COSC 3750
More Utilities

Kim Buckner

University of Wyoming

Jan. 25, 2022

File control and Management

ls – list.

- This is sometimes aliased to *dir*.
- Has a LARGE number of options from what to list to what to show about the files.
- Most common options are *-a* (all), *-l* (long), *--color* and/or *-F*.
- Last two are used to show visually what type the files are.

(more . . .)

- Types are such as directory (/), executable (*), symbolic link (@), FIFOs (|) and sockets (=).
- Three versions I use are
  - ls --color=never -F
  - ls --color=never -alF
  - ls --color=never -lF

chmod

- Change mode (permissions).
- All files are assigned a set of permissions. These are associated with three sets of people. The **owner**, the **group**, and the **others** (or the world).
- The permissions are normally displayed as a string like -rw-rw-rw-. These are grouped from left to right.

(more . . .)

- The left-most position is a special indicator, it is **d** for a directory and **l** for a link, **b** for a block special, **c** for a character special and so on. See ***info ls*** for more.

- The next 3 are the **owner**'s permissions, r for read, w for write, and x for executable.

(more . . .)

- The 'middle' three are for the **group** permissions and the right three for the **others**.

- There are some special permissions, set-user-ID, set-group-ID, and the "sticky" bit (restricted deletion flag).

rm – remove

- Remove or "unlink" a component
- Linux and UNIX systems are unforgiving, removing is permanent! There is no "undo".
- This means that you need to be a little more careful. Also that you need to keep track of what you are doing and keep things neat so you do not get confused.

(more . . .)

- You may not remove any files for which you do not have permission/ownership.
- There are several options.
  - The *-i* interactive option prompts you before removing the file.
  - By default *rm* does not remove directories and cannot remove a non-empty directory. But using rm -rf solves this by (1) "forcing" the removal and (2) recursively removing all the components that are in a directory. Needless to say, this is VERY dangerous.

cp – copy

- This can be simple, *cp* **source destination**. Creates a new file, **destination**, with the user's default permissions.

- There a number of options, the *-i* (interactive) option means that you will prompted before overwriting an existing destination file. THIS IS NOT THE DEFAULT!

(more . . .)

- If the destination is a directory, all the listed source files, with the same names, will be created in that destination directory.

mv – move (rename)

- Similar to copy but does NOT create new files, simply renames the components.
- *mv* **source destination**. If the destination is a directory, all the source are "moved" there.
- This command does not change the location on disk, it just modifies the inode and directory entries.

ln – link

- Create a link to an existing file
- There are two types of links, "soft" and "hard". Soft or symbolic links are MUCH safer. If you remove a soft link, you do NOT remove the corresponding file. With hard links, that is not the case.

(more . . .)

- *ln* -s **source** [**destination**]. If the destination is a directory or the destination is not given, a link, with the same name as the source will be created in the directory (or the current directory).

- You cannot change the permissions of a link, its permissions are ignored and only the permissions of the 'real' file are used.

(more . . .)

- There can be any number of links to a file, and if the file is removed or renamed, the links are unaffected. BUT attempts to access the file with the link will result in "No such file or directory".

- You generally cannot create a hard link across physical filesystems.

touch

- Change file times
- In its simplest form, *touch* **name**, the file access and modification time is changed to "now". If **name** does not exist, it is created (empty).
- Options allow a time different from "now" to be used, only the access time to change, only the modification time to change, and more.

mkdir – make directory

- Create a directory
- Basic is just *mkdir* **name**.
- This creates an empty directory, **name**, with the user's default permissions, usually `rwx`. You cannot create a file in a directory if you do not have execute permission on that directory.

(more . . .)

- Directory permissions are somewhat odd, so do not mess with them until you really need to, then practice a little to see what happens.

- Using the -p allows you to create a directory tree. It says create the parent directories if they do not already exist.

(more . . .)

- So you can use mkdir first/second/last if the directories first and first/second do **not** already exist, they will be created then first/second/last will be created.

rmdir – remove directory

- Only if it is empty.
- Can also remove a directory and its parents.
- Most people only use /bin/rm because it can remove all the files and then the directory.

pwd – print working directory

- A nice simple command that just prints the path of the current location.
- The shell generally implements this command.
- There is a */bin/pwd* which accesses the environment variables for the current directory.

Miscellaneous Utilities

uptime

- How long the system has been running
- Also gives current time and number of users
- Maybe the most important is the load averages of the last 1, 5, and 15 minutes.
- See also **w**.

uname

- Has nothing to do with **users**.
- If used as *uname -a* it prints
  - kernel name
  - host name
  - kernel version
  - hardware name
  - processor type
  - hardware platform
  - O/S

hostname

- Very simply gets the name of the computer you are logged into.
- If you need the name in a script which might run on all the machines in an organization, this will do it.
- It is also root's way of modifying info on the fly, if needed.
- Can also give you the fully qualified name and the IP address.

which

- This prints to standard output the full path of a (shell) command.
- Doesn't seem really useful until you need it
- If the command is aliased, you will find that out.
- And you can find out if something is installed (or in your path), without trying to execute the command.

df – display filesystems

- This is used to show the actual filesystems the user is accessing.
- This does not mean that you can modify them, only that they are mounted.
- The display also includes the usage and freespace of the filesystems.
- When you cannot create a file for some reason, this is the first place you look.

du – disk usage

- Display how much space you (or some other user) is using.

- Can only check on things that are readable by you.

- When the system administrator complains to you about your usage, use this to figure out where those 15 gigabytes is stored.

file

- An odd utility that determines the type of a file
- Performs 3 checks:
    - filesystem checks (stat (2)), determines if it exists and is a special device.
    - magic checks, looks at the beginning of the file for a "magic number".
    - language checks, look and see what (like ASCII text) is in the file.

§§