# *Introduction to AFS*
# *IMSA Intersession 2003*

## Basic AFS User Commands

Brian Sebby, IMSA ‘96

# *Types of user commands*

- There are two types of commands you will use to interact with AFS on a regular basis: simple and compound commands.
- Simple commands are stand alone commands that are used primarily for the initial authentication to AFS. They are of the form <command> <arguments>.
- Compound commands make up the majority of commands you will use to interact with AFS. They are of the form <command> <subcommand> <arguments>.
- The compound commands make up the AFS command suites which are used to interact with the various AFS servers.
- Almost all commands have a –help switch to give the user more information about that command.

# *An aside about AFS commands*

- One feature of AFS commands is that a great many arguments are optional; when you view a command's –help options, any arguments you see in [] are optional. They will default to expected values, whether it be the current cell, current directory, etc.
- Another interesting feature of many AFS commands is that they have "shortcut" names that can be used. These can either be an alias (like la for listacl) or the shortest distinct string of letters to differentiate that command from the rest in the command suite.
- One final unusual feature is that if you provide AFS commands with the data they want in the correct order, giving the argument names is optional. If a command wants its data in the form –directory <directory> -id <id> –cell <cell>, you can simply type <directory> <id> <cell>.
- Many commands will have similar arguments, such as –directory, -cell, -id (the volume ID or volume name), etc. All commands have the –help argument to get usage information.

# *Simple commands - `klog`*

- `klog`
  The `klog` command is used to authenticate to the server and receive a token. Its full usage is:

```
% klog -help
Usage: 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] [-tmp] [-help]
```

- Generally, typing klog by itself will allow you to get a token, if your AFS UID is the same as your Unix UID. Otherwise, you simply need to type "`klog <username>`".

# *Simple commands - `tokens`*

- `tokens`
  The `tokens` command allows you to list what tokens you currently hold.
- If you have no tokens, you will see information like this:

```
% tokens
Tokens held by the Cache Manager:

--End of list—
```

- If you do have a token, however, you will see information like this:

```
% tokens
Tokens held by the Cache Manager:
User's (AFS ID 12345) tokens for afs@mycell.org
[Expires Jan  6 06:54]

--End of list--
```

# *Simple commands - `unlog`*

- `unlog`
  The `unlog` command is used to discard your tokens.
- You simply need to type `unlog` on the command line, and your tokens for your cell will be removed. If you wish to destroy tokens from another cell, use the –cell switch to unlog.

# *Simple commands - `kpasswd`*

- `kpasswd`
  The `kpasswd` command allows you to change your AFS password. Its full usage is:

```
% kpasswd -help
Usage: kpasswd [-x] [-principal <user name>]
[-password <user's password>] [-newpassword <user's
new password>] [-cell <cell name>] [-servers
<explicit list of servers>+] [-pipe] [-help]
```

- You do not need to give it the passwords on the command line, however, as it will prompt you for data that was not supplied:

```
% kpasswd
Changing password for 'bob' in cell 'mycell.org'.
Old password:
New password (RETURN to abort):
Retype new password:
Password changed.
```

# *A bit about PAGs*

- As stated previously, tokens can be associated with a UID or a Process Authentication Group, or PAG. A PAG is a structure that is wedged into the Unix groups structure.
- If you associate tokens with a UID, all of your connections that have the same UID can use that token. This could be dangerous if one connection is hijacked, however.
- If it is associated with a PAG, however, the token is limited to just that shell.
- You can get a token associated with a PAG two ways.
- One, you can specify the `-setpag` switch to `klog` when authenticating. That will set up the PAG in your current shell.
- Two, you can use the simple command `pagsh` to create a subshell that has the PAG set up. You can then klog as normal, but the token will be limited to the current shell.

# *Compound Commands*

- The compound commands will be the commands that are used the most often in dealing with AFS. Each compound command is used for interacting with a different AFS service, and is of the form <compound command> <subcommand> <arguments>
- The `fs` commands are used when dealing with the file server.
- The `vos` commands are used when dealing with volumes and the VLDB.
- The `pts` commands are used when dealing with the protection database.
- The `kas` commands are used when dealing with the authentication database.
- Other command suites include `bos`, for dealing with server control, `backup`, for dealing with AFS backups, and `uss`, for creating user accounts.

# *Compound commands - `fs`*

- The `fs` command suite is the most commonly used suite in AFS. To get a listing of all fs commands, type "`fs help`" on the command line. A (truncated) list of the most common subcommands is below:
- 

```
% fs help
fs: Commands are:
checkservers    check local cell's servers
checkvolumes    check volumeID/name mappings
examine         display volume status
help            get help on commands
listacl         list access control list
listquota       list volume quota
lsmount         list mount point
mkmount         make mount point
rmmount         remove mount point
setacl          set access control list
setquota        set volume quota
```

# *An aside about ACLs*

- Before showing how to set ACLs, it is important to know what ACLs are supported by AFS. AFS has seven ACLs:

```
l  lookup  List directory contents and examine the
           ACL.
r  read    Read contents of files in the directory.
i  insert  Insert new files or directories.
w  write   Modify contents of files and use chmod.
d  delete  Delete or move files from the directory.
k  lock    Run programs that set locks in the
           directory.
a  administer    Change the ACL.
```

- Remember that AFS ACLs are set on a per-directory basis.

# *The wrath of* `chmod`

- Even though AFS uses its own ACL scheme for access control, that doesn't stop `chmod` from still affecting files.
- AFS looks at the owner bits of a file to see if it should be allowed to be read, written, or executed.
- Since most files are set to give the owner full access rights, this usually doesn't affect AFS operation, but occasionally you'll get a "permission denied" when you try to access a file due to the owner bits of the Unix protection scheme being set incorrectly. This occasionally causes problems when executables need to have the "x" privilege set.
- However, if you have w (write) ACL privileges for the directory, you can use `chmod` and change the file to whatever setting you need.

# *Another aside - system groups*

- We will discuss groups later, but there are a few predefined AFS groups that should be mentioned.
- The group `system:administrators` is the group of AFS accounts that have administrative (root) access to AFS. Most directories give them full access, but that is mostly irrelevant as system:administrators always have a (change ACL) access on all directories and can change the ACLs as they like.
- The group `system:anyuser` is the AFS "anonymous" account. This can be used to grant access rights to any user of AFS, authenticated or not. It is often used to give read access to public data.
- The group `system:authuser` is similar to the anyuser group, but it only applies if a user is authenticated to your cell. This can be used to restrict data to access from your cell, and not anonymously or from foreign cells.

# *Compound commands – `fs la`*

- The fs listacl (or the short form, fs la) is the AFS equivalent of ‘ls’. It lists the ACLs for a directory:

```
% fs la .
Access list for . Is
Normal rights:
  system:administrators rlidwka
  system:anyuser rl
  bob rlidwka
```

- In this directory, the user ‘bob’ has full access, while the groups system:administrators and system:anyuser have full and lookup rights, respectively.

# *Compound commands – `fs sa`*

- The `fs setacl` (or the short form, fs sa) is the AFS equivalent of `chmod`. It modifies or sets the ACLs for a directory. Its full usage is:

```
% fs sa -help
Usage: fs sa -dir <directory>+ -acl <access
list entries>+ [-clear] [-negative] [-id] [-if]
[-help]
```

- The general way of using `fs sa` is the following: `fs sa <directory> <user> <ACL>`. As an example, to give the user "bob" the rl (read and lookup) ACLs, you would type: `fs sa . bob rl`
- There are a few ACL shortcuts: "`read`" is defined as "`rl`", "`all`" is defined as "`rlidwka`", and "`none`" is defined as "".

# *Compound commands – more* `fs`

- The `fs examine` (or `fs exa` for short) command gives information about the volume that the directory specified (by default the current directory) is in. This is useful to find the volume name that a file or directory is in.
- The `fs listquota` (or `fs lq` for short) command lists the quota for the directory given.
- The `fs setquota` (or `fs sq` for short) command sets the quota for the entire volume based on the directory given.
- The `fs checkservers` command checks the status of the AFS servers.
- The `fs checkvolumes` command checks all volume mappings currently being accessed by the cache manager, and will update them as needed. This can be useful if a volume is moved while it is being used.

# *Compound commands – even more `fs`*

- The `fs mkmount` (or `fs mkm` for short) command creates a new mount point to a volume in the current directory. To mount the volume user.bob to look like the directory "bob" in the current directory, you would type:
  `fs mkm bob user.bob`
- The `fs rmmount` (or `fs rmm` for short) command deletes a mount point to a volume.
- The `fs lsmount` (or `fs lsm`) command lists information about a given mount point.
- There are a great number of other `fs` commands, but these are the most commonly used. Please refer to the documentation or use the –help switch to learn more.

# *A bit about file servers*

- We will talk about AFS servers in more detail later, but there are a few things you need to know before using the `vos` command suite.
- AFS files are stored on the file servers in “vice” partitions, which are mounted locally as /vicepa, /vicepb, … /vicepz, etc.
- Many `vos` commands will need to know what partition a volume is on to perform requested operations.
- The /vicep<x> partitions can be referred to as a, b, c, … z, etc. for most commands.
- For the next few slides, assume we have file servers machine1, machine2, and machine3, all of which have /vicepa, /vicepb, and /vicepc. This means we can refer to their partitions as a, b, and c.

# *Compound commands - vos*

- The vos command suite is used to perform commands on volumes in AFS. To get a listing of all vos commands, type "vos help" on the command line. A (truncated) list of the most common subcommands is below:
- 

```
% vos help
vos: Commands are:
addsite          add a replication site
backup           make backup of a volume
create           create a new volume
delentry         delete VLDB entry for a volume
examine          everything about the volume
help             get help on commands
listvldb         list volumes in the VLDB
move             move a volume
partinfo         list partition information
release          release a volume
remove           delete a volume
remsite          remove a replication site
zap              delete the volume, don't bother with
                 VLDB
```

# *Compound commands – `vos create`*

- The command `vos create` is used to create a new AFS volume. Its usage is:

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

- To create a new volume "user.bob" on partition b of machine2 with a quota of 5000 KB, we would issue the command (as mentioned before, argument names are optional):

```
vos create machine2 b user.bob 5000
```

- You could then use the `fs mkm` command to mount this volume in your file space.
- This new volume would initially be a read-write volume until it is replicated.

# *Compound commands – `vos addsite`*

- The command `vos addsite` is used to create a read-only replica of a volume. Its usage is:

```
% vos addsite -help
Usage: vos addsite -server <machine name for new
site> -partition <partition name for new site> -id
<volume name or ID> [-cell <cell name>] [-noauth] [-
localauth] [-verbose] [-help]
```

- If you create a replica on the same partition of the read-write volume, it only copies things that are different than the read-write volume when creating the new volume. It is in effect a "free" copy.
- A read-only replica on a separate partition is a complete copy of the volume; you will need "x" times the size of the original volume's size in disk space to store "x" copies.
- To make a replica of our "user.bob" volume on partition c of machine3, we would use the command:

```
vos addsite machine3 c user.bob
```

- Once you have added a replica, you will need to go through read-write mount points to modify the data in that volume.

# *Compound commands – `vos release`*

- The vos release command is what is used to update the read-only replicas of a volume to reflect changes made to the read-write volume. Its usage is:

```
% vos release -help
Usage: vos release -id <volume name or ID> [-f]
[-cell <cell name>] [-noauth] [-localauth] [-
verbose] [-help]
```

- If you do not specify the –f (full) flag, it will only update things that have changed since the last release; if you include the –f flag it will force a complete recloning of the volume.
- If we changed our read-write "user.bob" partition and wanted to update its replicas, we would type: `vos release user.bob`

# *Compound commands – `vos examine`*

- The `vos examine` (or `vos exa` for short) command will tell you information about the current volume. This is often used in conjunction with the `fs examine` command. For instance, if we were in /user/bob (where bob is the mount point for user.bob), we could do an `fs exa` to get that information:

```
% fs exa bob
Volume status for vid = 1818560247 named
user.bob.readonly
Current disk quota is 2000000
Current blocks used are 1766178
The partition has 1678667 blocks available out
of 35249332
```

- Note that the read-only volume can be specified as user.bob.readonly.
- (Example continued on next slide.)

# *Compound commands – `vos examine` continued*

- Then we can use vos exa to get information about the volume:

```
% vos examine user.bob
user.bob                  1818560246 RW    1766178 K  On-line
    machine2.imsa.edu /vicepb
    RWrite 1818560246 ROnly 1818560247 Backup 1818560248
    MaxQuota    2000000 K
    Creation    Tue Jul 28 17:01:44 1998
    Last Update Thu Dec 12 19:50:35 2002
    0 accesses in the past day (i.e., vnode references)

    RWrite: 1818560246    ROnly: 1818560247    Backup:
1818560248
    number of sites -> 2
       server machine2.imsa.edu partition /vicepb RW Site
       server machine3.imsa.edu partition /vicepc RO Site
```

- This gives you almost all the information you would need to know to work with this volume, including the location of the read-write and read-only copies. Note that the numbers after “RWrite”, “ROnly”, and “Backup” above are the actual volume IDs for the user.bob volume that AFS uses internally. All read-only replicas share the same ROnly volume ID.

# *Compound commands – `vos move`*

- The `vos move` command is used to move a volume from one server and partition to another. Its usage is:

```
% vos move -help
Usage: vos move -id <volume name or ID> -fromserver
<machine name on source> -frompartition <partition
name on source> -toserver <machine name on
destination> -topartition <partition name on
destination> [-cell <cell name>] [-noauth] [-
localauth] [-verbose] [-help]
```

- To move our "user.bob" volume from machine2, partition b to machine1, partition a, we would type:

```
vos move user.bob machine2 b machine1 a
```

- Moving volumes is often done to balance the load on disks or to free up disk space on an overused partition.
- Moving volumes is completely transparent to users. Their volumes can move to completely different server machines while they are using them, and they should not notice the change.

# *Compound commands – more `vos` commands*

- The `vos backup` command is used to create a backup volume for the volume specified. The backup volume is a static snapshot of the volume that is commonly generated daily and used when backing up the AFS file space.
- Backup volumes are created on the same partition as the read-write volumes and only contain the original copies anything that's changed on the volume since the backup command was issued – it merely has a link to files that have not changed.
- The `vos remove` command is used to remove a volume from AFS. See its –help flag for usage information.
- The `vos remsite` command is used to remove a replication site from AFS. See its –help flag for usage information.

# *Compound commands – even more `vos` commands*

- The `vos listpart` command lists information about partitions on a host. This information includes the amount of free space on each partition to allow administrators to see where space may be an issue.
- The `vos listvldb` command lists all volumes in the volume database for a specified server or partition
- The `vos delentry` and `vos zap` commands are "oops" commands used when a vos remove or other action only partially removes a volume. Vos delentry removes the VLDB entry for a volume, and vos zap removes the volume on disk (you must have the physical ID) without touching the VLDB.
- There are several more obscure `vos` commands available; check `vos help` for more information.

# *A few words about groups*

- For AFS users, the primary reason for using the pts command suite has to do with the use of AFS groups.
- Each user can create a number of groups (20 by default) of the form <username>:<groupname> that you can assign members to, and then use in setting ACLs.
- A group will have a unique negative UID associated with it (while normal users have positive UIDs.
- Members of AFS groups can either be AFS UIDs or IP addresses of specific machines.
- A group can be listed as a member of another group.
- By using IP addresses, you can limit ACLs to use on certain machines only.
- In the following examples, let's have our user "bob" create the AFS group "newgroup". This would then have an AFS group name of "bob:newgroup".

# *Compound commands - `pts`*

- The following pts commands are the main ones you will use with groups:

```
% pts help
pts: Commands are:
adduser          add a user to a group
chown            change ownership of a group
creategroup      create a new group
delete           delete a user or group from
database
examine          examine an entry
help             get help on commands
membership       list membership of a user or
group
removeuser       remove a user from a group
rename           rename user or group
```

# *Compound commands – more* `pts`

- Most of the `pts` commands are fairly self explanatory. Please look at their help messages for further information. The following example should show you how you can use groups.
- Create a new group:

```
% pts createg bob:newgroup
group bob:newgroup has id -344
```

- Add a user to the group:

```
% pts adduser bob bob:newgroup
```

- Examine the group:

```
% pts exa bob:newgroup
Name: bob:newgroup, id: -344, owner: bob,
creator: bob,
  membership: 1, flags: S-M--, group quota: 0.
```

- Example continued on next slide.

# *Compound commands – even more* `pts`

- Examine the membership of the group:

```
% pts mem bob:newgroup
Members of bob:newgroup (id: -344) are:
  bob
```

- Assign an ACL using the group:

```
% fs sa . bob:newgroup all
```

- View the ACL:

```
% fs la .
Access list for . Is
Normal rights:
  bob:newgroup rlidwka
  system:administrators rlidwka
  system:anyuser rl
  bob rlidwka
```

- Remove the group:

```
% pts rem bob bob:newgroup
```

# *The other command suites*

- We will discuss the other command suites, which include `bos`, `kas`, `backup`, and `uss` later.
- These commands are primarily used in administering the AFS cell.
- In addition, the `fs`, `vos`, and `pts` command suites have additional commands for administration that will be discussed later.

# *Conclusion*

- The commands listed in these slides are the main ones you will use when dealing with AFS. In addition, there are less commonly used ones that you may also find useful from time to time.
- As the AFS file space is intended to look like a standard Unix file system to users, regular commands like ls, rm, and cd will still be used to navigate and handle files in AFS space.
- For more details on using AFS, please read the *AFS User's Guide* in the OpenAFS documentation collection at: http://www.openafs.org/doc/index.htm