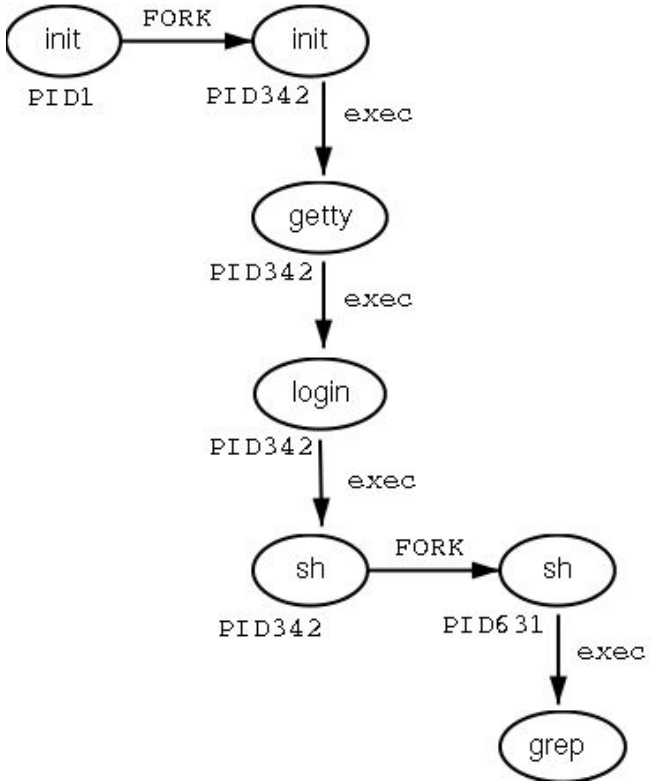
Advanced Unix Commands

Kameswari Chebrolu

Process Management

- A process is a program in execution
  - Hierarchy of processes
- Init has process id 1
  - parent of all processes
  - executed by the kernel during the booting of a system
- From a process, another process can be created
  - Achieved via fork system call
  - Parent-child relationship between the two processes
- Two main states: running, suspended



- ps (process status): displays a list of the processes currently executing
  - default output is a list of the processes associated with the command-line
  - Shows unique process id (pid), terminal used, amount of CPU time, and the program name
  - ps aux more informative
    - “a”: display the processes of all users
    - “u”: user-oriented format → more details
    - “x”: list the processes without a controlling terminal
      - started on boot time and running in the background

- USER - The user who runs the process.
- %CPU - The cpu utilization of the process.
- %MEM - The percentage of the process’s resident set size to the physical memory on the machine.
- VSZ - Virtual memory size of the process in KiB.
- RSS - The size of the physical memory that the process is using.
- STAT - The process state code, such as Z (zombie), S (sleeping), and R (running)
- START - The time when the command started.

- Processes can run in foreground or background
  - Foreground processes: interactive processes
    - Cannot initiate a new process from the same terminal.
  - Background processes: non interactive processes
    - Can initiate other processes within the same terminal
  - Can move programs to foreground and background via fg and bg commands; Can stop via ctrl+Z

- jobs is another command, a shell builtin that tells you about the jobs that the current shell is managing
- Kill: can terminate a process if it hangs and not responding
  - Needs PID; -9 option kills it with no questions asked (e.g. kill -9 12876)
  - Often a process running in command-line in fg can be killed via ctrl-C

Access Control

- UNIX is a multi-user system
- Every file and directory (in your account) can be protected from or made accessible to other users. How?
- Permissions for a file or directory may be any or all of
  - r - read; w - write; x - execute
  - a directory must have both r and x permissions if the files it contains are to be accessed
- Each permission (rwx) can be controlled at three levels:
  - u (user = yourself)
  - g (group, a set of users)
  - o (others, everyone else)
- File access permissions are displayed using “ ls -l”

- First field: - for File, d for Directory, l for Link
- Second,third,fourth fields: permissions for owner, group and others
- Fifth field: specifies the number of links or directories inside this directory
- Sixth field: user
- Seventh field: group
- Eighth field: size in bytes (use -lh option for better understanding)
- Ninth field: date of last modification
- Tenth field: name of the file/directory

- “chmod” command helps change access permissions for files you own
  - chmod [OPTIONS] MODE FILE(s)/directory(s)
  - Only root, the file owner or user with sudo privileges can change the permissions of a file
  - Symbolic Mode:
    - u - The file owner.
    - g - The users who are members of the group.
    - o - All other users.
    - a - All users, identical to ugo
    - - Removes the specified permissions.
    - + Adds specified permissions.
    - = Changes the current permissions to the specified permissions
      - If no permissions are specified after the = symbol, all permissions from the specified user class are removed
    - E.g. chmod g=r filename; chmod a-x filename;

- Numeric mode:
  - r (read) = 4
  - w (write) = 2
  - x (execute) = 1
  - no permissions = 0
  - chmod 640 samplefile (octal notation, user has r+w, group has read, others have none)
  - chmod -R 700 dirname (Recursively set read, write, and execute permissions to the file owner and no permissions for all other users on a given directory)

Superuser

- Superuser: user with super powers
  - A real user account (often root) that can do just about anything (modify/delete files, run any programs etc)
- For security reasons, “su” was introduced
  - Can mean ‘superuser’ or ‘switch user’
  - Helps change to another user without having to log out and in
    - Terminal session switched to the other user
    - Requires password of the other user
  - Administrators spend most time using normal account, when needed switch to superuser, do task and logout

- Further improvement, “sudo” was introduced
  - “switch user and do this command”
  - Prevents long-lived terminal sessions with dangerous powers
  - sudo is used to prefix a specific command that has to be run with superuser privileges
  - Requires user’s own password with some caching for a period of time
- Be very careful when using sudo or su

Regular Expressions (regex)

- Used in text editors, programming languages, and command-line tools (grep, sed, awk etc)
- regex: a pattern that matches a set of strings
- Metacharacters: characters with special meaning
  - "^" beginning of a line
    - Can also mean "not" if inside []. Coming up
  - "$" end of line
  - "." match any single character
  - "\" escape a special character
  - "|" or operation i.e. match a particular character set on either side

Quantifiers: specifying the number of occurrences of a character

- “*” Match the preceding item zero or more times
- “?” Match the preceding item zero or one time
- “+” Match the preceding item one or more times
- “{n}” Match the preceding item exactly n times
- “{n,} Match the preceding item at least n times
- “{,m}”Match the preceding item at most m times
- “{n,m}” Match the preceding item from n to m times

Groups and Ranges

- “ ( )” group patterns together
- “{ }” match a particular number of occurrences (seen before)
- “[ ]” match any character from a range of characters
  - ab[xyz]c "abxc" and "abyc"and "abzc"
- [^…..] matches a character which is not defined in the square bracket
- [a-z] matches letters of a small case from a to z
- [A-Z] matches letters of an upper case from A to Z
- [0-9] matches a digit from 0 to 9.

Commands

- man: manual for commands
  - E.g. man ls
- clear: helps clear the screen to reduce clutter
- history: shows history of all commands you entered
  - At terminal use !number to run that command in history
    - !! runs previous command
  - history 5 (shows last 5 commands)
- du: disk usage, used to estimate file space usage
  - Checkout options: -h, -a, -s
- head/tail: print n lines from head or tail
  - head file; head -n 2 file
  - Checkout: -n and -c option
- which: identify the location of a given executable(s)

More Commands

- grep: “global regular expression print”
  - Searches files for a particular pattern of characters
  - Lots of options: -v, -r, -w, -n, -c, -A, -B, -o, -i etc
- find: locate files/directories with known pattern
  - Options: -name, -type, -size, -mtime, -perm, -delete
- wc: print word, character, line count in a file
  - can accept zero (reads from standard input) or more input
  - Default output: lines, words, characters
  - “-l” number of lines.
  - “-w” number of words.
  - “-m” number of characters.
  - “-c” number of bytes

- tr: performs basic character replacement and removing
  - Checkout options: -c, -d, -s

- cut: helps cut parts of a line by delimiter, byte position, and character
  - “-f” specifies field(s)
  - “-b” specifies bytes(s)
  - “d” specify a delimiter that will be used instead of the default “TAB”
  - --complement - Complement the selection. When using this option cut displays all bytes, characters, or fields except the selected.
  - --output-delimiter - The default behavior of cut is to use the input delimiter as the output delimiter. This option allows you to specify a different output delimiter string
- paste: helps merge lines of files horizontally
  - Checkout options: -s, -d

- sort: sorts the records in a file
  - Checkout options: -r, -n, -k, -t
- zip/unzip: archive/unarchive files with compression
  - For zip, checkout options: -r, -s, -e
  - For unzip, checkout options: -d, -l

Input/Output

Stream: a special file that either continuously receives text in or pushes text out

- Whenever a process starts, that process is given access to three “standard” streams
  - Standard input (stdin; fd is 0)
  - Standard output (stdout, fd is 1)
  - Standard error (stderr, fd is 2); used when an error has occurred
- When a process starts
  - stdout and stderr are configured to print whatever they receive to the terminal screen
  - stdin is configured to read input from the user’s keyboard

Redirection (>, >>, <)

- >: Output of a command redirected to a file
  - command > file same as command 1> file (stdout redirected, stderr still screen)
  - command 2> file (send stderr to file, stdout is screen)
  - command 2> error.txt 1> out.txt (send both to different files)
  - command > file 2>&1 (send both to same file)
  - command 2> /dev/null (suppress error messages)
    - /dev/null is a special file that discards anything written to it
- >> : same as above but append instead of replace
- <: stdin of “command” read from “file.txt
  - command < file.txt

Pipe (|)

- How many files and folders in /etc ?
  - ls /etc > temp.txt; wc -l temp.txt; rm temp.txt
- Pipe takes output from one command and feeds it as input to another command
  - ls /etc | wc -l
  - Operates entirely in memory
  - Unidirectional: flows from left to right
- command1 | command2 | command 3…

Command Substitution

- How to use the output of a command in the arguments of another
  - Note: pipes are great for sharing stdin and stdout between processes
- Command substitution: Output of a command replaces the command itself
  - $(command) or
  - `command`

apt: Software Installation

- apt: helps install, update, remove packages on Ubuntu, Debian etc
  - Most of the apt commands need sudo privileges
- apt update: pulls the latest changes from the APT repositories
- apt upgrade: upgrades packages to their latest versions
  - doesn't upgrade packages that require removal of installed packages; use apt full-upgrade
  - apt upgrade package_name (to upgrade specific package)
- apt install: installs a package(s)
  - Normally pull from repo, but if path specified installs local copy
- apt remove: removes package
  - Can use apt purge for more complete removal including config files
- apt list: lists all packages
  - apt list --installed
  - apt list --upgradeable
- apt show: retrieves information (dependencies, size etc) about a given package

References

- https://ubuntu.com/tutorials/command-line-for-beginners#1-overview
- https://linuxize.com/ (good resource, use search box for info on different commands!)