Advanced Linux Commands

ENC Isungga

September 1, 2023

Advanced Linux Commands

Learning Outcomes

At the end of this laboratory session, the students should be able to:

1. explain how the bash shell works; and
2. execute shell commands.

Bourne Again SHell (bash)

Users are given access to the services of an operating system through a user interface. Types of user interface include command line interface (CLI), graphical user interface (GUI), touchscreen interface, gesture-based interface, and speech-based interfaces. Power users, such as system administrators, prefer to use the CLI because it provides greater flexibility and control. In addition, when managing servers remotely, CLI is the best user interface to use. In Linux, CLI is implemented as *shells*.

A shell is basically a program that accepts and executes commands. It is both a command interpreter and a programming language. The default shell in the Linux distribution is called the *Bourne Again SHell* (bash). Other shells that can be used are csh, tcsh, zsh, ksh. You can even write your own shell.

System Programs

Most commands are **system programs**. Compared to application programs which are user-oriented, say a word processor, system programs perform lower level operations such as disk formatting. The following is a broad category of system programs with example commands:

1. *Disk/File/Filesystem manipulation* (`mkdir`, `cat`, `dd`, `cp`, `mv`, `chmod`, `chown`, `mkfs.msdos`)
2. *Status information*, sometimes stored in a file (`free`, `top`, `/proc/cpuinfo`, `df`)
3. *Programming language support* (`objdump`, `gas`, `nasm`, `gcc`, `hexdump`)
4. *Program loading and execution* (`ld`, `gdb`, `bash`)
5. *Communications* (`ftp`, `ssh`, `curl`, `wget`, `talk`, `lynx`)
6. *Background services* (`cron`, `syslog`, `dmesg`)
7. *Application programs* (`nano`, `vim`, `cal`)

Shell Commands

The shell accepts commands when a user types the command line on a prompt (`$`) then presses [`Enter`]. It processes the command line following its syntax and semantic rules (similar to a programming

language). An integer value exit status is then returned after execution which can later be inspected. The shell can accept *simple commands*, *pipelines*, *lists*, or *compound commands*.

Simple Commands

A simple command begins with the *command name*, followed by its *argument/s*. Some of the commonly used commands are shown below:

Command	Description	Examples
`man <command>`	Most commands are shipped with their reference manuals. This is used to view it.	`man ls# use 'q'to quit ' man`
`ls`	This command is used to list all files and directories in the current/specified directory.	`lsls -l /etc`
`cp <src> <dest>`	This command is used to copy a file or directory to the same or different directory.	`cp file.txt Downloads/ filecopy.txtcp -r Downloads/ dloads/`
`mv <src> <dest>`	This command is used to move or rename files and directories.	`mv file.txt filecopy.txt mv filecopy.txt dloads/`
`rm <file/dir>`	This command is used to delete files and directories.	`rm dloads/filecopy.txtrm -rf dloads/`

Pipeline

A pipeline is a sequence of one or more commands separated by '|' and '|&'. The commands are connected via pipe where every output of a previous command, becomes an input to the next.

Examples

```
$ ls -l # list in long format
$ ls -l | tail -n 5 # list only the last 5
$ ls -l | tail -n 5 | sort -r # list in reversed order
```

Command List

A command list is a sequence of one or more pipelines separated by ';', '&', '&&', or '||', and optionally terminated by one of ';', '&', or a newline. The meaning of the symbols are given below:

Symbol	Description
&	run in the background
;	execute commands sequentially
&&	previous command must have an exit status of 0 (means success) before this command is executed
||	previous command must have a non-zero exit status (means failure) before this command is executed

Example

```
$ sudo apt-get update && sudo apt-get upgrade -y && sudo apt autoremove
    -y
```

The example command above performs an upgrade only if the update operation succeeded. It also removes unused packages only when the upgrade was successful.

Compound Command

A compound command provides scripting capabilities to the shell through looping and conditional constructs.

Examples

```
$ for ((i=10;i>=0;i-=2)); do echo "$i"; done
$ for i in /bin/*; do file $i; done
```

Additional Features

Shell Expansion

Some characters in the command line can be used as shorthands but are expanded by the shell to their actual meanings. There are several types of expansions that can be performed:

Expansion	Description	Example
Brace expansion	Expansion of expression into multiple text strings from a pattern containing braces	`$ echo a{b,c,d,e}f`
Tilde expansion	Expansion into the name of the home directory of the current or specified user	`$ echo ~`
Parameter expansion	Expansion of parameter values	`$ echo "${USER}"`
Command substitution	Allows the use of the output of a command as an expansion	`$ ls -l $(which cp)`
Arithmetic expansion	Allows arithmetic operations to be performed as an expansion	`$ echo $(($((2**3))+5))`
Filename expansion	Allows the use of wildcards as an expansion	`$ echo /usr/*/share`

I/O Redirection

An input or output of a command line can be redirected to different streams.

Symbol	Description
>	Redirect output of a command into a new file. If the file already exists, overwrite it.
»	Redirect the output of a command onto the end of an existing file.
<	Redirect a file as input to a program

Examples:

```
1 $ ls > file.txt
2 $ echo "ICS 536-1234" >> file.txt
3 $ more < file.txt
```

Quotes

The purpose of quotes is to prevent the interpretation of certain characters or words on the command line.

Character	Description
\ (backslash)	escape character that preserves the literal value of the character next to it
' (single quote)	preserves the literal value of each character inside
" (double quotes)	preserves the literal value of each character inside except $, \, ‘

Examples:

```
$ mkdir "${USER} folder"
$ cd $USER\ folder/
```

Comments

Any line that starts with ‘#’ will cause all characters on that line to be ignored.

Learning Experiences

Try this introductory activity: http://web.mit.edu/mprat/Public/web/Terminus/Web/main.html.

Assessment Tool

A four-part shell command activity sheet for exploring other shell commands will be given.

References

[1] https://www.gnu.org/software/bash
[2] https://www.gnu.org/software/bash/manual/html_node/Shell-Operation.html#Shell-Operation
[3] https://www.gnu.org/software/bash/manual/html_node/Shell-Expansions.html#Shell-Expansions
[4] http://linuxcommand.sourceforge.net/lc3_lts0080.php
[5] http://www.mprat.org/Terminus/

Acknowledgment

This material builds on top of the contributions by former CMSC 125 instructors: Joman Encinas, Chris Templado, Betel de Robles, Zenith Arnejo, Berna Pelaez

License