Introduction to AFS and its Administration

Alf Wachsmann (SLAC)
Today's Plan

9:00 – 10:00 General Introduction to AFS
10:00 – 10:30 AFS Client Topics
10:30 – 11:00 Coffee Break
11:30 – 12:30 AFS Server Topics
12:30 – 13:30 Lunch Break
13:30 – 15:00 AFS Server Topics cont'd.
15:00 – 15:30 Coffee Break
15:30 – 17:00 Advanced AFS Topics
Copyright Note

• This material is Copyright © 2004-2009 by Alf Wachsmann.
• All rights reserved.
• Use without permission is prohibited.
• Supported in part by U.S. Department of Energy contract DE-AC02-76SF00515.
• Version 8.0
I. General Introduction to AFS
History of AFS

• 1983: Andrew project started at CMU
• 1988: AFSv3 at CMU; first use outside of CMU
• 1989: original design team left CMU to form Transarc Corporation and AFS went commercial
• 1993: Arla project at KTH in Sweden started
• 1998: Transarc becomes wholly-owned subsidiary of IBM
• 2000: IBM releases OpenAFS as open source

• AFS stood for “Andrew File System” after the “Andrew Computing Project” named after Andrew Carnegie and Andrew W. Mellon, the founders of CMU
What is AFS?

• “AFS is a distributed filesystem that enables co-operating hosts (clients and servers) to efficiently share filesystem resources across both local area and wide area networks” (AFSWiki)

/afs/

openafs.org/  ...  cern.ch/  ...  slac.stanford.edu/

sw/  ...  user/a/alf/  package/  ...  u/sf/alfw/

• openafs.org, cern.ch etc. are AFS Cells
What is AFS?

- AFS uses Kerberos for authentication
- AFS client software
  - Cache manager
- AFS server software
  - Database servers
    - Authorization
    - Data location
  - File servers
    - Data
Benefits of using AFS...

- **Cache Manager**
  - maintains information about the identities of the users logged into the machine, finds and requests data on their behalf, and keeps chunks of retrieved files on local disk

- **Location Independence**
  - user does not need to know which fileserver holds the file, the user only needs to know the pathname of a file

- **Scalability**
  - With location independence comes scalability. An architectural goal of the AFS designers was client/server ratios of 114,000:1. A ratio of 2000:1 has been successfully exceeded at some sites.
...Benefits of using AFS...

- **Improved Security**
  - AFS makes use of Kerberos to authenticate users. This improves security for several reasons:
    - passwords do not pass across the network in plaintext
    - encrypted passwords no longer need to be visible
    - AFS uses mutual authentication - both the service provider and service requester prove their identities
    - federated authentication system
  - Secondly, AFS uses access control lists (ACLs) to enable users to restrict access to their own directories.

- **Same view of file store from each client**
  - No matter where users are logged in, they see the same files
...Benefits of using AFS...

- Replicated AFS volumes
  - frequently accessed data can be read-only replicated on several servers
  - Cache managers make use of replicated volumes to load balance
  - AFS will automatically start accessing the same data from a different replica
  - AFS client will access the closest volume copy

- Improved robustness to server crash
  - Cache Manager maintains local copies of accessed files
  - If a fileserver crashes, the client's locally cached file copies remain readable (as long as callback is valid (Unix) or fileserver comes back (Windows)) but updates to cached files fail while the server is down (to change with disconnected mode)
  - replicated read-only volumes on alternate file servers can satisfy requests for files from those volumes
...Benefits of using AFS...

• “Easy to use” networking
  • Remote files look like local ones

• Communications protocol
  • AFS communications protocol is optimized for Wide Area Networks. Retransmitting only the single bad packet in a batch of packets

• Improved system management capability
  • configuration changes can be made from any client in the AFS cell
  • AFS volumes can move from one server to another without users noticing it

Quoted from http://grand.central.org/twiki/bin/view/AFSLore/GeneralFAQ
...Benefits of using AFS

- Operating system independent:
  - **AFS client software runs on many systems**
    - AIX, HP-UX, Solaris (sparc, x86), MacOS X (ppc, intel)
    - Linux (incl. ARM), OpenBSD, NetBSD, FreeBSD
    - SGI Irix
AFS in Real Life

- AFS is not (only) used for esoteric and research purposes but in heavy duty production!
- Some example users:
  - Universities:
    - CMU, Stanford, MIT, KTH (Sweden), Chemnitz (Germany)
    - K-12 schools in Germany
  - Research Labs:
    - SLAC, CERN, DESY, JPL (NASA)
  - Several (large) Companies:
    - Intel, Morgan Stanley, Pictage
AFS from a User's View...

- Access files in AFS is transparent:

  `% cd /afs/cern.ch/user/a/alf/src/`
  `% cp hello.c /afs/slac.stanford.edu/u/sf/alfw/`

- `@sys` in path names is automatically translated on a per-client basis into a string of characters dependent on client's architecture
  - Example: `@sys/bin/` will be resolved
    - On 64 bit Linux with 2.6 kernel: `amd64_linux26/bin/`
    - On Solaris 10/SPARC: `sun4x_510/bin/`
    - On Solaris 9/x86: `sunx86_59/bin/`
    - On Windows2000: `i386_w2k/bin/`
  - Works only inside AFS file space
The command `fs sysname` shows what `@sys` will be replaced with.

- `@sys` can be a list of values:
  ```
  % fs sysname
  Current sysname list is 'i386_rh9' 'i386_linux24'
  ```

- In a file search, this list is traversed in sequential order without backtracking:
  ```
  /somewhere/i386_rh9/i386_linux24/bin/myprog
  /other/i386_linux24/i386_rh9/bin/<empty>
  /other/i386_linux24/i386_linux24/bin/mybin
  ```

- `fs sysname -newsys <name>` sets `@sys` to new value `<name>` on this client.
...AFS from a User's View

- Authentication is Kerberos 5
  - credentials expire after some time
  - AFS service ticket in kernel memory: AFS token
  - Getting a new AFS token:
    kinit, aklog, afslog, afs5log
  - tokens displays AFS tokens:

Tokens held by the Cache Manager:
User's (AFS ID 2171) tokens for afs@desy.de [Expires Mar 2 10:36]
User's (AFS ID 388) tokens for afs@cern.ch [Expires Mar 2 10:36]
User's (AFS ID 5828) tokens for afs@slac.stanford.edu [Expires Mar 2 10:36]
--End of list--

- You can have only one AFS token per cell per PAG
Process Authentication Group...

• On Unix, AFS tokens should be confined inside a Process Authentication Group (PAG)
• On Windows there is a single PAG per logon session
• There are no PAGs on MacOS X
• Container that child processes inherit from parent
• Without PAG, the cache manager uses UID for identification and another user with root can su to you and therefore use your token
...PAGs...

• How to tell whether you have a PAG:
  
  • **Linux: groups(1)**  
    
    alfw@Linux> groups  
    sf uucp 33536 32803  
  
  • **Linux/Solaris: id(1)**  
    
    alfw@Solaris> id  
    uid=5828(alfw) gid=1051(sf)  
    groups=33536,32803,1051(sf)  
  
  • Look for the two 32-bit/one 64-bit integer entries
...PAGs

• With OpenAFS-1.4.4 and later on Linux with kernels with keyring support, only one number to identify a PAG: 0x41000000 plus some 24-bit number

• PAGs on newer Linux kernels are now implemented via keyrings

alfw@Linux > uname -r
2.6.27.21-170.2.56.fc10.x86_64

alfw@Linux > id
uid=5828(alfw) gid=1051(sf) groups=1051(sf),1091036662

alfw@Linux > keyctl show
Session Keyring
          -3 --alswrv 0 1051 keyring: _ses.2881
  793038738 ----s--v 0 0  \_ afs_pag: _pag
Access Control List (ACL)...

• Access Control Lists (ACLs):

    % cd /afs/ir.stanford.edu/users/a/l/alfw/
    % fs listacl .
    Access list for . is
    Normal rights:
    system:backup rl
    system:administrators rlidwka
    system:anyuser l
    alfw rlidwka
### ACLs...

- **ACLs determine access rights to AFS directories**

<table>
<thead>
<tr>
<th>ACL</th>
<th>Permission</th>
</tr>
</thead>
<tbody>
<tr>
<td>lookup (l)</td>
<td>List contents of directory</td>
</tr>
<tr>
<td>insert (i)</td>
<td>Add files or directories</td>
</tr>
<tr>
<td>delete (d)</td>
<td>Delete entries in directory</td>
</tr>
<tr>
<td>administer (a)</td>
<td>Manipulate ACL for directory</td>
</tr>
<tr>
<td>read (r)</td>
<td>Read file content, query file status</td>
</tr>
<tr>
<td>write (w)</td>
<td>Write file content, change Unix permissions</td>
</tr>
<tr>
<td>lock (k)</td>
<td>Full-file advisory lock</td>
</tr>
</tbody>
</table>
...ACLs

- ACL inheritance:
  - AFS copies ACL on a parent directory over to a new subdirectory at the time of creation

- ACL awareness:
  - Not many (if any) tools are aware of ACLs
  - `cp -R` or `tar` are NOT ACL aware
  - Most backup software is NOT ACL aware

- Setting ACLs with `fs setacl . alfw rl`
ACL aware tools...

- AFS aware "copy": up
  - Comes with OpenAFS
  - Recursively copies directories and keeps AFS permissions and all other file attributes (owner, group, times)
  - Preserves AFS volume mount points if used with "-m"
  - Great for copying directories into new volumes
  - Usage: up -m <olddirectory> <newdirectory>
ACL aware tools

- AFS aware "find": Walk Subtree ws
  - Much more efficient than find. Takes advantage of inode number associated with file or directory: In AFS, directories have odd inode numbers and files have even inode numbers
  - Common usage: set directory permissions in all sub-directories: ws -d "fs sa %f system:anyuser rl"
  - Source code: http://web.mit.edu/~enomoose/www/ws.tar.gz
PTS Groups...

- PTS (protection server) groups allow you to create groups of users to place in the ACL as opposed to listing each user individually.
- Form groups of principals
- `% pts membership alfw
Groups alfw (id: 5828) is a member of:
  owner-pkg-heimdal
  maint-pkg-heimdal
  systems:group
  systems:unix-admin
  g-scs
  boeheim:systems
  ...

User's can create their own groups:
\% pts creategroup alfw:friends
group alfw:friends has id -5944

Add members to the group:
\% pts adduser ksa alfw:friends
\% pts examine alfw:friends
Name: alfw:friends, id: -5944, owner: alfw, creator: alfw,
membership: 1, flags: S-M--, group quota: 0.

And then use it in ACLs:
\% mkdir ~/friends
\% fs setacl ~/friends/ alfw:friends rl
There are special groups not owned by an individual:

- must be created by an administrator
- multiple members of this group allow multiple people to control
- good for group access areas: if an individual leaves others in the group retain control
- Example:

```bash
% pts membership facilities
Members of facilities (id: -905) are:
  ksa
  alfw

% pts listowner facilities
Groups owned by facilities (id: -905) are:
  facilities::root
  facilities::afs
  facilities::unix
[...]
...PTS Groups

- There are some special PTS groups:
  - `system:anyuser` literally everybody
  - `system:authuser` every user with valid AFS token for your cell
  - `system:administrators` group of PTS “super user” who can change ACLs even when explicitly forbidden
- These groups are automatically created at cell creation time
- OpenAFS-1.3.71 and later allows for nested groups (if compiled with `--enable-supergroups`)
AFS Command Suites...

- AFS comes with a set of commands for using, administering, monitoring, and debugging it
- The following list is not complete
- User related commands:
  - \texttt{klog}, \texttt{unlog}  obtain and delete AFS token
  - \texttt{aklog}  since OpenAFS 1.4
  - \texttt{tokens}  list existing AFS tokens
  - \texttt{pagsh}  obtain a new Shell in a new PAG
  - \texttt{kp passwd}  password change command (AFS/Krb4)
- \url{http://www.openafs.org/manpages/}
- Each of the following commands has many more sub-commands available
  - \texttt{Use <command> help and <command> <subcommand> \ -h}
• Administration related commands:
  
  • `fs <command> [options]`
    
    • `listacl` list access control list
    • `setacl` set access control list
    • `copyacl` copy access control list
    • `listquota` list volume quota
    • `setquota` set volume quota
    • `mkmount` make mount point
    • `lsmount` list mount point
    • `rmmount` remove mount point
    • `quota` show volume quota usage
    • `getcacheparms` get cache usage info
    • `setcachesize` set cache size
    • `sysname` get/set sysname (i.e. @sys) value
...AFS Command Suites...

* vos <command>  [options]  
  * addsite  add a replication site  
  * remsite   remove a replication site  
  * release  release a volume  
  * backup  make backup of a volume  
  * backupsys en masse backups  
  * restore  restore a volume  
  * examine everything about the volume  
  * listpart list partitions  
  * create create a new volume  
  * move  move a volume  
  * remove delete a volume  
  * rename rename a volume  

* new -noresolve option for all commands: shows actual IP address instead of DNS name
...AFS Command Suite...

- **pts <command> [options]**
  - *createuser* create a new user
  - *removeuser* remove a user from a group
  - *creategroup* create a new group
  - *adduser* add a user to a group
  - *rename* rename user or group
  - *delete* delete a user or group from database
  - *examine* examine an entry
  - *listentries* list users/groups in the protection database
  - *listowned* list groups owned by an entry
  - *membership* list membership of a user or group
AFS Command Suites...

- `bos <command> [options]`
  - `setcellname` set cell name
  - `addkey` add keys to key database
  - `listkeys` list keys
  - `removekey` remove keys from key database
  - `adduser` add users to super-user list
  - `listusers` list super-users
  - `removeuser` remove users from super-user list
  - `create` create a new server instance
  - `start` start running a server
  - `status` show server instance status
  - `stop` halt a server instance
  - `delete` delete a server instance
  - `salvage` salvage partition or volumes
...AFS Command Suites

• Monitoring and Debugging
  • afsmonitor monitors File Servers and Cache Managers
  • cmdebug examine status of Cache Manager: cache manager addresses and UUID, lock state, and stat cache entry state, read in-memory CellServDB data
  • rxdebug debugging trace of Rx activity
  • scout monitors the File Server process
  • udebug reports status of Ubik process associated with a database server process
  • fstrace traces Cache Manager operations
  • xstat_cm_test displays data collections from Cache Managers
  • xstat_fs_test displays data collections from File Servers
Differences UNIX vs. AFS...

- **File timestamps**
  - servers can't store last access time because AFS client caches data. `stat(3)` will show last time of file data change for `atime`, `ctime`, and `mtime`

- **File ownership**
  - AFS ID (not principal name!) determines file ownership, not UNIX UID
  - `chown(1)`, `chgrp(1)`
    - `root` is not sufficient to change these file attributes in AFS. Need to be member of `system:administrators`
...Differences Unix vs. AFS...

- **UNIX mode bits**
  - Group and Other permission bits are ignored
  - User permission bits determine final access permissions
  - This is very confusing to users!
  - See example on the next slide
...Unix mode bits

% fs listacl .
Access list for . is
Normal rights:
  system:authuser rl
  alfw rlidwka

% ls -l
total 290
-rw-rw-rw-  1 alfw sf   6394 Oct 4 09:26 file1
writable only by alfw despite group/other bits

-rw-------  1 alfw sf    348 Oct 4 09:26 file2
still readable by system:authuser

-r-xr-xr-x  1 alfw sf   1722 Oct 4 09:26 script1
not writable even for alfw

-rwxr-x---  1 alfw sf 285894 Oct 4 09:27 script2
still executable by system:authusers
Differences UNIX vs. AFS

- ACLs are changed after commands like `cp -R` or `tar`
  No Unix command is ACL aware
- No cross-directory hardlinks
- Quota
  - `df` does not work in AFS.
  - Use `fs listquota` or `du` instead
Missing Features in AFS

- AFS does not allow certain type of files: pipes, device files, sockets
- AFS cannot do byte-range locking in files (yet)
  - This makes AFS unusable as back-end for databases
  - Windows:
    - Client has working byte range locks
    - Full file locks on the server
- ACLs only work on directories not files (yet)
- AFS does not support mandatory file locks; only advisory locks are supported
- Strong encryption for file transport to clients (yet)
Weaknesses of AFS

• AFS is not so well suited for these situations (in increasing order of badness):
  • No reuse of read data
  • Access to files larger than cache (cache thrashing)
  • Mostly write access
  • Large numbers of directory entry changes from multiple clients

• No built-in fine grain privilege delegation mechanism (separate tools are available)
Arla

- Arla is an alternative AFS client implementation done by KTH in Sweden
- http://www.stacken.kth.se/projekt/arla/
- Supports: OpenBSD, FreeBSD, NetBSD, Linux (2.2, 2.4, 2.6), MacOS X, Tru64 4.0, 5.x (aka Digital Unix) also: AIX 4, IRIX 6, HPUX, SunOS
- Does not cache chunks of files like AFS but larger pieces
"Alternatives" to AFS

- There is a lot of confusion about possible "alternatives" to AFS:
  - **Coda**
    - Pure research project
    - Will never have production quality
  - **Intermezzo**
    - Is in Linux kernel
    - Is available only for Linux
  - **NFSv4**
    - Many similar features
    - Still too immature
AFS Documentation...

- There are several documents describing the ins and outs of AFS in great detail:
  http://www.openafs.org/doc/index.htm
  - AFS Administrator's Guide
    - Everything you need to know to administer AFS
  - AFS man pages (http://www.openafs.org/manpages/)
    - All commands in alphabetical order with all options they understand
  - AFS Quick Start Guide for UNIX
    - How to install and configure AFS clients and servers on Unix
  - AFS Quick Start Guide for Windows NT/2000/XP
    - How to install and configure AFS clients and servers on Windows
  - AFS User's Guide

- Ignore hyperlinks, read the texts sequentially!
AFS Documentation...

- Two books about AFS:
  Managing AFS: The Andrew File System
  by Richard Campbell
  Prentice Hall, 1998 (out of print)

  Distributed Services with OpenAFS:
  For Enterprise and Education
  by Franco Milicchio and Wolfgang A. Gehrke
  Springer; 2007
...AFS Documentation

- IBM's documentation as well as the book only describe AFS, not OpenAFS
- Some new options and parameters are not documented (example: `fs getcrypt/setcrypt`)
- Unix man pages (in separate package; up to date)
- OpenAFS Wiki: http://www.dementia.org/twiki/bin/view/AFSLore/
- Documentation located at http://doc.openafs.org
AFS Training and Support

- For commercial support and training, consult http://www.openafs.org/support.html

- AFS related events:
  - German AFS Workshops: http://www-dat.lrz.de/dat/afs/ws/
  - This tutorial
II. AFS Client Topics
AFS Client

• AFS client software is available for a long list of Operating Systems: Debian based Linux, Red Hat based Linux, Suse based Linux, AIX, MacOS X, SGI IRIX, Solaris, Windows, ...
• And many versions of each of these
• On many different CPUs (x86, ia64, amd64, S390, SPARC, PA-RISC, ARM, etc.)
• Nokia N800 and N810 Internet tablets
Getting Started w/o AFS Server

• You can use AFS without installing a server first
• OpenAFS daemon `afsd` has an option to allow this mode: `-dynroot` (Windows: freelance mode)
  • When `afsd` starts up, it does not mount `/afs` from any AFS server but populates it with data from its own local CellServDB file
• This mode is also useful for machines where you would like to have AFS for convenience but the machine needs to be standalone (no external dependencies) in case of an emergency
AFS Cache...

• On Linux:
  • Needs proper inode layer and operation structs
  • Diskless:
    • tmpfs does not work, use ramdisk instead
    • afsd understands -memcache
  • On Disk:
    • ext2 or ext3 work; reiserfs, xfs, jfs do NOT work

• Should be its own disk partition
• Size depends on your workload
  • Unix: up to 20GB works just fine
  • Windows: ~1GB max. on 32-bit; ~1TB max. on 64-bit
• Adjust the number of vcache/stat cache entries and volume cache entries to match the working set requirements!
...AFS Cache...

- Cache directory contains
  - V-files
    - Organized in subdirectories D0 - DXX
    - Store chunks of data
    - Size of each V-file: up to chunksize
    - Number of V-files: depends on cache size
  - Cachefiles file (binary-format)
    - Records information about V-files
  - Volumefiles file (binary-format)
    - Records information about volumes accessed:
      - Volume-to-mount point relationship
      - Volume-location info
  - Cellfiles file (binary-format)
    - Records information about cells accessed
...AFS Cache

- Cache consistency is guaranteed through **callbacks** from the file servers:
  - Server notifies clients when the
    - data in the file has changed (data version change)
    - meta data for the file has changed (no DV change)
    - lock state of the file has changed (no DV change)
    - file server ran out of space in the callback table and had to drop the callback registration (no DV change)
  - Client invalidates the status information
  - Next access goes to the file server, not the cache

- Side Note: When O(n) client machines write only one file into a single directory, then there are O(n^2) callbacks.
The directory `/usr/vice/etc/` contains most of the client configuration information:

- **ThisCell**
  - This file contains the name of the local AFS cell.  
    Example: `slac.stanford.edu`

- **cacheinfo**
  - Defines size and location of the AFS cache.  
    Example: `/afs:/usr/vice/cache:828528`

- **CellAlias**
  - Short-cuts for cell names.  
    Example:  
    `slac.stanford.edu slac`  
    `ir.stanford.edu ir`

- **CellServDB**
  - This file contains a list of all publicly known AFS cells

- The Windows equivalents are in the registry
CellServDB File

- The file `/usr/vice/etc/CellServDB` has a list of all AFS cells the client knows about.

- Here an excerpt:

```
...  
>slac.stanford.edu   #Stanford Linear Accelerator Center
134.79.18.25          # afsdb1.slac.stanford.edu
134.79.18.26          # afsdb2.slac.stanford.edu
134.79.18.27          # afsdb3.slac.stanford.edu
>ir.stanford.edu      #Stanford University
171.64.7.222           # afsdb1.stanford.edu
171.64.7.234           # afsdb2.stanford.edu
171.64.7.246           # afsdb3.stanford.edu
...  
```

- The fields behind '#' are mandatory!
- About 180 public AFS cells in there.
• All modern computers are large enough for the set of $LARGE options in /etc/sysconfig/afs
• Other important afsd options that can be set in /etc/sysconfig/afs:
  • -nosettime
    • This behavior is now the default; set it for older clients
    • Leaves the time on the client alone
    • It is better to use ntp on all your machines
  • -chunksize X
    • $x$ is log(2) of the blocksize with which AFS transfers and caches data. Default is 16 (= $2^{16\text{B}} = 64\text{KB}$). At SLAC, we are using 19 (= $2^{19\text{B}} = 512\text{KB}$) everywhere
  • -fakestat
    • Allows for $\text{ls} -l$ in /afs without doing a $\text{stat}(3)$ on foreign cell entries
...AFS Client Configuration

- `afsdb`
  - use DNS's resource record "afsdb" to locate a cell's database servers if CellServDB does not have a matching entry
  - To find out whether a cell has this configured, use `host -t afsdb <cell>`
- `mountdir /some/dir`
  - mounts AFS on `/some/dir` instead of `/afs`
- `fakestat-all`
  - uses fakestat on all mountpoints not just on mountpoints to foreign cells
AFS Client Installation - Linux

- Best starting point is official documentation: AFS Quick Start Guide for Unix Chapter “Installing Additional Client Machines”
  [http://www.openafs.org/pages/doc/QuickStartUnix/auqbg007.htm#HDRWQ1](http://www.openafs.org/pages/doc/QuickStartUnix/auqbg007.htm#HDRWQ1)

- The short version:
  - Go to [http://www.openafs.org/release/latest.html](http://www.openafs.org/release/latest.html) and download the latest binaries (RPMs for Linux)
  - Install the following RPMs on your machine:
    - openafs: contains all command suites
    - openafs-client: contains afsd and /usr/vice/etc/
    - openafs-kernel: contains kernel modules
  - Adapt /etc/sysconfig/afs, /usr/vice/etc/ThisCell, and /usr/vice/etc/cacheinfo
  - Execute /etc/rc.d/init.d/afs start
AFS Client Installation - Windows

• Newer Windows clients need servers with OpenAFS-1.2.8 or higher
• AFS client (and server) are installed via MSI
  • Easy to follow configuration (see following slides)
• Integrates with MIT Kerberos for Windows 3.2.2 or higher
• Comes with System Tray and Control Panel tools
• Context sensitive menus in Explorer for ACLs etc.
• Also comes with “Control Center” which is not functioning properly at the moment
• Release Notes:
  http://docs.openafs.org/ReleaseNotesWindows/
AFS Client Installation - Windows

Client Cell Name Configuration
Please enter the name for your default cell:

Enter AFS cell name: slac.stanford.edu

You may now configure some installation options that control how the AFS Client behaves.

- Enable AFS crypt security (Recommended)
- Enable AFS Freelance client (Recommended)
- Use DNS to Search for Cell Servers
- Use Integrated Logon

Nullsoft Install System v2.07
AFS Client Installation - Windows

Do not use the Drive mapping. Use the Explorer Shell instead and \AFS UNC paths
AFS Client - Windows

Address: \afs\slac.stanford.edu\al\alfw
AFS Client - Windows

Introduction to AFS and its Best Practices
AFS/KfW Client - Windows

MIT Kerberos for Windows 3.2.2

Configuration file: C:\WINNT\krb5.ini
AFS Client Installation – Mac OS X

- Download the package, start it and follow the instructions
AFS Client Installation – Mac OS X

- AFS space in Mac browser
AFS Client Maintenance...

- Enable encryption with `fs setcryption -crypt on` (in AFS start-up script)
  - Important: No AFS token = no encryption despite setcryption
- Side Note: Unix clients need to get a new token before setcryption takes effect! Or use `aklog -force`
...AFS Client Maintenance

- Changes in CellServDB
  - afsd *does not recognize changes automatically*
  - CellServDB is only read the first time VLDB server lists are required for a cell.
  
  If the list changes, use:

  ```
  # fs newcell bis.example.edu \ 
  afs01.bis.example.edu afs02.bis.example.edu \ 
  afs03.bis.example.edu
  ```

  - No way to remove obsolete cells
  - There are scripts that do these operations
Integrated Login...

- Automatically obtain an AFS token when logging in
- Best done with PAM on Linux/Unix:
  ```
  auth required   /lib/security/pam_env.so
  auth sufficient /lib/security/pam_krb5.so ignore_root
  auth sufficient /lib/security/pam_unix.so likeauth \ 
                      try_first_pass
  auth required   /lib/security/pamdeny.so
  ```
- **Use** `pam_krb5` **with AFS+Kerberos 5**
  - Red Hat's latest version works fine (with minor source code modifications even on Solaris)
  - Russ Allbery has one for Debian that also works on other Linuxes and Solaris
- **Use** Russ Allbery's `pam_afs_session` **to obtain AFS token from Krb5 credentials**
...Integrated Login...

- Later versions of OpenAFS for Windows work nicely with Network Identity Manager from MIT's Kerberos for Windows

- NIM is currently Windows only. Desired for Mac and Linux.
OpenSSH and AFS

- OpenSSH used to be able to forward AFS tokens
- It now can forward “only” Kerberos 5 credentials
- Install PAM to generate AFS token out of forwarded Kerberos 5 credential
  - Either Red Hat's later `pam_krb5` or
  - Russ Allbery's Debian `pam_krb5 + pam_afs_session`
PAG Clean-Up

- Garbage collection of PAGs on Linux is turned off by default in OpenAFS < 1.4.2:
  
  ```
  # cat /proc/sys/afs/GCPAGs
  2
  ```

- To turn it on:
  ```
  echo 1 > /proc/sys/afs/GCPAGs
  ```

- There are only these two values:
  ```
  AFS_GCPAGS_OK = 1
  AFS_GCPAGS_USERDISABLED = 2
  ```

- Other values mean errors, like
  ```
  # cat /proc/sys/afs/GCPAGs
  8
  => AFS_GCPAGS_EPROCWALK
III. AFS Server Topics
Planning a new AFS cell

• Cell name considerations
  • It can be anything if you don't want contact from the outside world

• Conventions and best practices
  • Use the name of your DNS domain name if you want to be visible
  • Use something like eng.example.edu if example.edu is not opportune
  • Look at CellServDB file for examples

• Pick carefully, cell name is very hard to change
• Kerberos 5 realm name should be your AFS cell name
• Register your new cell name (see http://www.central.org/csdb.html) so that others know about your new cell
# AFS Server Processes

<table>
<thead>
<tr>
<th>Server Process</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>bosserver</td>
<td>Basic OverSeer Server</td>
</tr>
<tr>
<td>fileserver</td>
<td>serves the files</td>
</tr>
<tr>
<td>volserver</td>
<td>serves volume data</td>
</tr>
<tr>
<td>vlserver</td>
<td>Volume Location Server</td>
</tr>
<tr>
<td></td>
<td>Kerberos Authentication Server</td>
</tr>
<tr>
<td>ptserver</td>
<td>Protection Server</td>
</tr>
<tr>
<td>buserver</td>
<td>Backup Server</td>
</tr>
<tr>
<td>upserver</td>
<td>Update Server</td>
</tr>
</tbody>
</table>
Types of AFS Servers

- **File Servers:**
  - Run the `bosserver fs` job (`fileservers, volserver, salvager`)
  - Have `/vicepXX` partitions
  - As many as needed

- **Database Servers:**
  - Run the processes that maintain AFS databases (`vlserver, kdc, ptserver, buserver`)
  - More than one; typically three of them

- **System Control Machine:**
  - Single server that distributes configuration data (like KeyFile)

- **Binary Distribution Machines:**
  - Distributes AFS binaries
  - One server per architecture
AFS Server Communication

- AFS is based on its own RPC system: Rx over UDP
- Designed for efficiency over WAN
- Database servers use “ubik”, a replicated database mechanism which is implemented using RPC
  - Ubik guarantees that copies of AFS databases on multiple server machines remain synchronized and consistent
  - Ubik elects on synchronization (sync) site
  - Usually this is the DB server with smallest IP address
  - Sync site sends “beacon” packet to secondary sites (about every 15 seconds)
  - If sync site is lost, new vote is cast to elect new sync site
  - If unable to find sync site: no DB updates possible
  - To make ubik work best: three DB servers
AFS and Kerberos

- AFS does not come with its own Kerberos KDC
- MIT's Kerberos 5, Heimdal and Microsoft AD work with AFS
- Give your Kerberos principals the same name as your Unix accounts
- Side note: Use your Unix UIDs as PTS AFS IDs!
There are two major open-source Kerberos 5 implementations: MIT Kerberos 5 and KTH's Heimdal

Both have advantages and disadvantages

Advice:

- Just pick one server
- Make your krb5.conf compatible for both client versions
- Install both client packages at your site
- Compile tools against whatever version they need

At SLAC: Heimdal KDC with some MIT client software (PAM, Perl module, etc.)
...MIT Kerberos 5...

• Available from: [http://web.mit.edu/kerberos/dist/](http://web.mit.edu/kerberos/dist/)
• Widely used, lots of additional software
• Supports “policies”
• AFS integration is okay now
  (`asetkey` comes with newer OpenAFS releases)
• Out-of-the-box password quality checking but awkward interface for plugging in your own check (better in versions ≥ 1.5)
...Heimdal...

- [http://www.h5l.org/](http://www.h5l.org/)
- Very easy AFS integration
- Missing “policies”:
  - Password change date in DB available since 0.8
  - No automatic account disabling after X failed authentications
- No out-of-the-box password quality checking but very nice interface to plug-in your own check
...AFS-Kerberos 5 Integration...

• See the excellent documentation in Ken Hornstein's AFS Migration Toolkit

• Only tool still required from it is kaserver database to MIT migration tool

• Quick overview:
  • Get Krb5 TGT
  • Get AFS Service Ticket with it
  • Call \texttt{aklog} to translate Krb5 AFS Service Ticket into AFS token (≈ Krb5 Service Ticket in kernel memory)
    • Works locally without talking to 524 ticket converter service
    • In fact, 524 converter is no longer required for anything (beware of old \texttt{aklog}'s!)
Here are the basic steps to integrate AFS into existing Kerberos 5 realm:

- **Add a new “afs/cellname@REALM” principal**
- **Export this principal's key into an AFS KeyFiles**

**MIT:**
- `kadmin> addprinc -randkey afs/example.edu`
- **Extract key into keytab file first:**
  ```
  kadmin> ktadd -e des-cbc-crc:afs3 afs/example.edu
  Note the "kvno <n>" in the output.
  ```
- **Convert it to an AFS KeyFiles:**
  ```
  asetkey add <n> /etc/krb5.keytab afs/example.edu
  ```

**Heimdal:**
- `kadmin> add -random-key afs/example.edu`
- **Remove all encryption types but des-cbc-crc**
  ```
  kadmin> del enctype afs/example.edu des3-cbc-sha1 ...
  ```
- **Extract KeyFile**
  ```
  kadmin> ext -k AFSKEYFILE:/usr/afs/etc/KeyFile afs/example.edu
  ```
...AFS-Kerberos 5 Integration...

• Usually, your AFS cell has same name as Krb5 realm
  • However, Kerberos 5 realm name is completely independent from the cell name
  • One realm can authenticate multiple cells
  • One cell can be authenticated by multiple realms
  • The important thing is use of \texttt{afs/<cell>@<REALM>} service key notation
  • \texttt{Current aklog will always try afs/<cell>@<TGT-REALM>} first

• If this is not the case, put the Krb5 realm name in \texttt{/usr/afs/etc/krb.conf} on all your AFS servers
AFS and Microsoft AD...

- You can use Microsoft's Active Directory Kerberos 5 with AFS
  - Create service principal `afs/cellname@REALM` in AD and set the account to “use DES only”
  - Export its key to a keytab file with Window's `ktpass` (use WinSCP or similar to transfer securely it to Unix)
  - Import the key using `asetkey` which now comes with OpenAFS ≥ 1.4.x

- Warning: The 2003 SP1 Support Tools version 5.2.3790.1830 of `ktpass` does NOT work!

- See Derrick Brashear's talk “Utilizing Active Directory as the AFS KDC”:
  [http://workshop.openafs.org/afsbpw06/talks/shadow](http://workshop.openafs.org/afsbpw06/talks/shadow)
...AFS and Microsoft AD

- A good summary is also

- To make things more efficient, you can turn off Privilege Attribute Certificate (PAC) information in tickets from the AD KDC per account following these instructions:
  http://support.microsoft.com/kb/832572
AFS File Space...

- Each file server should have partitions
  `/vicepa, ..., /vicepXY`
  - Max. 255 partitions per server
    `/vicepa - /vicepz, /vicepaa - /vicepiv`
    WARNING: `/vicepiv` is not safe to use!
  - Size limit of a partition: ~2TB in OpenAFS <= 1.4
  - Partition in OpenAFS >= 1.4.8: over 2TB
- Size of partitions is not important (at SLAC ~60GB); time to `fsck` depends on size
- Each partition can hold volumes
  - No limit on number of volumes per partition
    (well, it's 4,294,967,295 and VLDB <2GB)
  - ~2TB size limit per volume
  - The more volumes on a file server, the longer it takes to start it
Each volume can hold directories and files

- You can have 64,000 files in an AFS directory if the filenames are all less than 16 characters long. If the filenames are between 16 and 32 characters than this number decreases. There are 64,000 slots per directory. Each filename < 16 characters takes 1 slot. Each filename > 16 and <32 takes 2 slots, etc...

Each volume has a quota

- % fs listquota $HOME
  Volume Name    Quota    Used  %Used     Partition
  user.alfw      100000   25875   26%         81%

Volumes are mounted in AFS directory tree:

- % fs lsmount /afs/ir.stanford.edu/users/a/l/alfw
  '/afs/ir.stanford.edu/users/a/l/alfw' is a mount point for volume '#user.alfw'
namei vs. inode Interface

• On some OSes, the AFS file server comes with two different interfaces to the file system layer
  • AFS' native inode interface
    • Requires special `fsck`
    • Does NOT work with journaling file systems
    • a little faster
  • namei interface
    • Uses `fsck` from OS
    • Works on all file systems
    • almost as fast as inode in new OpenAFS versions

• Choice needs to be made at compile time
• They are incompatible (no easy migration)
• Linux: only namei; Solaris: both
AFS File Server File Systems

- with namei interface, a server's `/vicepXX “partitions”` can be any (local or non-local) file system
- Linux
  - XFS is excellent choice
  - ext3 is slower
- Solaris
  - Logging UFS works well
  - so does ZFS with OpenAFS >= 1.4.6 (fsync!)
- If `/vicepXY's are not real mountpoints, they need `/vicepXY/AlwaysAttach to make sure they get mounted by the file server (AlwaysAttach is required anytime the partition contains any data on it that is not AFS data)
Hardware Capacity Planning

• You should plan on (eventually) having three AFS database servers
  • These can be small machines but need to be very secure (i.e. don't run any other services on them!)
• File servers: no limits here.
  • Use RAID 6 disks or a SAN
  • Decide on inode vs. namei interface
  • Use journaling file system for /vicepXX partitions with namei
  • Advise: more smaller servers is better than fewer larger!
    • IBM/Transarc recommends 200 clients per server
    • At SLAC: 1000-2000 clients per server
    • 2000-3000 clients per server is pushing it too hard!
AFS Administrator Roles

- No shared administrator accounts please!
- Use Kerberos 5 convention:
  alfw/admin@ENG.EXAMPLE.EDU
- Types of administrators in AFS:
  - Members of system:administrator group can run privileged PTS commands and change ACLs
  - Members of server's admin list can perform all vos/bos/backup commands.
    % bos adduser server1 alfw.admin
  - Local authority ("root") on servers (access to KeyFile)
  - Can also add alfw/admin to KDC ACL file for Kerberos admin privileges if desired
- No finer admin delegation possible (→tools)
AFS Server Installation...

- Install Kerberos 5 first (see Heimdal/MIT/MS documentation)
  - Use your AFS cell name as Kerberos 5 realm name!

- Create a principal
  \texttt{afs/eng.example.edu@ENG.EXAMPLE.EDU}

- Create a KeyFile (here: Heimdal):
  \texttt{# ktutil -k AFSKEYFILE:KeyFile get \}
  \texttt{afs/eng.example.edu@ENG.EXAMPLE.EDU}

- Do the AFS server installation (skip all kas/kaserver \textbf{and} \texttt{bos addkey} steps)

- If you convert an existing AFS cell to Heimdal, see
  \texttt{http://www.dementia.org/~shadow/ka2heim.txt}

- List of resources:
  \texttt{http://www.slac.stanford.edu/~alfw/kerb5.html}
• IBM/Transarc documentation recommends restarting all AFS servers once a week. This is not necessary any more. Many memory leaks have been plugged.

N.B.: Restarting the servers cause the clients to flush all of their callbacks. This is not true for the Demand Attach server which preserves callback information across reboots in fsstat.dat
AFS Server Software...

- AFS server software on Unix looks like this:

```
/usr/afs/:
total 9
-rwxr-xr-x  2 root  bin   2048 Jan 10 19:39 bin
-dwrx-------  2 root  other  512 Jul  1 2003 db
-rwxr-xr-x  2 root  sys   512 Oct  6 11:06 etc
-dwrx-------  2 root  other  512 Feb  5 14:06 local
-dwrx-------  2 root  other  512 Feb  5 14:01 logs

/usr/afs/bin:
total 48284
-rwxr-xr-x  1 root  root  1513500 Jan 10 11:30 bos
...
-rwxr-xr-x  1 root  root  1430308 Jan 10 11:23 vos
```
...AFS Server Software...

```
/usr/afs/db:
total 11636
-rw------- 1 root other  7349312 Mar  8 10:32 bdb.DB0
-rw------- 1 root other     64 Mar  8 10:32 bdb.DBSYS1
-rw------- 1 root other  629824 Mar  8 10:26 kserver.DB0
-rw------- 1 root other     64 Mar  8 10:26 kserver.DBSYS1
-rw------- 1 root other 1329216 Mar  8 08:35 prdb.DB0
-rw------- 1 root other     64 Mar  8 08:35 prdb.DBSYS1
-rw------- 1 root other 2549824 Mar  8 10:08 vldb.DB0
-rw------- 1 root other     64 Mar  8 10:08 vldb.DBSYS1

/usr/afs/etc:
total  5
-rw-r--r-- 1 root other     156 Sep 15 09:16 CellServDB
-rw------- 1 root other     100 Sep 15 09:16 KeyFile
-rw-r--r-- 1 root other      9 May 18   1997 License
-rw-r--r-- 1 root other     17 Jun 12  2003 ThisCell
-rw-r--r-- 1 root root      231 Jan 26  12:55 UserList
```
...AFS Server Software...

/usr/afs/local:
total 27
-rw-r--r-- 1 root root  464 Feb  5 14:01 BosConfig
-rw-r--r-- 1 root root   0 Feb  5 14:01 SALVAGE.fs
-rw------- 1 root other 23872 Mar  8 10:32 kaserverauxdb
-rw-r--r-- 1 root other   0 Jul  7  2003 salvage.lock
-rw-r--r-- 1 root root   32 Feb  5 14:06 sysid
-rw-r--r-- 1 root root   32 Jan 10 20:01 sysid.old
...AFS Server Software

/usr/afs/logs:
total 11331
-rw-r--r-- 1 root root  8778 Feb 11 00:58 AuthLog
-rw------- 1 root other  4096 Mar  4 15:35 AuthLog.dir
-rw------- 1 root other  524288 Mar  8 10:32 AuthLog.pag
-rw-r--r-- 1 root other  74184 Mar  7 02:01 BackupLog
-rw-r--r-- 1 root root  111 Feb  5 14:01 BosLog
-rw-r--r-- 1 root root  841 Feb  5 14:01 FileLog
-rw-r--r-- 1 root root  318 Feb  5 14:01 PtLog
-rw-r--r-- 1 root root  451 Feb  5 14:01 SalvageLog
-rw-r--r-- 1 root root  430 Feb  5 14:01 VLLog
-rw-r--r-- 1 root root   77 Feb  5 14:01 VolserLog
-rw------- 1 root other 8944840 Jul  7 2003 corefile.fs
-rw------- 1 root other 1989480 Jul 17 2003 corevol.fs
AFS Security...

- The "afs" and "krbtgt.ENG.EXAMPLE.EDU" principals are two very important entries in your KDC database.
- `/usr/afs/etc/KeyFile` stores the password (`string2key(passwd)` to be exact) for "afs".
- A compromise of these principals compromises your Kerberos realm/AFS cell.
- Change password of "afs" regularly (use OpenAFS-1.2.11 or later on your servers!). Key for "krbtgt" was changed automatically by kaserver.
- The `KeyFile` is used:
  - by servers to authenticate themselves
  - by all commands using the `-localauth` flag.
...AFS Security...

- All AFS commands work over the Internet
- All AFS commands that don't require authentication can be executed by everybody against your AFS servers/clients on the Internet
- Do not encode confidential information in volume names!
- Example (see next slides):
...AFS Security...

% fs listquota /afs/grand.central.org/software
Volume Name             Quota   Used % Used Partition
root.software.readonly  5000     6   0%    8%

% vos examine root.software -cell grand.central.org
vsu_ClientInit: Could not get afs tokens, running unauthenticated.
root.software 536870930 RW 6 K On-line
     GRAND.MIT.EDU /vicepa
  RWrite 536870930 ROOnly 536870931 Backup 536870932
 MaxQuota  5000 K
 Creation  Wed Dec 5 12:44:37 2001
      Copy   Wed Apr 8 08:15:56 2009
   Backup   Thu May 14 21:05:55 2009
 Last Update Wed Dec 31 01:18:22 2008
 12 accesses in the past day (i.e., vnode references)

RWrite: 536870930 ROOnly: 536870931 Backup: 536870932
number of sites -> 4
   server GRAND.MIT.EDU partition /vicepa RW Site
   server andrew.e.kth.se partition /vicepa RO Site
   server PENN.CENTRAL.ORG partition /vicepa RO Site
   server GRAND.MIT.EDU partition /vicepa RO Site
% vos listpart -server GRAND.MIT.EDU -cell grand.central.org -noauth
The partitions on the server are:
  /vicepa  /vicepb
Total: 2

% vos listvol GRAND.MIT.EDU /vicepa -cell grand.central.org
vsu_ClientInit: Could not get afs tokens, running unauthenticated.
Total number of volumes on server GRAND.MIT.EDU partition /vicepa: 249
a.mailman                                   536871082 RW       1603 K On-line
a.mailman.backup                       536871084 BK        1605 K On-line
a.mailman.readonly                    536871083 RO       1754 K On-line
a.mb.comerrers                           536871133 RW          40 K On-line
a.mb.comerrers.backup               536871135 BK           40 K On-line
a.mb.comerrers.readonly            536871134 RO           40 K On-line
...
...AFS Security

- AFS clients can access volumes by name without the use of the full path from the `root.cell` volume.
- Do not assume that because the directory containing a mount point cannot be listed that the volume cannot be accessed!
- Remember: AFS volumes can have multiple mount points even outside your own cell.
# AFS' UDP Ports

<table>
<thead>
<tr>
<th>Port</th>
<th>Machine Type</th>
<th>Process</th>
</tr>
</thead>
<tbody>
<tr>
<td>7000</td>
<td>file server</td>
<td>fs</td>
</tr>
<tr>
<td>7001</td>
<td>client</td>
<td>afsd (for callbacks)</td>
</tr>
<tr>
<td>7002</td>
<td>database server</td>
<td>ptserver</td>
</tr>
<tr>
<td>7003</td>
<td>database server</td>
<td>vlserver</td>
</tr>
<tr>
<td>7004</td>
<td>database server</td>
<td>kaserver</td>
</tr>
<tr>
<td>7005</td>
<td>file server</td>
<td>volserver</td>
</tr>
<tr>
<td>7007</td>
<td>server</td>
<td>bosserver</td>
</tr>
<tr>
<td>7008</td>
<td>server</td>
<td>upserver</td>
</tr>
<tr>
<td>7009</td>
<td>AFS/NFS Translator</td>
<td>-</td>
</tr>
<tr>
<td>7021</td>
<td>backup server</td>
<td>buserver</td>
</tr>
<tr>
<td>7025 – 7032+</td>
<td>backup tape coordinator</td>
<td>butc</td>
</tr>
<tr>
<td>7101</td>
<td>clients/servers</td>
<td>xstat</td>
</tr>
</tbody>
</table>
AFS and Firewalls...

- AFS client
  - Allow established UDP connections with a source port of 7000-7007 to any non-restricted destination port
  - Allow all UDP traffic with a destination port of 7001 from your AFS file servers to any AFS client
  - If kaserver: allow established UDP traffic with a source port of 88 or 750 to any high-numbered port on the client

- AFS file servers
  - Allow UDP traffic to ports 7000, 7005

- AFS VLDB servers
  - Allow UDP traffic to ports 7002 and 7003

- Allow connections from your master to upclient
- 7007 only if you want to allow bos access
...AFS and Firewalls

- Kerberos 5 UDP ports: 88, 464
  - 464 only needs to be open on the master
- You don't need all ports open on your perimeter
- AFS and NAT:
  - Make sure the state of connections is stored long enough
  - Callbacks can come much later
  - 2 hours or more seems a good value
- `net.ipv4.netfilter.ip_conntrack_udp_timeout=480`
  - how long the firewall will permit inbound packets to be delivered after the last outbound packet between a given set of endpoints
- `net.ipv4.netfilter.ip_conntrack_udp_timeout_stream=900`
  - how long an idle port mapping will be maintained before it can be reused by a new client
AFS File Space Layout

- **Best Practices:**
  
  `/afs/eng.example.edu/ = $cell
  public/
sun4x_510/usr/afsws/

  i386_linux26/usr/afsws/

  ....
  common/usr/local/
  user/a..z/{a..z}/
group/
software/
project/

- **User home directories would be:**
  
  `/afs/eng.example.edu/user/a/alfw/ or
  /afs/eng.example.edu/user/a/l/alfw/`
Volume Naming Conventions

• AFS has no way of keeping track of mount points for volumes
• Best Practices: 
  Encode mount point name into the volume name!
• Examples:
  u.alfw   -> $cell/user/a/alfw
  sw.gcc   -> $cell/software/gcc
  prjct.AFS -> $cell/project/AFS
• Example from SLAC:
  g.babar.detect.emc.bin -> /afs/slac/g/babar/detector/emc/bin/
• Volume names can only be 22 characters long
• Legal characters for volume names: .ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789-
ACLs on AFS File Space...

- /afs/ and for /afs/@cell/ should be open to anybody:

  system:administrators rlidwka
  system:anyuser rl

- **Note**: @cell is a short-cut used in some cells:
  /afs/@cell is symbolic link to
  /afs/eng.example.edu

- **Better**: use client's CellAlias file
...ACLs on AFS File Space...

- `$cell/software/` should be available
  - from all your machines without AFS token
  - from off-site with valid AFS token
  - ACLs:
    - `system:administrators` `r` `lidwka`
    - `system:authuser` `r`
    - `system:slac` `r` `l`

- Where `system:slac` is a PTS group with SLAC's full IP address range (more later)
- The same for `$cell/common/usr/local/

- Set ACLs in `$cell/public/` to
  - `system:administrators` `r` `lidwka`
  - `system:anyuser` `r` `l`
ACLs on AFS File Space

• User home directories are tricky!
• Stanford University uses these ACLs for Unix home directories:
  system:backup rl
  system:administrators rlidwka
  system:campushosts l
  system:www-servers l
  alfw rlidwka

• Other ACLs are possible
  • At SLAC: more open but much more complicated to make secure
### ACLs on AFS File Space

Each Stanford home directory has some default subdirectories with default ACLs:

<table>
<thead>
<tr>
<th>Directory</th>
<th>ACLs</th>
</tr>
</thead>
<tbody>
<tr>
<td>$HOME/public/</td>
<td>system:backup rl</td>
</tr>
<tr>
<td></td>
<td>system:administrators rlidwka</td>
</tr>
<tr>
<td></td>
<td>system:anyuser rl</td>
</tr>
<tr>
<td></td>
<td>alfw rlidwka</td>
</tr>
<tr>
<td>$HOME/WWW/</td>
<td>system:backup rl</td>
</tr>
<tr>
<td></td>
<td>system:administrators rlidwka</td>
</tr>
<tr>
<td></td>
<td>system:www-servers rl</td>
</tr>
<tr>
<td></td>
<td>alfw rlidwka</td>
</tr>
<tr>
<td>$HOME/private/</td>
<td>system:backup rl</td>
</tr>
<tr>
<td></td>
<td>system:administrators rlidwka</td>
</tr>
<tr>
<td></td>
<td>alfw rlidwka</td>
</tr>
<tr>
<td>$HOME/Mail/</td>
<td>system:backup rl</td>
</tr>
<tr>
<td></td>
<td>system:administrators rlidwka</td>
</tr>
<tr>
<td></td>
<td>alfw rlidwka</td>
</tr>
<tr>
<td>$HOME/.ssh/</td>
<td>system:backup rl</td>
</tr>
<tr>
<td></td>
<td>system:administrators rlidwka</td>
</tr>
<tr>
<td></td>
<td>system:anyuser l</td>
</tr>
<tr>
<td></td>
<td>alfw rlidwka</td>
</tr>
</tbody>
</table>
Machine Groups in PTS...

- A machine group is a PTS group with IP addresses as members
- Cannot use IP based ACL unless it is in a group
- First, create a PTS entry for your machine(s). Set aside a range if PTS “user” IDs for this!
  pts create 192.168.2.43 -id 942
  - Wildcard value, 0, allowed in certain positions of IP address:
    - A.B.C.0 matches IP addresses A.B.C.0 – A.B.C.255
    - A.B.0.0 matches IP addresses A.B.0.0 – A.B.255.255
    - A.0.0.0 matches IP addresses A.0.0.0 – A.255.255.255
  - For SLAC's full IP address range: 134.79.0.0
...Machine Groups in PTS...

- Then create a PTS group:
  ```bash
  % pts creategroup \
      -name host-www \
      -owner system:administrators
  ```

- And add the machine user to it:
  ```bash
  % pts adduser -user 192.168.2.43 \
      -group host-www
  ```

- Now you can use it like: `fs sa . host-www rl`

- Conventions:
  - Use a separate PTS ID range for the machine users (SLAC is using the 800 – 999 range for “special” PTS entries)
  - Name the groups `host-<machine name>`
...Machine Groups in PTS

- These groups are not really advisable
- Use Kerberos keytab and (maybe) Russ' k5start instead
  - encrypted AFS traffic
  - privilege only for a service and not entire machine
Software Packages in AFS Space

• Suggestion for structure of packages under $cell/software/

• Criteria:
  • Different versions of same package for same architecture
  • Different versions for different architectures
  • One /usr/local/ for all architectures
Software Packages in AFS Space

$cell/software/mypackage/

- **Version per platform**
- **Equivalent platform**
- **Platform dependent**
- **Platform independent**

<table>
<thead>
<tr>
<th>Platform</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>rs aix42</td>
<td>version1</td>
</tr>
<tr>
<td>i386 linux24</td>
<td>version2</td>
</tr>
<tr>
<td>sun4x 58</td>
<td>version2</td>
</tr>
<tr>
<td>sun4x 59</td>
<td>sun4x 58</td>
</tr>
<tr>
<td>bin</td>
<td>@sys/bin</td>
</tr>
<tr>
<td>man</td>
<td>@sys/man</td>
</tr>
<tr>
<td>version1/</td>
<td></td>
</tr>
<tr>
<td>version2/</td>
<td></td>
</tr>
</tbody>
</table>

| sun4x 59         | sun4x 58    |
| bin             |             |
| man             |             |
| version1/        |             |
| version2/        |             |

/sun4x_59/bin/mybin

- symlinks to
  - $cell/software/mypackage/bin/mybin
- resolves to
  - $cell/software/mypackage/@sys/@sys/bin/mybin
- resolves to
  - $cell/software/mypackage/version2/sun4x_58/bin/mybin

/usr/local/bin/mybin

- symlinks to
  - $cell/software/mypackage/bin/mybin
- resolves to
  - $cell/software/mypackage/@sys/@sys/bin/mybin
- resolves to
  - $cell/software/mypackage/version2/sun4x_58/bin/mybin

Introduction to AFS and its Best Practices
Software Packages in AFS Space

- To make this scheme work, software packages need to be compiled with special care
- Recommendation: Create two PTS groups for each software package: `owner-sw-mypackage` and `maint-sw-mypackage` with the owner and maintainer as members
- Set ACLs in `$cell/software/mypackage/` to:
  - `owner-sw-mypackage` rlidwka
  - `maint-sw-mypackage` rlidwk
  - `system:administrators` rlidwka
  - `system:authuser` rl
  - `system:slac` rl
AFS Volume Replication...

• AFS volumes can be replicated on demand
  % vos create afs04 /vicepa sw.gcc
  % vos addsite afs04 /vicepa sw.gcc
  % vos addsite afs05 /vicepb sw.gcc

• RO copy on same server/partition eats almost no additional space

• No automatic update mechanism; by hand with:
  % vos release sw.gcc

• Clients will load-balance access among all RO copies

• Clients will switch to another RO copy in case of failure (after timeout)

• There can be up to 11 RO copies per volume
• **Traversal rules:**
  • If root.afs is replicated, use RO volume. RW otherwise
  • As path is traversed
    • Current volume is RO and next volume is
      • Replicated: RO volume is used
      • Not replicated: use RW volume
    • Current volume is RW: must access RW volume
  • Assumption: you did not force mount point to RO volume
  • If you forget to replicate volume, replicated copies of volumes mounted beneath it will never be accessed
If everything is set up correctly, clients will prefer the replicated RO path!

Once they run across a RW volume, their bias will switch.

You can mount RW and RO volumes:

```
% fs mkmount \
  -dir /afs/.eng.example.edu/software/gcc \n  -vol sw.gcc
```

or

```
% fs mkmount \
  -dir /afs/.eng.example.com/software/gcc \n  -vol sw.gcc.readonly
```

but don't do this! It will confuse everybody.
...AFS Volume Replication

- Namei file servers only:
  - In case you lose a RW volume, a RO volume can easily be converted to a RW volume with:
    
    ```bash
    vos convertROtoRW -server <machine name> \ 
    -partition <partition name> \ 
    -id <volume name or ID>
    ```
AFS Volume Mountpoints

• You can have many mountpoints to the same AFS volume
• You can mount AFS volumes from foreign cells and your volumes can be mounted elsewhere
  • access to content is still enforced by local AFS server
• Enable auditlogs to track mountpoints from creation time onward
Volume Moves...

- % vos move u.alfw server1 /vicepc \ server2 /vicepf

- Very complicated operation behind the scenes
  1. Volume gets locked and special backup clone is generated (very efficient: fraction of a second) and unlocked again
  2. Volume is created at destination; check for enough space
  3. Clone of original volume is dumped and restored at destination
  4. Clone volume gets deleted
  5. Lock original volume and incrementally copy any missed data
  6. New volume is brought on-line, original volume is put off-line, VLDB is updated, volume state is unlocked
  7. Original volume and all backup volumes are deleted
...Volume Moves

- Use `-verbose` to see each single step
- Due to very short locking times, transparent to users
- Location independent path names don't change
- All backup volumes have to be re-created by hand!
Backup Volumes

- AFS can be configured to create a full or incremental backup of volumes at a specific time of the day.
- You can create a backup volume for a single read/write volume with
  \% vos backup volume.name
- To create backup volumes for multiple read/write volumes, use
  \% vos backupsys -prefix “user.” -prefix “sw.”
  - Many more options available
- Issuer of these commands must be in
  /usr/afs/etc/UserList
Backup

• Most backup tools don't know about AFS' ACLs
• Exceptions: Amanda (http://www.amanda.org/), TiBS (Teradactyl), older versions of TSM (IBM)
• Simple backup mechanism:
  • Use the .backup volumes (see previous slide)
  • Do a % vos dump u.alfw.backup /tmp/u.alfw.<date>
  • Backup these volume dumps instead
  • Best Practice: mount the backup volume in each user's AFS home directory
    % fs mkmount ~alfw/.backup u.alfw.backup
  • Restore: read volume dump from tape, then
  • % vos restore afs01 /vicepa u.alfw.<date> /tmp/u.alfw.<date>
VI. Advanced AFS Topics
Some Tips & Tricks...

• AFS client cache corruption
  • Used to be common on Linux (not any more)
  • Two clean-up methods:
    • Flush the cache with
      \texttt{fs setcachesize 1} \& \texttt{fs setcachesize 0}
    • Delete \texttt{/usr/vice/cache/CacheItems} and reboot

• Deleting a volume is \textbf{not} \texttt{vos del}:
  \texttt{vos delentry} delete VLDB entry for a volume
  \texttt{vos remove} delete a volume

• If you have a replicated volume and forget to release it, changes made to it and the volumes underneath it, including replications, will not be seen.
...Some Tips & Tricks

• For speeding up volume meta operations:
  vos changeaddr -oldaddr <IP address> -remove
  useful after AFS servers changed IP address or were decommissioned
AFS and Java

- There is a Java API for AFS
- Complete binding of C AFS API for Java
- For details, see http://www.dementia.org/twiki/bin/view/AFSLore/JavaAdminAPI
AFS and Perl

• There are two very different Perl modules on CPAN dealing with AFS:
  • `AFS::Command` by Phil Moore is an OO wrapper around the command line utilities
  • `AFS` by Norbert Grüner is an OO API written in XS interfacing the AFS libraries
• `AFS::Monitor` interfaces the AFS monitoring and debugging utilities
• MIT: `Authen::Krb5::Admin`, Heimdal: `Heimdal::Kadm5`
The AFS Perl module

- Perl functions for all AFS command line tools: klog, tokens, fs, kas, pts, vos, bos
- separated into 15 Perl classes:
  - Access to Cache Manager routines: AFS::CM
  - Access to File Server routines: AFS::FS
  - Managing ACLs: AFS::ACL
  - Manipulating Cell Configuration: AFS::Cell
  - Access to Protection Server routines: AFS::PTS
  - Access to Authentication Server routines: AFS::KAS
  - Manage Kernel Token Cache: AFS::KTC_TOKEN
  - Deal with principals: AFS::KTC_PRINCIPAL
  - Deal with encryption keys: AFS::KTC_EKEY
What does the Perl module do?

- Access to share utility functions: AFS::Utils
- Error handling, debugging, etc.: AFS
- Access to VLDB related functions: AFS::VLDB
- Access to Volume Location Server: AFS::VOS
- Access to Basic Overseer Server: AFS::BOS
Heimdal Perl Example...

Equivalent of `kadmin -p $admin list -l $user`  
or `kas examine $user -admin $admin`
...Heimdal Perl Example

Output from previous example:

alfw/admin@SLAC.STANFORD.EDU's Password:
  Principal: alfw@SLAC.STANFORD.EDU
  Principal Expires: never
  Password Expires: 2008-06-16 23:59:59 UTC
  Last Password Change: never
    Kvno: 21
    Mkvno: 0
  Max Ticket Life: 90000 seconds
  Max Renewable Life: 2147483647 seconds
  Keytypes(salts): aes256-cts-hmac-sha1-96(pw-salt),
                  arcfour-hmac-md5(pw-salt)
**AFS Perl Examples...**

*Equivalent of* `vos listvol afs05 /vicepa -extended`

```perl
#!/usr/bin/perl -w

use strict;
use AFS::VOS;

die "Usage: $0 server [partition]\n" if $#ARGV < 0;

my $server = shift;
my $part   = shift; $part = '' unless $part;
my $fast   = 0;
my $extend = 1;

my $vos = AFS::VOS->new;
print "Error: ", $AFS::CODE, "\n" if ($AFS::CODE);

my $partlist = $vos->listvol($server, $part, $fast, $extend);
print "Error: ", $AFS::CODE, "\n" if ($AFS::CODE);

print_ext($partlist);
```
...AFS Perl Examples...

Output from previous example:

Partition /vicepa:
......

Volume: root.afs
Key: backupID, Value: 536870914
Key: cloneID, Value: 536870913
Key: creationDate, Value: Mon May 19 13:41:40 1997
Key: inUse, Value: On-line
Key: maxquota, Value: 5000
Key: parentID, Value: 536870912
Key: partition, Value: /vicepa
Key: server, Value: afs05.slac.stanford.edu
Key: type, Value: RW
Key: updateDate, Value: Thu Jul  5 15:04:27 2007
Key: volid, Value: 536870912
    Key 0-60sec:
        Key: dirDiffAuth, Value: 0
        Key: dirSameAuth, Value: 0
        Key: fileDiffAuth, Value: 0
        Key: fileSameAuth, Value: 0

......
...AFS Perl Examples...

Equivalent of `vos listvldb root.afs`

```perl
#!/usr/bin/perl -w
use strict;
use AFS::VLDB;

my $vol = shift;
$vldb = AFS::VLDB->new;
my $vldblist = $vldb->listvldbentry($vol);

print "Error: ", $AFS::CODE, "\n" if ($AFS::CODE);
print_vldblist($vldblist);
```

Key: `root.afs`
  Key: `nServers`, Value: 9
  Key: `flags`, Value: 28672
  Key: `Backup`, Value: 536870914
  Key: `ROnly`, Value: 536870913
  Key: `RWrite`, Value: 536870912

Server number 1:
  Key: `serverFlags`, Value: 4
  Key: `name`, Value: `afs05.slac.stanford.edu`
  Key: `type`, Value: `RW`
  Key: `partition`, Value: `/vicepa`

Server number 2:
  Key: `serverFlags`, Value: 2
  Key: `name`, Value: `afs10.slac.stanford.edu`
  Key: `type`, Value: `RO`
  Key: `partition`, Value: `/vicepa`

.......
#!/usr/bin/perl -w
# basic script to create a new AFS account

use strict;
use AFS::VOS;
use AFS::PTS;
use AFS::ACL;
use AFS::Cell qw(localcell);
use AFS::FS qw(setquota);
use Heimdal::Kadm5;

my $user = shift;
my $initial = substr($user, 0, 1);

# create PTS entry
my $pts = AFS::PTS->new;
my $id = $pts->createuser($user);

# create user volume
my $vos = AFS::VOS->new;
$ok = $vos->create('server1', '/vicepa', 'u.$user');

# continue next slide
...AFS Perl Examples...

# cont'd. of basic AFS account creation

# create volume mountpoint
my $cell = localcell;
$ok = mkmount("/afs/$cell/user/$initial/$user", "u.$user");
$ok = setquota("/afs/$cell/user/$initial/$user", 50000);

# set ACLs in new directory
my $acl = AFS::ACL->new('');
$acl->set($user => 'rlidwka');
$acl->set('system:administrators' => 'rlidwka');
$acl->set('system:backup' => 'rl');
$acl->set('system:anyuser' => 'l');
my $ok = $acl->apply("/afs/$cell/user/$initial/$user");

# create all subdirectories and set ACLs
# copy .dot files into new home directory

# continue next slide
# cont'd. of basic AFS account creation

# create Krb5 principal
my $ADMIN = shift;

# this line will ask for $ADMIN's password:
my $client = Heimdal::Kadm5::Client->new(
    Principal => $ADMIN);

my $new_princ = $client->makePrincipal($user);
$new_princ->setMaxLife(25*60*60);
$new_princ->setMaxRenewableLife(14*24*60*60);
$new_princ->setAttributes(0);

my $newpass = shift;  # new user's password
$client->createPrincipal($new_princ, $newpass, 0);

$client->handle->c_flush();
AFS Perl: Further Information

- AFS Perl module comes with excellent documentation
- a lot of helpful example scripts included
- plays nicely with other modules
- Kerberos 5:
  - `AFS::KAS` needs to be replaced with Perl module for your Kerberos 5 distro
    - MIT: `Authen::Krb5::Admin` (from CPAN)
    - Heimdal: `Heimdal::Kadm5` (from Heimdal's home page)
  - `AFS::KTC_PRINCIPAL`, `AFS::KTC_TOKEN` need to be replaced with `Authen::Krb5`
uss vs. AFS Perl

- **uss** is a tool that comes with AFS and allows creation and deletion of AFS accounts.
- I like AFS Perl much better:
  - More flexible
  - More robust
  - It's Perl
Perl Module AFS::Monitor

- AFS::Monitor interfaces the following AFS monitoring and debugging utilities: afsmonitor, cmdebug, rxdebug, scout, udebug, xstat_cm_test, xstat_fs_test

available via http://www.slac.stanford.edu/~alfw/AFS-Monitor/

or CPAN http://www.cpan.org/modules/by-authors/id/A/AL/ALFW/
AFS::Monitor Example...

- **check_ubik.pl**:
  - **AFS database server ubik tester**:
    - Is sync site still sync site?
    - Are non-sync sites still non-sync sites?
    - Is recovery state still “0x1f”? (best possible state)
  - **uses udebug()**

- **This and other applications are available via**
#!/usr/bin/perl
use AFS::Monitor qw(udebug);

my $result = udebug(server => $opt_H, port => 7003);
my $status = 'CRITICAL' if ($AFS::CODE);
my $sync_site_name = 'afsdb1'; # IP: 134.79.18.25
my $afsdb_server_pattern = 'afsdb';

if ($result) {
    if ($opt_H eq $sync_site_name) {
        $status = 'WARNING' unless $result->{amSyncSite};
        $status = 'WARNING' unless ($result->{recoveryState} == 0x1f);
    } elsif ($opt_H =~ /$afsdb_server_pattern/) {
        $status = 'WARNING' if $result->{amSyncSite};
        $status = 'WARNING' unless ($result->{lastYesHost} eq '134.79.18.25')
    }
}

print "AFS $status\n";
exit $ERRORS{$status};
tklife...

- Users are surprised when AFS tokens expire
- Little script that reminds them

#!/usr/local/bin/perl

# File: tklife
# Description: Check the token expiration
# Author: C. Bondila (CERN/CN/DCI)
# Contact: Setup Support <Setup.Support@cern.ch>
# Version: 3.2
# Date: 17 May 1995
# Last Change: Wed Sep 9 17:35 1998 by W. Friebel <friebel@ifh.de>

$ENV{"PATH"} = '/usr/bin:/usr/afsws/bin';

use Date::Parse;
use Getopt::Std;

&getopts('l:') || die "Usage: tklife [-l <warning time in hh[:mm:ss]>]";
$warn_time=3600;
$warn_time = $1*3600 + $2*60 + $3 if $opt_l =~ /\(\d+):?(\d+):?\(\d+\)/;

#get token value
$token=`tokens`;

if ($token !~ /Expire/) {
  print ">>> No AFS token! <<<\n";
} else {
  for (split(/\n/, $token)) {
    next unless /Expire/;
    if ( /t?T?okens for (.*) \[Expires (.*)\]/ ) {
      $cell=$1;
      $date=$2;
      if ( ($secs = mystr2time($date) - time()) <= $warn_time) {
        printf ">>> AFS token is going to expire in %d minutes on $cell! <<<\n", int($secs/60);
      }
    } else {
      print ">>> AFS token expired on $1 ! <<<\n"
      if /t?T?okens for (.*) \[>> Expired <<\]/;
    }
  }
}
sub mystr2time {
    # this routine is identical to str2time except it prefers the future
    # instead of the past, if the year is not given

    my @t = strptime(@_);
    return undef unless @t;

    my($ss,$mm,$hh,$day,$month,$year,$zone) = @t;
    my @lt = localtime(time);

    $hh ||= 0;
    $mm ||= 0;
    $ss ||= 0;

    $month = $lt[4] unless(defined $month);
    $day = $lt[3] unless(defined $day);
    $year = ($month < $lt[4]) ? ($lt[5] + 1) : $lt[5] unless(defined $year);

    return defined $zone ? timegm($ss,$mm,$hh,$day,$month,$year) - $zone :
        Date::Parse::timelocal($ss,$mm,$hh,$day,$month,$year);
}

• Put the following into your $HOME/.zshrc
• Other shells work too (syntax may vary, though)

periodic_tklife() {
    sched +00:51 periodic_tklife
    ${HOME}/scripts/tklife
}

if [ $USER != "root" ]; then
    periodic_tklife
fi
...tklife

alfw@melmac 1> pagsh

<<<< No AFS token! <<<<<<

alfw@melmac 2> kinit -l 5min alfw; afslog

alfw@melmac 3> tklife

<<<< AFS token is going to expire in 4 minutes on \ afs@slac.stanford.edu! <<<<<<
AFS and Cron jobs

- How can a cron job securely acquire an AFS token?
- The following uses Russ Allbery's `k5start`:
  - New Kerberos 5 principal `alfw/cron@SLAC.STANFORD.EDU`
  - Keytab file readable only by `alfw` locally on a machine: `/var/keytabs/alfw_cron.keytab`
  - Create AFS user `alfw.cron` in PTS
  - Add `alfw.cron` to the appropriate AFS ACLs (eg. one directory in `alfw's` AFS home directory)
  - `crontab entry with k5start:`
    ```
    5 5 * * 2 /usr/bin/k5start -qtU -f /var/keytabs/alfw_cron.keytab /afs/slac/a/l/alfw/scripts/cronjob1.sh
    ```
  - `cronjob1.sh` now runs with Kerberos 5 ticket for `alfw/cron` and an AFS token (in PAG) for `alfw.cron`
Delegating AFS Administration...

- AFS comes with a very coarse grain delegation scheme
- Tools for remote AFS aware “sudo” functionality:
  - DESY's new arc v2
    - Written in Perl
    - Uses SASL
  - Russ Allbery's remctl:
    http://www.eyrie.org/~eagle/software/remctl/
Delegating AFS Administration

- Protocol Schema:

  **Client**
  - Krb TGT
  - Send Command
  - Receive Result

  **Server**
  - /etc/krb5.keytab
  - Receive Command
  - Authorization for Principal
  - Set new PAG
  - Obtain AFS admin token from srvtab file
  - Execute Command
  - Send Result

  Mutual Authentication
**More Tools...**


- **AFS**
  - **Balancing**: Balance volumes, usage, and accesses among servers
  - **frak**: Show differences between AFS file trees or volumes
  - **fsr**: Recursive wrapper around fs directory commands
  - **lsmounts**: Search through a directory for mount points
  - **Monitoring**: Nagios-compatible probes to monitor AFS
  - **Mount points**: Maintain a database of volume mount points
  - **mvto**: Smart vos move with usage balancing
  - **partinfo**: Nicely formatted partition usage report
  - **Reporting DB**: Load AFS metadata into a SQL database for reporting
  - **volcreate**: Smart vos create with automatic volume placement
  - **volnuke**: Smart vos remove that finds volume location
...More Tools...

- **Kerberos**
  - `kadmin-remctl`: kadmin remctl backend and password reset client
  - `krb5-strength`: Kerberos v5 kadmind strength checking plugin
  - `krb5-sync`: Kerberos v5 kadmind account synchronization plugin
  - `kstart`: Daemon versions of kinit for Kerberos v4 and v5
  - `remctl`: Run specific commands remotely with Kerberos and ACLs
  - `runauth`: Run a command with Kerberos and AFS credentials
  - `S/Ident`: Kerberos callback authentication via ident
  - `wallet`: Secure credential management system
  - `WebAuth`: Kerberos web authentication system for Apache
...More Tools

• Jeffrey Hutzelman's Volume Dump Utilities
  http://grand.central.org/dl/software/dumpscan
  - afsdump\_scan: scanning and repairing volume dumps
  - afsdump\_dirlist: lists contents of an AFS directory file
  - afsdump\_extract: extracting contents of a volume dump into a local filesystem
  - Some more experimental tools
AFS and Batch Systems

- Various batch systems have interfaces to obtain and renew AFS tokens and/or Kerberos tickets
  - Mechanism for Sun Grid Engine: http://dvinfo.ifh.de/SGEwithAFS
  - Torque will get GSSAPI interface soon: http://www.clusterresources.com/pipermail/torqueusers/2008-February/006870.html
- Some general hints and remarks: http://www.kapet.de/kb/krb_afs_batch/
AFS-to-NFS Translator...

- Provides AFS access for machines without AFS support
- Can run on a single AFS client “translator” machine
- Runs as extension of the AFS client on that machine (as of now: Solaris only; maybe Linux support in OpenAFS >= 1.5)
- Needs special kernel module extension to pass along authentication (“knfs”)
- Linux support in 1.5 is kernel version specific due to necessary symbols being changed to GPL_ONLY
...AFS-to-NFS Translator...

- OpenAFS client on Solaris can re-export AFS as NFS:
  - Kernel module `libafs64.o` or `libafs.o` instead of `libafs64.nonfs.o` or `libafs.nonfs.o`
  - Additional `afsd` option required: `-rmtsys`
  - Entry in `/etc/dfs/dfstab`:
    ```
    share -F nfs -o rw /afs/slac.stanford.edu
    ```
  - In AFS startup script:
    ```
    /usr/afsws/bin/fs exportafs
    -type nfs
    -convert off
    -uidcheck off
    -submounts on
    -start on
    echo "shareall for afs translator"
    /usr/sbin/shareall
    ```
...AFS-to-NFS Translator

• You can use a user-space NFS server to re-export AFS as NFS

• Look at “User-space NFSv3 Server” on Sourceforge:
  http://unfs3.sourceforge.net/
  • Works on Linux and Solaris
AFS-to-SMB Translator

• Samba can re-export AFS space as a SMB share
• See Samba documentation (smb.conf(5))
• http://www.samba.org
• Useful: kimpersonate
  • Prints Kerberos tickets for a user given a keytab or AFS KeyFile
Web Access to AFS Space

- “[File Drawers] is a web-based file manager which talks to an AFS filesystem back-end”

- http://www.filedrawers.org/
- Also see Simon Wilkinson's talk from AFSBPW08:
  http://workshop.openafs.org/afsbpw08/wed_3_1.html
Filedrawers
Use of AFS PTS Names...

- Use AFS' PTS name entries with Name Service Switch (nss):
  - Todd Lewis'  
    http://tarna.oit.unc.edu/~utoddl/nss_pts_0.2.tgz
  - Frank Burkhardt's  
    ftp://instantafs.cbs.mpg.de/instantafs/debian/sarge/libnss-ptdb/
Use of AFS PTS Groups...

• If you don't have the AFS PTS groups also in (e.g.) LDAP
• Authorization to login to a PAM enabled Unix system via
  `pam_afs_groups`
  `http://chu.in-chemnitz.de/download/pam_afs_groups.tgz`
• Example configuration:

  ```bash
  auth required pam_afs_groups.so \
  groups chu:test,foo:bar
  ```
...Use of AFS PTS Groups

- Using AFS PTS groups in Apache2 `.htaccess` files via `mod_auth_afs_group`
  
  http://chu.in-chemnitz.de/download/mod_auth_pam_2.0.tgz

  - Example `.htaccess` for use with `mod_auth_kerb`:

  ```
  AuthType Kerberos
  AuthAuthoritative off
  AuthName "ACM.UIUC.EDU"
  KrbMethodNegotiate on
  KrbAuthRealms ACM.UIUC.EDU
  Krb5Keytab /etc/www.keytab
  AuthAFSGROUP_StripRealm on
  require afsgroup cclausen:self
  ```

- Example `.htaccess` for use with Basic Auth:

  ```
  SSLRequireSSL
  AuthType Basic
  AuthName AFS-Groups
  AuthAuthoritative off
  AuthPam_Enabled on
  AuthAFSGROUP_VaidateGroups off
  require afsgroup chu:test foo:bar
  ```
Multi-Homed AFS Clients

• To control which addresses the Cache Manager (CM) registers with File Servers:
  • /usr/vice/etc/{NetInfo, NetRestrict}
  • $prefix/etc/openafs/{NetInfo, NetRestrict}

• NetInfo specifies the IP addresses that the client should register with the AFS Servers it connects to
• NetRestrict specifies the IP addresses that the client should not register with the AFS Servers it connects to

• Display list of addresses CM is registering with File Servers: fs getclientaddrs
Multi-Homed AFS Servers...

- **NetInfo**
  - local interfaces the File Server (fileserv process) can register in the Volume Location Database (VLDB) at initialization time
  - DB servers: local interfaces the Ubik uses when communicating with the database server processes running on other database server machines

- **NetRestrict**
  - local interfaces the File Server (fileserv process) does not register in the Volume Location Database (VLDB) at initialization time
  - DB servers: local interfaces Ubik does not use when communicating with the database server processes running on other database server machines
...Multi-Homed AFS Servers

- Location of the two files on AFS servers:
  - /usr/afs/local/{NetInfo, NetRestrict}
  - $prefix/var/openafs/{NetInfo, NetRestrict}
Different Realm vs. Cell Name

• If your Kerberos realm name is different from your AFS cell name, put the Krb name in
/usr/afs/etc/krb.conf

• In OpenAFS-1.4.10, this will also support multiple different but equivalent realm names
Cross-Cell Authentication

- Cross-cell authentication is when one cell recognizes the AFS tokens of another cell and allows access to AFS file space.
- Set up cross-realm authentication in Kerberos first:
  - Both sides create Kerberos principals:
    - `krbtgt/CELL1.EDU@CELL2.EDU`
    - `krbtgt/CELL2.EDU@CELL1.EDU`
  - With the same key (not password!) and kvno.
- If you wish users from the other cell to be able to be authentic users in yours, create the PTS group system: `authuser@other.cell.name`
- If you want this, you need Kerberos 5 due to a security problem with this in Kerberos 4.
AFS Debugging Tools...

- **Rx protocol**: `rxdebug`, `kdump`
- **Cache Manager**: `cmdebug`, `kdump`
- **Ubik protocol**: `udebug`
- There used to be a very good documentation by IBM with a complete list of tools and explanations how to use them (Google for “IBM AFS debugging”)
- See also the AFS Administration Reference
- **All these tools work without authentication, i.e. everybody on the Internet can use them against your machines!**
...AFS Debugging Tools

• Known bug:
  If any of these tools runs or talks to a 64 bit Linux system, several statistic counters are always zero ("0")!
  AFS::Monitor Perl module suffers from the same problem because it uses the OpenAFS libraries.
• People are working on a fix.
• `rxdebug` command provides a trace of Rx activity for servers or client machines

• Simple example:
  
  ```
  % rxdebug afsclient 7001 -version
  Trying 134.79.24.169 (port 7001):
  AFS version: 1.4.4 built 2007-05-02
  ```
What vlserver is a client talking to:

% rxdebug afsclient 7001 -allcon -onlyport 7003

Trying 134.79.24.169 (port 7001):
Free packets: 130, packet reclaims: 4, calls: 4528, used FDs: 64
not waiting for packets.
o calls waiting for a thread
1 threads are idle
Showing only connections on port 7003
Connection from host 134.79.18.27, port 7003, Cuid 9332979e/c7792c
  serial 266, natMTU 1444, security index 0, client conn
    call 0: # 133, state not initialized
    call 1: # 2, state not initialized
    call 2: # 0, state not initialized
    call 3: # 0, state not initialized
Connection from host 128.2.13.199, port 7003, Cuid 9332979e/c77944
  serial 12, natMTU 1444, security index 0, client conn
    call 0: # 7, state not initialized
    call 1: # 0, state not initialized
    call 2: # 0, state not initialized
    call 3: # 0, state not initialized
Connection from host 128.2.13.199, port 7003, Cuid 9332979e/c77948
  serial 6, natMTU 1444, security index 0, client conn
    call 0: # 4, state not initialized
    call 1: # 0, state not initialized
...rxdebug

- List which clients have cached files from a server:

```bash
#!/bin/ksh -
# NAME        afsclients
# AUTHOR      Rainer Toebbicke <Rainer.Toebbicke@cern.ch>
# DATE        June 1994
# PURPOSE     Display AFS clients which have grabbed files from a server

if [ $# = 0 ]; then
    echo "Usage: $0 <afs_server 1> ... <afsserver n>"
    exit 1
fi
for n; do
    /usr/afsws/etc/rxdebug -servers $n -allconn
done | grep '^Connection' | while read x y z ipaddr rest; do echo $ipaddr;
done | sort -u |
while read ipaddr; do
    ipaddr=${ipaddr%%,}
    n="`host $ipaddr | cut -d' ' -f5`" # changed by A. Wachsmann
    n="${n##*Name: }
    n="${n%%Address:*}
    n="${n##*([ ])}"
    n="${n%?}"
    echo "$n ($ipaddr)"
done
```
cmdebug

- `cmdebug` is used to examine the status of the cache manager and cache entries on a particular AFS client machine
- This command is useful when you suspect that the cache manager is hung or is performing poorly
- Was used quite often in the old days of AFS cache corruption problems
kdump...

- **kdump** allows to gather detailed information about the internal structures of a cache manager.
- **cmdebug** only dumps stat and lock information while **kdump** dumps the state of all internal structures.
- **cmdebug** relies on having the cache manager (and the network) running while **kdump** does not.
- The **kdump** program is particularly useful when AFS is loaded via a dynamic loader.
- Cryptic output that needs a lot of processing before it reveals anything.
...kdump

- Easy to try: # /usr/afsws/etc/kdump -stats

Printing count references to cm-related functions..

[....]
1747546 Dread
  11517 FixupBucket
  11517 afs_newslot
1747215 DReLease
  21128 DFlush
    0 DFlushEntry
379858 DVOffset
  383 DZap
    0 DNew
355463 afs_RemoveVCB
  1335 afs_NewVCache
21128 afs_FlushActiveVcaches
[....]

Printing some cm struct performance stats..

  File Server up/downtimes, same cell: [...]
  File Server up/downtimes, diff cell: [...]
  VL Server up/downtimes, same cell: [...]
  VL Server up/downtimes, diff cell: [...]

Introduction to AFS and its Best Practices  © 2004-2009 by A. Wachsmann  182
afsmonitor

- **afsmonitor** is a UI showing the same information as `kdump -stat`
- Does not need root privileges and works remotely
- To monitor Cache Managers (AFS clients):
  
  \[
  \text{% afsmonitor -cmhosts afsclient1 afsclient2 ...}
  \]

  press `c` to see the statistics

  press `r` (right) to page through the statistics

  press `o` (overview) and `Q` (quit)

- To monitor File Server:
  
  \[
  \text{% afsmonitor -fshosts fileserver1 fileserver2 ...}
  \]

  press `f` to see the statistics
• Scout monitors the File Server process

```bash
% scout -server afs05 afs06
```

<table>
<thead>
<tr>
<th>Conn</th>
<th>Fetch</th>
<th>Store</th>
<th>Ws</th>
<th>Disk attn: &gt; 95% used</th>
</tr>
</thead>
<tbody>
<tr>
<td>8081</td>
<td>943934009</td>
<td>82244650</td>
<td>2401</td>
<td>afs05</td>
</tr>
<tr>
<td></td>
<td>a:21099389</td>
<td>b:10549484</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>c:1833989</td>
<td>d:14294476</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>e:45078456</td>
<td>f:19345850</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>g:62531139</td>
<td>h:34125002</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8812</td>
<td>*22770637</td>
<td>31391370</td>
<td>2140</td>
<td>afs06</td>
</tr>
<tr>
<td></td>
<td>a:10392121</td>
<td>b:7295946</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>c:8775881</td>
<td>d:32971594</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>e:19630754</td>
<td>f:33167000</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>g:31499205</td>
<td>h:62531131</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Probe 2 results

Conn: number of active connections  Ws: number of active clients (last 15min)
% udebug afsdb1 7003
Host's addresses are: 134.79.18.25
Host's 134.79.18.25 time is Fri Mar  5 08:52:42 2004
Local time is Fri Mar  5 08:52:42 2004 (time differential 0 secs)
Last yes vote for 134.79.18.25 was 11 secs ago (sync site);
Last vote started 11 secs ago (at Fri Mar  5 08:52:31 2004)
Local db version is 1076018780.5365
I am sync site until 49 secs from now (at Fri Mar  5 08:53:31 2004) (3 servers)
Recovery state 1f
Sync site's db version is 1076018780.5365
0 locked pages, 0 of them for write
Last time a new db version was labelled was:
  2486782 secs ago (at Thu Feb  5 14:06:20 2004)

Server (134.79.18.26): (db 1076018780.5365)
  last vote rcvd 11 secs ago (at Fri Mar  5 08:52:31 2004),
  last beacon sent 11 secs ago (at Fri Mar  5 08:52:31 2004), last vote was yes
dbcurrent=1, up=1 beaconSince=1

Server (134.79.18.27): (db 1076018780.5365)
  last vote rcvd 11 secs ago (at Fri Mar  5 08:52:31 2004),
  last beacon sent 11 secs ago (at Fri Mar  5 08:52:31 2004), last vote was yes
dbcurrent=1, up=1 beaconSince=1
% udebug afsdb2 7003
Host's addresses are: 134.79.18.26
Host's 134.79.18.26 time is Fri Mar 5 09:01:59 2004
Local time is Fri Mar 5 09:01:59 2004 (time differential 0 secs)
Last yes vote for 134.79.18.25 was 13 secs ago (sync site);
Last vote started 13 secs ago (at Fri Mar 5 09:01:46 2004)
Local db version is 1076018780.5365
I am not sync site
Lowest host 134.79.18.25 was set 13 secs ago
Sync host 134.79.18.25 was set 13 secs ago
Sync site's db version is 1076018780.5365
0 locked pages, 0 of them for write
tcpdump, wireshark

- Tools like tcpdump and wireshark know AFS

```bash
% tcpdump udp port 7001
...
14:29:07.084391 client.afs3-callback > server.afs3-fileserver:
    rx ack (65) (DF)
14:29:07.175243 client.afs3-callback > server.afs3-fileserver:
    rx data fs call fetch-status fid 536936023/162/1722 (44) (DF)
14:29:07.175779 server.afs3-fileserver > client.afs3-callback:
    rx data fs reply fetch-status (148) (DF)
14:29:07.196632 client.afs3-callback > server.afs3-fileserver:
    rx ack (65) (DF)
14:29:07.206665 client.afs3-callback > server.afs3-fileserver:
    rx data fs call fetch-status fid 536936023/194/647 (44) (DF)
14:29:07.207273 server.afs3-fileserver > client.afs3-callback:
    rx data fs reply fetch-status (148) (DF)
```
Salvager

- Salvager is usually run by bos to fix problems when AFS file server has crashed
- It can be run by hand without taking volumes offline
- Usages:
  - Find all volume mountpoints inside a volume:
    % salvager -partition /vicepa -volumeid sw.gcc -showmounts
  - Find all SUID/SGID files inside a volume:
    % salvager -partition /vicepa -volumeid sw.gcc -showsuid
  - Only tell but not do anything:
    % salvager -partition /vicepa -volumeid sw.gcc -nowrite
- Warning: If you run the salvager by hand and do not use one of -nowrite, -showmounts, or -showsuid, you must shut down the file server first. Otherwise you risk damaging volumes!
AFS Fileserver Internals

• If you need to debug or want to performance tune your OpenAFS fileserver, look at these two talks by Thomas Müller (TU Chemnitz):
  • OpenAFS: Debugging Methods and Tools  
    http://archiv.tu-chemnitz.de/pub/2002/0118/
  • OpenAFS Fileserver Debugging/Tuning  
    http://archiv.tu-chemnitz.de/pub/2003/0130/
  • Both are in German but there is not much real text on the slides anyway
Read-Write Disconnected AFS...

- Not quite ready for prime time but very close
- Problems:
  - If you reboot your machine for any reason whilst disconnected, you lose all of the changes made whilst in the disconnected state.
  - User interface is currently pretty primitive
- Reconnect may fail either due to conflicts on the server, or due to transient errors such as connection timeouts
  - If it does: retry
  - Server conflicts currently have to be repaired by hand.
  - If you're in a situation where you just can't fix the problem, and you're happy to discard the data, `fs discon online -force` will reconnect you, throwing away any stored changes.
...Read-Write Disconnected AFS

• Please provide bug reports to Simon Wilkinson <sxw at inf.ed.ac.uk>

• Before disconnecting, you must first prime your cache with all of the files you wish to use whilst disconnected with (e.g.) `find . | xargs cat > /dev/null`

• Take the cache manager disconnected by running as root
  % fs discon offline

• You may then read and write files in the primed set as you see fit.

• Go back online by running, as root, and with appropriate tokens:
  % fs discon online
AFS + Object Storage

- Project created in 2005
- Co-operation between
  - Rainer Többicke from CERN
  - Andrei Maslennikov, Ludovico Giammarino, Roberto Belloni from CASPUR, and
  - Hartmut Reuter from RZ Garching of MPG IPP
- Allows RW replication and striping of data
- Uses modified client and servers
- New behavior needs to be explicitly turned on on client
- `/afs/ipp-garching.mpg.de/.cs/openafs/openafs-1.4.10-osd`
- This work was supported and funded by ENEA, CASPUR, CERN and RZG
Questions?