

Using PTS Groups and Setting Directory Protections in Andrew An Overview of AFS

This document contains the following sections:

- [Using PTS Groups](#)
- [Protection Rights](#)
- [Protection Groups](#)
- [File Protection Using CHMOD](#)

For information related to this topic refer to:

- [Andrew Files and Directories](#)
(<http://www.cmu.edu/computing/doc/unix/afs-files.html>)
- [Andrew Linux Support Statement](#)
(<http://www.cmu.edu/computing/doc/os/linux.html>)
- [Andrew Solaris Support Statement](#)
(<http://www.cmu.edu/computing/doc/os/solaris.html>)
- [Increasing and Managing Your AFS Quota](#)
(<http://www.cmu.edu/computing/doc/accounts/quota/index.html>)
- [Logging into and out of AFS](#)
(<http://www.cmu.edu/computing/doc/unix/afs-login.html>)
- [Project Volumes](#)
(<http://www.cmu.edu/computing/doc/accounts/project-volume/index.html>)
- [Using Unix and the Andrew System](#)
(<http://www.cmu.edu/computing/doc/unix/unix-andrew/index.html>)
- [Your Andrew Unix Account](#)
(<http://www.cmu.edu/computing/doc/unix/andrew-acct/index.html>)

Last Updated: 6/27/05

Using pts Groups

PTS groups provide a way to grant a list of people access to your files and directories on Andrew. Once a pts group has been created, you can grant it access to an Andrew directory just like you would an individual userid.

The **pts** program lets you create, view, and maintain PTS groups. To view a full list of pts options, type the following at the Andrew Unix prompt:

```
pts -help
```

Note: You can only create, delete, view, and maintain the pts groups that you own, or pts groups that you are a member of that are self-owned (e.g., the pts group owns the group so any group member can maintain the pts group).

Creating pts Groups

To create a pts group, enter the following (where *userid* is your Andrew userid and *name* is the unique name for this group).

```
pts createg userid: name
```

For example:

```
pts createg ju32:mygroup
```

creates the group ju32:mygroup.

Viewing Group Membership

To view the list of group members, enter the following:

```
pts members groupname
```

For example:

```
pts members ju32:mygroup
```

To view the list of groups that you are a member of, type the following (where *userid* is your Andrew userid).

```
pts members userid
```

For example:

```
pts members ju32
```

Adding and Removing Group Members

To add someone to a PTS group, type:

```
pts add userid groupname
```

For example:

```
pts add jd4 ju32:mygroup
```

adds the user jd4 to the group ju32:mygroup.

To remove a person from a PTS group, type:

```
pts remove userid groupname
```

For example:

```
pts remove jd4 ju32:mygroup
```

removes the user jd4 from the group ju32:mygroup.

Last Update: 6/27/05

Protection Rights

File and directory protection provides the ability to determine who can access your files and directories and what operations these users can perform on them. These protection settings are called privileges or "rights" that you can give or take away from users who join a particular group.

To view the Andrew directory protections on a directory, type the following (where directory is the directory you want to view).

```
fs la directoryname
```

For example:

```
fs la .fs la ~ju32fs la /afs/andrew/org/salsa
```

To change the privileges or rights to directories for a user, type:

```
fs sa directorynameuserIDrights
```

For example:

```
fs sa /afs/andrew/org/salsa ju32 rights
```

To remove a user or group's access to a directory, type:

```
fs sa directoryuserid_OR_ptsgroup none
```

For example:

```
fs sa /afs/andrew/org/salsa ju32:mygroup none
fs sa /afs/andrew/org/salsa jd4 none
```

To give back rights that have been taken away:

Re-setting

```
fs sa directoryname userIDrights
```

With -negative:

```
fs sa directoryname -negative userID none<
```

To give or take away rights to a group, substitute **groupname** with **userID**

Do my directories automatically come with protections?

When your account was created, several directories were placed in it automatically. Each of these directories come with default protections. You can add to these default protections, but you should not take away any of them. If you do, you may prevent some basic system functions (like mail or printing) from working for you.

By default, any subdirectories you create have the same protections as their parent directories. If you change the parent directory's protection after creating subdirectories, though, the change will not affect the subdirectories.

Who can work with my directories?

You can set rights for individual users or groups of users. The access control list (ACL) for a directory tells you which users or groups have rights for that particular directory. See the section "Access control lists: seeing the protections on a directory."

Types of Rights

There are two different kinds of rights, positive and negative.

- Positive rights allow other users or groups special access to a directory.
- Negative rights prevent access for users who would otherwise have access.

Following is a list of rights you can set for other users or groups. Each of these rights can be positive (allowing the action) or negative (denying the action):

- Read (r): read any file in the directory.
- Lookup (l): list all files in the directory and obtain status information about the files.
- Insert (i): add new files to the directory.
- Delete (d): remove files.
- Write (w): create or edit files in the directory.
- Lock (k): place write locks on any file in the directory. This is used mainly by application programs.
- Administer (a): modify the access list and ownership of a directory. The owner of the directory always has Administer rights, even if he or she appears to have no rights to the directory. Therefore, the owner of a directory can always reset the protections.

Commonly Used Aliases of Rights

To keep you from having to remember exactly what codes to use, you can combine a group of user rights into an alias (see the aliases help file):

- read (rl): allows users to lookup and read any files in the directory.
- write (rlidwk): allows users all the read rights above, as well as the right to add, change, and delete files in the directory.
- all (rlidwka): allows users all rights to the directory, including Administer. You should be careful about assigning another user all rights to one of your directories.
- none: allows users no rights to the directory (users will not be listed at all on the access control list for the directory). Note that users given no rights may still have access to the directory if they are members of a protection group that has access to the directory. See the section "Removing rights from directories."

See the following sections for examples.

Giving Rights to Directories

There may be times when you want to give certain users rights to your subdirectories. To do this, use the fs command with the sa (setting access) parameter. At the system prompt, type:

```
fs sa directorynameuserIDrights
```

Directoryname is the pathname of the directory to which you are setting rights, userID is the user ID of the person to whom you are giving rights, and rights are the abbreviations or codes for the rights you wish to set.

For example, if you wanted to give someone whose userID is "pat" read and lookup access to a directory of yours called notes (remember that read can be used to indicate both read and lookup "rl" rights), at the system prompt you would type:

```
fs sa ~/notes pat read
```

or

```
fs sa ~/notes pat rl
```

To verify that the correct rights were added, use the **fs la** *directoryname* command explained earlier.

Giving Rights to Users from Other Cells

It is possible to add users from other cells to an access control list, provided they have obtained a cross-realm token for your cell and have had this token registered in the protection database for your cell. (See the cklog help file for further information on cross-realm tokens and authentication.)

To add a user from another cell, use the **fs sa** command and supply the cross-realm identity for this person as the userID (to add a protection group you would use the pts command). A cross-realm identity is in the form: <userID>@<cellname> So the fs sa command would look like:

```
fs sa directoryname <userID>@<cellname> rights
```

For example, to add Harry Bovik from Computer Science (userID of bovik@cs.cmu.edu) to an access control list of yours in the Andrew cell, you would add his cross-realm identity to your ACL.

Harry's cross-realm identifier would be bovik@andrew.cmu.edu (note the difference between this and his CS user ID); this is what you would add to your access control list.

```
fs sa directoryname bovik@andrew.cmu.edu rl
```

The individual you wish to add to your access control list must have obtained the cross-realm token and registered it with the protection database for your cell; you cannot do this for the individual.

Removing Rights from Directories

There are two possible ways to take away rights to a directory. One way is to use none with the fs sa command and the other is to use -negative with the fs sa command.

The main difference between none and -negative is that none merely erases a userID or group name from the access control list, while -negative adds the name to the list with a special kind of permission, negative rights.

If you remove a user from the ACL of one of your directories using none, but he or she is a member of another group that has rights, the user will still have rights to the files in that directory. However, if you use -negative to deny rights, he or she will appear on

the ACL as having negative rights. Then, even though the user is still a member of the other group, he or she will be denied access to your directory because of those negative rights.

Using None

To take away all rights to a particular directory (to set none rights) you place `none` at the end of an `fs sa` command line:

```
fs sa directoryname userID none
```

Where *directoryname* is the name of the directory you are denying access to and *userID* is the userID of the person to whom you are denying access.

For example, if you had previously given user pat read rights ("rl" rights) to your notes directory, but decided now that you don't want pat to have any rights, at the system prompt you would type:

```
fs sa ~/notes pat none
```

Remember, pat will still have the same access that system:anyuser has, because pat is a member of that group.

Setting negative rights, or using -negative

When you use `-negative` with the `fs sa` command, you are setting negative rights or denying a user specific access to a directory. To assign negative rights, at the system prompt type:

```
fs sa directoryname -negative userID rights
```

Where *directoryname* is the pathname of the directory you are setting negative rights for, *userID* is the user ID of the person to whom you are denying rights, and *rights* are the abbreviations for the rights you are taking away.

To give pat negative read and lookup rights (to take away those rights) to your notes directory, use the word read for "rl"; at the system prompt type:

```
fs sa ~/notes -negative pat read
```

If you use the `fs la ~/notes` command, you will see that a list of negative rights has been added to the rights list for notes:

Normal rights:

```
system:anyuser rl<userID> rlidwka
```

Negative rights:

```
pat rl
```

Giving Back Rights you have Taken Away

There are two ways to give back rights that you have taken away. Which you use depends on how you removed the rights initially.

- If you used **none** to take away a user's rights, use the same command described earlier for setting rights:

fs sa *directoryname* *userID* *right*

- If you used **-negative** to take away a user's rights, then you must use **-negative** combined with **none** to remove those negative rights:

fs sa *directoryname* -negative *userID* none

Here, you are telling the system that the user has no negative rights.

Last Updated: 6/27/05

Protection groups

A protection group is a collection of users or groups who have the same access rights to a directory. You may belong to a group created by another user or you may create your own groups of users for certain directories (see the protection-groups help file for details).

There are two major protection groups in the AFS distributed file system:

- system:anyuser - anyone using the system.
- system:authuser - a user with a valid token. (For more information, see the How Authentication Works section below.)

How Authentication Works

Membership for AFS groups is decided on the basis of authentication (except system:anyuser, where anyone using the system has access). When you log in, a cache manager checks your password to verify that you are a valid AFS user. You then receive a set of tokens allowing you access to AFS files and directories (you are accepted into the group system:authuser).

Access Control Lists: Seeing the Protections on a Directory

The `fs la` command shows you the protections for a directory. At the system prompt, type:

```
fs la
directoryname
```

This command (or the **fs listacl** command) shows you an access control list (ACL), a list of users and protection groups who have access rights to a specified directory.

You can see the protections on your home directory by typing the `fs la` command followed by the tilde (~) as an abbreviation for your home directory pathname:

```
fs la ~
```

If you are already in your home directory, you do not need to include the tilde; without a directory pathname, AFS will default to the directory you are currently in. See the section on default protections for examples of access control lists for several different directories.

The `fs` (file system interface) command does not work by itself. It must be followed by a parameter, like `la`, to make it perform a specific action. See the `fs` help file for more command options.

Default Protections

When your Andrew account is created, it comes with several default directories including your home directory, private, or public. Default protections are set for each of these directories.

The following is a list of the commands you can use to see the default protections on each of your default directories, and what the protections should look like:

Home Directory

```
% fs la ~  
Normal rights:  
system:anyuser l  
<your user ID> rlidwka
```

This indicates that any user on the Andrew system can lookup any file in your home directory but cannot read any files. Therefore, be careful about the type of material you keep in your home directory. You, as the owner, have all possible rights.

Private

```
% fs la ~/private  
Normal rights:  
<your user ID> rlidwka
```

This indicates that you have exclusive rights to this directory. No other user can see files or subdirectories listed in this directory.

Public

```
% fs la ~/public  
Normal rights:  
system:anyuser rl  
<your user ID> rlidwka
```

This gives all Andrew users read and lookup rights to files in your public directory. You have all rights.

Last Updated: 6/27/05

File Protection Using CHMOD

Files in the AFS distributed file system use access control lists for the directories they are in, and most protections are set on directories using the `fs sa` command (or `pts` command for groups). Files on a local disk, however, do not have access control lists, so the `chmod` (change file mode) command must be used to set access to an individual file.

`Chmod` allows you to control whether a file can be read or written at all. If you turn off writing for a file, you will prevent anyone, including yourself, from writing to that file; the same goes for reading a file. If you turn on writing for a file, you allow anyone write access to that file if they already have write access to the file's directory; the same goes for read and execute access. See the `chmod` help file for more information.

To turn off write access for a file

```
chmod -w filename
```

To turn off read access for a file

```
chmod -r filename
```

To turn off execute access for scripts and executable files

```
chmod -x filename
```

To turn on write access for a file

```
chmod +w filename
```

To turn on read access for a file

```
chmod +r filename
```

To turn on execute access for scripts and executable files

```
chmod +x filename
```

Most protections on Andrew are set on directories. Therefore, even if you change the protections on an individual file using the `chmod` command, other users may still have access rights (or regain access rights) to that file based on their access rights to the directory in which the file resides.

Last Updated: 6/27/05