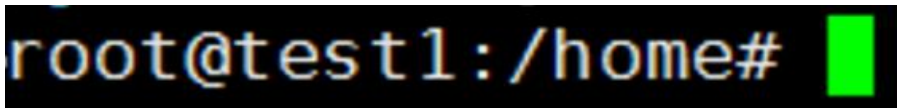
Linux Commands and Tools

Chung-Ang University

Linux commands and tools

- **Linux Shell or Terminal**
  - A shell is a program that receives commands from the user and gives it to the OS, and it shows the output
  - This shell interact with users in a terminal emulation window
  - To open the terminal, press Ctrl + Alt + T in Ubuntu



  - You will learn **shell script programming** with TA

Basic Linux commends - cd

- cd command
  - The cd command (change directory) is a command-line shell command used to change the current working directory in various operating systems
  - $ cd file_path
  - Ex)
    - $ cd Desktop
    - $ cd /home/user_name/Desktop

Basic Linux commends - ls

- ls is a command to list files in operating systems
  - ls lists the files in the current working directory
- Depending on the parameters, various results can be checked
  - l : detailed listing view
  - a : to include hidden files

Ex)

$ ls

$ ls –a

$ ls –l

$ ls –al

```
pungki@dev-machine:~$ ls
Desktop    lynis-1.3.8       Music                          Public
Documents  lynis.log         Pictures                       Templates
Downloads  lynis-report.dat  PlayOnLinux's virtual drives   Videos
pungki@dev-machine:~$
```

```
pungki@dev-machine:~$ ls -l
total 508
drwxr-xr-x 2 pungki pungki   4096 Des 10 16:36 Desktop
drwxr-xr-x 2 pungki pungki   4096 Des 10 19:52 Documents
drwxr-xr-x 5 pungki pungki   4096 Jan  2 00:27 Downloads
drwxrwxr-x 6 pungki pungki   4096 Des 28 09:53 lynis-1.3.8
-rw-r----- 1 root   root   363167 Des 28 09:23 lynis.log
-rw-r----- 1 root   root   115339 Des 28 09:23 lynis-report.dat
drwxr-xr-x 2 pungki pungki   4096 Des  1 10:48 Music
drwxr-xr-x 2 pungki pungki   4096 Des  1 10:48 Pictures
lrwxrwxrwx 1 pungki pungki     38 Des  1 11:14 PlayOnLinux's virtual drives -> /
home/pungki/.PlayOnLinux//wineprefix/
drwxr-xr-x 2 pungki pungki   4096 Des  1 10:48 Public
drwxr-xr-x 2 pungki pungki   4096 Des  1 10:48 Templates
drwxr-xr-x 2 pungki pungki   4096 Des  1 10:48 Videos
pungki@dev-machine:~$
```

Basic Linux commends - mv

- mv (short for move) is a linux command that moves one or more files or directories from one place to another
- Rename the file, you “move” it into a new file with the new name
- File move and rename action could have been achieved in one step

$ mv original_filepath target_filepath

Ex)
$ mv /Documents/Ukulele/Apache.pdf /test/ (file move)
$ mv Apache.pdf move_Apache.pdf (rename)
$ mv /Documents/Apache.pdf /test/move_Apache.pdf
(filemove + rename)

Basic Linux commends - cp

- Copy files and directories from directory to directory
- File copy and rename action could have been achieved in one step

- $ cp original_filepath target_filepath

Ex)

$ cp /Documents/Apache.pdf /test/Apache.pdf

Basic Linux commends – sudo

- sudo is a command for linux operating systems that allows users to run programs with the security privileges
  - It originally stands for “superuser do”
  - The sudo command is required when performing actions that require root permissions

$ sudo Linux_command

Ex)

$ sudo passwd root (set password for root)

$ sudo su (change the user to root)

$ apt-get install “something”

Basic Linux commends - pwd

- pwd command (print working directory) write the full pathname of the current working directory to the standard output

```
$ pwd
```

Basic Linux commends – mkdir

- create new directories in the filesystem. You must provide the name of the new directory to mkdir

  $ mkdir new_folder_name

  Ex)

  $ mkdir ysson

  $ mkdir linux_system

Basic Linux commends – touch

- Create new file in the filesystem. You must provide the name of the new file name to touch

  $ touch new_file_name

  Ex)

  $ touch example.py

  $ touch syslab.c

Basic Linux commends – shutdown

- The shutdown command lets you shut down or reboot your Linux system.

  $ shutdown

  (You must have administrator privileges to run it.)

  Ex)

  $ sudo shutdown

Basic Linux commends – passwd

- Change the password for a user.
- Just type passwd to change your own password.

$ passwd

$ passwd username

(You can also change the password of another user account, but you must use sudo)

Ex)

$ sudo passwd mary

Basic Linux commends – ps

- ps (process status) command is used to provide information about the currently running processes, including their process identification numbers (PIDs)

$ ps

-e : print all processes

-f : show in full format (UID, PID, etc.)

Ex)

$ ps -ef

Basic Linux commends – kill

- Terminate a process from the command line

  $ kill PID

  Ex)

  $ ps -e | grep process_name //process_name's PID: 1692

  $ kill 1692

  (The grep filter searches a file for a particular pattern of characters, and displays all lines that contain that pattern)

Basic Linux commends – apt

- The apt command is a command-line tool, which works with Ubuntu's Advanced Packaging Tool (APT) performing such functions as installation of new software packages, upgrade of existing software packages, updating of the package list index, and even upgrading the entire Ubuntu system.

$ apt options

Ex)

$ sudo apt update

$ sudo apt upgrade

$ sudo apt install “something”

$ sudo apt remove “something”

Basic Linux commands – ssh (secure shell)

- Make a connection to a remote Linux computer and log into user account

  $ ssh user_account@server_domain or IP_address

  Ex)

  $ ssh linux@192.168.10.109

Basic Linux commands – tar

- Create an archive file that can contain many other files

  $ tar –options file_path

  - Common options
    - cvf : when compressing
    - xvf : when extracting

  Ex)

  $ tar -cvf foo.tar files

  $ tar -xvf foo.tar

Basic Linux commands – top

- top command is used to show the Linux processes. It provides a dynamic real-time view of the running system. Usually, this command shows the summary information of the system and the list of processes or threads which are currently managed by the Linux Kernel.
- First line shows you the time and how long your computer has been running
- Second line shows the number of tasks and their states: running, stopped, sleeping and zombie
- Third line shows CPU information
- Forth line shows the total amount of memory, and how much is free and used
- Fifth line shows the total amount of swap memory, and how much is free and used

- $ top

Basic Linux commands – uname

- uname (short for unix name) is a computer program in Unix and Unix-like computer operating systems that prints the name, version and other details about the current machine and the operating system running on it

- $ uname
  - Common options.
    - a : see everything
    - s : when extracting
    - r : see the kernel release
    - v : see the kernel version

  Ex)

  $ uname -r

Basic Linux commands – history

- History command lists the commands you have previously issued on the command line

  $ history

Basic Linux commands – less

- View files without opening an editor

  $ less file_path

  Ex)
  $ less core.c
  $ less example.py
  $ less pci.c

Basic Linux commands – man

- Displays the “manual pages” for a command

```
$ man linux_command
```

Ex)

```
$ man ps
$ man top
```

Useful tools – ctags

- **Why use**
  - Utility of making indexes to recognize and find where certain functions and variables
  - Make database(tags files) of tags of source code
  - Can move to where certain functions and variables are declare

- Install

  $ sudo apt install ctags

- Usage

  $ ctags -R

Useful tools – cscope

- **Why use**
  - cscope is a programming tool which works in console mode
  - used on very large projects to find source code, functions, declarations, definitions and regular expressions given a text string
- Install

  $ sudo apt install cscope
- Usage

  $ cscope

Useful tools – cscope + ctags

- **Finding vfs_read() function**
  - vfs_read() function → a function in the virtual file system
    1) Run cscope

```
root@test1:/home/syslab/ysson/linux-5.2.11# cscope
```

    2) Enter vfs_read

```
Find this C symbol:
Find this global definition:
Find functions called by this function:
Find functions calling this function:
Find this text string: vfs_read
Change this text string:
Find this egrep pattern:
Find this file:
Find files #including this file:
Find assignments to this symbol:
```

    3) Choose vfs_read which you want

Useful tools – cscope + ctags

- **Finding vfs_read() function**
  - vfs_read() function → a function in the virtual file system
    3) Choose vfs_read which you want

```
  File            Line
0 cache.c          225 static void v9fs_vfs_readpage_complete(struct page *page, void *data,
1 cache.c          252 v9fs_vfs_readpage_complete,
2 cache.c          290 v9fs_vfs_readpage_complete,
3 vfs_addr.c        76 * v9fs_vfs_readpage - read an entire page in from 9P
4 vfs_addr.c        83 static int v9fs_vfs_readpage(struct file *filp, struct page *page)
5 vfs_addr.c        89 * v9fs_vfs_readpages - read a set of pages from 9P
6 vfs_addr.c        98 static int v9fs_vfs_readpages(struct file *filp, struct address_space *mapping,
7 vfs_addr.c       326 .readpage = v9fs_vfs_readpage,
8 vfs_addr.c       327 .readpages = v9fs_vfs_readpages,
9 dax.c            1160 * validated via access_ok() in either vfs_read() or
a exec.c           1003 ssize_t res = vfs_read(file, (void __user *)addr, len, &pos);
b namei.c          4704 * vfs_readlink - copy symlink body into userspace buffer
c namei.c          4713 int vfs_readlink(struct dentry *dentry, char __user *buffer, int buflen)
d namei.c          4742 EXPORT_SYMBOL(vfs_readlink);
e nfs4xdr.c        3648 * XXX: By default, vfs_readlink() will truncate symlinks if they
f nfs4xdr.c        3650 * easy fix is: if vfs_readlink() precisely fills the buffer, assume
g read_write.c      421 ssize_t __vfs_read(struct file *file, char __user *buf, size_t count,
h read_write.c      440 result = vfs_read(file, (void __user *)buf, count, pos);
i read_write.c      446 ssize_t vfs_read(struct file *file, char __user *buf, size_t count, loff_t *pos)
j read_write.c      461 ret = __vfs_read(file, buf, count, pos);
k read_write.c      587 ret = vfs_read(f.file, buf, count, ppos);
l read_write.c      639 ret = vfs_read(f.file, buf, count, &pos);
m read_write.c      987 ssize_t vfs_readv(struct file *file, const struct iovec __user *vec,
n read_write.c     1034 ret = vfs_readv(f.file, vec, vlen, ppos, flags);
o read_write.c     1089 ret = vfs_readv(f.file, vec, vlen, &pos, flags);
p splice.c          359 res = vfs_readv(file, (const struct iovec __user *)vec, vlen, &pos, 0);
q stat.c            411 error = vfs_readlink(path.dentry, buf, bufsiz);
r xfs_ioctl.c       294 error = vfs_readlink(dentry, hreq->ohandle, olen);
s fs.h             1885 extern ssize_t __vfs_read(struct file *, char __user *, size_t, loff_t *);
t fs.h             1886 extern ssize_t vfs_read(struct file *, char __user *, size_t, loff_t *);
u fs.h             1888 extern ssize_t vfs_readv(struct file *, const struct iovec __user *,
v fs.h             3222 extern int vfs_readlink(struct dentry *, char __user *, int);
w sysctl_binary.c   923 result = vfs_read(file, oldval, oldlen, &pos);
x iint.c            200 ret = __vfs_read(file, buf, count, &offset);
```

Useful tools – cscope + ctags

- **Finding vfs_read() function**
  - vfs_read() function → a function in the virtual file system
    4) Enter “ Ctrl +] ” on vfs_read function

```
ssize_t ksys_pread64(unsigned int fd, char __user *buf, size_t count,
                     loff_t pos)
{
        struct fd f;
        ssize_t ret = -EBADF;

        if (pos < 0)
                return -EINVAL;

        f = fdget(fd);
        if (f.file) {
                ret = -ESPIPE;
                if (f.file->f_mode & FMODE_PREAD)
                        ret = vfs_read(f.file, buf, count, &pos);
                fdput(f);
        }

        return ret;
}
```

Useful tools – cscope + ctags

- **Finding vfs_read() function**
  - vfs_read() function → a function in the virtual file system
    5) You can find body of vfs_read() function

```
ssize_t vfs_read(struct file *file, char __user *buf, size_t count, loff_t *pos)
{
        ssize_t ret;

        if (!(file->f_mode & FMODE_READ))
                return -EBADF;
        if (!(file->f_mode & FMODE_CAN_READ))
                return -EINVAL;
        if (unlikely(!access_ok(buf, count)))
                return -EFAULT;

        ret = rw_verify_area(READ, file, pos, count);
        if (!ret) {
                if (count > MAX_RW_COUNT)
                        count =  MAX_RW_COUNT;
                ret = __vfs_read(file, buf, count, pos);
                if (ret > 0) {
                        fsnotify_access(file);
                        add_rchar(current, ret);
                }
                inc_syscr(current);
        }

        return ret;
}
```

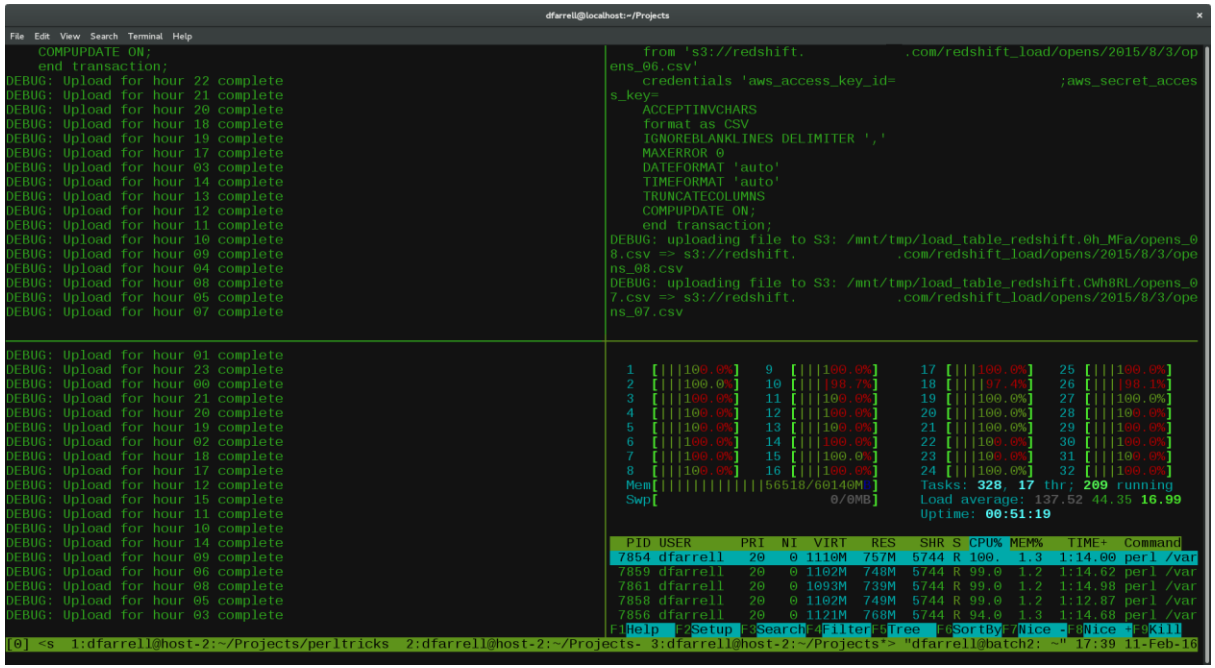
Useful tools – cscope + ctags

- **Finding vfs_read() function**
  - vfs_read() function → a function in the virtual file system
    5) Enter “ Ctrl + t ” → you can back the previous function

```
ssize_t ksys_pread64(unsigned int fd, char __user *buf, size_t count,
                     loff_t pos)
{
        struct fd f;
        ssize_t ret = -EBADF;

        if (pos < 0)
                return -EINVAL;

        f = fdget(fd);
        if (f.file) {
                ret = -ESPIPE;
                if (f.file->f_mode & FMODE_PREAD)
                        ret = vfs_read(f.file, buf, count, &pos);
                fdput(f);
        }

        return ret;
}
```

Tmux

- **Tmux is a tool to manage virtual concoles**
  - It allows multiple terminal sessions to be accessed simultaneously in a single window
  - It is useful for running more than one command-line program at the same time



Tmux

- **Essential tmux commands**
  - tmux: start a new tmux session
  - ctrl-b + %: split a pane vertically
  - ctrl-b + “ : split a pane horizontally
  - ctrl-b + o : move to the next pane
  - ctrl-b + z : zoom (or unzoom) a pane
  - ctrl-b + c: create a new window
  - ctrl-b + N: go to window N (0~9)
  - ctrl-b + d: detach from a session
  - tmux a: attach to an existing session