Working with Directories

Working with Directories

Every file on Athena (and elsewhere) is located in a directory. Directories keep files organized; through directories, you can find your files, as well as other files you are interested in. This section explains directories on Athena, how to create and remove directories, how to change from one directory to another, and other commands relevant to directories.

Directories, Pathnames, and the File System

The *file system* is a set of many files organized into a tree of directories and subdirectories. Every directory and every file has a pathname that specifies its location within the file system tree. Everybody's files are somewhere in this tree, occupying some sub-branch: the system's files, your files, other users' files, and the online documentation files all are stored in this tree.

The path to a file or directory is usually listed from top-most directory down, with intermediate directories separated by slashes. For instance, when you log into an Athena workstation, the system puts you into your *home directory.* (Your home directory, and all the subdirectories beneath it, constitute your *locker*.) Your home directory is located in the overall directory tree as:

> */afs/athena.mit.edu/user/* first-letter/ second-letter/ username

Here, *username* is the name you enter when you login, *first-letter* is the first letter of your username, and *second-letter* is the second letter of your username. For instance, if your username were **jruser**, your home directory would have the following pathname:

> */afs/athena.mit.edu/user/j/r/jruser*

This means that someone starting at the root of the directory tree would have to go down into the directory tree through the following directories to get to your home directory:

- the *root* directory (indicated as the first slash /);
- the directory **afs**, which has subdirectories for many different AFS sites ("cells") around the world;
- the directory **athena.mit.edu**, which holds subdirectories at MIT related to Athena (for example, courseware and development lockers are held in subdirectories of athena.mit.edu);
- the directory **user**, which holds all the directories for individual user accounts on Athena;
- the directory **j**, which holds all the directories belonging to users whose usernames start with the letter "j";
- the directory **r**, which holds all the directories belonging to users whose usernames start with the letter "j" and whose second letter is "r" (i.e., only usernames that start "jr");
- the directory **jruser**, which is the user's home directory itself (note that the name of your home directory is your username).

You can find out the full pathname of your own home directory by typing **printenv HOME** at your joeuser@athena:~$ prompt.

Because the full pathname is rather long, Athena lets you specify your home directory in an alternative shorthand as follows:

> */mit/*username

The home directory is still in its original location, but there is a *link* in the /mit directory that lets you get to the home directory through the shorter path.

The Working Directory

To identify a file, the system needs to know its exact location (i.e., its given name preceded by the path of directories one must follow to find it). Nevertheless, you do not have to give a file's full name every time you specify it because the system keeps a value for you called the *working directory* (also known as a "default directory" or "the current directory"). A working directory is, simply, the directory you are currently "in".

From the time you log in to the time you log out, you are in a current working directory. You start out in your home directory (e.g., /afs/athena.mit.edu/user/j/r/jruser, also known as /mit/jruser), because that is where the login process puts you when it lets you use the workstation.

Whenever you specify a simple filename, the system assumes that you are talking about a file in your working directory, and so can locate the file. You can change your working directory at any time. Each of the windows on your screen has its own current working directory.

The system provides a few short-hand synonyms relative to the working directory, which you can use in file and directory commands:
. the current working directory
.. the directory above the current working directory
~ your home directory

Finding Out Where You Are (pwd)

To find out the pathname of your current working directory, use the **pwd** (print working directory) command at your joeuser@athena:~$ prompt.

If you use the command immediately after logging in, the transaction proceeds as follows:

```
joeuser@athena:~$ pwd
/afs/athena.mit.edu/user/ first-letter/ second-letter/ username
```

where *username* is your username, *first-letter* is the first letter of your username, and *second-letter* is the second letter of your username.

As you start hopping around the tree with **cd** commands (see Changing the Working Directory), it is easy to forget where you are. You can always find out your current working directory with **pwd**.

Note that the results might seem a little confusing if you go to a directory via a link pathname rather than its full pathname. For example, if you **attach** the sipb locker, it creates a link in /mit such that you can refer to the directory as /mit/sipb – however, this pathname is just a convenient alias, not the actual path; the sipb locker is still actually located in the /afs branch of the file system tree. The **pwd** command returns the real pathname, not the link pathname, with results that might seem a little counterintuitive until you understand that links are not real paths.

Creating a Directory (mkdir)

Use the **mkdir** command to create new directories.

For example, suppose you are in your home directory and you want to create a series of directories in which to store your programs. You want a directory called Programs in your home directory. To do this, type:

```
joeuser@athena:~$ mkdir Programs
```

in the directory where you want the Programs directory to be (here, your home directory).

Note that you must have appropriate access permission to create new directories under an existing directory; by default, you have this permission in every subdirectory of your home directory, but you may not have this permission in most other locations of the file system tree. Also, even for the directories you create in your own home directory, you will want to make sure the access permissions are set appropriately (e.g., you may not want any other users to be able to list the names of the files in your new subdirectory). For information about how to check and set access permissions, see the document AFS on Athena.

Changing the Working Directory

You will often want to temporarily change your working directory from your home directory to somewhere else on the tree. Use the **cd** (for "change directory") command.

For example, suppose your username is jruser and your current working directory is your home directory. You want to modify some of the files in the subdirectory of your home directory called Private – you want to "work in" that directory, as the saying goes. You could specify the files of interest by their full pathnames (e.g., /afs/athena.mit.edu/user/j/r/jruser/Private/resume.tex) or you could specify the files by their somewhat simpler but still tedious "relative pathnames" (i.e., pathnames relative to the current working directory, such as Private/resume.tex). However, if you intend to do any serious work in that directory, you probably want to change your working directory from your home directory to Private, then refer to the files by their local names (e.g., resume.tex). To do this, type:

```
joeuser@athena:~$ cd Private
```

If you are in a directory other than your home directory, you can use **cd** without an argument to change the working directory back to your home directory. Thus the **cd /mit/jruser** command at the end of the previous example could be shortened to just **cd**:

```
joeuser@athena:~$ cd
joeuser@athena:~$ pwd /afs/athena.mit.edu/user/j/r/jruser
```

You may be working in somebody else's directory, somewhere else on the tree. Rather than always typing out /mit/ *otheruser*, you can just change the working directory to their home directory after attaching their locker:

```
joeuser@athena:~$ cd /mit/otheruser
```

There are several potential oddities to note as you use **cd**. Any of these of these conditions can produce the illusion that something is seriously wrong with your files or your login session, but in fact some simple explanation lies behind the difficulty:

- While each user's home directory is always accessible via its full pathname (i.e., under the /afs/athena.mit.edu/user branch of the file tree system), home directories are available under /mit *only* if they have had a link created under /mit by the **attach** or **add** command – they are not available there by default. This is also true of other lockers you may wish to access. In other words, in the last example above, you would need to **attach** the other user's directory first before you would be allowed to **cd** to it under the /mit directory.
- Even if a directory you want to change to exists (whether under /mit or elsewhere), you may not be allowed to **cd** to it because you have insufficient access permission to look at that directory.

Removing a Directory (rmdir)

If you want to remove a directory, you will want to use the rmdir command. (You could also use the **delete** command; see Removing a File.)

You cannot remove a directory unless all of the files underneath it are gone. This prevents you from accidentally wiping out important subtrees with one careless command. A quick way to delete all the files in a directory you want to get rid of is to change to that directory with **cd**, then:

```
joeuser@athena:~$ delete *
```

Be careful before you do this! If the directory also has . dot files in it, you must *also* say:

```
joeuser@athena:~$ delete .[^.]*
```

Now you can check what you've deleted with **lsdel**, then go ahead and **expunge** if you're certain about the files you're deleting. You can then delete the directory itself. Move out of the directory back to the one above it, then type:

```
joeuser@athena:~$ rmdir dirname
```

To prevent accidentally erasing files, the **rmdir** command only removes empty directories. If a directory isn't empty, **rmdir** displays an error message. You must then **cd** to that directory and remove all of its files and subdirectories.

Sample File/Directory Specifications

It takes a little practice to get the knack of correctly entering filenames and directories so that you get the files you want. You could specify all files by their complete pathnames, but that is awfully tedious; also, there are some cases where full pathnames are not what you want (e.g., if you want to enter the same command at different points of the tree, you will want to be able to use relative references to files).

The following table lists some of the common ways to specify directories or files. (All of these have synonyms that would work just as well.) These specifications might be used, for example, in a command of the form **cd *specification***. Use the following specifications to indicate:

- **.**(one period): the current working directory
- **..** (two periods): the parent directory of the current working directory (i.e., the directory "above" the current working directory)
- **~** (a tilde): your home directory
- **foo**: a directory called **foo** inside your current directory
- **foo/:** all the files and directories in directory **foo** below your current directory
- **foo/stuff** : a file or directory **stuff** in the directory **foo** inside your current directory
- **stuff**: a file or directory **stuff** in your home directory, when you are in your home directory
- **~/stuff**: a file or directory **stuff** in your home directory, when you are in another directory
- **../stuff**: a file **stuff** in the directory above you
- **../foo/stuff**: a file **stuff** in the directory **foo** in the directory above the one you are in

Here's a common mistake: Suppose you wanted to look at the files in your Mail directory and issued the following command:

```
joeuser@athena:~$ ls /Mail /Mail not found (No such file or directory)
```

This is one manifestation of a common mistake people make when learning about directory specifications. You were trying to list out the files under your Mail directory, but now it seems that your mail files have disappeared.

The problem is the use of the */* character. A slash at the beginning of a directory specification means the whole system's root directory. Thus, you did not name your Mail directory, you named a Mail directory immediately under the root directory. This directory does not exist on Athena. (If it did, it would belong to the operating system, and it would be unreadable anyway.)

The rule, then, is to never start a directory specification with */* unless you explicitly mean one of the directories immediately below the whole system's root directory, for example: /etc, /mit, and /usr.

Summary of File/Directory Commands

The following commands let you list, examine, create, delete, copy, and rename files and directories:

- **ls**: list contents of directory
- **cat**: catenate and display file(s)
- **more**: display contents of file one screenful at a time
- **tee**: pipe copy of output into file
- **cp**: copy file/directory
- **mv**: move (rename) file
- **delete**: mark file/directory for later permanent removal
- **expunge**: permanently remove files marked for deletion
- **lsdel**: list files marked for deletion
- **purge**: permanently remove files under ~ marked for deletion
- **undelete**: recover files marked for deletion but not yet removed

- **rm**: permanently remove file
- **pwd**: display name of current working directory
- **cd**: change to the specified directory
- **mkdir**: create new directory
- **rmdir**: remove empty directory
- **echo**: displays the typed text, expanding wildcards