Lab 5: **The Linux Labs – Standard Linux Commands**

CSI4103 – Web Application Software Design

Faculty of Engineering – University of Ottawa

Objective:

Understand and practice how to use linux command line instructions. This is useful when working on server backends through a terminal window (e.g. using telnet or an SSH connection). This part of the lab continues on with a review of basic linux commands that are needed when navigating and using the file system from a linux command line

Instructions:

Linux files are organized within a hierarchical directory structure. We will work with two entities that exist within a directory structure, files and directories.

Each file and each directory is accessed via a path. That path can be expressed in two ways, as an absolute path and as a relative path. When working with files, Linux does not know where the files are located; you need to tell the operating system, the EXACT LOCATION of the file using either an absolute or relative path.

Most of the Linux commands will follow the following standard structure:

- `command –options –other_options argument(arguments…)`

  Example: `ls -la --color=never /etc/`
  where `ls` is the command, `-la` are two options, `--color=never` is also an option and `/etc` is an argument

To explore the directory structure, the following commands will be used:

- `cp` – copy files
- `mv` – move or rename files
- `rm` – remove or delete files
- `cat` – concatenate files. Used to display file contents
- `passwd` – to change a password
- `find` – to find files
- `man/info` - to display help
- `sleep` – to pause or wait
- `touch` – to change the timedate stamp of a file. Used to create an empty file.
- `clear` – to clear the screen

touch and cp

The **touch** command updates different timedate stamps. It is also frequently used to create empty files.

The **cp** command makes a copy of an existing set of files or directories (remember that directories are just a type of file) into another area of the file system. The syntax for the cp command is:

- **`cp [-options]` *`source destination`***
- Where source is often a directory and file specification and destination is often just a directory

The mkdir command

1. *touch clock*
2. *ls –l clock*

   -rw-rw-r-- 1 ubuntu ubuntu 0 Feb 7 17:08 clock

   Record the timedate stamp: _____

3. **sleep 61*

   Command '*sleep' not found, did you mean: command 'sleep' form deb coreutils (8.32-4.1ubuntu1))

   What does this command do? _____

4. *touch clock*
5. *ls –l clock*

   -rw-rw-r-- 1 ubuntu 0 Feb 7 17:11 clock

   Record the timedate stamp: _____

Copying files to a directory

a. **cd**

b. **touch f1 f2 f3**

c. **ls**

What does this show? the current directory with the three new empty files added "f1", "f2", and "f3" added

d. **mkdir lab**

e. **ls**

What does this show? the current directory with the new directory "lab" added to it

f. **cp f1 f2 f3 lab**

g. **ls lab**

What does this show? the contents of the lab folder with the newly copied empty files f1, f2, and f3

h. **mkdir coffee**

i. **cd coffee**

j. **touch cream sugar**

k. **cd ..**

l. **cp coffee/cream coffee/sugar lab**

m. **ls –l lab**

Have the files been copied? Yes the files have been copied

2. **<u>Copying directories to a directory (-r option) recursive</u>**

a. **mkdir dir1 dir2 dir3**

Record the command used to verify the directories have been created: ___

b. **cp dir1 dir2 dir3 lab**

Record one of the messages displayed: cp: -r not specified; omitting directory 'dir1'

c. **ls lab**

Have the directories been copied? No the directories have not.

d. cp –r dir1 dir2 dir3 lab

Record one of the messages displayed: None.

e. ls lab

Have the directories been copied? Yes the directories have.

3. Copying directories to a directory (-r & --parents options)

a. mkdir parent/child

Record one of the messages displayed: mkdir: cannot create directory 'parent/child': No suh file or directory

b. mkdir –p parent/child

4. cp –r --parents parent/child lab?????

List the contents of the lab directory:

cream, dir1, dir2, dir3, f1, f2, f3, parent, sugar

5. Now try the *verbose* option of the copy command:

`cp –v source destination`

What is different? After moving the file "f1" into dir1, the console returned the message 'f1' -> 'dir1/f1'

mv and rm

The **mv** command moves or renames files. Recall that directories are just a type of file. The syntax for the **mv** command is:

- `mv` *`source destination`*

1. **Renaming files**
    a. **cd ~/lab**
    b. **ls**
    c. **mv f1 m1**
    d. **ls**
    Has the file been renamed? Yes, the file once named 'f1' is now 'm1'

2. **Moving files**
    a. **touch red green blue**
    b. **mkdir colours**
    c. **mv red green blue**
    Record the error message: mv: target 'blue' is not a directory
    d. **mv red green blue colours**
    e. **ls**
    f. **ls colours**
    What have you observed? The files 'red', 'green', and 'blue' have been added to the folder 'colors' (and I only just noticed files are listed in alphabetical order)

3. Try exercises 1 and 2 again, this time using mv with the **–i** option
    What does this option do? -i for command mv does nothing, but for other commands an overwrite prompt is presented

4. Try exercises 1 and 2 again, this time using mv with the **–u** option
    What does this option do? -u for command mv does nothing, but for other commands it moves the files only when the source file is newer than tl

5. Try exercises 1 and 2 again, this time using mv with the **–b** option
    What does this option do? -b for command mv does nothing,

6.
    a. **mkdir toddlers children sandbox**
    b. **mv toddlers children sandbox**
    c. **ls**
    Are there toddlers and children in the sandbox? Yes there are directories 'toddlers' and 'children' in the directory '

7. Try Exercise 6 again with the **–v** option

What does this option do? This option renamed each directory to its now parent path (e.g., 'toddlers' -> 'sandbox/toddlers')

The **rm** or remove file command allows you to delete the *content* of any directory. It is both a dangerous and useful command because of it flexibility. Unlike DOS or Windows, a file deleted in Linux is gone. (Is this always true? What do you think can prevent this from happening? When people have a problem, they usually invent a solution.) The syntax for the **rm** command is:

- `rm file_specification`

1. **Removing files**
   a. **cd**
   b. **cd lab/sandbox**
   c. **touch child1 child2 child3**
   d. **ls**

      Are there children in the sandbox? Yes, files 'child1', 'child2', and 'child2' are in the directory 'sandbox'

   e. **rm child1 child2**

      Are there children in the sandbox? file 'child3' is still in the directory 'sandbox', but files 'child1' and 'child2' have been removed

      How do you know? Used command 'ls' to list the directory and files 'child1' and 'child2' are missing, but 'child3' is still there

   f. **rm child3**

      Are there children in the sandbox? No, there are no more files with the prefix 'child' in the directory 'sandbox'

      What command did you enter to confirm your answer? ls

   g. **cd ..**
   h. **rmdir sandbox**
   i. **cd**
   j. **rmdir lab**

      Record the error message: i.) rmdir: failed to remove 'sandbox': Directory not empty, j.) rmdir: failed to remove 'lab': Directory not empty

   k. **rm –r lab**

      Has the directory been deleted? Yes, the directory 'lab' has been deleted

      How do you know? Using command 'ls' in home directory and directory 'lab' is no longer listed

cat

Cat is a utility which will concatenate files and print on the standard output. It's power actually lies in the fact that is uses standard input and standard output which can be redirected and with pipes. It is often used to display small files to the screen (which is standard output). The syntax of the cat command is:

- `cat [options] [file]`

`clear`

- The clear command simply clears the terminal window of output. Try it! Remember there are still ways to review the previous output. You can use SHIFT-PG UP / SHIFT-PG DOWN or if you are in a graphical environment, you would normally have a scroll bar available.

Viewing files with cat.

a. **cat /etc/hosts**
b. **cat /etc/fstab**
c. **cat /etc/inittab**
d. **cat /var/log/messages**
e. **cat /var/log/messages | more**
f. **cat /var/log/messages | less**

passwd

The `passwd` command will change your password if you enter it without a username argument. If you are root or superuser, you can change any user's password.

If you are root, you can also use any password without regard to any password policies which may otherwise restrict certain password choices.

If you are a regular user trying to change your own password, the default password policies should prevent you from choosing trivial passwords.

1. **passwd**
   a. Record the message here: New password: _____
2. Enter the following password: **sky**
   a. Record the message here: BAD PASSWORD: The password is shorter than 8 characters. New password: _____
3. **passwd**
   a. Enter your current UNIX password:
   b. Try to change your password to linux
   c. Were you successful? Explain: No. I installed Linux using VirtualBox on the lab computer, and I read that the default password was "cha _____
   Both times I received a message saying "BAD PASSWORD: The password is shorter than 8 characters" and after three attempts I received the message "passwd: Have exhausted the r _____
4. **passwd**
   a. Try to change your password to 56?1Lin
   b. Were you successful? Explain: No. It returned the message "56?Lin: command not found" _____
   _____
5. You can now close the terminal window used for this exercise and open a new one to continue

Executing commands that are in the search path

1. **`echo $PATH`**
   - Record the path:
     /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin:/snap/bin
   - What is this?
     Shows which directories your shell is set to check for executable files.
2. **`whereis ls`**
   - Record the output:
     ls: /usr/bin/ls /usr/share/man/man1/ls.1.gz
   - What is the purpose of the whereis command?
     Used to find the location of source/binary file of a command and manuals sections for a specified file in Linux system.
3. **`which ls`**
   - Record the output?
     /usr/bin/ls
   - What is the purpose of the which command?
     The which command allows users to search the list of paths in the $PATH environment variable and outputs the full path of the command specified as an argument.
4. Describe the circumstances you would use these commands?
5. What are the differences between the **`whereis`** and **`which`** commands?
   whereis - locates binary, source, and manual page files for a command
   which - shows the full path of executables
6. **`ls`**
   - Did the command work? Why? Yes, because there are both binary/source/manual page files and executables to list the path for
7. **`whereis runlevel`**
   - Record the output: runlevel: /usr/sbin/runlevel /usr/share/man/man8/runlevel.8.gz
8. **`runlevel`**
   - Record the output: N 5
   - Did the command work? Why? Yes, because the machine is in a current state and it prints that state
9. **`/sbin/runlevel`**
   - Record the output: N 5
   - Did the command work? Why? Yes, because it has a current and previous runlevel for this operating system
   -

Executing commands that are not in the search path.

- **`echo $PATH`**
  - Record the output: /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin:/snap/bin
- **`PATH=`**
- **`echo $PATH`**
  - Record the output: Empty.
- **`ls`**
  - Did the command work? Why? No, it returned bash ls: No such file or directory. This is because the path has been set to itself.
- **`/bin/ls`**
  - Did the command work? Why? Yes, it returned the list of directories because bin displays the binary files.

10. **`exit`**
    - To log out. This will "fix" your path.
11. Log back in (or su – if appropriate)

- **`echo $PATH`**
  - Record the output? /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin/:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin:/snap/bin
- **`mkdir bin`**
- **`cp /bin/pwd  mypwd`**
  - What did the command do? Seems to have made a copy of the directory '/bin/pwd' into directory 'mypwd'
- **`mypwd`**
  - Did the command work? Why? No, it returned 'mypwd: command not found.' Because a
- **`./mypwd`**
  - Did the command work? Why? Yes, it returned '/home/ubuntu', because
- **`cp /bin/pwd  bin/mybinpwd`**
- **`mybinpwd`**
  - Did the command work? Why? _____
- _____
- **`rm  mypwd  bin/mybinpwd`**
- **`whoami`**
  - What is your current login id? _____
  - Record your current prompt

- **`id`**
  - Record the output: _____
- **`echo $PATH`**
  - Record the output: _____
- **`su -`**
  - Enter the root password
- **`whoami`**
  - Record the output: _____
- **`id`**
  - Record the output: _____
- **`echo $PATH`**
  - Record the output: _____
- **`runlevel`**
  - Did the command work? _____
- **`exit`**
- **`whoami`**
- **`id`**
  - What is your current login id? _____

Review

This exercise assumes that the commands listed below are executed in a **regular** user's home directory. If you want to keep things neat, create a new user just for the purpose of performing this review. Keep notes so that you can answer the questions at the end of the exercise.

Not all commands will be successful! Make SURE that you have entered everything correctly!

```
mkdir   ~/labex

cd  labex

mkdir  ./orchard

touch  apple  orange

mv  orange  orchard/lemon

cat  orange

touch  lettuce  tomato  cucumber

cp tomato  lettuce  garden

mkdir  jardin  forest

mv lettuce  cucumber  jardin

rmdir  garden

cd  ..

touch  test

cd  orchard

cd  labex/forest

mv  tomato  forest
```

1. How many **directories** are **created** during the review exercise? _____
    a. List them: _____

    _____
2. How many **directories** are **deleted** during the review exercise? _____
    a. List them: _____

    _____

3. How many **regular files** remain in the directory `labex`?
   Do not include subdirectories. _____

   a. List them: _____

   _____

4. How many **regular files** remain in the directory **orchard**?
   Do not include subdirectories. _____

   a. List them: _____

   _____

5. How many **tomato** files remain in the **labex** directory and all its **subdirectories**? _____

   a. List the directories: _____

   _____

6. How many **unsuccessful** attempts creating (using **touch** or **cp**) or removing **regular files**? _____

   a. List the errors: _____

   _____

7. How many unsuccessful attempts creating and removing **directories**? _____

   a. List the errors _____

   _____

8. What is the **current** directory at the end of the exercise?

   _____

1. How do you know? What command did you enter to find out?

   _____