

CS 344 – UNIX OS

Fundamentals – Lecture #6

Purushotham Bangalore
Department of Computer and Information
Sciences
University of Alabama at Birmingham (UAB)

“history” command

- Displays a list of previous commands executed by the shell
- Each command has an associated number
- To repeat the last command enter **!!**
- To execute a command by event number use *!number* (e.g., !19 executes the command associated with event # 19)
- To execute a command that begins with a specific string use *!string* (e.g., !ca will execute the last command starting with ca)

More use of “history” command

- To add history event number to command prompt enter *export PS1='[N] \$ '*
- To select all arguments from previous command use *!** as argument of new command

```
cat quizscores homework  
wc !*
```

- To select the last argument of previous command use *!\$* as argument of new command
- To add an argument to a previous command use *!! NewArgument* or *!string NewArgument*

```
!! -l  
!cat hw5
```

2/28/2005

3

Modify history using editing commands

- Editing commands can be used modify previous commands
- The shell must be first instructed which editor commands to use, this is done by typing “set -o vi” at the command prompt
- After this step we can think of history list as a file and use vi commands to access and change the history
- By default we are in the insert mode, to enter command mode press ESC, then enter the appropriate vi command

2/28/2005

4

File Permissions

- Typical UNIX user performs the following operations on files:
 - Read files (using more, cat, etc.)
 - Write files (using >, >>, cat, vi, etc.)
 - Execute commands in a file (shell scripts, executables, etc.)
- Correspondingly each file has three permissions read, write, and execute (rwx)
- On UNIX systems there are three classes of users: the *owner*, other members of owner's *group*, and all *other* users
- The owner can modify permissions for each of these three classes of users
- To examine file permissions use `ls -l`

```
$ ls -l myfile
-rw-r--r-- 1 puri staff 2093 Feb 28 10:52 myfile
$ ls -ld cs344
drwxr-xr-x 3 puri staff 4096 Feb 23 12:02 cs344
```

2/28/2005

5

Determine user and group

- To determine login name of user type “echo \$USER” or “who am i”
- To determine what groups you belong type “groups” (first group is your default group)
- To change to a new group type “newgrp groupname”, any new files created now will have this group name
- To determine your user id (UID) and group id (GID) type “id”
- To change group use the command “chgrp” and to change owner use the command “chown”

2/28/2005

6

Changing File Permissions

- To change current file permissions use “chmod” (change mode) command
- To add specific permission use chmod +
 - To add write permission to all users use: chmod +w filename
 - To add read permission to only to users in your group use: chmod g+r filename
- To remove specific permission use chmod –
 - To remove read permission for all users use: chmod –r filename
 - To remove read, write, and execute permission for the group and others use: chmod go-rwx filename
- You can also combine add and remove permissions (e.g., chmod u+x,g+r,o-rwx filename)

2/28/2005

7

Using numerical permissions

- Instead of using u,g,o for user, group, and others we can also specify file permissions using numbers
 - rwX = 111 = 7
 - rw- = 110 = 6
 - r-x = 101 = 5
 - r-- = 100 = 4
 - wx = 011 = 3
 - w- = 010 = 2
 - x = 001 = 1
 - = 000 = 0
- chmod go+rx filename = chmod 755 filename (assuming current user permission is rwX = 7, if it is rw- = 6, then use chmod 655 filename)

2/28/2005

8

Directory Permissions

- To list contents of a directory with `ls` command we need read permissions
- To add/remove files in a directory we need write and execute permissions
- To change to a directory or go through the directory we need execute permissions
- To list files with `ls -l` we need read and execute permissions for the directory, since information about permissions, owner, group, etc. are in the directory entry

2/28/2005

9

Set/Get Default Directory Permissions

- When new file/directory is created the shell uses default permissions determined by `umask` value
- To obtain your default `umask` value, at command prompt enter: `umask`
- To change current `umask` value, enter `umask <new-mask-value>`
- Based on the `umask` value appropriate permissions are unmasked (allowed)
- Changing `umask` value has no effect on existing files, only new files will be effected

Umask	New Directory Permissions	
000	<code>rw-rw-rw-</code>	777
022	<code>rw-r--r--</code>	755
027	<code>rw-r--x--</code>	750
017	<code>rw-rw----</code>	760

2/28/2005

10

Get/Set File Permissions

- Set umask to 000, create a new file, list the file using `ls -l`, this will indicate the default file permission (typically `rw-rw-rw-` = 666)
- Execute permission are never granted when files are created hence setting the mask on execute bit has to effect
- Set umask to 022, create a new file, list file, the new file permissions will be `rw-r--r--` = 644
- Set umask to 023, create a new file, list file, the new file permission will be still `rw-r--r--`
- To retain file permissions during file copy use `cp -p` option