date The date and time at which the volume was dumped (for example, *Wed Jul 13 05:59:01 1994*).

The command displays multiple lines of information for a volume if one or more incremental dumps were performed since the last full dump of the volume. The command displays one line of output for the last full dump and one line of output for each incremental dump. It displays the lines in the order in which the dumps would need to be restored, beginning with the full dump. It does not necessarily present all of the lines for a volume consecutively in the order in which the incremental dumps occurred.

If you intend to write the output of the **-n** flag to a file for use with the **-file** argument, you may have more than one entry for a volume; the command ignores any additional lines for

the volume, but if you wish to exclude a volume you must remove all existing entries for that volume in the file. You do not need to remove the number sign (**#**) and the information that follows it; the command ignores any characters that follow the third argument on a line.

When the **-n** flag is included, no task ID and job ID numbers are reported because none are assigned.

Notes:

The amount of time required for the **backup volsetrestore** command to complete depends on the number of volumes to be restored. However, a restore operation that includes a large number of volumes can take hours to complete. To reduce the amount of time required for the operation, you can execute multiple instances of the command simultaneously, specifying disjoint volume sets with each command if you use the **-name** argument, or indicating files that list different volumes with each command if you use the **-file** argument. Depending on how the volumes to be restored were dumped to tape, specifying disjoint volume sets can also enable you to make the most efficient use of your backup tapes when many volumes need to be restored.

Examples:

The following restores all volumes included in entries in the volume set named *data.restore*, which was created expressly to restore data to a pair of file server machines on which all data was corrupted due to a software error. All volumes are restored to the sites recorded in their entries in the VLDB.

```
% backup vols data.restore
Starting restore
backup: task ID of restore operation: 112
backup: Finished doing restore
```

The following restores all volumes that have entries in the file named */tmp/restore*:

```
% backup vols -file /tmp/restore -p 1
Starting restore
backup: task ID of restore operation: 113
backup: Finished doing restore
```

The */tmp/restore* file has the following contents:

```
fs1.abc.com b user.morin
fs1.abc.com b user.vijay
fs1.abc.com b user.pierette
fs2.abc.com c user.frost
fs2.abc.com c user.wvh
fs2.abc.com c user.pbill
 . . .
 . . .
```

Privilege Required:

The issuer must be listed in the */usr/afs/etc/UserList* file for the specified cell.

5.6.4. The backup labeltape Command

AFS 3.4 Changes

The **backup labeltape** command allows users to label tapes explicitly with a permanent name in AFS 3.4. The Backup System remembers this permanent name across dumps in the Backup System. If a user supplies a permanent name to a tape with the **backup labeltape** command using the **-name** argument, the Backup System remembers the permanent name as the tape is re-used or recycled. The Backup System remembers the permanent name until the user explicitly changes it using the **backup labeltape** command with the **-name** argument. It is recommended that permanent tape names be unique since that is the tape name which is recorded in the Backup Database and which is requested on backup restore operations. The **-name** argument has been changed to

-name <*permanent_tape_name*>

where *permanent_tape_name* specifies the permanent name that the user assigns to the tape.

The new syntax for the **backup labeltape** command is

backup labeltape [**-name** <*permanent_tape_name*>]
[**-size** <*tape size in Kbytes, defaults to size in tapeconfig*>]
[**-portoffset** <*TC port offset*>] [**-localauth**] [**-cell** <*cell name*>] [**-help**]

If the user does not explicitly name a tape with a permanent name, AFS assigns a non-permanent name to the tape as it did previously. The Backup System produces this non-permanent name by concatenating the volume set and dump level with a tape sequence (for example, *guests.monthly.3*). This name is not permanent and changes whenever the tape label is re-written by a **backup** command (for example, when using the **backup dump**, **backup labeltape**, and **backup savedb** commands). The AFS-assigned non-permanent name is listed in the `AFS name` field in the output resulting from the **backup readlabel** command.

As in AFS 3.3, the **backup labeltape** command overwrites the existing tape label and destroys any data on the tape; for example, when the user wishes to recycle a tape that was previously used to store other dumped volumes. If the **-name** argument is not supplied with the **backup labeltape** command, the tape keeps its permanent name. A user can enter a null name to remove the permanent name as shown in the following example:

backup labeltape -name ""

5.6.5. The backup readlabel Command

AFS 3.4 Changes

In AFS 3.4, the **backup readlabel** command lists the permanent tape name, which users can assign with the **backup labeltape** command, and the AFS tape name, which is assigned by AFS, in the output of the command. If you designated a permanent tape name with the **backup labeltape** command, the command displays the permanent tape name (`tape name`) and the AFS-assigned tape name (`AFS tape name`), as shown in the following output:

```
Tape label
tape name = monthly.guest.dump
AFS tape name = guests.monthly.3
creationTime = Fri Nov 11 05:31:32 1994
cell = abc.com
size = 2097152 Kbytes
dump path = /monthly
dump id = 697065340
useCount = 44
-- End of tape label --
```

If you did not designate a permanent tape name, the **backup readlabel** command displays only the AFS-assigned tape name, as shown in the following output:

```
Tape label
AFS tape name = guests.monthly.3
creationTime = Fri Nov 11 05:31:32 1994
cell = abc.com
size = 2097152 Kbytes
dump path = /monthly
dump id = 697065340
useCount = 44
-- End of tape label --
```

6. The bos Commands

This chapter describes changes to the **bos** command suite for AFS 3.4. In particular, AFS 3.4 contains changes to the **bos addkey** command.

These changes are marked with the heading "**AFS 3.4 Changes**."

This chapter also contains changes from the AFS 3.3 release that have not been incorporated into the full AFS documentation set. These changes are marked with the heading "**AFS 3.3 Changes**."

6.1. The bos addkey Command

AFS 3.4 Changes

In AFS 3.4, the **bos addkey** command has been updated to prompt you twice for the key in the same manner that you are prompted to enter a password during a password change. The prompt follows:

```
# bos addkey -server <machine name> -kvno 0
Input key:
Retype input key:
```

If you type the key incorrectly the second time, the command displays the following error message and exits without adding a new key:

```
Input key mismatch
```

In AFS 3.3, if the **-key** argument was not provided on the command line, the command prompted you to enter the key in the same way that you are prompted to enter a password during a password change.

The **bos addkey** command has also been updated to prevent you from reusing a key version number currently found in the */usr/afs/etc/KeyFile* file. This ensures that users who still have tickets sealed with the current key are not prevented from communicating with the file server because the current key is overwritten with a new key.

AFS 3.3 Changes

In earlier versions of AFS, the **bos addkey** command required the entry of a new key on the command line. This approach posed many obvious security problems because the key was visible on the screen, in the process entry for the **ps** command, and in the command **history** of the issuer's shell.

To prevent these security risks, the **-key** argument has been made optional on the **bos addkey** command. If you do not provide the argument on the command line, you are prompted to enter the key in the same way that you are prompted to enter a password during a password change.

The new syntax of the **bos addkey** command follows:

bos addkey -server <*machine name*> [**-key** <*key*>] **-kvno** <*key version number*> [**-cell** <*cell name*>] [**-noauth**] [**-localauth**] [**-help**]

6.2. The bos status Command

AFS 3.3 Changes

The **-long** flag with the **bos status** command now displays the pathnames of notifier programs associated with processes via the **bos create** command when the processes terminate.

7. The fs Commands

This chapter describes changes to the **fs** command suite for AFS 3.4. In particular, AFS 3.4 contains a new command, **fs storebehind**, and changes to the following **fs** commands:

- The **fs setserverprefs** command
- The **fs getserverprefs** command
- The **fs checkservers** command
- The **fs exportafs** command

These changes are marked with the heading "**AFS 3.4 Changes**."

This chapter also contains changes from the AFS 3.3 release that have not been incorporated into the full AFS documentation set. These changes are marked with the heading "**AFS 3.3 Changes**."

7.1. Setting and Getting Cache Manager Server Preferences

AFS 3.4 Changes

AFS 3.4 supports default preferences and ranks for Volume Location (VL) servers, in addition to default preferences and ranks for file servers. These preferences are file server or VL server machines from which the client machine's Cache Manager prefers to access ReadOnly volumes or VLDBs, respectively. The Cache Manager bases its preference on a numerical rank; the lower the numerical rank, the greater the Cache Manager's preference for selecting that file server or VL server. The numerical rank can be set by the Cache Manager or by the user explicitly with the **fs setserverprefs** command.

Each Cache Manager stores a table of preferences for file server and VL server machines. A preference is stored as a file server or VL server machine's Internet Protocol (IP) address and an associated "rank." A file server or VL server machine's rank is an integer in the range from *1* to *65,534* that determines the Cache Manager's preference for selecting the server machine when the Cache Manager must access a ReadOnly replica or VLDB that resides on it. Preferences can bias the Cache Manager to access ReadOnly replicas or VLDBs from machines that are "near" rather than from those that are "distant" ("near" and "far" refer to network distance rather than physical distance). Effective preferences can generally reduce network traffic and result in faster access of data.

A replicated AFS volume typically has multiple ReadOnly copies. Each ReadOnly copy provides the same data, but each resides on a different file server machine. When the Cache Manager needs to access a ReadOnly copy or VLDB, it first contacts the VL server to determine the names of the file server machines on which the ReadOnly copy resides or it first contacts the

VLDB on the preferred VL server machine. The Cache Manager then checks its internal table to determine the rank associated with each of the machines. After comparing the ranks of the machines, the Cache Manager attempts to access the ReadOnly copy on the machine that has the lowest integer rank.

If the Cache Manager cannot access the ReadOnly copy or VLDB on the machine with the lowest rank (possibly because of a server process, machine, or network outage), the Cache Manager attempts to access the ReadOnly copy or VLDB on the machine with the next lowest rank. The Cache Manager continues in this manner until it either accesses the ReadOnly copy or VLDB, or determines that all of the file server machines on which the ReadOnly copy is housed or all of the VL server machines on which the VLDB is housed are unavailable.

The Cache Manager assigns preferences to file server machines in the local cell and to machines from foreign cells as it accesses files from volumes on those machines; the Cache Manager can only assign preferences to VL server machines in the local cell. The Cache Manager stores the preferences as IP addresses and associated ranks in the kernel of the client machine. Because they are stored in the kernel of the client machine, the preferences are recalculated when the client machine is rebooted. To rebuild its preferences following initialization, the Cache Manager assigns a default rank to each file server or VL server machine that it contacts or that houses a copy of a ReadOnly volume from which it accesses data. To display the Cache Manager's current set of file server or VL server machine preferences, use the **fs getserverprefs** command. By default, the command displays its output on standard output, but you can direct the output to a specified file.

If the Cache Manager encounters a bad ReadOnly volume (possibly because the volume was deleted by the **salvager** process), the Cache Manager moves the machine address and rank of that volume/server combination to the end of its preferences list. This prevents the Cache Manager from repeatedly attempting to access a bad ReadOnly volume. If the Cache Manager is unable to access a particular database server machine, it will move that database server machine to the end of the preferences list.

AFS provides commands for displaying and modifying a Cache Manager's preferences for server machines. The **fs getserverprefs** command can be used to display a Cache Manager's preferences for file server and VL server machines. The command's **-file** argument can be used to direct the output to a file rather than to standard output (stdout), which is the default.

The **fs setserverprefs** command can be used to set the preference rank for one or more file server or VL server machines. Preferences are specified with the command as pairs of file server machines and ranks. The first value is the name or IP address of the file server or VL server machine; the second is the rank to be associated with the machine.

A Cache Manager's file server preferences are potentially derived from four different sources:

- Explicitly specified with the **-servers** argument of the **fs setserverprefs** command
- Received from an input file specified with the **-file** argument of the **fs setserverprefs** command
- Received from standard input specified with the **-stdin** flag of the **fs setserverprefs** command

- Default values selected by the Cache Manager on the basis of the algorithm described in Section 7.1.1

The arguments and flag are not mutually exclusive, so multiple preferences can be specified with one issuance of the command. You can include the **fs setserverprefs** command in a machine's initialization file (the */etc/rc* file or equivalent) to load server preferences at reboot.

Otherwise, the Cache Manager uses the preferences input via the **fs setserverprefs** command, when they exist, over existing default preferences. The Cache Manager uses the last preference entered from the combined input for a particular server machine. For example, consider the following sequential input is given for the file server *fs1.abc.com* via the **fs setserverprefs** command:

- The **-servers** argument sets *fs1.abc.com* to a rank of *20000*
- The file specified by the **-file** argument sets *fs1.abc.com* to a rank of *25000*
- The **-stdin** flag produces an entry for *fs1.abc.com* set to a rank of *21000*

The Cache Manager previously had a rank for the file server *fs1.abc.com* of *25000*. The resulting rank for *fs1.abc.com* is *21000* because the Cache Manager uses the last rank entered with the **fs setserverprefs** command (with the **-stdin** flag).

A Cache Manager's VL server preferences are potentially derived from two different sources:

- Explicitly specified with the **-vlservers** argument of the **fs setserverprefs** command
- Default values randomly selected by the Cache Manager

The **fs setserverprefs** command contains a **-vlservers** argument that allows you to explicitly set VL server preferences and ranks. The **fs getserverprefs** command contains a **-vlservers** flag that allows the Cache Manager's VL server preferences and ranks to be displayed. AFS 3.4 supports default preferences for VL servers only; it does not support preferences for Protection, Authentication, or Backup Database servers.

7.1.1. The fs setserverprefs Command

AFS 3.4 Changes

In addition to file server preferences, the **fs setserverprefs** command can set preferences for Volume Location (VL) servers via the **-vlservers** argument. This section contains the revised command reference page for the **fs setserverprefs** command.

fs setserverprefs [**-servers** <*fileserver names and ranks*>$^{+}$]
[**-vlservers** <*VL server names and ranks*>$^{+}$]
[**-file** <*input from named file*>] [**-stdin**] [**-help**]

Acceptable Abbreviations/Aliases:

fs sets [**-se** <*fileserver names and ranks*>+] [**-vl** <*VL server names and ranks*>+] [**-f** <*input from named file*>] [**-st**] [**-h**]

fs sp [**-se** <*fileserver names and ranks*>+] [**-vl** <*VL server names and ranks*>+] [**-f** <*input from named file*>] [**-st**] [**-h**]

Description:

Sets the Cache Manager's preferences for one or more file server or VL server machines. A preference consists of the hostname or IP address of a file server or VL server machine followed by the machine's numerical rank; the machine's numerical rank is an integer in the range from *1* to *65,534*. A file server machine's or VL server machine's rank determines the Cache Manager's preference for electing to access ReadOnly volumes that reside on the file server machine over ReadOnly volumes that reside on other file server machines or for electing to access the Volume Location Database (VLDB) that resides on the VL server machine over VLDBs that reside on other VL server machines.

When the Cache Manager needs to access data from the ReadOnly version of a volume or to access data from a VLDB, it first contacts the VL server and accesses the VLDB to determine the names of the file server machines on which a ReadOnly volume resides. If multiple file server machines house the ReadOnly volume, the Cache Manager consults its preferences for file server machines and VL server machines and attempts to access the ReadOnly volume on the file server or VLDB on the VL server machine that has the lowest recorded rank. If multiple file server machines have the same rank, the Cache Manager selects them in the order in which it received their names from the VL server.

If the Cache Manager cannot access the ReadOnly volume on the file server machine or the VLDB on the VL server machine with the lowest rank (possibly because the machine or the network on which the machine resides is down), the Cache Manager attempts to access the ReadOnly volume on the file server machine or the VLDB on the VL server machine that has the next-lowest rank. The Cache Manager continues in this manner until it either succeeds in accessing the ReadOnly volume or VLDB or determines that all of the file server machines on which the ReadOnly volume is stored or that all of the VL server machines on which the VLDB is stored are unavailable.

The Cache Manager stores its server preferences in the kernel of the local machine. The preferences are lost each time the Cache Manager is initialized with the **afsd** command (each time the client machine is rebooted). After it is initialized, the Cache Manager rebuilds its collection of preferences by assigning a rank to each file server machine that it contacts or that houses a ReadOnly volume from which it accesses data. The Cache Manager makes no distinction between preferences for server machines from the local cell and those for server machines from a foreign cell. Default preferences bias the Cache Manager to select ReadOnly volumes from file server machines that are in the same subnetwork or network as the local machine. You can use the **fs setserverprefs** command to alter the default preferences.

If the **fs setserverprefs** command specifies a rank for a file server or VL server machine for which the Cache Manager has no rank, the command defines the machine's initial rank. If the command specifies a rank for a machine for which the Cache Manager already has a rank, the command changes the current rank to match the specified rank. You can include the **fs setserverprefs** command in a machine's initialization file to load a predefined collection of server preferences when the machine is rebooted.

Specifying Preferences:

Using the **fs setserverprefs** command, you specify preferences as pairs of values. The first value of the pair is the hostname (for example, *fs1.abc.com*) or IP address, in dotted decimal format, of a file server or VL server machine; the second value of the pair is the machine's numerical rank, an integer in the range from *1* to *65,534*.

To minimize the chances that multiple file server or VL server machines are assigned the same rank and to ensure maximum load balancing among file server and VL server machines, the Cache Manager adds a random number in the range from *0* (zero) to *15* to each rank that you specify. For example, if you specify a rank of *15,000* to a server machine, the Cache Manager records the rank as an integer in the range from *15,000* to *15,015*.

You can specify pairs of file server machines and their ranks

- On the command line via the **-servers** argument. Use the argument to tune the preferences manually in response to system or network adjustments.
- From a file via the **-file** argument. Use the option to configure one or more Cache Managers with a fixed set of preferences, specifying a file created manually or generated automatically. You can use the **fs getserverprefs** command to generate a file of preferences that has the proper format.
- From standard input via the **-stdin** flag. Use the option to pipe preferences to the command from a user-defined process that generates preferences in an acceptable format.

Note: You can specify a unique preference for any of the multi-homed addresses available at a file server machine using the **fs setserverprefs** command.

You cannot specify VL server preferences with the **-file** argument or the **-stdin** flag. You can specify pairs of VL server machines and their ranks explicitly via the **-vlservers** argument only.

The **-servers**, **-file**, and **-stdin** arguments are not mutually exclusive. You can include any combination of these options with the command. Note that the command does not verify hostnames or IP addresses specified with any of its options. You can add a preference for an invalid hostname or IP address; the Cache Manager stores such preferences in the kernel, but it ignores them (the Cache Manager never needs to consult such preferences).

Allowing the Cache Manager to Assign Preferences to File Server Machines:

The Cache Manager bases default ranks that it calculates on IP addresses rather than on actual physical considerations such as location or distance. It uses the following heuristic to calculate default ranks *for file server machines only*:

- If the local machine is also a file server machine, the machine receives an initial rank of *5000*.

- Each file server machine in the same subnetwork as the local machine receives an initial rank of *20,000*.
- Each file server machine in the same network as the local machine receives an initial rank of *30,000*.
- Each file server machine on the distant ends of point-to-point links from the local machine receives an initial rank of *30,000*.
- Each file server machine in a different network from the local machine receives an initial rank of *40,000*.
- Each file server machine for which the Cache Manager can determine no network information receives an initial rank of *40,000*.

As it does with ranks specified with the **fs setserverprefs** command, the Cache Manager adds a random number in the range from *0* (zero) to *15* to each initial rank that it determines. For example, while it assigns an initial rank of *20,000* to a file server machine in the same subnetwork as the local machine, the Cache Manager records the actual rank as an integer in the range from *20,000* to *20,015*.

Arguments:

-servers Specifies one or more pairs of file server machines and their respective ranks. Separate each machine specification and each rank with one or more spaces. Refer to the section of this command reference page titled "Specifying Preferences" for information about specifying file server machines and ranks.

-vlservers Specifies one or more pairs of VL server machines and their respective ranks. Separate each machine specification and each rank with one or more spaces. Refer to the section of this command reference page titled "Specifying Preferences" for information about specifying VL server machines and ranks.

-file Specifies the full pathname of a file from which the command is to read pairs of file server machines and their respective ranks. Separate each machine specification from its rank with one or more spaces, and include each paired machine specification and rank on a separate line. You cannot specify VL server preferences with the **-file** argument. Refer to the section of this command reference page titled "Specifying Preferences" for information about specifying file server machines and ranks.

-stdin Directs the command to read pairs of file server machines and their respective ranks from standard input (stdin). Separate each machine specification and each rank with one or more spaces. Refer to the section of this command reference page titled "Specifying Preferences" for information about specifying file server machines and ranks.

-help Prints the online help for this command. All other valid options specified with this option are ignored.

Examples:

1. The following command uses the **-servers** argument to set the Cache Manager's preferences for the file server machines named *fs3.abc.com* and *fs4.abc.com*, the latter of which is specified by its IP address, *128.21.18.100*. The two file server machines reside in a different subnetwork in the same network as the local machine, so by default the Cache Manager would assign each a rank of *30,000* plus an integer in the range from *0* to *15*. To make the Cache Manager prefer these file server machines over file server machines in other subnetworks in the local network, you can use the **fs setserverprefs** command to assign these machines ranks of *25,000*, to which the Cache Manager adds an integer in the range from *0* to *15*.

   ```
   # fs sets -se fs3.abc.com 25000 128.21.18.100 25000
   ```

2. The following command uses the **-servers** argument to set the Cache Manager's preferences for the same two file server machines, but it also uses the **-file** argument to read a collection of preferences from a file that resides on the local machine at */etc/fs.prefs*:

   ```
   # fs sets -se fs3.abc.com 25000 128.21.18.100 25000 -f /etc/fs.prefs
   ```

   The file */etc/fs.prefs* has the following contents and format:

   ```
   128.21.16.214     7500
   128.21.16.212     7500
   121.86.33.41     39000
   121.86.33.34     39000
   121.86.33.36     41000
   121.86.33.37     41000
   ```

3. The following command uses the **-stdin** flag to read preferences from standard input (stdin). The preferences are piped to the command from a program, **calc_prefs**, which was written by the issuer to calculate preferences based on values significant to the local cell.

   ```
   # calc_prefs | fs sets -stdin
   ```

4. The following command uses the **-vlservers** argument to set the Cache Manager's preferences for the VL server machines named *fs1.abc.com*, *fs3.abc.com*, and *fs4.abc.com* with ranks of *1*, *30000*, and *65534*, respectively:

   ```
   # fs sets -vl fs1.abc.com 1 fs3.abc.com 30000 fs4.abc.com 65534
   ```

   If you want VL server preferences to survive a reboot, you can add the **fs setserverprefs** command to your startup files on your client machine.

Privilege Required:

The issuer must be logged in as "root" on the local machine.

7.1.2. The fs getserverprefs Command

AFS 3.4 Changes

In AFS 3.4, the **fs getserverprefs** command can display preferences for Volume Location (VL) servers via the **-vlservers** flag, in addition to file servers. This section contains the revised command reference page for the **fs getserverprefs** command.

fs getserverprefs [**-file** <*output to named file*>] [**-numeric**] [**-vlservers**] [**-help**]

Acceptable Abbreviations/Aliases:

fs gets [**-f** <*output to named file*>] [**-n**] [**-vl**] [**-h**]
fs gp [**-f** <*output to named file*>] [**-n**] [**-vl**] [**-h**]

Description:

Displays the Cache Manager's preferences for file server or VL server machines. A preference consists of the hostname or IP address of a file server or VL server machine, followed by the machine's rank. The rank is an integer in the range from *1* to *65,534*.

A server machine's rank determines the Cache Manager's preference for accessing data from ReadOnly volumes that reside on a file server or for accessing data from the Volume Location Database (VLDB) that resides on a VL server. When the Cache Manager needs to access data from the ReadOnly version of a volume or data from the VLDB, the Cache Manager determines which file server machines house ReadOnly versions by consulting the VLDB and which machines are VL servers that house a VLDB and then attempts to access the ReadOnly volume or VLDB on the machine that has the lowest recorded rank.

If it cannot access the ReadOnly volume on the file server machine or the VLDB on the VL server machine that has the lowest rank (possibly because the machine or the network on which the machine resides is down), the Cache Manager attempts to access the ReadOnly volume or the VLDB on the machine that has the next-lowest rank. It continues in this manner until it either accesses the ReadOnly volume or VLDB, or determines that all of the file server or VL server machines on which ReadOnly versions or the VLDB resides are unavailable.

The Cache Manager records a rank for each file server machine that it contacts or that houses a ReadOnly volume from which it has accessed data. The Cache Manager also records each rank specified for a file server machine by the **fs setserverprefs** command. The Cache Manager stores the IP addresses and ranks in the kernel of the local machine. The Cache Manager makes no distinction between preferences for machines from the local cell and those for machines from a foreign cell. Refer to the Description section of the **fs setserverprefs** command for a discussion on how the Cache Manager assigns preferences to file servers.

The **fs getserverprefs** command displays file server rank information on standard output (stdout) by default. Use the **-file** argument to specify the complete pathname of a file to which the command is to write its output. If you include the **-file** argument, the command displays no output on standard output. To display VL server preferences, you must specify the **-vlservers** flag with the **fs getserverprefs** command.

Arguments:

-file Specifies the full pathname of a file to which the server preferences (either file server or VL server) are to be written. If the specified file already exists, the command overwrites it. The command fails if the specified pathname is invalid. Omit this option to display the preferences on standard output.

-numeric Directs the command to display the IP addresses rather than the hostnames of the file server or VL server machines in the server preferences that it reports. Omit this option to display the hostname (for example, *fs1.abc.com*) of each machine.

-vlservers Directs the command to display the hostnames and rankings of VL server machines rather than file server machines.

-help Prints the online help for this command. All other valid options specified with this option are ignored.

Output:

The **fs getserverprefs** command displays a separate line of output for each file server or VL server machine for which it maintains a preference. By default, each line consists of the name of a file server machine followed by the machine's rank, as follows:

hostname rank

where *hostname* is the name of a file server machine, and *rank* is the rank associated with the machine. If the **-numeric** flag is included with the command, the command displays the IP address, in dotted decimal format, of each file server machine instead of the machine's name. The command also displays the IP address of any machine whose name it cannot determine (for example, if a network outage prevents it from resolving the address into the name).

Examples:

1. The following command displays the preferences (the list of file server machines and their respective ranks) associated with the Cache Manager on the local machine. The local machine belongs to the AFS cell named *abc.com*; the ranks of the file server machines from the *abc.com* cell are lower than the ranks of the file server machines from the foreign cell, *def.com*. The command shows the IP addresses, not the names, of two machines from the foreign cell because it cannot currently determine their names.

```
% fs gets
fs2.abc.com         20007
fs3.abc.com         30002
fs1.abc.com         20011
fs4.abc.com         30010
server1.def.com     40002
121.86.33.34        40000
server6.def.com     40012
121.86.33.37        40005
```

2. The following command displays the same Cache Manager's preferences, but the **-numeric** flag is included with the command to display the IP addresses rather than the names of all file server machines. The IP address of the local machine is *128.21.16.221*. The two file server machines on the same subnetwork as the local machine have ranks of *20,007* and *20,011*; the two file server machines on a different subnetwork in the same network as the local machine have ranks of *30,002* and *30,010*; the remainder of the file server machines are in a different network, so their ranks range from *40,000* to *40,012*.

```
% fs gets -n
128.21.16.214      20007
128.21.18.99       30002
128.21.16.212      20011
128.21.18.100      30010
121.86.33.41       40002
121.86.33.34       40000
121.86.33.36       40012
121.86.33.37       40005
```

3. The following command displays the preferences for VL servers (the list of VL server machines and their respective ranks) associated with the Cache Manager on the local machine. The **-vlservers** flag is required to display the preferences for VL servers.

```
% fs gets -vl
fs2.abc.com            1
fs3.abc.com        30000
fs1.abc.com        65534
```

Privilege Required:

No privileges are required.

7.2. The fs storebehind Command (New)

AFS 3.4 Changes

The **fs storebehind** command is a new command for data storage from the Cache Manager to the file server. The **fs storebehind** command performs a delayed asynchronous write to the file server for specified file(s). The **fs storebehind** command allows the Cache Manager to return control to a closing application program before a the final chunk of a file is completely transferred to the file server. You can specify a particular file or all files to be closed (by using the **-files** argument or using the command without the **-files** argument, respectively) after control has been returned to the closing application program. You can also indicate the amount of data that is stored to the file server after returning control to the closing application. If you issue the **fs storebehind** command with or without arguments, the command displays the current status of any files in the Cache Manager. This command is useful for accessing and writing very large files in AFS. For example, if you have finished working on a large database file, you can issue

the **fs storebehind** command to close the file in the background and asynchronously write it to the file server while you move on to work on something else.

Note: If the delayed close and write on the specified file fails, the **fs storebehind** command does not notify the application or inform the application that the close and write operations failed. Make certain that you check the disk quota for the volume to which the specified file belongs and that you do not exceed the disk quota when using the **fs storebehind** command; if you exceed the disk quota when writing the specified file, the portion of the file that exceeds the disk quota will be lost.

The syntax for the **fs storebehind** command follows:

fs storebehind [**-kbytes** <*asynchrony for specified names*>] [**-files** <*specific pathnames*>+] [**-allfiles** <*new default (KB)*>] [**-verbose**] [**-help**]

In AFS 3.4, the default for the Cache Manager (**afsd**) operation is to complete the transfer of a closed file to the file server before returning control to the application invoking the close. In AFS 3.3, the default for the Cache Manager operation was to return control to a closing application program before the final chunk of a file was completely written to the file server.

The default functionality of the **fs storebehind** command was previously provided by the default setting of the **afsd** command, while the default functionality of the AFS 3.4 Cache Manager was previously provided by the **-waitclose** flag of the **afsd** command. For this reason, the **-waitclose** flag has no effect on the operation of the Cache Manager in AFS 3.4.

Arguments:

-kbytes Specifies the amount of data that is written to the file server after returning control to the closing application program on the client machine. For example, setting this argument to *100* kbytes means that there can be up to 100 kbytes of data still waiting to be written to the file server after returning control to the closing application program. A value of *0* means that no data is written to the file server after returning control to the closing application. The default value is *60*.

-files Specifies a particular file or files to be closed and written to the file server after returning control to the closing application program. The specified file is closed in this manner only as long as it is cached. When the file is no longer cached, AFS returns to the default Cache Manager operation; that is, files are closed completely and written to the file server before control is returned to the closing application program.

-allfiles Specifies that all files are to be closed and written to the file server after returning control to the closing application program. The initial default value is *0*, which means files are closed completely and written to the file server before control is returned to the closing application program. The values for this argument range from *0* to the maximum AFS file size (in AFS 3.4, the maximum file size is 2 GB).

-verbose Tells the file server to report on what it is doing as it executes the command.

-help Prints the online help entry for this command. All other valid options specified with this option are ignored.

Example:

1. The following command uses the **-servers** argument to set the Cache Manager's preferences for the file server machines named *fs3.abc.com* and *fs4.abc.com*, the latter of which is specified by its IP address, *128.21.18.100*. The two file server machines reside in a different subnetwork in the same network as the local machine, so by default the Cache Manager would assign each a rank of *30,000* plus an integer in the range from *0* to *15*. To make the Cache Manager prefer these file server machines over file server machines in other subnetworks in the local network, you can use the **fs setserverprefs** command to assign these machines ranks of *25,000*, to which the Cache Manager adds an integer in the range from *0* to *15*.

```
# fs storebehind -se fs3.abc.com 25000 128.21.18.100 25000
```

Privilege Required:

No privileges are required.

7.3. The fs checkservers Command

AFS 3.4 Changes

Through the **fs checkservers** command, an AFS client machine can probe file server machines in a specified cell or all cells to determine if those file servers are available. Each client machine reports any file servers that did not respond to its probe. The output of this command has been modified for AFS 3.4.

Each AFS client machine probes file server machines to determine if any file servers it has accessed in the requested cell since the client has been "up" are available. Specifically, each client machine probes those file servers that house data that the client has cached in its local cell by default or a cell specified by the **-cell** argument. If a file server does not respond to a probe, the client assumes the file server is unavailable due to server or network problems. If you want the time interval specified by the **-interval** argument to survive a reboot, you can put the **fs checkservers** command in the startup files.

AFS 3.3 Changes

In previous versions of AFS, the interval between probes was automatically set to 3 minutes. For some client machines, a 3-minute probe interval may be too long or too short. Therefore, a new argument, **-interval**, has been added to the **fs checkservers** command to allow you to specifically set this interval. The default value is 180 seconds; the maximum and minimum values are 10 minutes (600 seconds) and 1 second, respectively. To check the current length of the interval, specify 0 with the **-interval** argument.

Only the user "root" can issue the **fs checkservers** command with the **-interval** argument. Once set, the probe interval remains set until it is changed via this command or until the client machine is rebooted (at which time it returns to the default setting).

7.4. The fs exportafs Command

AFS 3.4 Changes

Several modifications have been made to the **fs exportafs** command syntax for AFS 3.4.

- The **-state** argument of the **fs exportafs** command has been changed to the **-start** argument since this argument is used to start and stop the NFS/AFS Translator.
- The **-uidcheck** and **-submounts** arguments of the **fs exportafs** command now support an *on* or *off* selection. When issuing any of these three arguments, you must select *on* or *off*. If you do not specify a certain argument, the value of that argument is either of the following:
  - — If the argument has never been specified before the default value
  - — The value assigned to it in a previous execution of the **fs exportafs** command
- The **-noconvert** argument has been changed to the **-convert** argument so that it is more compatible with the *on* and *off* selections for the other switches in the **fs exportafs** command.

The modified **fs exportafs** command syntax follows:

fs exportafs -type <*exporter name*> [**-start** <*start/stop translator (on | off)*>]
[**-convert** <*convert from afs to unix mode (on | off)*>]
[**-uidcheck** <*run on strict 'uid check' mode (on | off)*>]
[**-submounts** <*allow nfs mounts to subdirs of /afs/ . . (on | off)*>] [**-help**]

Arguments:

-type Names the alternate file system for which the setting is to be changed or reported. Only lowercase letters are acceptable. The only legal value is *nfs*.

-start Controls whether the workstation is accessible as a server of the non-AFS file system or not. The legal values are *on*, which enables the workstation as a server, and *off*, which makes it inaccessible as a server. If the issuer omits this argument, the output reports the current setting.

-convert Determines whether the "group" and "other" UNIX mode bits on exported files and directories are converted to match the "user" UNIX mode bits. The default value of this argument is *on*. When the **-convert** argument is *on*, the "group" and "other" UNIX mode bits are made to match the "user" UNIX mode bits. When the **-convert** argument is *off*, the bits are left as they are in AFS.

-uidcheck Prevents users from deleting the tokens of other users with the **knfs** command. The default value of this argument is *off*. When the **-uidcheck** argument is *on*, the command prevents users from deleting the tokens of other users with the **knfs** command. When the **-uidcheck** argument is *off*, the command does not prevent users from deleting the tokens of other users with the **knfs** command. You can use this feature only if your users have the same UIDs in the */etc/passwd* files (or equivalent) on both the NFS client and the NFS server (the NFS/AFS Translator machine).

-submounts Allows a user to NFS-mount any directory in the AFS filespace, rather than */afs*. The default value of this argument is *off*. When the **-submounts** argument is *on*, the command allows a user to NFS-mount any directory in the AFS filespace, rather than */afs*. When the **-submounts** argument is *off*, the command does not allow a user to NFS-mount any directory in the AFS filespace other than */afs*.

-help Prints the online help entry for this command. Do not provide any other arguments or flags with this one.

To find out the current status of the **fs exportafs** command arguments, execute the following command:

```
% fs exportafs nfs
```

To reset all of the arguments to their default values, execute the following commands:

```
% fs exportafs nfs off
% fs exportafs nfs on
```

AFS 3.3 Changes

In the past, when using the NFS/AFS Translator, it was often easy to assign a token mistakenly to the wrong user or to delete the wrong user token by entering the wrong UID with the **knfs** command. A new flag, **-uidcheck**, has been added to the **fs exportafs** command that, when used, prevents users from assigning and deleting the tokens of other users with the **knfs** command. You can use this feature only if your users have the same UID's in the */etc/passwd* files (or equivalent) on both the NFS client and the NFS server (the NFS/AFS Translator machine).

Another new flag, **-submounts**, has also been added to the **fs exportafs** command. This flag allows a user to NFS-mount any directory in the AFS filespace, rather than just */afs*. To allow NFS-mounting at any point in the AFS filespace, specify the **-submounts** flag with the command; to restrict NFS-mounting to the */afs* directory only, omit the **-submounts** flag.

7.5. The -cell Argument on fs Commands

AFS 3.3 Changes

The **-cell** argument on **fs** commands now fully expands shortened versions of a cell name provided the shortened version is unique. The Cache Manager determines if a shortened version is unique by consulting the *CellServDB* file and the list of new cells identified using the **fs newcell** command.

The following **fs** commands are affected by this change:

- The **fs checkservers** command
- The **fs getcellstatus** command
- The **fs mkmount** command
- The **fs setcell** command

7.6. The fs copyacl, fs listacl, and fs setacl Commands

AFS 3.3 Changes

Two new flags, **-id** and **-if**, have been added to the **fs copyacl**, **fs listacl**, and **fs setacl** commands to allow AFS interaction with Transarc Corporation's AFS/DFS Migration Toolkit™. The new flags provide no functionality outside of the Migration Toolkit.

The new syntax of the commands follows:

fs copyacl -fromdir <*source directory (or DFS file)*>
-todir <*destination directory (or DFS file)*>$^{+}$ [**-clear**] [**-id**] [**-if**] [**-help**]

fs listacl [**-path** <*dir/file path*>$^{+}$] [**-id**] [**-if**] [**-help**]

fs setacl -dir <*directory*>$^{+}$ **-acl** <*access list entries*>$^{+}$ [**-clear**] [**-negative**] [**-id**] [**-if**] [**-help**]

Refer to the *AFS/DFS Migration Toolkit Administration Guide and Reference* for more information about these commands.

7.7. The fs newcell Command

AFS 3.3 Changes

A new argument, **-linkedcell**, has been added to the **fs newcell** command to allow AFS interaction with Transarc Corporation's AFS/DFS Migration Toolkit.

The new syntax of the command follows:

fs newcell -name <*cell name*> **-servers** <*primary servers*>$^{+}$ [**-linkedcell** <*linked cell name*>] [**-help**]

Refer to the *AFS/DFS Migration Toolkit Administration Guide and Reference* for more information about the **fs newcell** command.

8. The fstrace Commands (New)

AFS 3.4 Changes

The **fstrace** command suite allows you to record, or *trace*, in detail the processes executed by the AFS Cache Manager (**cm**). These processes, or *events*, executed by the Cache Manager comprise the **cm** *event set*. Examples of **cm** events are fetching files and looking up information for an **ls** listing.

When AFS tracing is enabled, each time a **cm** event occurs, a message is written to the trace log, **cmfx**. To diagnose a problem, you may read the output of the trace log and analyze the processes executed by the Cache Manager. The trace log has a default size of 60K; however, its size can be increased or decreased.

Following are the **fstrace** commands and their respective functions:

- The **fstrace apropos** command provides a short description of commands.
- The **fstrace clear** command clears the trace log.
- The **fstrace dump** command dumps the contents of the trace log.
- The **fstrace help** command provides a description and syntax for commands.
- The **fstrace lslog** command lists information about the trace log.
- The **fstrace lsset** command lists information about the event set.
- The **fstrace setlog** command changes the size of the trace log.
- The **fstrace setset** command sets the state of the event set.

All **fstrace** commands, except **fstrace apropos** and **fstrace help**, require you to be logged in as "root" on the local machine. Before issuing an **fstrace** command, verify that you have the necessary privilege.

To use **fstrace**, you must first enable tracing and reserve, or *allocate*, space for the trace log with the **fstrace setset** command. With this command, you can set the **cm** event set to one of three states to enable or disable tracing for the event set and to allocate or deallocate space for the trace log in the kernel:

active Enables tracing for the event set and allocates space for the trace log.

inactive Temporarily disables tracing for the event set; however, the event set continues to allocate space occupied by the log to which it sends data.

dormant Disables tracing for the event set; furthermore, the event set releases the space occupied by the log to which it sends data. When the **cm** event set that sends data to the **cmfx** trace log is in this state, the space allocated for that log is freed or unallocated.

When a problem occurs, set the **cm** event set to **active** using the **fstrace setset** command. When tracing is enabled on a busy AFS client, the volume of events being recorded is significant; therefore, when you are diagnosing problems, restrict AFS activity as much as possible so that unrelated fstrace logging is minimized. Because tracing may have a negative impact on system performance, leave **cm** tracing in the **dormant** state when you are not diagnosing problems.

If a problem is reproducible, clear the **cmfx** trace log with the **fstrace clear** command and reproduce the problem. If the problem is not easily reproduced, keep the state of the event set **active** until the problem recurs.

To view the contents of the trace log and analyze the **cm** events, use the **fstrace dump** command to copy the content lines of the trace log to standard output (stdout) or to a file.

Note: If a particular command is causing problems, it may be helpful to determine the UNIX process id (pid) of that command. The output of the **fstrace dump** command can later be searched for the given pid to show only those lines associated with the process of the command that exhibits the problem with AFS.

8.1. Setting the State of an Event Set

The **fstrace setset** command allows you to specify the state of the **cm** event set. The state of an event set determines whether information on the events in that event set is logged. The syntax of the command is as follows:

fstrace setset [**-set** <*set_name*>+] [**-active**] [**-inactive**] [**-dormant**] [**-help**]

Arguments:

-set Specifies the name of the event set to be set. The only valid argument is **cm**. If the **-set** argument is omitted, the default is **cm**.

-active Enables tracing for the event set.

-inactive Temporarily disables tracing for the event set; however, the event set continues to allocate space occupied by the log to which it sends data.

-dormant Disables tracing for the event set; furthermore, the event set releases the space occupied by the log to which it sends data. When the **cm** event set that sends data to the **cmfx** trace log is in this state, the space allocated for that log is freed or unallocated.

You must be logged in as “root” on the local machine to use this command.

8.2. Changing the Size of Trace Logs

The trace log occupies 60K of kernel by default. You can change the size of the log with the **fstrace setlog** command. If the specified log already exists, it is cleared when this command is issued and a new log of the given size is created. The syntax of the command is as follows:

fstrace setlog [**-log** <*log_name*>+] [**-buffersize** <*1-kilobyte_units*>] [**-help**]

Arguments:

-log Specifies the name of the trace log to be affected. The only valid argument is **cmfx**. If the **-log** argument is omitted, the default is **cmfx**.

-buffersize Specifies the number of 1-kilobyte blocks to allocate. The default value is 60K.

Because log data is stored in a finite, circular buffer, some of the data can be overwritten before being read. If this happens, the following message is sent to standard output (stdout) when data is being dumped:

```
Log wrapped; data missing.
```

Note: If this message appears in the middle of a dump, which can happen under a heavy work load, it indicates that not all of the log data is being written to the log or that some data is being overwritten. Increasing the size of the log with the **fstrace setlog** command can alleviate this problem.

You must be logged in as "root" on the local machine to use this command.

8.3. Dumping the Contents of Trace Logs

To view the information in a trace log, you must copy the content lines of the log to standard output (stdout) or to a file. The **fstrace dump** command dumps trace logs to standard output (stdout) or to a specified file to allow you to analyze the Cache Manager processes. The syntax of the command is as follows:

fstrace dump [**-set** <*set_name*>+] [**-follow** <*log_name*>+] [**-file** <*output_filename*>] [**-sleep** <*seconds_between_reads*>] [**-help**]

Arguments:

-set Specifies the name of the event set for which you wish to dump the corresponding log. The only valid argument is **cm**. You can specify the **-set** argument or the **-follow** argument, but not both. If you omit both arguments, the default trace log to be dumped is **cmfx**.

-follow Specifies the name of a trace log to dump continuously (the command continues to dump log information as changes to the log occur); a log that is continuously dumped is also cleared after the dump is stopped. The only valid argument is **cmfx**. You can specify the **-follow** argument or the **-set** argument, but not both. If you omit both arguments, the default trace log to be dumped is **cmfx**.

-file Indicates the destination for the output. You can specify the full or relative pathname of the output file. If the **-file** argument is omitted, the output is directed to standard output (stdout).

-sleep Specifies the number of seconds to wait between dumps. The default value is 10 seconds. This argument is meaningful only when used with the **-follow** argument.

At the beginning of the output of each dump is a header specifying the date and time at which the dump began. The number of logs being dumped is also displayed if the **-follow** argument is *not* specified. The header appears as follows:

```
AFS Trace Dump –
Date: Fri Nov 18 10:44:38 1994
Found 1 logs.
```

The contents of the log follow the header and are comprised of messages written to the log from an **active** event set. The messages written to the log contain the following three components:

- The timestamp associated with the message (number of seconds from an arbitrary start point)
- The process ID or thread ID associated with the message
- The message itself

For example, each message is formatted as follows:

```
time 749.641274, pid 3002:Returning code 2 from 19
```

A catalog file needs to be installed when AFS is installed in order to format the messages that are written to a log file. If your message looks similar to the following, verify that the catalog file was installed in */usr/vice/etc/C/afszcm.cat*:

```
raw op 232c, time 511.916288, pid 0
p0:Fri Nov 18 10:36:31 1994
```

Every 1024 seconds, a current time message is written to each log. This message has the following format:

time *timestamp*, pid 0: Current time: *unix_time*

where *timestamp* is the number of seconds from an arbitrary start point and *unix_time* is standard UNIX time format since January 1, 1970.

The current time message can be used to determine the actual time associated with each log message. Determine the actual time as follows:

1. Locate the log message whose actual time you want to determine.
2. Search backward through the dump record until you come to a current time message.
3. If the current time message's timestamp is smaller than the log message's timestamp, subtract the former from the latter. If the current time message's timestamp is larger than the log message's timestamp, add 1024 to the latter and subtract the former from the result.
4. Add the resulting number to the current time message's *unix_time* to determine the log message's actual time.

Because log data is stored in a finite, circular buffer, some of the data can be overwritten before being read. If this happens, the following message appears at the appropriate place in the dump:

```
Log wrapped; data missing.
```

Note: If this message appears in the middle of a dump, which can happen under a heavy work load, it indicates that not all of the log data is being written to the log or some data is being overwritten. Increasing the size of the log with the **fstrace setlog** command can alleviate this problem.

You must be logged in as "root" on the local machine to use this command.

8.4. Listing Information about Trace Logs

The **fstrace lslog** command displays information about the **cmfx** trace log. It optionally displays size and allocation information. The syntax is as follows:

fstrace lslog [**-set** <*set_name*>+] [**-log** <*log_name*>+] [**-long**] [**-help**]

Arguments:

-set Specifies the name of the event set whose corresponding log you want to display. The only valid argument is **cm**. You may specify the **-set** argument or the **-log** argument, but not both. If you omit both arguments, the default trace log to be displayed is **cmfx**.

-log Specifies the name of each log you want to display. The only valid argument is **cmfx**. You may specify the **-log** argument or the **-set** argument, but not both. If you omit both arguments, the default trace log to be displayed is **cmfx**.

-long Displays additional information about the log: the size of the log in 1-kilobyte units and the allocation state of the log. There are two allocation states:

- **allocated** — Space is reserved for the log in the kernel. This indicates that the event set that writes to this log is either **active** (tracing is enabled for the event set) or **inactive** (tracing is temporarily disabled for the event set; however, the event set continues to reserve space occupied by the log to which it sends data).
- **unallocated** — Space is *not* reserved for the log in the kernel. This indicates that the event set that writes to this log is **dormant** (tracing is disabled for the event set; furthermore, the event set releases the space occupied by the log to which it sends data).

When run without the **-long** flag, the **fstrace lslog** command displays only the name of the log.

You must be logged in as "root" on the local machine to use this command.

8.5. Listing Information about Event Sets

The **fstrace lsset** command displays information about the state of the **cm** event set. The syntax of the command is as follows:

fstrace lsset [**-set** <*set_name*>$^{+}$] [**-help**]

The **-set** argument specifies the name of the event set about which information is to be specified. The only valid argument is **cm**. If you omit the **-set** argument, the default is **cm**.

The three event set states are:

- **active** — Enables tracing for the event set.
- **inactive** — Temporarily disables tracing for the event set; however, the event set continues to claim space occupied by the log to which it sends data.
- **dormant** — Disables tracing for the event set; furthermore, the event set releases the space occupied by the log to which it sends data. When the **cm** event set that sends data to the **cmfx** trace log is in this state, the space allocated for that log is freed or unallocated.

You must be logged in as "root" on the local machine to use this command.

8.6. Clearing Trace Logs

The **fstrace clear** command clears trace log data. When you are no longer concerned with the information in a trace log, you can clear that log. The syntax of the command is as follows:

fstrace clear [**-set** <*set_name*>+] [**-log** <*log_name*>+] [**-help**]

Arguments:

-set Specifies the names of event set whose log you wish to clear. The only valid argument is **cm**. You can specify the **-set** argument or the **-log** argument, but not both. If you omit both arguments, the default trace log to be cleared is **cmfx**.

-log Specifies the name of log to be cleared. The only valid argument is **cmfx**. You can specify the **-log** argument or the **-set** argument, but not both. If you omit both arguments, the default trace log to be cleared is **cmfx**.

You must be logged in as "root" on the local machine to use this command.

8.7. Getting Help for Command Usage

The **fstrace apropos** command and the **fstrace help** command display the name and a short description for every **fstrace** command. If the **-topic** argument is specified, the commands provide the short description for only the command names listed. The **fstrace help** command provides the syntax along with the short description when the **-topic** argument is specified. The syntax of the commands is as follows:

fstrace apropos -topic <*help string*> [**-help**]

fstrace help [**-topic** <*help string*>+] [**-help**]

9. The kas Commands

This chapter describes changes to the **kas** command suite for AFS 3.4. In particular, AFS 3.4 contains changes to the following **kas** commands:

- The **kas examine** command
- The **kas setfields** command

AFS 3.4 also contains a change to the **kas** command ticket lifetime. These changes are marked with the heading "**AFS 3.4 Changes**."

9.1. The kas Command Ticket Lifetime

AFS 3.4 Changes

The **kas** command ticket is the ticket you receive from the Authentication server when using any command in the **kas** command suite. Previously, the ticket lifetime was set to 1 hour. In AFS 3.4, the ticket lifetime has been changed to 6 hours to enable you to work on extended operations such as large Authentication Database listings.

9.2. The kas examine Command

AFS 3.4 Changes

The **-reuse** argument of the **kas setfields** command specifies whether you can reuse any of your last twenty passwords. The **kas examine** command, which displays information for an Authentication Database entry, has been updated to display the value of the **-reuse** argument.

9.3. The kas setfields Command

AFS 3.4 Changes

The **-attempts** argument of the **kas setfields** command specifies the maximum number of times that you can consecutively fail to enter your password correctly during authentication before being locked out of your account. In AFS 3.4, the default value has been changed to *6*, which limits you to six failed authentication attempts. Previously, the default value was *0* (no limit for failed authentications). The command also has been changed so that it logs a failed login message in the */var/log/syslog* file after the sixth failed login attempt rather than after the tenth.

10. The package Command

This chapter describes changes to the **package** command and configuration file lines for AFS 3.4. In particular, AFS 3.4 allows relative pathnames and contains changes to the following arguments on configuration lines:

- The *minor device number* argument
- The *owner name* argument
- The *group name* argument

These changes are marked with the heading "**AFS 3.4 Changes**."

10.1. The package Command Allows Relative Pathnames

AFS 3.4 Changes

In AFS 3.4, the **package** command interprets relative pathnames beginning with "`./`", "`../`", or "`/`" specified by the *actual file* argument of the "**L**" configuration line. The **package** command also interprets "`:`" and "`!`" characters contained within a pathname.

10.2. Changes to *minor device number* Argument

AFS 3.4 Changes

The *minor device number* argument is specified on the "**B**" and "**C**" configuration file lines with the **package** command. In AFS 3.4, the **package** command interprets the number specified by the *minor device number* argument as a hexadecimal number, an octal number, or a decimal number.

Previously, the **package** command interpreted the *minor device number* as a decimal number only.

The **package** command continues to interpret the *major device number* as a decimal number only.

10.3. Changes to *owner name* Argument

AFS 3.4 Changes

The *owner name* argument is specified on the "**B**", "**C**", "**D**", "**F**", "**L**", and "**S**" configuration file lines with the **package** command. In AFS 3.4, the **package** command attempts to locate the user name specified by the *owner name* argument in the local UNIX file system. If the **package** command cannot locate the user name, it translates the user name to its corresponding user ID and attempts to locate the user ID instead. Previously, if the **package** command could not locate the user name, the **package** command failed.

10.4. Changes to *group name* Argument

AFS 3.4 Changes

The *group name* argument is specified on the "**B**", "**C**", "**D**", "**F**", "**L**", and "**S**" configuration file lines with the **package** command. In AFS 3.4, the **package** command attempts to locate the group name specified by the *group name* argument in the local UNIX file system. If the **package** command cannot locate the group name, it translates the group name to its corresponding group ID and attempts to locate the group ID instead. Previously, if the **package** command could not locate the group name, the **package** command failed.

11. The uss Commands

This chapter describes changes to the **uss** command suite for AFS 3.4. In particular, AFS 3.4 contains changes to the **uss bulk** command.

These changes are marked with the heading "**AFS 3.4 Changes**."

This chapter also contains changes from the AFS 3.3 release that have not been incorporated into the full AFS documentation set. These changes are marked with the heading "**AFS 3.3 Changes**."

11.1. The uss bulk Command

AFS 3.4 Changes

A new flag, **-pipe**, has been added to the **uss bulk** command. The **-pipe** flag has been added to assist you in running batch jobs without displaying the password prompt. The **-pipe** flag allows the **uss bulk** command to accept input piped in from another program.

The new syntax for the **uss bulk** command follows:

uss bulk -file <*bulk input file*> [**-template** <*pathname of template file*>] [**-verbose**]
[**-cell** <*cell name*>] [**-admin** <*administrator to authenticate*>] [**-dryrun**] [**-skipauth**]
[**-overwrite**] [**-pwexpires** <*password expires in [0..254] days (0 => never)*>] [**-pipe**] [**-help**]

AFS 3.3 Changes

The documentation correctly states that each type of line in the **uss bulk** command has a syntax order similar to its corresponding **uss** command. The syntax of the **delete** line corresponds to the syntax of the **uss delete** command and the syntax of the **add** line corresponds to the syntax of the **uss add** command.

However, both the *AFS System Administrator's Guide* and the *AFS Command Reference Manual* provide incorrect information on the syntax of the **add** line. The correct syntax of the **add** line follows:

add <*login name*> [**:**<*full name*>][**:**<*initial passwd*>][**:**<*password expires*>]
[**:**<*FileServer for home volume*>][**:**<*FileServer's disk partition for home volume*>]
[**:**<*home directory mount point*>][**:**<*uid to assign the user*>][**:**<*var1*>][**:**<*var2*>]
[**:**<*var3*>][**:**<*var4*>][**:**<*var5*>][**:**<*var6*>][**:**<*var7*>][**:**<*var8*>][**:**<*var9*>]

11.2. The uss add Command

AFS 3.3 Changes

The syntax of the **uss add** command is incorrect in the AFS documentation. The correct syntax follows:

uss add -user <*login name*> [**-realname** <*full name in quotes*>] [**-pass** <*initial password*>]
[**-pwexpires** <*password expires in [0..254] days (0 => never)*>]
[**-server** <*FileServer for home volume*>] [**-partition** <*FileServer's disk partition for home volume*>]
[**-mount** <*home directory mount point*>] [**-uid** <*uid to assign the user*>]
[**-template** <*pathname of template file*>] [**-verbose**] [**-var** <*auxiliary argument pairs (Num val)*>+]
[**-cell** <*cell name*>] [**-admin** <*administrator to authenticate*>]
[**-dryrun**] [**-skipauth**] [**-overwrite**] [**-help**]

12. The vos Commands

This chapter describes changes to the **vos** command suite for AFS 3.4. In particular, AFS 3.4 contains changes to the following **vos** commands:

- The **vos restore** command
- The **vos backup** command
- The **vos create** command
- The **vos release** command
- The **vos rename** command
- The **vos syncserv** command

In AFS 3.4, the **vos** command also can perform dump and restore operations from a named pipe.

These changes are marked with the heading "**AFS 3.4 Changes**."

This chapter also contains changes from the AFS 3.3 release that have not been incorporated into the full AFS documentation set. These changes are marked with the heading "**AFS 3.3 Changes**."

12.1. The vos restore Command

AFS 3.4 Changes

In AFS 3.3, the **vos restore** command determined whether the volume specified by the **-name** argument already existed on the partition specified by the **-server** and **-partition** arguments. If the volume existed on the specified partition, the **vos restore** command asked whether you wanted to overwrite the volume. If you entered a *yes* response, the **vos restore** command completely overwrote the existing volume; if you entered a *no* response, the command aborted. It was impossible to perform an incremental restore operation. If the volume did not exist on the specified partition, the **vos restore** command aborted.

In AFS 3.4, the **vos restore** command determines whether the volume specified by the **-name** argument already exists on the partition specified by the **-server** and **-partition** arguments. If the volume exists, the **vos restore** command prompts you to determine which of the following actions it is to perform:

- *abort* — aborts the **vos restore** command.
- *full* — performs a full restore by completely overwriting the existing volume.
- *incremental* — performs an incremental restore over the existing volume.

The following abbreviations are valid responses to the prompt:

- *a* for *abort*
- *f* for *full*
- *inc* or *increment* for *incremental*

If the volume exists, but not on the specified partition, the **vos restore** command prompt you for either a full restore or to abort the restore operation. An incremental restore cannot be done.

If standard input cannot be used for a prompt, the default action is to abort the restore operation.

The **vos restore** command also includes a new **-overwrite** argument for situations where you do not want to be prompted or standard input (stdin) is redirected and cannot be used for a prompt. The new command syntax follows:

vos restore -server <*machine name*> **-partition** <*partition name*>
-name <*name of volume to be restored*> [**-file** <*dump file*>] [**-id** <*volume ID*>]
[**-overwrite** <*abort* | *full* | *incremental*>] [**-cell** <*cell name*>]
[**-noauth**] [**-localauth**] [**-verbose**] [**-help**]

The valid abbreviations for the **-overwrite** argument are the same as those listed as valid responses to the prompt. The default action for the **-overwrite** argument is *abort*.

The following are rules for using the **vos restore** command:

- If the volume specified with the **-name** argument exists on the specified partition *and* the **-overwrite** argument is specified on the command line, the command performs the action indicated by the **-overwrite** argument.
- If the volume specified with the **-name** argument does not exist, the **-overwrite** argument has no effect and a full restore is done. If the **-overwrite** argument was set to *incremental* or *abort* on the command line, a message is displayed stating that a full restore will be done instead.
- If the volume specified with the **-name** argument exists, but is not on the specified partition, *and* the **-overwrite** argument is set to *full*, the command performs a full restore. If the **-overwrite** argument is set to *incremental* or *abort*, the command fails to replace the volume and a message is displayed stating that the operation is being aborted.
- If the volume specified with the **-name** argument exists, but is not on the specified partition, *and* the **-overwrite** argument is *not* specified, the command prompts you for a restore action (*abort* or *full*).
- If the volume specified with the **-name** argument exists on the specified partition and the **-overwrite** argument is *not* specified, the command performs one of the following actions:
  - — If the **-file** argument was omitted, the **vos restore** command aborts and no restore occurs.
  - — If the **-file** argument was specified, the **vos restore** command prompts you for a restore action (*abort*, *full*, or *incremental*). The default action is to abort the restore operation.

12.2. The vos backup Command

AFS 3.4 Changes

When you issue the **vos backup** command to create a Backup volume for a given ReadWrite volume, the command now exits without creating the Backup volume if the system cannot locate the ReadWrite volume at the site listed in the VLDB. The command displays the following message telling you that the operation aborted without creating the Backup volume:

```
RW volume not found in VLDB entry for volume id
```

Previously, the **vos backup** command exited without indicating that the Backup volume was not created.

12.3. The vos create Command

AFS 3.4 Changes

The **vos create** command has a new **-maxquota** argument. The new syntax for the command follows:

vos create -server <*machine name*> **-partition** <*partition name*>
-name <*volume name*> [**-maxquota** <*initial quota (KB)*>]
[**-cell** <*cell name*>] [**-noauth**] [**-localauth**] [**-verbose**] [**-help**]

The **-maxquota** argument specifies the maximum amount of disk space the volume can use. Express the **-maxquota** argument in kilobyte blocks (a value of *1000* is one megabyte). A value of *0* grants an unlimited quota, but the size of the disk partition that houses the volume places an absolute limit on the volume's maximum size. The default value for the **-maxquota** argument is *5000*.

Previously, when creating a volume, you had to use the **vos create** command to create a volume, use the **fs mkmount** command to create the mount point for the volume, and then use the **fs setquota** command to set the quota for the volume. The **-maxquota** argument has been added to the **vos create** command to allow you to create the volume and set the quota in the same step. There is no change in the requirement for creating a mount point for a volume; that is, you still need to create a mount point for the volume using the **fs mkmount** command after creating it with the **vos create** command.

12.4. The vos release Command

AFS 3.4 Changes

In AFS 3.4, the **vos release** command can now update up to half of a ReadWrite volume's replicas simultaneously. This is done automatically and internally; no arguments have been added to the **vos release** command. Previously, the **vos release** command previously updated one replica at a time.

12.5. The vos rename Command

AFS 3.4 Changes

In AFS 3.4, a message has been added to inform you that the **vos rename** command failed because the specified volume does not exist. The message follows:

```
vos: Could not find entry for volume <oldname>
```

Previously, if you specified a nonexistent volume with the **vos rename** command, the command did not inform you that it had failed; the command appeared to have executed properly.

12.6. The vos syncserv Command

AFS 3.4 Changes

In AFS 3.4, the **vos syncserv** command continues to check all remaining servers even if it cannot contact one or more of the servers. Each time a server cannot be contacted, a message identifying the server it could not contact is displayed, as in the following example:

```
Transaction call timed out for server 'fs1.abc.com'
```

Previously, the **vos syncserv** command attempted to contact every server on which a ReadOnly volume resided. If the command could not contact a particular server, it failed without attempting to contact the remaining servers. The command displayed a message stating that it could not contact a server without specifying which server it attempted to contact.

12.7. Restoring from a Named Pipe

AFS 3.4 Changes

The **vos dump** and **vos restore** commands allow you to dump and restore volumes from a named pipe without timing out the **fileserver** process. This feature allows AFS to interoperate with third-party backup systems.

12.8. The vos changeaddr Command

AFS 3.3 Changes

Changing the IP address of a simple file server was a difficult task in earlier versions of AFS. After changing the IP address, you had to run the **vos syncserv** and **vos syncvldb** commands. Then you had to issue the **vos remsite** command to remove site information associated with the ReadOnly volumes under the old IP address. A new **vos** command, **vos changeaddr**, allows you to change a simple file server's IP address easily.

The syntax of the new command follows:

vos changeaddr -oldaddr <*original IP address*> **-newaddr** <*new IP address*> [**-cell** <*cell name*>] [**-noauth**] [**-localauth**] [**-verbose**] [**-help**]

Arguments:

-oldaddr Specifies the old IP address of the file server whose IP address you are changing.

-newaddr Specifies the new IP address.

-cell Specifies the cell in which to run the command; the default is the local cell. Do not use this flag with the **-localauth** flag; the **-cell** argument and the **-localauth** flag are mutually exclusive.

-noauth Tells the Volume Location (VL) servers to assign the identity *system:anyuser* to the issuer.

-localauth Constructs a server ticket using a key from the */usr/afs/etc/KeyFile* file. Do not use this flag with the **-cell** argument; the **-cell** argument and the **-localauth** flag are mutually exclusive.

-help Prints the online help for this command. Do not provide any other arguments or flags with this one.

Note: This command does not change IP addresses contained in any protection groups that you have defined. Use the **pts rename** command to change IP addresses in existing groups.
Changing the IP address of a Ubik database server involves additional changes. Refer to the *AFS System Administrator's Guide* for more information.

13. Miscellaneous AFS Commands

This chapter describes changes to miscellaneous (non-suite) AFS commands for AFS 3.4. In particular, AFS 3.4 contains changes to the following miscellaneous commands:

- The **afsd** command
- The **butc** command
- The **fileserver** command
- The **klog** command
- The **knfs** command
- The **pagsh** command
- The **salvager** command
- The **scout** command
- The **upclient** command
- The **vldb_convert** command
- The **vlserver** command
- The **volserver** command

These changes are marked with the heading "**AFS 3.4 Changes**."

This chapter also contains changes from the AFS 3.3 release that have not been incorporated into the full AFS documentation set. These changes are marked with the heading "**AFS 3.3 Changes**."

13.1. The afsd Command

AFS 3.4 Changes

AFS 3.4 contains four changes to the **afsd** command:

- The new "cache-truncate" daemon flushes the AFS cache when it is required.
- AFS compares the cache size to the partition size so that the cache cannot be set to a value that is too large.
- The **afsd** command correctly interprets white space in the */usr/vice/etc/cacheinfo* file.
- The **-waitclose** flag has no effect on the operation of the AFS 3.4 **afsd** command.

13.1.1. New Cache Manager Daemon

The new Cache Manager "cache-truncate" daemon flushes the AFS cache when requested by the Cache Manager. When the cache-truncate daemon flushes the AFS cache, it removes cached data and status information from the AFS cache and writes it to the file server.

In previous versions of AFS, the process of truncating or flushing the AFS cache was handled by the Cache Manager's "background" daemons. The background daemon blocked until the required amount of data was flushed from the AFS cache to the file server. This function of flushing the AFS cache is now handled exclusively by the cache-truncate daemon, resulting in improved Cache Manager performance.

If you check the AFS cache size using the **fs getcacheparms** command before the cache-truncate daemon has finished flushing data from the AFS cache to the file server, you will see a message similar to the following:

```
% fs getcacheparms
AFS using 52640 of the cache's available 50000 1K byte blocks.
Cache guideline temporarily deliberately exceeded; it will be
adjusted down but you may wish to increase the cache size.
```

The AFS Cache Manager allows the cache to be exceeded deliberately in this instance because it is aware that the cache-truncate daemon is flushing the AFS cache. The AFS Cache Manager allows this situation only for a short period of time until the AFS cache is flushed.

13.1.2. AFS Compares Cache Size to Partition Size

AFS clients can panic and create warnings and error messages if the cache size is set too close to or higher than the size of the underlying partition. The *AFS Command Reference Manual* recommends the following cache sizes: for a disk cache, you should devote no more than 95% of the partition on which the cache resides; for a memory cache, you should determine the maximum amount of memory required to run processes and commands and require at least this amount of memory for processes and commands to run.

To avoid problems resulting from a disk cache that is too large, AFS now compares the disk cache size to the partition size when you issue the **afsd** command. If the disk cache size is greater than 95% of the partition size, AFS returns an appropriate message to standard output (stdout) and exits without starting the Cache Manager. You cannot start the Cache Manager until you reduce the size of the disk cache to less than 95% of the partition size.

13.1.3. Correct Interpretation of White Space in the *cacheinfo* File

Changes have been made to the **afsd** command's interpretation of the */usr/vice/etc/cacheinfo* file, which contains all of the information needed to run the Cache Manager. Previously, the **afsd** command could not interpret spaces, carriage returns, tabs, or blank lines that were inadvertently inserted into the file. The **afsd** command failed if it found extra white space while attempting to read the */usr/vice/etc/cacheinfo* file. In AFS 3.4, the **afsd** command ignores extra white space.

13.1.4. The -waitclose Flag Has No Effect on the afsd Command

In AFS 3.4, the default for the Cache Manager (**afsd**) operation is to complete the transfer of a closed file to the file server before returning control to the application invoking the close. In AFS 3.3, the default for the Cache Manager operation was to return control to a closing application program before the final chunk of a file was completely written to the file server.

The default functionality of the **fs storebehind** command was previously provided by the default setting of the **afsd** command, while the functionality of the AFS 3.4 Cache Manager was previously provided by the **-waitclose** flag of the **afsd** command. For this reason, the **-waitclose** flag has no effect on the operation of the Cache Manager in AFS 3.4.

13.2. The butc Command

AFS 3.4 Changes

AFS 3.4 contains two enhancements to the **butc** command:

- A **-localauth** flag has been added to the **butc** command.
- There is a new debugging level associated with the **-debuglevel** argument.

13.2.1. New -localauth Flag

The **butc** command now includes the **-localauth** flag, which assigns the issuer a token that never expires and displays an expiration date of NEVER, when the issuer wants to run a backup process in the background.

The new syntax of the **butc** command follows:

butc [**-port** <*port offset*>] [**-debuglevel** < *0* | *1* | *2* >] [**-cell** <*cell name*>]
[**-aixscsi**] [**-noautoquery**] [**-localauth**] [**-help**]

The **-localauth** flag assigns the **butc** command a token that never expires and needs to be run from a file server machine as "root." This flag instructs the **butc** command interpreter running on the local file server machine to construct a server ticket using the server encryption key with the highest key version number in the */usr/afs/etc/KeyFile* file on the local file server machine. The **butc** command presents the ticket to the Volume and/or Volume Location (VL) server to use in mutual authentication. This flag is only useful for commands issued on file server machines, since client workstations do not have a */usr/afs/etc/KeyFile* file. It is intended for **cron**-type processes or jobs included in the machine's */usr/afs/local/BosConfig* file. The flag can also be used if the issuer is unable to authenticate to AFS but is logged into the local UNIX file system as "root."

13.2.2. Change to the -debuglevel Argument

In AFS 3.4, the **-debuglevel** argument of the **butc** command, which determines the amount of information the Tape Coordinator reports, has three legal values: *0*, *1*, and *2*. The following describes the information supplied by the three legal values:

- *0* displays the minimum level of detail required to describe Backup Tape Coordinator (**butc**) operations at the Tape Coordinator window. The information includes error messages, tape start and finish messages, and prompts for placing new tapes in the drive. This is the default value.
- *1* displays names of volumes at the Tape Coordinator window as they are being dumped to and restored from tape.
- *2* displays all messages sent to the tape log file (*TL_<device_name>*) at the Tape Coordinator window.

In AFS 3.3, the **-debuglevel** argument had two legal values: *0* and *1*.

13.3. The fileserver Command

AFS 3.4 Changes

The **-m** argument of the **fileserver** command has been modified; the **-m** argument only affects machines running the AIX version of AFS. The **-m** argument specifies the percentage by which the **fileserver** process allows partitions on the file server machine to exceed their quotas. Previously, the default value for this option was *5*. The default value has been increased in AFS 3.4 to *10*. This change was necessary because AIX does not use the BSD standard of keeping a disk reserve.

The **fileserver** process now alerts you sooner when partitions on the file server machine are approaching their quota by returning the following error message:

```
No space left on device
```

AFS 3.3 Changes

Each File Server (**fileserver**) process generates a key with an infinite lifetime (using the AFS key), which it uses to communicate with the Protection Server (**ptserver**) process. In earlier versions of AFS, if the AFS key on which the File Server key was based was removed, the File Server could not communicate with the Protection Server because the File Server was still using the old key, which the Protection Server could no longer access. The only way to break this deadlock was to restart the File Server. (When the File Server was restarted, it generated a new key based on the latest AFS key.)

The **fileserver** program has been changed to remove this deficiency. Now, if a **fileserver** process is unable to authenticate with the **ptserver** process, the **fileserver** process generates a new key based on the latest AFS key and attempts to authenticate again. This change affects cells whose administrators followed Transarc's recommendations on AFS key changes and retirement but did

not restart the **fileserver** processes on a regular basis (if ever). These cells whose administrators no longer need to restart their **fileserver** processes as a result of an AFS key change.

This change does not affect cells whose administrators

- Never changed AFS keys (not recommended)
- Never retired old AFS keys (not recommended)
- Restarted the **fileserver** processes on a regular basis and retained old AFS keys for at least as long a period of time as the time between consecutive **fileserver** process restarts.

13.4. The klog Command

AFS 3.4 Changes

The **-tmp** flag has been removed from the **klog** command. The **-tmp** flag is no longer necessary because there is a **klog.krb** program available to authenticate to AFS from a Kerberos database. The new syntax of the **klog** command follows:

klog [**-x**] [**-principal** <*user name*>] [**-password** <*user's password*>] [**-cell** <*cell name*>] [**-servers** <*explicit list of servers*>$^{+}$] [**-pipe**] [**-silent**] [**-lifetime** <*ticket lifetime in hh[:mm[:ss]]*>] [**-setpag**] [**-help**]

Use the **klog.krb** program for Kerberos authentication rather than the **klog** command with the **-tmp** flag.

AFS 3.3 Changes

A new flag, **-setpag**, has been added to the **klog** command. When run with this flag, the **klog** command creates a process authentication group (PAG) prior to requesting authentication. The tokens created are then placed in this newly created PAG.

13.5. The knfs Command

AFS 3.4 Changes

In AFS 3.4, if you run the **knfs** command without the **-id** argument, the command uses the **getuid()** function to identify the issuer and grant appropriate permissions to the issuer of the command.

Previously, if you omitted the **-id** argument from the **knfs** command, the command defaulted to granting *system:anyuser* permissions to the issuer.

13.6. The pagsh Command

AFS 3.4 Changes

The **pagsh** command invokes the Bourne shell by default. If you prefer the C shell over the Bourne shell, issue the following command to invoke the C shell:

```
# pagsh -c /bin/csh
```

13.7. The salvager Command

AFS 3.4 Changes

Two new flags have been added to the **salvager** command.

-showlog Instructs the Salvager to display on standard output (stdout) all log data that is being written to the */usr/afs/logs/SalvageLog* file. This is useful if the user wants to use pipes to search for certain log output or wants to avoid the additional step of looking at the log file.

-showsuid Displays a list, for each partition, of the pathname of all **setuid** and **setgid** files that reside on that partition. This information is useful for administrative purposes.

The new syntax of the **salvager** command follows:

salvager [**initcmd**] [**-partition** <*Name of partition to salvage*>]
[**-volumeid** <*Volume Id to salvage*>] [**-debug**] [**-nowrite**] [**-inodes**]
[**-force**] [**-oktozap**] [**-rootinodes**] [**-salvagedirs**] [**-blockreads**]
[**-parallel** <*# of max parallel partition salvaging*>] [**-tmpdir** <*Name of dir to place tmp files*>]
[**-showlog**] [**-showsuid**] [**-help**]

13.8. The scout Command

AFS 3.4 Changes

In AFS 3.4, the **scout** command includes the name of the server in a message when a problem exists on a partition. An example of the new message for a partition named */vicepx* on a server named *fs1.abc.com* follows:

```
Could not get information on server fs1.abc.com partition /vicepx
```

Previously, when a problem existed on a partition, the **scout** command displayed the following message:

```
Could not get information on partition /vicepx
```

If the server name listed at the top of the screen had scrolled off, the user might not know which server was involved.

13.9. The upclient Command

AFS 3.4 Changes

The **-level** argument of the **upclient** command has been removed because its functionality is duplicated by the **-clear** and **-crypt** flags. The new syntax for the command follows:

upclient <*hostname*> [**-clear**] [**-crypt**] [**-t** <*retry time*>] [**-verbose**] <*dir*>+ [**-help**]

Note: The **-crypt** flag is not available in the international version of this command.

13.10. The vldb_convert Command

AFS 3.4 Changes

In addition to its previous Volume Location Database (VLDB) conversion values, the **vldb_convert** command now converts the VLDB from AFS version 3.4 (*4*) format to AFS version 3.3 (*3*) format. The value of 3 is only used with the **-from** argument.

VLDB upgrade conversions from AFS version 3.3 format to AFS version 3.4 format are not necessary. The version 3.3 VLDB is automatically converted to a version 3.4 VLDB when you upgrade the **vlserver** binaries.

AFS 3.3 Changes

A new flag, **-dumpvldb**, has been added to the **vldb_convert** command. The flag directs the command to produce debugging output. The new syntax of the **vldb_convert** command follows:

vldb_convert [**initcmd**] [**-to** <*goal version*>] [**-from** <*current version*>] [**-path** <*pathname*>] [**-showversion**] [**-dumpvldb**] [**-help**]

13.11. The vlserver Command

AFS 3.4 Changes

AFS 3.4 contains two changes to the **vlserver** command:

- A change in the values for the **-p** argument
- A new log file (*/usr/afs/logs/VLLog*) for the **vlserver** process

13.11.1. Change in Values for -p Argument

The **-p** argument of the **vlserver** command allows you to set the number of server lightweight processes to run. The default value for the **-p** argument of the **vlserver** command has been changed from 4 to 9. The minimum value for this argument is 4, and the maximum value is 16.

13.11.2. New Log File for the vlserver Process

AFS 3.4 supports a log file for the Volume Location (VL) Server (**vlserver** process). When the **vlserver** process is started, the VL Server creates an activity log file named */usr/afs/logs/VLLog*, if the file does not already exist. When the **vlserver** process creates a new *VLLog* file, it copies the existing *VLLog* file to a file named *VLLog.old*. You can examine this log file using the **bos getlog** command. By default, no logging is done by the **vlserver** process.

The *VLLog* file can be set to record three different information levels. You can enable logging in the *VLLog* file by using the following command:

kill -TSTP <*process id for vlserver*>

In the following example, the **ps** command is run to find the process id of the **vlserver** process and the **kill -TSTP** command is run to enable logging in the *VLLog* file:

```
# ps -axwu | more
USER   PID  %CPU  %MEM  SZ  RSS  TT  STAT  START   TIME   COMMAND
root    93   0.0   0.0  60    0   ?   IW   Feb 27  0:00   vlserver
# kill -TSTP 93
```

Use the same command to increase the current level of logging information (that is, to change from the first level of logging information to the second level or from the second level to the third level). A log entry is created in the *VLLog* file to indicate any change in the *VLLog* file detail level.

The first level of information contained in the *VLLog* file can include the following messages:

- Create Volume <volume-id>
- Delete Volume <volume-id>
- Change Addr <addr1> <addr2>
- Replace Volume <volume-id>
- Update Volume <volume-id>
- SetLock Volume <volume-id>
- ReleaseLock Volume <volume-id>
- GetNewVolid newmax=<volume-id>

The second level of information contained in the *VLLog* file can include messages related to standard lookup operations, such as the following messages:

- GetVolumeById <volume-id> (id)
- GetVolumeByName <volume-id> (id)
- ListAttrs nentries=<count>
- GetStats
- GetAddrs

The third level of information contained in the *VLLog* file can include messages related to infrequent lookup operations, such as `ListEntry index=<id>`.

You can disable logging for the **vlserver** process with the following command:

kill -HUP <*process id for vlserver*>

You can decrease the level of logging for the **fileserver** process by issuing the following command:

kill -HUP <*process id for vlserver*>

Afterwards, issue the following command to obtain the desired level of logging:

kill -TSTP <*process id for vlserver*>

13.12. The volserver Command

AFS 3.4 Changes

AFS 3.4 contains two changes to the **volserver** command:

- A change to the **-log** flag
- A new **-p** argument to set the number of lightweight processes (LWPs) that the **volserver** command is to run

13.12.1. Change to the -log Flag

The **-log** flag causes the Volume Server to record the names in the */usr/afs/logs/VolserLog* file of all users who successfully initiate a **vos** command. In AFS 3.4, the *VolserLog* file also contains file entries for any file removal activity resulting from using the **vos release** command with the **-f** flag.

13.12.2. New -p Argument

A new argument has been added to the **volserver** command. The **-p** argument of the **volserver** command sets the number of server lightweight processes (LWPs) to run. The minimum value for this argument is 4, and the maximum value is 16. The default is 9.

The new syntax of the **volserver** command follows:

/usr/afs/bin/volserver [**-log**] [**-p** <*lwp processes*>] [**-help**]

AFS 3.3 Changes

The **-verbose** flag has been removed from the **volserver** command because the flag generates only two possible messages. The functionality of the flag is now part of the base functionality of the command. In other words, the AFS 3.3 version of the **volserver** command, when run without any flags or arguments, behaves like the AFS 3.2 version of the command when run with the **-verbose** flag.

13.13. The dlog Command

AFS 3.3 Changes

The new **dlog** command is for use with Transarc Corporation's AFS/DFS Migration Toolkit. The **dlog** command authenticates to the DCE Security Service to obtain an AFS-compatible DCE ticket in order to access a DCE cell from an AFS client via the Translator Server. The command provides no functionality outside of the Migration Toolkit.

The syntax of the new command follows:

dlog [**-principal** <*user name*>] [**-cell** <*cell name*>] [**-password** <*user password*>] [**-servers** <*explicit list of servers*>+] [**-lifetime** <*ticket lifetime in hh[:mm[:ss]]*>] [**-setpag**] [**-pipe**] [**-help**]

Refer to the *AFS/DFS Migration Toolkit Administration Guide and Reference* for more information on the **dlog** command.

13.14. The dpass Command

AFS 3.3 Changes

The new **dpass** command is for use with Transarc Corporation's AFS/DFS Migration Toolkit. The **dpass** command returns the DCE password created for a user with the **dm pass** command. The command provides no functionality outside of the Migration Toolkit.

The syntax of the new command follows:

dpass [**-cell** <*original AFS cell name*>] [**-help**]

Refer to the *AFS/DFS Migration Toolkit Administration Guide and Reference* for more information on the **dpass** command.

13.15. The up Command

AFS 3.3 Changes

The **up** command now returns a *0* only if it succeeds; otherwise, the command returns a *1*. Formerly, the command always returned a *0*, regardless of success or failure.

13.16. The xstat Utility

AFS 3.3 Changes

Two new arguments, **-frequency** and **-period**, have been added to the two **xstat** programs (**xstat_cm_test** and **xstat_fs_test**):

- The **-frequency** argument sets the frequency in seconds at which the program initiates probes to the Cache Manager or file server; formerly, this value was hard coded at 60 seconds; 60 seconds is now the default.
- The **-period** argument sets the program's run time duration in minutes. At the end of this period of time, the program exits. Formerly, this value was hard coded at 10 minutes; 10 minutes is now the default.

The new syntax of the two programs follows:

xstat_cm_test [**initcmd**] **-cmname** <*Cache Manager name(s) to monitor*>+
-collID <*Collection(s) to fetch*>+ [**-onceonly**] [**-frequency** <*poll frequency, in seconds*>]
[**-period** <*data collection time, in minutes*>] [**-debug**] [**-help**]

xstat_fs_test [**initcmd**] **-fsname** <*File Server name(s) to monitor*>+
-collID <*Collection(s) to fetch*>+ [**-onceonly**] [**-frequency** <*poll frequency, in seconds*>]
[**-period** <*data collection time, in minutes*>] [**-debug**] [**-help**]

13.17. The -help Flag

AFS 3.3 Changes

In previous versions of AFS, the miscellaneous commands were inconsistent in their use of the **-help** flag. These commands now consistently use the **-help** flag to provide information on their syntax:

- The **afsd** command
- The **afsmonitor** command
- The **budb_convert** command
- The **butc** command
- The **fileserver** command
- The **fms** command
- The **kaserver** command
- The **kdb** command
- The **klog** command
- The **knfs** command
- The **kpasswd** command
- The **ptserver** command
- The **runntp** command
- The **scout** command
- The **tokens** command
- The **unlog** command
- The **upclient** command
- The **upserver** command
- The **vldb_convert** command
- The **vlserver** command
- The **volserver** command

14. Additional Functional Changes

This chapter lists additional functionality added to AFS for the 3.4 release, including:

- Multi-homed file servers
- Support for unlinking open files
- OSF/1 data conversion on DEC Alpha machines
- The **fileserver** process checks for *FORCESALVAGE* flag
- 8-bit character support for international characters
- Support for file server partitions larger than 2 GB
- Improved message when the host name is missing from the *CellServDB* file
- The volume name in response to an AFS command, if known by the Cache Manager
- New rights for the *system:administrators* group
- Improved database access during Ubik elections
- An increase in the number of server partitions
- Additional AIX 3.2 support
- Improvements to the NFS/AFS Translator

These changes are marked with the heading "**AFS 3.4 Changes**."

This chapter also contains changes from the AFS 3.3 release that have not been incorporated into the full AFS documentation set. These changes are marked with the heading "**AFS 3.3 Changes**."

14.1. Multi-Homed File Servers

AFS 3.4 Changes

Multi-homed file servers have multiple IP addresses. A multi-homed file server can respond to an RPC via a different IP address than one initially addressed by a client machine. By providing multiple paths through which a client machine's Cache Manager can communicate with it, a multi-homed file server can increase the availability of computing resources and improve performance. A multi-homed file server makes several addresses available to service requests for client machines. These additional addresses provide multiple paths with which a client machine's Cache Manager can communicate with the multi-homed file server.

A multi-homed file server could choose to service an RPC through a different IP address if there is heavy network traffic at the original IP address which serviced a previous RPC. For example, assume a multi-homed file server originally responds to a client machine's service

request at the IP address of *199.206.34.562*. When the client machine sends to the file server another service request, the file server's IP address of *199.206.34.562* is busy servicing the requests of other client machines. When the client machine's Cache Manager realizes that this IP address is busy, it selects another IP address, *199.206.34.564*, belonging to the multi-homed file server. The client machine's Cache Manager then attempts to send the RPC to that IP address with the service request.

AFS 3.4 supports up to 16 addresses per multi-homed file server machines (8 interface boards). File servers register their network addresses with the Volume Location Database (VLDB) upon startup. In AFS 3.3 and earlier versions, file servers were identified in the VLDB by a single IP address. In AFS 3.4, file servers are represented in the VLDB by a unique host identifier, which is created by the **fileserver** process during the startup process. This file server host identifier contains information about all known IP addresses for that file server. These IP addresses are updated whenever the **fileserver** process is restarted.

Note: You can specify a unique preference for any of the multi-homed addresses available at a file server machine using the **fs setserverprefs** command.

Note: Authentication, Protection, Volume Location, and Backup Database servers cannot be multi-homed.

Note: AFS 3.4 does not support multi-homed clients or database servers.

For more information about starting a multi-homed file server, refer to Section 3.3.

14.2. Support for Unlinking Open Files

AFS 3.4 Changes

AFS 3.4 allows you to unlink open files in the AFS file space. Unlinking an open file is a technique for creating short-lived temporary files when you want to keep these files hidden from other users or do not want to keep a permanent record of the file in your file system. When you unlink an open file, the file server renames the file with a unique file name, .__.afs*xxxx*, where *xxxx* is a random numeric or numeric/alphabetic string generated by the file server. The unlinked file's vcache maintains the credentials from the former file and the new filename created by the file server.

The unlinked file does not appear to users that can view the contents of the directory using the UNIX **ls** command. The renamed, unlinked file appears in the output of the UNIX **ls -la** command, but can be accessed only by those users that have the correct credentials to view the file in AFS (that is, only those users with "superuser" permissions).

When the unlinked temporary file is closed, the file server removes the file from the disk on the file server machine permanently.

14.3. Converting OSF/1 Data on DEC Alpha Machines

AFS 3.4 Changes

Before you can use AFS 3.4 on DEC Alpha file server machines running OSF/1 version 3.0 binaries (*alpha_osf30*), you must convert AFS partitions containing data from OSF/1 software earlier than version 3.0 to OSF/1 version 3.0. Transarc provides a data conversion program, **fs_conv_osf30**, with AFS 3.4 to convert this data in AFS partitions. The syntax for the **fs_conv_osf30** conversion program follows:

fs_conv_osf30 [**convert** | **unconvert**] [**-part** <*AFS partition name or device*>+] [**-verbose**] [**-help**]

Description:

The **fs_conv_osf30** program converts OSF/1 data in the AFS partitions specified by the **-partition** argument on the file server machine on which the command is run. Omit the **-partition** argument if you wish to convert the OSF/1 data in all AFS partitions on the file server machine on which you are running the **fs_conv_osf30** program.

The **fs_conv_osf30** program can perform two conversions:

- Use the **convert** command to convert OSF/1 data from a version earlier than 3.0 in an AFS partition to an OSF/1 version 3.0 data format.
- Use the **unconvert** command to convert OSF/1 version 3.0 data in an AFS partition to the data format of a version earlier than OSF/1 version 3.0.

Arguments:

convert Converts OSF/1 data from a version earlier than 3.0 in an AFS partition to an OSF/1 version 3.0 data format.

unconvert Converts OSF/1 version 3.0 data in an AFS partition to the data format of a version earlier than OSF/1 version 3.0.

-part Specifies the name or names of partitions which are to be converted by the command specified by the **fs_conv_osf30** program. If you omit this option, the **fs_conv_osf30** program converts the OSF/1 data in all AFS partitions on the file server machine.

-verbose Tells the file server to report on what it is doing as it executes the command.

-help Prints the online help for this command. All other valid options specified with this option are ignored.

Output:

The **fs_conv_osf30** program converts the OSF/1 data in the AFS partitions specified by the conversion command. If the **convert** command is specified, the program converts the OSF/1 data from a version format earlier than version 3.0 to version 3.0 format. If the **unconvert** command is specified, the program converts the OSF/1 data from version 3.0 format to the data format of a version earlier than OSF/1 version 3.0.

Examples:

1. The following command converts OSF/1 data from a version earlier than 3.0 in the */vicepa* partition to OSF/1 version 3.0 data on the file server machine on which the command is run. The **-verbose** flag is included to report on the status of the command.

   ```
   # fs_conv_osf30 /vicepa convert -verbose
   ```

2. The following command converts OSF/1 version 3.0 data in all AFS partitions to OSF/1 version 2.0 data on the file server machine on which the command is run. The **-verbose** flag is included to report on the status of the command.

   ```
   # fs_conv_osf30 unconvert -verbose
   ```

Privilege Required:

The issuer must be logged in as “root” on the machine on which the command is issued.

14.4. The fileserver Process Checks for FORCESALVAGE Flag

AFS 3.4 Changes

When the **vfsck** process determines that a partition needs to be salvaged, **vfsck** creates a *FORCESALVAGE* flag on that partition. Previously, the **fileserver** process did not check for a *FORCESALVAGE* flag when rebooting the file server machine after a clean shutdown. The file server attached all volumes even if a partition had a *FORCESALVAGE* flag.

In AFS 3.4, when the file server machine is rebooting, the **fileserver** process looks for a *FORCESALVAGE* flag. If the **fileserver** process detects such a flag in a partition, it detaches all of the volumes in the partitions already attached and aborts, sending an appropriate message to the */usr/afs/logs/FileLog* file. The **fileserver** process passes responsibility to the **bosserver** process, which causes the **salvager** process to run. After the salvage is complete, the **fileserver** process attaches all of the volumes properly.

14.5. AFS Supports 8-Bit Characters

AFS 3.4 Changes

For international character support, AFS 3.4 supports 8-bit characters in file and directory names. AFS file and directory names were previously restricted to 7-bit characters (ASCII).

14.6. AFS Supports Partitions Larger Than 2 GB

AFS 3.4 Changes

AFS 3.4 allows you to create vice partitions larger than 2 GB. The maximum size of a vice partition is the same as the maximum partition size of the local UNIX file system supported by the operating system. Refer to your operating system vendor's documentation for details.

Although you can create vice partitions larger than 2 GB, AFS 3.4 does not fully support volumes larger than 2 GB. You can read from and write to AFS volumes that are larger than 2 GB, but you cannot perform typical AFS volume operations, such as dumping, restoring, moving, or replicating the volume. Files in an AFS volume are still limited to a maximum of 2 GB.

Previous versions of AFS restricted partition sizes to under 2 GB.

14.7. New CellServDB Error Message

AFS 3.4 Changes

AFS commands with the **-cell** argument produce an error message when the host name is missing from the */usr/vice/etc/CellServDB* file. In AFS 3.4, these commands produce an error message indicating that the command failed and that there is a problem with the *CellServDB* file. The new message also tells which line of the *CellServDB* file caused the failure. For example, the **klog** command now issues the following message:

```
Can't properly parse host line xxx.xx.xx.xx in configuration file
/usr/vice/etc/CellServDB
klog: error reading cell database Can't get local cell name!
```

Previously, this message explained that the command had failed but did not indicate that there was a problem with the *CellServDB* file.

14.8. Cache Manager May Show Volume Name

AFS 3.4 Changes

In AFS 3.4, if the volume name information is available in the cache, the Cache Manager displays the volume name along with the volume ID when reporting information about a particular volume. The following message is an example of how the Cache Manager may display volume information:

```
Waiting for busy volume XXX (name) in cell XXX.
```

Previously in AFS, the Cache Manager displayed only the volume ID when reporting information about a particular volume.

14.9. New Rights for the system:administrators Group

AFS 3.4 Changes

In AFS 3.4, members of the *system:administrators* group have both **administer** (**a**) and **lookup** (**l**) rights on the access control list of every directory in the system.

Previously, members of the *system:administrators* group had only implicit **administer** (**a**) rights on the access control list of every directory in the system.

14.10. Improved Database Access During Elections

AFS 3.4 Changes

In AFS 3.4, users are now able to access ReadOnly data from all database servers, even when Ubik cannot attain a quorum; however, they cannot update the data until Ubik has a quorum.

Previously, all database servers except the Protection Server provided ReadOnly data, even when no quorum existed. Users were able to read data but were not able to update the data until Ubik established a quorum.

14.11. Increase in Server Partitions

AFS 3.4 Changes

AFS 3.4 supports up to 256 partitions per server. The names of these partitions range from */vicepa* to */vicepiv*.

In earlier versions of AFS, each server had a maximum of 26 partitions. The names of these partitions ranged from */vicepa* to */vicepz*.

14.12. Additional AIX 3.2 Support

AFS 3.4 Changes

For AIX 3.2 systems only, Transarc has updated the AIX **rstar** commands. AFS 3.4 now supports all group permission features. For example, entries of the following type are supported:

```
+@NetGroup
-@NetGroup
-@HostName
```

Previously, AFS did not support group permission features in the */.rhosts* or */etc/hosts.equiv* file.

14.13. Changes to the NFS/AFS Translator

AFS 3.4 Changes

In AFS 3.4, it is no longer necessary to load the AFS kernel extensions before starting the NFS daemons for the NFS/AFS Translator.

In AFS 3.4, source customers can build AFS Cache Managers that function as translators.

14.14. Improved Networking Support

AFS 3.3 Changes

In AFS 3.3, the **Rx** Remote Procedure Call (RPC) system can take better advantage of networks with large Maximum Transfer Unit (MTU) values. Previously, the Ethernet MTU of 1500 bytes limited the efficiency of AFS running on high-speed networks such as FDDI. The modifications allow for higher throughput between machines directly attached to the high-speed network.

14.15. Modification to fsync()

AFS 3.3 Changes

Previously, executing **fsync(2)** on an AFS file caused changes to the file to be written to the cache device and to the file server machine, but it did not cause the changes to be written to the file server's non-volatile storage. To provide maximum security for user data, **fsync(2)** now does the latter. This modification further reduces the amount of changed user data that can be jeopardized by a file server crash.

With these changes, **fsync(2)** consumes slightly more CPU and considerably more disk I/O resources on the file server machine than it previously did. In practice, this facility is infrequently used and the impact of the change is negligible; however, any application that uses **fsync(2)** heavily will suffer a performance penalty.

14.16. Version Strings in Binaries

AFS 3.3 Changes

Every AFS binary file includes a version string identifying the configuration of sources used to produce the binary. This allows AFS Product Support to more quickly determine which AFS release is being used and which patches (if any) have been applied. Use the UNIX **what** command to display the version string of an AFS binary file. If you do not have the **what** command, type the following command:

'strings *filename* **| grep afs3'**

where *filename* is the name of the appropriate binary file.

14.17. File Locking Operations

AFS 3.3 Changes

AFS 3.3 is limited with respect to file locking, as follows:

- AFS does not support byte-range locks. This includes all **lockf()** calls and those **fcntl()** calls that specify a byte offset to a file. However, all operations on byte range locks return a success value of 0. In addition, the first time a byte-range locking operation is called by a program, AFS displays the following message:

```
afs: byte-range lock/unlock ignored; make sure no one else
is running this program.
```

- File locking semantics in AFS are not distributed. Processes from the same workstation competing for locks on the same file obey proper locking semantics. However, processes from different machines competing for locks on the same file would get `EWOULDBLOCK`.
- Deadlock avoidance does not work in AFS. The **fcntl()** lock calls that can result in deadlock do not return `EDEADLK`.

15. Bug Fixes

AFS 3.4 Changes

AFS 3.4 includes fixes for many bugs in AFS 3.3, a subset of which are described in this chapter. This chapter describes only the most visible fixes included with AFS 3.4. Unless otherwise noted, these bug fixes do *not* affect the documentation.

The following **backup** bugs have been fixed:

- The Backup System now reduces the number of calls made to the Volume Location Database (VLDB) and the Backup Database (BUDB). This fix improves the performance of the Backup System when starting a dump operation (especially full dumps).
- The Backup System does not read information from the Backup Database until it requires the information. This fix improves the startup time of the backup process.
- In AFS 3.4, backup dump operations now use the most recent dump hierarchy. If the **-dump** (dump level) argument is skipped, the Backup System will recognize it and base the dump on the most recently created dump hierarchy. It previously based the dump on a dump hierarchy that was not the most recently created dump hierarchy.
- The error message produced when the Backup System cannot locate a host entry in the Backup Database has been improved. Previously, the error message stated:

  ```
  backup: Unable to connect to tape coordinator at port TC_port_offset
  ```

  Now, the error message states:

  ```
  backup: No such host/port entry; Can't connect to tape coordinator
  at port TC_port_offset
  ```

- The **backup interactive** command now displays the `Waiting for job` message once and waits silently until the backup job completes. Previously, the **backup interactive** command displayed the `Waiting for job` message continually while waiting for the backup job to complete.
- The Backup System now waits for text configuration locks. Previously, the Backup System returned an error when it encountered configuration locks.
- When restoring a volume by date using the **-date** argument with the **backup volrestore** command, the Backup System restores the volume that was cloned closest to the specified date (and not past it). Previously, the Backup System restored the volume whose dump started closest to the specified date (and not past it).

- The maximum number of volumes that you now can dump or restore using the Backup System is unlimited. In previous versions of AFS, the maximum number of volumes that you could dump or restore was 10,000.
- The Backup System now can display tapes with an unlimited number of volumes per dump and volumes per tape. In previous versions of AFS, the Backup System displayed only tapes with less than 1,000 volumes per tape.
- Before dumping volumes to tape, the Backup System now sorts the volumes to be dumped by server and partition. This fix reduces the number of tape changes required to complete disk restores or restores of volumes on the same partition.
- You now can abort the **backup scantape** command using a command that can kill the operation like **^c** (Control-c). In previous versions of AFS, it was impossible to abort the **backup scantape** command.
- If you specify the **-dbadd** flag with the **backup scantape** command, the command now adds minor entries to the Backup Database as it encounters the information. Previously, the command added entries to the Backup Database after completing a full scan of the tape.
- If a volume fails to dump, the Tape Coordinator checks to see if the volume was moved. If the Tape Coordinator determines that the volume was moved, the Tape Coordinator now dumps the volume from its new location. Previously, the Tape Coordinator did not check to see if the volume was moved and omitted that volume from the dump.
- Unmounting a tape no longer causes the entire tape operation to fail. Previously, unmounting a tape could cause the entire tape operation to fail.
- The Tape Coordinator now recognizes and correctly reports error messages associated with the **volserver** process. Previously, the Tape Coordinator did not recognize and correctly report error messages associated with the **volserver** process.
- The Backup System now determines a volume's clone date at the time of the dump. Previously, the Backup System did not take into account a change of date.
- The Tape Coordinator no longer times out the connection during a rewind-on-close tape operation. Previously, the Tape Coordinator did time out the connection during a rewind-on-close tape operation.
- The Backup Server now returns the correct host ID for DEC Paramax machines with the **backup dbverify** command.
- The **backup dbverify** command now prints clearer status and error messages in the Tape Coordinator's log and error files.
- The Backup Server no longer times out a database dump too early with the **backup savedb** command.

The following **fs** command bugs have been fixed:

- Previously in AFS, the **fs newcell** command allowed **setuid** privileges by default. No other commands defaulted to allow these privileges. In AFS 3.4, the **fs newcell** command does not default to allow these privileges. The **fs newcell** command defaults to no **setuid** privileges.
- The **fs mariner** command is an alias for the **fs monitor** command.

The following **package** command bug has been fixed:

- The **-rebootfiles** flag of the **package** command now works as described in the *AFS Command Reference Manual*.

The following **pts** command bug has been fixed:

- The **pts createuser** command does not allow you to create a user with an ID of 0. If you attempted to specify an ID of 0 using the **-id** argument prior to AFS 3.4, the **pts createuser** command created an ID other than 0 and reported the created ID on standard output. For example, if you issued the following command:

  ```
  % pts createuser -name root -id 0
  ```

  the command returned a message similar to the following:

  ```
  User root has id 100232323
  ```

  In AFS 3.4, the previous command does not allocate an ID, but rather displays the following error message and aborts without creating a user:

  ```
  0 isn't a valid user id; aborting
  ```

The following bugs in the **vos** command have been fixed:

- Previously, the **vos release** command did not take into account any files deleted from the ReadWrite volume. When the ReadWrite volume was released for replication, these deleted files became zero-length files with the same disk allocation given to the original files in the ReadOnly volumes. This resulted in lost disk space. The only way to recover the lost disk space was removing the ReadOnly volume and generating it again.

  As a workaround to this problem in AFS 3.4, issuing the **vos release** command with the **-f** flag takes into account any files deleted from the ReadWrite volume and does not copy them into the ReadOnly volume; this recovers some of the lost disk space without removing the ReadOnly volume and releasing it again.

  The **-log** argument of the **volserver** command has changed so that the *VolserLog* file now contains all removal activity as a result of the **vos release** command if the user specified the **-f** flag.

- The **-server** and **-partition** arguments of the **vos backupsys** command now work as described in the *AFS Command Reference Manual*.

  The **vos backupsys** command has been fixed so that now when a user specifies a volume with the **-server** and **-partition** arguments, the command checks whether the volume is a ReadOnly or a ReadWrite and only creates a Backup for the ReadWrite version. If the user does not specify a volume with the **-server** and **-partition** arguments, the command automatically creates backup volumes for every ReadWrite volume for specific servers and partitions.

  Previously in AFS, if a user issued the **vos backupsys** command specifying a volume with the **-server** and **-partition** arguments, the command did not check whether the specified

volume was a ReadOnly or a ReadWrite. If the user issued the command specifying a ReadOnly volume, the command may have backed up the ReadOnly volume to the site of the ReadWrite version. The user may think that the backup copy is the most recent version when it may not be, depending on when the ReadOnly version was last updated.

The following bugs have been fixed for the AFS miscellaneous commands:

- In AFS 3.4, the default value for the **-volumes** argument is 128. The **afsd** command now accepts values within the range of 50 to 3000 for the **-volumes** argument.

  Previously, the **-volumes** argument of the **afsd** command used the value of 128, regardless of the value indicated on the command line. Even if you did not specify the **-volumes** argument on the command line and wanted to use the default value of 50 (as stated in the *AFS Command Reference Manual*), the option used the value of 128.
- The **butc** command now contains a safety check and informs the operator when important data may be destroyed with a requested **backup dump** operation. The **butc** command checks the following conditions when recycling a tape:

  — The dump is overwriting a tape within the most recent dump set. The Backup System displays a warning on standard output (stdout) and in the *TE_<device_name>* and *TL_<device_name>* log files:

  ```
  Warning: Overwriting most recent dump before current one has
  finished
  ```

  The Backup System then proceeds with the dump operation.

  — The dump is overwriting a tape belonging to the current dump set. The Backup System does not allow you to overwrite a tape belonging to the current dump set.

  ```
  Can't overwrite tape containing the dump in progress
  ```

  Instead, the Backup System prompts you for another tape.

  — The dump is overwriting a tape in this dump set's hierarchy. The Backup System displays a warning on standard output (stdout) and in the *TE_<device_name>* and *TL_<device_name>* log files:

  ```
  Warning: Overwriting parent dump (DumpID number)
  ```

  The Backup System then proceeds with the dump operation.

  — Any master dump is not found in the database:

  ```
  Warning: Can't find parent dump number in backup database
  ```

AFS 3.3 Changes

AFS 3.3 includes fixes for many bugs in AFS 3.2, a subset of which are described in this chapter. This chapter describes only the most visible fixes included with AFS 3.3. Unless otherwise noted, these bug fixes do *not* affect the documentation.

The following bug has been fixed for the AFS miscellaneous commands:

- The maximum number of **afsd** stat entries has been set to 3600 on AIX 3.2 machines. It was found that if a site specified a large number of stat entries for the **afsd** command, the Cache Manager on an AIX 3.2 machine was more likely to panic. To protect against these Cache Manager panics, a maximum limit of 3600 stat entries has been established. If you attempt to exceed this limit by specifying a number greater than 3600 with the **-stat** argument of the **afsd** command, the command automatically sets the number of entries to 3600. Although this limit does not eliminate panics when large numbers of stat entries are requested, it should substantially reduce the number of panics.

The following bugs have been fixed for modified UNIX commands:

- The equivalent of the **rsh** command is now called the **remsh** command on HP-UX machines.
- The **rlogin** command must be located in the */usr/bin* directory rather than the */usr/ucb* directory on HP-UX machines.

16. Documentation Corrections

AFS 3.4 Changes

Previous versions of the AFS documentation contained some incorrect or misleading information. Unless otherwise noted, these documentation errors have *not* been corrected in the AFS 3.4 documentation:

- Step 2 of Section 2.4.1, "Using the Kernel Extension Facility on AIX Systems," on page 2-9 of the *AFS Installation Guide* states, "If the machine is not going to act as an NFS/AFS translator:

  # **cd /usr/vice/etc**
  # **./cfgexport -a export.ext.nonafs**"

  Change the command lines to

  # **cd /usr/vice/etc/dkload**
  # **./cfgexport -a export.ext.nonfs**"

- The section on the operation of the AFS **login** program on page 2-40 of the *AFS System Administrator's Guide* states, "If no AFS token was granted [because of an incorrect password], the **login** program attempts to log the user into the local UNIX file system..." The AFS **login** program and documentation is not consistent with some vendors' operating systems. If you cannot authenticate into AFS for any reason, the rules for logging into your local system apply. See the documentation for your particular operating system.
- The AFS 3.4 **login** program includes support for secondary authentication for AIX 3.2; however, if you cannot authenticate to AFS for any reason, the rules for logging into your local system apply. See the AIX vendor's operating system documentation for additional information.
- The REQUIREMENTS/RESTRICTIONS section of the **inetd**, **rcp**, and **rsh** command descriptions on pages 10-5, 10-12, and 10-16, respectively, of the *AFS Command Reference Manual* contain the following information in a bulleted list:

  "The following two lines must appear in the */etc/services* file on the local machine (as well as on the remote machine running modified [**inetd**]). On NeXT machines, this information must appear in the NetInfo database rather than in */etc/services*.

  ```
  auth 113/tcp authentication
  ta-rauth 601/tcp rauth"
  ```

TCP ports 113 and 601 no longer are used by AFS. Remove this information from the *AFS Command Reference Manual*.

- AFS does *not* support sockets. (This note is added for the purpose of clarification.)

AFS 3.3 Changes

Previous versions of the AFS documentation contained some incorrect or misleading information. Unless otherwise noted, these documentation errors have *not* been corrected in the AFS 3.3 documentation:

- On page 17-15 of the *AFS System Administrator's Guide* and page 4-38 of the *AFS Command Reference Manual*, the following recommendation is given for limiting consecutive failed login attempts: "For most cells, Transarc Corporation recommends setting the limit on authentication to 5 attempts and the lockout time to 25 minutes."

  This is not a good recommendation. Instead, you should follow the recommendation listed in the *AFS 3.3 Release Notes*:

  Recommendation: Transarc Corporation recommends a limit of 9 consecutive failed authentication attempts and a 25-minute lockout time. Although some cells may want to use other limits, these should suffice for most cells.

Index

8-bit characters 101

A

acceptable abbreviations
- **fs getserverprefs** command 56
- **fs setserverprefs** command 52

AFS
- file size 101
- **login** program 27
- partition size 101
- volume size 101

AFS 3.3 Release Notes 112
AFS authentication method 29
AFS Backup System
- error handling 36
- error messages 36
- modifications 37

AFS Command Reference Manual . . . 111, 112
AFS Installation Guide 111
AFS System Administrator's Guide . 8, 111, 112
AFS/DFS Migration Toolkit
- **dlog** command 94
- **dpass** command 94

AFS/DFS Migration Toolkit Administration Guide and Reference 63, 94
afs_dynamic_auth program 28
AFSCELL environment variable 38–40
afsd command 85–87, 108, 109
- bug fixes 108, 109

AIX
- **login** program 27

AIX 3.2
- documentation corrections 111
- secondary authentication support 30
- support 103

aliases
- **fs getserverprefs** command 56
- **fs setserverprefs** command 52

ALTSHELL variable 30
arguments
- **backup interactive** command 38
- **backup volsetrestore** command 39
- **-cell** 37, 63
- **fs exportafs** command 61
- **fs getserverprefs** command 57, 59
- **fs setserverprefs** command 54
- **fstrace clear** command 71
- **fstrace dump** command 67
- **fstrace lslog** command 69
- **fstrace lsset** command 70
- **fstrace setlog** command 67
- **fstrace setset** command 66
- *group name* 76
- *major device number* 75
- *minor device number* 75
- *owner name* 76
- **vos changeaddr** command 83

ASK parameter 33
Authentication Database
- **kaserver** 27

authentication method
- *AFS* 29
- *DCE* 29

AUTOQUERY parameter 34

B

"**B**" configuration file line 75, 76
backup command suite 31
- **backup dbverify** 106
- **backup dump** 36
- **backup interactive** 38, 105

backup command suite
- **backup kill** 33

backup command suite
backup labeltape 36, 45
backup readlabel 36, 45
backup savedb 36, 106
backup scantape 37, 106
backup volsetrestore 39, 105
bug fixes 105
-cell argument 37
-localauth flag 37
backup configuration file 31
backup dbverify command 106
backup interactive command 38, 105
backup kill command 33
backup labeltape command 36, 45
backup parameters
ASK 33
AUTOQUERY 34
BUFFERSIZE 35
FILE 35
MOUNT 32
NAME_CHECK 34
UNMOUNT 33
backup readlabel command 36, 45
backup savedb command 106
backup scantape command 37, 106
backup volsetrestore command 39, 105
arguments 39
description 41
examples 44
-file argument
using the 42
-name argument
using the 41
notes 44
output 42
binary version strings 104
bos addkey command 47
bos command suite 47
bos addkey 47
bos create 48
bos status 48
bos create command 48
bos status command 48
BUFFERSIZE parameter 35
bug fixes 105
afsd command 108, 109
backup command suite 105
butc command 108
fs command suite 106
package command 107
pts command suite 107
rlogin command 109
rsh command 109
vos command suite 107
butc command 87, 88, 108
bug fixes 108

C

"**C**" configuration file line 75, 76
Cache Manager
volume name 102
Cache Manager daemon 86
Cache Manager preferences
fs setserverprefs command 53
-cell argument 37
CellServDB
error message 101
CellServDB file 63
characters
8-bit 101
international 101
commands
afsd 85–87, 108, 109
backup dbverify 106
backup dump 36
backup interactive 38, 105
backup kill 33
backup labeltape 36, 45
backup readlabel 36, 45
backup savedb 36, 106
backup scantape 37, 106
backup volsetrestore 39, 105
bos addkey 47
bos create 48
bos status 48
butc 87, 88, 108
dlog 94
dpass 94

fileserver 88
fs checkservers 60, 63
fs copyacl 63
fs exportafs 61
fs getcacheparms 86
fs getcellstatus 63
fs getserverprefs 56
fs listacl 63
fs mkmount 63
fs monitor 106
fs newcell 63, 106
fs setacl 63
fs setcell 63
fs setserverprefs 51
fs storebehind 58
fstrace apropos 71
fstrace clear 71
fstrace dump 67
fstrace help 71
fstrace lslog 69
fstrace lsset 70
fstrace setlog 67
fstrace setset 66
kas examine 73
kas setfields 73
klog 89
knfs 89
package 75, 107
pagsh 90
pts createuser 107
rlogin 109
rsh 109
rstar 103
salvager 90
scout 90
up 94
upclient 91
uss add 78
uss bulk 77
vldb_convert 91
vlserver 91, 92
volserver 93
vos backup 81
vos backupsys 107
vos changeaddr 82
vos create 81
vos release 81, 107
vos rename 82
vos restore 79
vos syncserv 82
xstat 95
configuration file lines
“**B**” 75, 76
“**C**” 75, 76
“**D**” 76
“**F**” 76
“**L**” 76
“**S**” 76
CONSOLE variable 30

D

“**D**” configuration file line 76
data conversion
OSF/1 99
database access
during elections 102
DCE authentication method 29
DEC Alpha OSF/1 99
DEC Alpha OSF/1 **login**
program 29
description
backup volsetrestore command 41
fs getserverprefs command 56
fs setserverprefs command 52
directories
/usr/vice/etc 28
dlog command 94
documentation corrections 111
AFS Installation Guide 111
AFS System Administrator’s Guide 111
AIX 3.2 111
inetd command 111
installation 111
local operating system 111
login program 111
rcp command 111
rsh command 111
sockets 112

dpass command 94

E

enhanced systems 7
environment variable
AFSCELL 38–40
error handling
AFS Backup System 36
error message
CellServDB 101
error messages
AFS Backup System 36
/etc/default/login file 30
/etc/passwd file 28, 29
/etc/security/login.cfg file 29
/etc/security/passwd file 30
/etc/security/user file 28
event set states
fstrace clear command 70
examples
backup volsetrestore command 44
fs getserverprefs command 57
fs setserverprefs command 55
fs storebehind command 60

F

"**F**" configuration file line 76
file locking
operations 104
FILE parameter 35
file servers
multi-homed 97
files
backup configuration 31
CellServDB 63
/etc/default/login 30
/etc/passwd 28, 29
/etc/security/login.cfg 29
/etc/security/passwd 30
/etc/security/user 28
syslog 27
unlinking 98
/usr/afs/backup/CFG_<tape_device> . . . 31
fileserver
process 100
fileserver command 88
fileserver process
FORCESALVAGE flag 100
flags
-localauth 37
fs checkservers command 60, 63
fs command suite 49
bug fixes 106
-cell argument 63
fs checkservers 60, 63
fs copyacl 63
fs exportafs 61
fs getcacheparms 86
fs getcellstatus 63
fs getserverprefs 56
fs listacl 63
fs mkmount 63
fs monitor 106
fs newcell 63, 106
fs setacl 63
fs setcell 63
fs setserverprefs 51
fs storebehind 58
fs copyacl command 63
fs exportafs command 61
arguments 61
fs getcacheparms command 86
fs getcellstatus command 63
fs getserverprefs command 56
acceptable abbreviations 56
aliases 56
arguments 57, 59
description 56
examples 57
output 57
privilege required 58
fs listacl command 63
fs mkmount command 63
fs monitor command 106
fs newcell command 63, 106
fs setacl command 63
fs setcell command 63

fs setserverprefs command 51
acceptable abbreviations 52
aliases 52
arguments 54
Cache Manager preferences 53
description 52
examples 55
privilege required 55
specifying preferences 53
fs storebehind command 58
examples 60
privilege required 60
fs_conv_osf30 program 99
fstrace apropos command 71
fstrace clear command 71
arguments 71
event set states 70
fstrace command suite 65
fstrace apropos 71
fstrace clear 71
fstrace dump 67
fstrace help 71
fstrace lslog 69
fstrace lsset 70
fstrace setlog 67
fstrace setset 66
fstrace dump command 67
arguments 67
fstrace help command 71
fstrace lslog command 69
arguments 69
fstrace lsset command 70
arguments 70
fstrace setlog command 67
arguments 67
fstrace setset command 66
arguments 66
fsync()
modification 103

G

getting
Volume Location (VL) server preferences 49, 51, 56
groups
system:administrators 102

H

-help flag 95
HZ variable 30

I

IDLEWEEKS variable 30
international characters 101

K

kas command suite 73
kas examine 73
kas setfields 73
ticket lifetime 73
kas examine command 73
kas setfields command 73
kaserver
Authentication Database 27
Kerberos authentication
support 27
klog command 89
knfs command 89

L

"**L**" configuration file line 76
-localauth flag 37
login program 111
login programs
AFS 27
AIX 27
DEC Alpha OSF/1 29
Solaris 30
ls -la command 98
ls command 98

M

machines 8
DEC Alpha 99
Maximum Transfer Unit (MTU)
values 103
miscellaneous AFS commands 85
modification
fsync() 103
modifications
AFS Backup System 37
MOUNT parameter 32
MTU
see Maximum Transfer Unit 103
multi-homed
file servers 97

N

NAME_CHECK parameter 34
names
permanent tape 36, 45
networking
support 103
new rights
system:administrators group 102
new systems 7
NFS/AFS Translator
changes 103
notes
backup volsetrestore command 44

O

obsolete systems 7
operating systems 8
operations
file locking 104
OSF/1 data conversion 99
output
backup volsetrestore command 42
fs getserverprefs command 57

P

package command
bug fixes 107
configuration file lines 75
group name argument 76
major device number argument 75
minor device number argument 75
owner name argument 76
relative pathnames 75
pagsh command 90
parameters
ASK 33
AUTOQUERY 34
BUFFERSIZE 35
FILE 35
MOUNT 32
NAME_CHECK 34
UNMOUNT 33
partition size
change 101
partitions
increase 102
partitions larger than 2 GB 101
PASSREQ variable 30
PATH variable 30
pathnames
relative 75
permanent tape names 36, 45
preferences
Volume Location (VL) server . . . 49, 51, 56
getting 49, 51, 56
setting 49, 51
privilege required
fs getserverprefs command 58
fs setserverprefs command 55
fs storebehind command 60
processes
fileserver 100
volserver 106
program
AFS **login** 27
Solaris **login** 30

programs
afs_dynamic_auth 28
AIX **login** 27
DEC Alpha OSF/1 **login** 29
fs_conv_osf30 99
login 111
pts command suite
bug fixes 107
pts createuser 107
pts createuser command 107

R

recommendations
failed login attempts 112
reducing
cache size 86
registry variable 28
relative pathnames 75
Remote Procedure Call
Rx 103
rlogin command 109
bug fixes 109
RPC
see Remote Procedure Call 103
rsh command 109
bug fixes 109
rstar commands 103
Rx
Remote Procedure Call 103

S

"**S**" configuration file line 76
salvager command 90
scout command 90
server
partitions 102
setting
Volume Location (VL) server preferences
. . . . . 49, 51
SLEEPTIME variable 30
sockets 112
Solaris
login program 30
Solaris variables
ALTSHELL 30
CONSOLE 30
HZ 30
IDLEWEEKS 30
PASSREQ 30
PATH 30
SLEEPTIME 30
SUPATH 30
TIMEOUT 30
TIMEZONE 30
ULIMIT 30
UMASK 30
specifying preferences
fs setserverprefs command 53
summary of changes 1
SUPATH variable 30
support
8-bit character 101
AIX 3.2 103
secondary authentication 30
international character 101
Kerberos authentication 27
networking 103
supported systems 7, 8
@sys variable 8
syslog file 27
SYSTEM file 28
system name 8
system:administrators group 102
new rights 102
systems
enhanced 7
new 7
obsolete 7
supported 7, 8

T

Tape Coordinator
    tape scanning 37
tape scanning
    Tape Coordinator 37
ticket lifetime
    kas command suite 73
TIMEOUT variable 30
TIMEZONE variable 30

U

Ubik database
    access 102
UDP port 750 27
UDP port 88 27
ULIMIT variable 30
UMASK variable 30
UNIX commands
    ls 98
    ls -la 98
unlinking
    open files 98
UNMOUNT parameter 33
up command 94
upclient command 91
/usr/afs/backup/CFG_<tape_device> file . . 31
    ASK parameter 33
    AUTOQUERY parameter 34
    BUFFERSIZE parameter 35
    FILE parameter 35
    MOUNT parameter 32
    NAME_CHECK parameter 34
    UNMOUNT parameter 33
/usr/vice/etc directory 28
uss add command 78
uss bulk command 77
uss command suite
    uss add 78
    uss bulk 77

V

variable
    registry 28
    SYSTEM 28
variables
    Solaris
        ALTSHELL 30
        CONSOLE 30
        HZ 30
        IDLEWEEKS 30
        PASSREQ 30
        PATH 30
        SLEEPTIME 30
        SUPATH 30
        TIMEOUT 30
        TIMEZONE 30
        ULIMIT 30
        UMASK 30
    @sys 8
version strings
    binary 104
VL server
    see Volume Location server 49, 51, 56
vldb_convert command 91
vlserver command 91, 92
volserver command 93
volserver process 106
volume name
    Cache Manager 102
volumes larger than 2 GB 101
vos backup command 81
vos backupsys command 107
vos changeaddr command 82
    arguments 83

vos command suite
bug fixes 107
vos backup 81
vos backupsys 107
vos changeaddr 82
vos create 81
vos release 81, 107
vos rename 82
vos restore 79
vos syncserv 82
vos create command 81
vos release command 81, 107
vos rename command 82
vos restore command 79
vos syncserv command 82

X

xstat command 95