Linux Academy: Managing Files In Linux – Course Notes

Video One – File Naming Basics and A Look At File Commands

- Naming conventions
  - case sensitive: unlike windows, linux is case sensitive
    - example: test.TXT and Test.txt and TEST.txt are all completely separate files
    - spaces: are permitted, but require escape character preceding them (\) when referencing the file
    - "dot" files: file names that are preceded with a 'dot' or period (.), can appear invisible when listing the filesystem with a plain 'ls' command
      - listing the filesystem with 'ls -al' will list ALL files in all formats in that directory
      - most commonly used as configuration files in a home directory
- Wildcard
  - *: will substituted any value from 1 to N characters
    - example: bul* would be valid to refer to filenames bulk, bull, bullshark
  - ?: one character substitute rather than 1 to N characters
    - example: bul? is valid for 'bulk' but not 'bulkrate'
  - [char]: character based substitution, allows specific character designation as substitutes for a file name
    - example: bu[a-z]k would be valid for files 'bulk, buok, bukk' but not 'bu3k, bu#k, bu@k'
- ls
  - lists files and directories
    - -al: list all files, directories and ‘hidden/system config’ files (.dir, .config, etc)
- cp
  - makes copies of existing files
    - –f: force overwrite
    - -r: recursive (copies directories too)

Video Two - File Archiving and the RM, MV Commands

- mv
  - cannot 'move' a directory to a file, or vice versa
  - mv file1 file2 - effectively renames the file1 to file2
    - -i: interactive, will prompt before an overwrite or move for example
  - same filesystem moves simply update the location and reference to the content, date/timestamp of the file will remain the same
  - moves across filesystems (devices or network) recreates the file in a new location, adds the directory references to its location and then deletes the original file, as a result, the date/timestamp will reflect the time/date of the move
- rm
  - removes files that you have permission to
    - -r: recursive (removes directories)
    - -f: force (removes with a ‘y’ answer to ‘Are You Sure?’)
- tar
  - used to archive or unarchive files and directories, can be used with or without additional compression
    - -x: extract files from the archive
    - -t: list files in the archive
    - -c: create the file

        - -v: verbose output to console
        - -z: (de)compress with gzip
        - -j: (de)compress with bzip2
        - -A: add to the existing archive
        - -f: name of tar file follows this
            - Example: tar zxvf list.tar.gz /home/username
                - This will extrac all files and directories in the file into the /home/username directory, uncompress them with gzip, output verbose information to the screen from from the list.tar.gz file

Video Three – Linux Links

- ln
    - equivalent to windows shortcuts or OSX aliases
    - give multiple names to a file, in a different location (or the same)
- Soft links vs hard links
    - –s: soft link creates a special file that 'refers' to the original file (path and name) but does not duplicate the file content. Deleting this does not delete the original.
    - Hard link: hard link creates a duplicate of the original file and cannot be used across filesystems, nor can they be used to refer to a directory
        - Removing the hard link does not remove the original, you would need to delete both files in both locations
        - Hard linked file updates are replicated to all hard linked locations

Video Four – Basic Directory and Group Commands

- mkdir
    - makes a new directory with the indicated name
    - makes one directory level at a time by default
    - making a full directory path that does not already exist is possible
        - -p: makes a base directory and creates all subs in the command
        - Example: mkdir –p /home/to/my/dir
            - Makes /home and all subs in the path if they do not exist
- rmdir
    - removes a directory that you have access to
        - -p: removes all indicated directories in the path that exist
    - only removes empty directories, removing directories that have content can be accomplished with the 'rm' command
- chgroup
    - changes the group ownership of a file or directory if you have rights to manipulate that objects ownership
    - the group you are changing to must be one that you are already a member of

Video Five – Special Permission Bits

- Easiest to manipulate as root user, although caution during bit manipulation in system directories – you could end up with an unbootable system
- sh (dash)
    - shell interpreter that will run any applications (the environment is independent of the currently logged in user shell, environment is unique, but the environment pulls parameters from the owner of the application)
- +s (chmod)

    - Will change the file/application to run with the same permissions of the user that owns the file
- setuid
    - run the file with the same permissions as the user that owns the file
- sgid
    - run the file with the same permissions as the group that owns the file
- sticky bit permission
    - +t: protects files or directories from being deleted by those users/groups that do not own the files
    - can override the normal file/directory permissions on a file

Video Six – Default Permissions, umask, newgrp and chattr

- newgrp groupname
    - will change the default group that files/folders are created under to the indicated group
- umask
    - will display the permissions that files and folders are created under
        - files = ####
        - folders = ####
        - #### (in octal notation)
            - 0 = rwx
            - 1 = rw
            - 2 = rx
            - 3 = r
            - 4 = wx
            - 5 = w
            - 6 = x
            - 7 = no permissions
    - Can be set by octal notation (above) or symbolic notation
        - Example a+r = allow read for all users
    - Default permissions are 666
    - umask values then subtract from the default
        - example: umask 277
        - leaves us files = 666 – 277 = 400
- chattr
    - changes the attributes on a linux filesystem
    - example: chattr –a filename.txt
        - remove the write attribute for a file allowing ONLY append
    - –i: immutable (cannot write, delete or link to file)
    - –s: set the file attribute for deletion so that recovery is not possible, the inode is overwritten with 0's
    - –A: do not update the modified time if file is written to

Video Seven – Linux Core Directories and What They Are Used For

- /etc
    - system files, configurations, start up information, locale, link to parameters, etc
    - /etc/init.d: initialization scripts and services that start on boot up
- /boot

    - grub, kernel parameters
    - sometimes (ideally) a separate partition from root
- /bin
    - common system scripts, applications and utilities (ping, kill, top, sh, mount, etc)
- /sbin
    - system administration scripts, applications and utilities (root level user access command like fdisk, format, reboot/shutdown, file system management, etc)
- /lib
    - system binary libraries (like windows DLLs) that are shared and linked to by applications installed on your system
        - ld.so.conf can reference various directories that contain library files for linking
- /usr
    - bulk of the linux base applications and scripts
    - common directories for all users, binary files, libraries, local binary files by user, etc
- /opt
    - user level or post installation user applications that are installed (often referred to as the equivalent of "program files" for windows)
- /root
    - the home directory of the root user
- /var
    - logs, spools (mail and print), html files for apache servers, libraries for applications, mysql default installation location
- /tmp
    - temporary system or application directory, this is cleaned up periodically and automatically
- /mnt
    - typical location for mounting external filesystems (NFS, Samba, Windows, cdrom, etc)
- /dev
    - device directory for linux, direct references to all the devices on the system
- /proc
    - list of files that contain system level information (cpu, disk space, kernel version, memory, existing filesystems, distribution type, etc)
    - used by system utilities to cleanly display system level behavior and information

Video Eight – Finding Files In Linux Using Find, Locate, Whereis, Which and Type

- find
    - example: find /home/username –name "file*"
    - look in the /home/username directory for a file with the name "file*", will display filename.txt files filemeaway.sh, etc
    - slower since it reads the file and directory system one by one, but more flexible
    - does not use an indexing system, rereads the entire structure each time
    - –size: can find files of a certain size
- locate
    - example: locate etc
        - will display any files or directories containing 'etc' in the name or path (very quick)
        - uses index files by running 'updatedb' to create the index used
- whereis
    - search path directories (binary, man pages, libraries) for the file asked for

    - example: whereis ls
        - will search and find the command ‘ls’ immediately in /bin
- which
    - search the binary path directories (/bin, /sbin, /usr/local/bin, etc) indicated in the shell PATH variable
- type
    - will display how the system will interpret the command
    - example: type man
        - will display “man is /usr/bin/man’ so you will know it is an executable