Basic Commands

- `ls [(optional) directory]`
- `cd [directory]`
- `touch [new file]`
- `mkdir [new directory]`
- `cat [file]`
- `mv [source] [destination]`
- `cp [source] [destination]`
- `rm [file(s)]`
- `rm -rf [directory]`

Special Symbols

- `*` ⇒ all/any
- `.` ⇒ current directory
- `..` ⇒ previous (parent) directory

User Management

- `useradd [user]`
- `passwd [user]`
- `userdel -r [user]`

- `groupadd [group]`
- `gpasswd [group]`
- `groupdel [group]`
- `groups [user]` ⇒ list groups a user belongs to
  - **Note:** to list users in a group, just `cat /etc/group`
- `gpasswd -a [user] [group]` ⇒ add user to group
- `gpasswd -d [user] [group]` ⇒ remove user from group

- `who` OR `w` ⇒ view who's currently logged in
- `usermod -s [shell] [user]` ⇒ change user shell
  - eg. `usermod -s /sbin/nologin root`
  - Remember to disable root login in ssh as well:
    - Change to `PermitRootLogin no` in `/etc/ssh/sshd_config`

Important Files:

- `/etc/passwd` ⇒ users
  - Human users have ids starting at 1000
- `/etc/shadow` ⇒ user passwords
- `/etc/group` ⇒ groups
- `/etc/lightdm/lightdm.conf` ⇒ login configuration
  - **Note:** disable guest account by adding line `allow-guest=false`

Sudo/Wheel

- `sudo -l -U [user]` ⇒ permissions for specific user
- `/etc/sudoers` ⇒ file with all sudo permissions

  - Edit with `visudo`, *not* `vi` or `nano`

```
[user] [hosts rule applies to]=([impersonatable users]:[impersonatable groups]) [commands]
```

User Command History

- `/home/[user]/.bash_history` ⇒ normal user commands
    - Can be redirected to `/dev/null` with
        - `rm ~/.bash_history`
        - `ln -s /dev/null ~/.bash_history`
- `tail /var/log/auth.log | grep username` ⇒ sudo commands

Permissions

- `ls -lah` ⇒ display all permissions info and hidden files
- `chown [user]:[group] [folder/file]` ⇒ set owner of folder/file
- `chmod [code] [folder/file]` ⇒ set permissions on folder/file
    - Code:
        - `[user][group][other]`
            - for each: `[read][write][execute]` then convert to decimal
                - 0 ⇒ does not have permission
                - 1 ⇒ does have permission

Add `-R` to either command to recursively change permissions of contents of a directory

Note: See *File Management* for finding files and directories with specific permissions

File Management

- `grep [contents] [file/directory]`
    - `-i` ⇒ ignore case
    - `-r` ⇒ recursive search
    - `-H` ⇒ list file names along with contents
    - `-l` ⇒ list file names instead of contents
    - `-o` ⇒ display only matching text in contents
    - `-n` ⇒ list line number
    - `-v [contents]` ⇒ unwanted contents

  examples:
    - `grep -Horn [directory]`
    - `grep -Hrn [directory]`
- `find [base dir]`
    - `-type d/f/l` ⇒ search directories/files/symbolic links
    - `-user [username]` ⇒ files owned by user
    - `-writable` ⇒ files writable by current user
        - **Note:** for another user, `sudo -u [user] find` … but this will ONLY display files that are both readable and writable, not -wx or -w-
        - eg. `sudo -u [user] find / -type d -writable 2>/dev/null`
    - `-perm [prefix][permission]`
        - Prefixes:
            - `/` ⇒ any permission bit set

      - `-` ⇒ all permission bits set
      - no prefix ⇒ exact permission specified
    - Permissions:
      - Standard code: `[1-7][1-7][1-7]`
      - `[entity]=[permission]`
        - Entities:
          - `u` ⇒ owner
          - `g` ⇒ group
          - `o` ⇒ other
          - `a` ⇒ all
        - Permissions:
          - `r` ⇒ read
          - `w` ⇒ write
          - `x` ⇒ execute
          - `s` ⇒ set id (setuid/setgid)
  - `-user root -perm -u=s` ⇒ setuid
  - `-user root -perm -g=s` ⇒ setgid
  - `-exec [command] [args] [ending]`
    - Args: to pass `find` results as arg use `{}`
    - Endings:
      - `\;` ⇒ one find arg per command
      - `+` ⇒ passes in as many args as possible

Package Management

Note: `apt` shown here, but `apt-get`, `yum`, etc. are similar with some minor differences. Notably, `yum update` both pulls updated repos and updates outdated packages.

Viewing Packages

- `apt list --installed` OR `dpkg -l`
  - `apt list --installed | grep -v automatic`
- `apt show [package]` OR `dpkg -p [package]` ⇒ get info on package
- `sudo aptitude search -F ' * %p -> %d ' --no-gui --disable-columns '?and(~i, !?automatic, !?section(libs), !?section(kernel), !?section(devel))'` ⇒ pretty list only important packages and info
- `apt update && apt list --upgradable`

Updating Packages

- `apt upgrade`
- `apt install --only-upgrade`
- `/etc/apt/apt.conf.d` ⇒ contains apt configuration
  - Check out `10periodic` for updating package lists and auto upgrades
- `/etc/apt/sources.list` ⇒ repositories list

Process Management

- `ps`

  - `e` ⇒ show environment variables
  - `-e` ⇒ show all processes
  - `a` ⇒ list all processes with tty
  - `-u` ⇒ user-oriented format
  - `-f` ⇒ full format
  - `-x` ⇒ list all processes without tty
  - `-o` ⇒ user-defined format

  examples:
  - `ps aux`
  - `ps -ef --forest`
  - `ps -eo user,pid,cmd --forest`
  - `ps ao user,tty,pid,cmd --forest`

- `lsof -p [pid]` ⇒ files opened by process
- `lsof -i :[port]` ⇒ files opened by process on specific port

- `pidof [name]` ⇒ get pid from name
- `pwdx [pid]` ⇒ get name from pid
- `kill [pid]`

All process information linked in `/proc/[pid]`:

- `cmdline` ⇒ command line arguments
  - `cat /proc/[pid]/cmdline | tr '\000' ' '` ⇒ get running command
- `cpu` ⇒ current and last cpu
- `cwd` ⇒ link to current working directory
- `environ` ⇒ environment variables
- `exe` ⇒ link to executable
  - `ls -l /proc/[pid]/exe` ⇒ get process exe file
- `fd` ⇒ file descriptors
- `maps` ⇒ maps executables to libraries
- `mem` ⇒ memory
- `root` ⇒ link to root directory
- `stat` ⇒ status
- `statm` ⇒ memory status
- `status` ⇒ human-readable status

Service Management

- `systemctl [list-unit-files/list-units]`
  - `-t service`
  - `-t timer`
  - `--state=enabled` (list-unit-files only)
  - `--state=running` (list-units only)
- `systemctl [enable/disable/start/stop/restart/status]`

Cron Jobs

Note: Also check `anacron` with the same files and directories but replacing `cron` with `anacron`.

- `crontab`
  - `-u [user]`
  - `-l` ⇒ list
  - `-e` ⇒ edit
    - Syntax: `[minute] [hour] [day of month] [month] [day of week] [command]`

Crontab Files

- `/etc/cron.allow` ⇒ users who can edit the crontab
- `/etc/cron.deny` ⇒ users who cannot edit the crontab

  Note: /etc/cron.allow overrides /etc/cron.deny
- `/var/spool/cron` ⇒ cron jobs for each user
- `ls /etc/cron*` ⇒ view all other directories (they're self-explanatory)

Kernel Modules

- `lsmod` ⇒ list modules
- `rmmod [module]`
  - `-f` ⇒ force (dangerous)

Networking

- `ip`
  - `address`
  - `route`
- `netstat/ss`
  - `-a` ⇒ show all
  - `-l` ⇒ show listening
  - `-n` ⇒ show numerical addresses
  - `-t` ⇒ show tcp
  - `-u` ⇒ show udp
  - `-p` ⇒ show pid and process name

  examples:
  - `netstat -antup`
  - `netstat -plunt`

Firewall

ufw

- `ufw`
  - `[enable/disable]`
  - `status`
  - `default [allow/deny] [outgoing/incoming]`
  - `allow [service or port]`
  - `allow [service or port]/[tcp/udp]`
  - `allow from [source] to [destination] port [port] proto [tcp/udp]`
    - Replace with any for all sources or destinations

    - `delete …`

firewalld

Zone Commands

- `firewall-cmd --list-all-zones` ⇒ shows all zone information
- `firewall-cmd --get-zones` ⇒ only shows zone names
- `firewall-cmd --list-all --zone=`[zone] ⇒ shows info for specific zone or `firewall-cmd --info-zone=`[zone]
- `firewall-cmd --new-zone=`[zone] `--permanent` ⇒ creates a new zone
- `firewall-cmd --set-default-zone=`[zone] ⇒ sets default zone
    - Default zone is used for everything that's not assigned to another zone

Rule Commands

- `firewall-cmd`
    - `--zone=`[zone] → if this is not specified, it will modify the default zone
    - `--permanent` ⇒ persist on service restart
    - [rules]

Rule	Description	Command Option (Flag)
Interface	The interface assigned to this zone	`--change-interface=`[interface] ⇒ assign interface to this zone **Note:** you can do this instead of adding `ZONE=`[zone] to the CentOS IP configuration file.
Source	Whitelist (accept connections from) IP addresses	`--add-source=`[ip] `--remove-source=`[ip]
Target	How to handle packets that don't match any other rules	`--set-target=`[accept/reject/drop]
Service	Services running on *this* machine that are accessible by this zone	`--add-service=`[service] `--remove-service=`[service] **Note:** use `firewall-cmd --get-services` to list available services.
Port	Ports running on *this* machine that are accessible by this zone. Use when a service is not available	`--add-port=`[port]`/`[`tcp/udp`] `--remove-port=`[port]`/`[`tcp/udp`]
Forward Port	Ports to be forwarded to *other* machines that are accessible by this zone	`--add-forward-port=port=`[source port]`:proto=`[`tcp/udp`]`:toport=`[destination port]`:toaddr=`[destination address] `--remove-forward-port=`[same options]

Masquerade	Allow masked outbound connections on this zone (useful for external)	`--add-masquerade` `--remove-masquerade`

Always load changes to rules with `firewall-cmd --reload`