Linux Administration Notes

T.J. Robinson

November 19, 2025

1 File Management

1.1 File Permissions

File Permissions are used to prevent unauthorized access by *users* to files and directories

1.2 Permission Classes

Permission Classes unique cataegorizes utilized by the kernel to maintain file security via access rights.

Users are assigned to 3 catagories:

1. **User Owner (u)**
2. **Group (g)**
3. **Other (o)**
4. **All (a)** - represents all 3 classes

1.3 Permission Types

There are 3 types of permission bits:

1. **Read (r)** - view and copy
2. **Write (w)** - modify
3. **Execute (x)** - run
4. **null (-)** - permission not granted

1.4 Permission Modes

1. Append permission bit (+)
2. Revoke permission bit (-)
3. Assign permission bit (=)

1.5 Modifying Permissions

chmod is used to change permissions of files & diectories

1.5.1 Symbolic vs. Octal Notation

- **Symbolic Notation** uses letters (ex. **u,g,o**) & sumbols (ex. **+,-,=**) to modify permissions.
- **Octal Notation** uses 3-digit numbering (ex. **766**) to modify permissions.

1.6 Default Permission

umask is used to set default permissions on a file without modify permissions on existing files and directories.

- The default *umask* value for all users including the root user is **0022**.
- The default initial permission value for files is **666** & **777** for directories.

1.7 Calulating Default Permission

Calculating default permissions for files:
Initial Permission 666
umask - 022
==========
Default Permission 044

Calculating default permissions for directories:
Initial Permission 777
umask - 022
==========
Default Permission 055

1.8 Special File Permission

There are 3 Special Permission Bits that can be configured for binary files and directories:

1. **SETUID** (SET User IDentifier) - applied to binary executable files at the *user owner (u)* level. It gives non-owners the same file permissions as the user owner.
2. **SETGID** (SET Group IDentifier) - applied to binary executable files at the *user owner (u)* level. It gives non-owners and group memebers the same file permissions as the user & group owner.
3. **Sticky Bit** - is set on public directories to prevent other users from deleteing or moving files.

1.9 File Searching

find is the command used to search for files on a Linux System and perform actions on found files.

After invoking the find command, the first option is the location path to search (ex. current (.), /tmp, /home/).

1.10 `find` Command Options

- use **-iname** to search for files that *begins* with a string.

  example input: `find /dev -iname usb*`

  example output:
  ```
  /dev/usb1
  /dev/monusb0
  /dev/monusb1
  ```

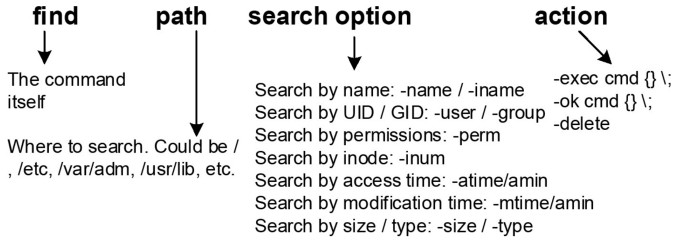

Figure 1: FIND Command Syntax

- use **-size** to search for files by size
  - use (-) to find items smaller then designated size
    example input: `find /dev -size -2M`
  - use (+) to find items larger then designated size
    example input: `find /dev -size +2M`
- find files owened by a specific user (*daemon*) and exclude specific group (*user1*).
  example input: `find /dev -user daemon -not -group user1`
- use **-type** to seach by filetype (d=directory, f=file)
  example input: `find /usr -type d -name src`
- use **-maxdepth** to seach set maximum subdirectory depth to search
  example input: `find /home -maxdepth 3 -type f -name src`

1.10.1 Using the `-exec` and `-ok` options

- `-exec` is used to perform actions on the files found by `find`.
- `-ok` is the same as `-exec`, but requires user confirmation to execute.

example input: `find /Documents -type f -name BLS* -exec ls -ld {} \;`

- (**{}**) represents each file found
- (**;**) terminates the command.
- (\) is used to escape (**;**)

2 Linux Processes and Job Scheduling