CL-2006Operating System	LAB- 01 Introduction to Linux Operating System and its basic commands

NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES

Fall 2024

Contents:

Introduction:

Lab#01 Introduction to Linux operating System and its basic commands

Linux is an open source operating system (OS). An operating system is the software that directly manages a system's hardware and resources, like CPU, memory, and storage. The OS sits between applications and hardware and makes the connections between all of your software and the physical resources that do the work.

Linux Vs Windows:

Windows is widely used operating system while Linux has its own advantages. Following are the advantages discussed in the table below.

LINUX	WINDOWS
Linux is an open source operating system i.e. user can change source code as per requirement	Windows OS is a commercial operating system and its closed source i.e. its source code is inaccessible.
Linux has Monolithic kernel i.e. Whole operating system works in the kernel space.	Windows has a hybrid kernel i.e. Combination of microkernel and monolithic kernel.
File names are case-sensitive in Linux.	File names are not case-sensitive in Windows.
Booting can be done from any disk.	Booting can only be done from the prime disk.
Linux is highly reliable and secure. It has a deeprooted emphasis on process management, system security, and uptime.	Windows is less reliable than Linux. Over the recent years, Windows reliability has been improved a lot. However, it still has some system instabilities and security weaknesses because of its oversimplified design.

Linux Components:

Major components of Linux are:

- Kernel
- System user space
- Applications **Kernel:**

Kernel is the base component of the OS. Without it, the OS doesn't work. The kernel manages the system's resources and communicates with the hardware. It's responsible for memory, process, and file management.

System user space:

System user space is the administrative layer for system-level tasks like configuration and software installation. This includes the shell or command line, processes that run in the background, and the desktop environment.

Applications:

A type of software that lets you perform a task. Applications include everything from desktop tools and programming languages to multiuser business suites. Most Linux distributions offer a central database to search for and download additional apps.

Linux Shell:

Shell is a program that receives commands from the user and gives it to the OS to process, and it shows the output. Shell is command line interface (CLI), a console that interacts with the system via texts and processes. It's similar to the Command Prompt application in Windows.

Linux Command:

A Linux command is a program or utility that runs on shell. Linux commands are executed on Terminal by pressing 'Enter' at the end of the line. User can run commands to perform various tasks, from package installation to user management and file manipulation.

Linux command's general syntax looks like:

CommandName [option(s)] [parameter(s)]

A command may contain an option or a parameter. In some cases, it can still run without them. These are the three most common parts of a command:

- **CommandName** is the rule that you want to perform.
- **Option** or **flag** modifies a command's operation. To invoke it, use hyphens (–) or double hyphens (—).

Parameter or **argument** specifies any necessary information for the command.

Basic Linux Commands:

From here the reader is exposed to the basic Linux commands. All the commands have to be tried in the terminal. Throughout the lab manuals Ubuntu will be used for explaining the concepts.

NOTE: All Linux commands are case sensitive i.e. 'cp' is not equivalent to 'CP'. Also, all the files and directories in linux are case sensitive so for example '/etc/hosts' is not equivalent to '/etc/Hosts' and so hosts and Hosts are two different files in the same directory. When executing multiple commands in a single line, use **;** to separate them.

1. Terminal Related Commands:

Following are the terminal related commands:

1. man
2. clear
3. exit
4. history
5. echo
6. alias, unalias
7. |
8. &&

1. man command

The man command provides a user manual of any commands or utilities you can run in Terminal, including the name, description, and options.

It consists of nine sections:

1. Executable programs or shell commands
2. System calls
3. Library calls
4. Games
5. Special files
6. File formats and conventions
7. System administration commands
8. Kernel routines
9. Miscellaneous

To display the complete manual, enter:

man man[command_name]

For example, you want to access the manual for the ls command:

man ls

Enter this command if you want to specify the displayed section: **man**

[option] [section_number] [command_name]

For instance, you want to see section 2 of the ls command manual

man man 2 ls

2. clear command

Enter the **clear** command to clean the Terminal screen.

3. **exit command**

Enter the **exit** command to exit the terminal.

4. **history command**

With history, the system will list up to 500 previously executed commands, allowing you to reuse them without re-entering. Keep in mind that only users with sudo privileges can execute this command. How this utility runs also depends on which Linux shell you use. To run it, enter the command below:

history [option]

This command supports many options, such as:

-c clears the complete history list.

-d offset deletes the history entry at the OFFSET position.

-a appends history lines.

5. **echo command**

The echo command is a built-in utility that displays a line of text or string using the standard output. Here's the basic syntax: **echo**

[option] [string]

For example, you can display the text Hostinger Tutorials by entering:

echo "Hostinger Tutorials"

This command supports many options, such as:

-n displays the output without the trailing newline.

-e enables the interpretation of the following backslash escapes:

\a plays sound alert.

\b removes spaces in between a text.

\c produces no further output.

-E displays the default option and disables the interpretation of backslash escapes.

6. **alias, unalias commands** alias allows you to create a shortcut with the same functionality as a command, file name, or text.

When executed, it instructs the shell to replace one string with another.

To use the alias command, enter this syntax:

alias Name=String

For example, you want to make e the alias for the exit command:

alias e=exit

On the other hand, the unalias command deletes an existing alias.

Here's what the general syntax looks like:

unalias [alias_name]

7. **|**

If a user in Linux likes to combine two or more commands, pipes is the option. Pipe is represented by the symbol '**|**'

8. **&&**

And command **&&** is used to only allow the next command to run if the previous one is successful.

2. System Related Commands:

Commands related to system includes some basic commands and commands for managing users and groups in Linux that require root access.

Basic System Related Commands:

Following are the basic system related commands:

1. date
2. shutdown, poweroff , reboot
3. ps
4. kill
5. df
6. du

7. top
8. htop
9. uname

1. **date command**
   The date command will display the system date.

2. **shutdown, poweroff and reboot commands:**

Command	Switch	Description	Example	Output
shutdown	None	Power off the computer	shutdown now	The system will shutdown now for maintenance
	-r	Reboots after shutdown	shutdown –r now	None
poweroff	none	Shutdowns computer	sudo poweroff	None
reboot	none	Restarts computer	sudo reboot	None

3. **ps command**

The process status or ps command produces a snapshot of all running processes in your system. The static results are taken from the virtual files in the /proc file system.

Executing the ps command without an option or argument will list the running processes in the shell along with:

- The unique process ID (PID)
- The type of the terminal (TTY)
- The running time (TIME)
- The command that launches the process (CMD) Here are some acceptable options you

can use:

-T displays all processes associated with the current shell session.

-u username lists processes associated with a specific user.

-A or -e shows all the running processes.

4. kill command

Use the kill command to terminate an unresponsive program manually. It will signal misbehaving applications and instruct them to close their processes.

To kill a program, you must know its process identification number (PID). If you don't know the PID, run the following command:

ps ux

After knowing what signal to use and the program's PID, enter the following syntax:

kill [signal_option] pid

There are 64 signals that you can use, but these two are among the most commonly used:

SIGTERM requests a program to stop running and gives it some time to save all of its progress. The system will use this by default if you don't specify the signal when entering the kill command.

SIGKILL forces programs to stop, and you will lose unsaved progress.

For example, the program's PID is 63773, and you want to force it to stop:

kill SIGKILL 63773

5. df command

Use the df command to report the system's disk space usage, shown in percentage and kilobyte (KB).

Here's the general syntax:

df [options] [file]

For example, enter the following command if you want to see the current directory's system disk space usage in a human-readable format:

df -h

These are some acceptable options to use:

df -m displays information on the file system usage in MBs.

df -k displays file system usage in KBs. **df -T** shows the file

system type in a new column.

6. du command

If you want to check how much space a file or a directory takes up, use the du command. You can run this command to identify which part of the system uses the storage excessively.

Remember, you must specify the directory path when using the du command. For example, to check /home/user/Documents enter: **du /home/user/Documents**

Adding a flag to the du command will modify the operation, such as:

-s offers the total size of a specified folder.

-m provides folder and file information in MB **-**

k displays information in KB.

-h informs the last modification date of the displayed folders and files.

7. top command

The top command in Linux Terminal will display all the running processes and a dynamic real-time view of the current system. It sums up the resource utilization, from CPU to memory usage.

The top command can also help you identify and terminate a process that may use too many system resources.

8. htop command

The htop command is an interactive program that monitors system resources and server processes in real time. It is available on most Linux distributions, and you can install it using the default package manager.

Compared to the top command, htop has many improvements and additional features, such as mouse operation and visual indicators. To use it, run the following command:

htop [options]

You can also add options, such as:

-d or –delay shows the delay between updates in tenths of seconds.

-C or –no-color enables the monochrome mode.

-h or –help displays the help message and exit.

To run the command, simply enter **top** into the CLI.

9. uname command

The uname or unix name command will print detailed information about your Linux system and hardware. This includes the machine name, operating system, and kernel. To run this command, simply enter uname into your CLI.

Here's the basic syntax:

uname [option]

These are the acceptable options to use:

-a prints all the system information.

-s prints the kernel name.

-n prints the system's node hostname.

Managing users and groups in Linux:

For root privilege, sudo command is used.

sudo command

Short for superuser do, sudo is one of the most popular basic Linux commands that lets you perform tasks that require administrative or root permissions.

When using sudo, the system will prompt users to authenticate themselves with a password. Then, the Linux system will log a timestamp as a tracker. By default, every root user can run sudo commands for 15 minutes/session.

If you try to run sudo in the command line without authenticating yourself, the system will log the activity as a security event.

Here's the general syntax:

sudo (command)

You can also add an option, such as:

-k or –reset-timestamp invalidates the timestamp file.

-g or –group=group runs commands as a specified group name or ID.

-h or –host=host runs commands on the host.

3. Network Related Commands:

Linux terminal provides commands for interfacing with networks, some of the basic commands are: 1.

ping

2. wget
3. hostname
4. ip

1. **ping command**

The ping command is one of the most used basic Linux commands for checking whether a network or a server is reachable. In addition, it is used to troubleshoot various connectivity issues.

Here's the general format:

ping [option] [hostname_or_IP_address]

For example, you want to know whether you can connect to Google and measure its response time:

ping google.com

2. **wget command**

The Linux command line lets you download files from the internet using the wget command. It works in the background without hindering other running processes.

The wget command retrieves files using HTTP, HTTPS, and FTP protocols. It can perform recursive downloads, which transfer website parts by following directory structures and links, creating local versions of the web pages.

To use it, enter the following command:

wget [option] [url]

For example, enter the following command to download the latest version of WordPress:

wget https://wordpress.org/latest.zip

3. **hostname command**
   Run the hostname command to know the system's hostname. You can execute it with or without an option. Here's the general syntax:

hostname [option]

There are many optional flags to use, including:

-a or –alias displays the hostname's alias.

-A or –all-fqdns displays the machine's Fully Qualified Domain Name (FQDN).

-i or –ip-address displays the machine's IP address.

For example, enter the following command to know your computer's IP address:

hostname –i

4. ip command

The ip command is a Linux net-tool for system and network administrators. IP stands for Internet Protocol and as the name suggests, the tool is used for configuring network interfaces.

Older Linux distributions used the ifconfig command, which operates similarly. However, ifconfig has a limited range of capabilities compared to the ip command.

To Use the ip command enter:

ip [OPTION] OBJECT {COMMAND | help}

OBJECTS (or subcommands) that you will use most often include:

1. link (**l**) – used to display and modify network interfaces.

2. address (**addr/a**) – used to display and modify protocol addresses (IP, IPv6).

3. route (**r**) – used to display and alter the routing table.

4. neigh (**n**) – used to display and manipulate neighbor objects (ARP table).

4. Managing Files & Directories:

Basic Commands for Files & Directory:

- **pwd command**

Use the pwd command to find the path of your current working directory. Simply entering pwd will return the full current path – a path of all the directories that starts with a forward slash (/). For example, /home/username.

The pwd command uses the following syntax: **pwd**

[option]

It has two acceptable options:

-L or –logical prints environment variable content, including symbolic links.

-P or –physical prints the actual path of the current directory.

- **cd command**

To navigate through the Linux files and directories, use the cd command. Depending on your current working directory, it requires either the full path or the directory name.

Running this command without an option will take you to the home folder. Keep in mind that only users with sudo privileges can execute it.

Let's say you're in /home/username/Documents and want to go to Photos, a subdirectory of Documents. To do so, enter the following command:

cd Photos

If you want to switch to a completely new directory, for example, /home/username/Movies, you have to enter cd followed by the directory's absolute path: **cd /home/username/Movies**

Here are some shortcuts to help you navigate:

cd ~[username] goes to another user's home directory.

cd .. moves one directory up. **cd-** moves to your

previous directory.

- **ls command**

The ls command lists files and directories within a system. Running it without a flag or parameter will show the current working directory's content

To see other directories' content, type ls followed by the desired path. For example, to view files in the Documents folder, enter:

ls /home/username/Documents

Here are some options you can use with the ls command:

ls -R lists all the files in the subdirectories. **ls -a** shows

hidden files in addition to the visible ones.

ls -lh shows the file sizes in easily readable formats, such as MB, GB, and TB.

- **mkdir command**

Use the mkdir command to create one or multiple directories at once and set permissions for each of them. The user executing this command must have the privilege to make a new folder in the parent directory, or they may receive a permission denied error.

Here's the basic syntax:

mkdir [option] directory_name

For example, you want to create a directory called Music:

mkdir Music

To make a new directory called Songs inside Music, use this command:

mkdir Music/Songs

The mkdir command accepts many options, such as:

-p or –parents create a directory between two existing folders. For example, mkdir -p Music/2020/Songs will make the new "2020" directory.

-m sets the file permissions. For instance, to create a directory with full read, write, and execute permissions for all users, enter mkdir -m777 directory_name.

-v prints a message for each created directory.

- **rmdir command**

To permanently delete an empty directory, use the rmdir command. Remember that the user running this command should have sudo privileges in the parent directory.

For example, you want to remove an empty subdirectory named personal1 and its main folder mydir:

rmdir -p mydir/personal1

- **cp command**

Use the cp command to copy files or directories and their content. Take a look at the following use cases.

To copy one file from the current directory to another, enter cp followed by the file name and the destination directory. For example:

cp filename.txt /home/username/Documents

To copy files to a directory, enter the file names followed by the destination directory:

cp filename1.txt filename2.txt filename3.txt /home/username/Documents

To copy the content of a file to a new file in the same directory, enter cp followed by the source file and the destination file:

cp filename1.txt filename2.txt

To copy an entire directory, pass the -R flag before typing the source directory, followed by the destination directory:

cp -R /home/username/Documents /home/username/Documents_backup

- **mv command**

The primary use of the mv command is to move and rename files and directories. Additionally, it doesn't produce an output upon execution.

Simply type mv followed by the filename and the destination directory. For example, you want to move filename.txt to the /home/username/Documents directory: **mv filename.txt /home/username/Documents.** You can also use the mv command to rename a file:

mv old_filename.txt new_filename.txt

- **rm command**

The rm command is used to delete files within a directory. Make sure that the user performing this command has write permissions.

Remember the directory's location as this will remove the file(s) and you can't undo it.

Here's the general syntax:

rm filename

To remove multiple files, enter the following command:

rm filename1 filename2 filename3

Here are some acceptable options you can add:

-i prompts system confirmation before deleting a file.

-f allows the system to remove without a confirmation.

-r deletes files and directories recursively.

File Editor and creation commands:

- **touch command**

The touch command allows you to create an empty file or generate and modify a timestamp in the Linux command line.

For example, enter the following command to create an HTML file named Web in the Documents directory:

touch /home/username/Documents/Web.html

- **cat command**

Concatenate, or cat, is one of the most frequently used Linux commands. It lists, combines, and writes file content to the standard output. To run the cat command, type cat followed by the file name and its extension. For instance: **cat filename.txt.**

Here are other ways to use the cat command: **cat**

> filename.txt creates a new file.

cat filename1.txt filename2.txt > filename3.txt merges filename1.txt and filename2.txt and stores the output in filename3.txt.

- **nano, vi, jed commands**

Linux allows users to edit and manage files via a text editor, such as nano, vi, or jed. nano and vi come with the operating system, while jed has to be installed.

The **nano** command denotes keywords and can work with most languages. To use it, enter the following command: **nano [filename] vi** uses two operating modes to work – insert and command. insert is used to edit and create a text file. On the other hand, the command performs operations, such as saving, opening, copying, and pasting a file.

To use vi on a file, enter:

vi [filename]

jed has a drop-down menu interface that allows users to perform actions without entering keyboard combinations or commands. Like vi, it has modes to load modules or plugins to write specific texts.

To open the program, simply enter **jed** to the command line.

- **Search commands:**

➢ locate command

The locate command can find a file in the database system.

Moreover, adding the -i argument will turn off case sensitivity, so you can search for a file even if you don't remember its exact name.

To look for content that contains two or more words, use an asterisk (*). For example:

locate -i school*not

The command will search for files that contain the words school and note, whether they use uppercase or lowercase letters.

➢ find command

Use the find command to search for files within a specific directory and perform subsequent operations. Here's the general syntax:

find [option] [path] [expression]

For example, you want to look for a file called notes.txt within the home directory and its subfolders:

find /home -name notes.txt

Here are other variations when using find:

find -name filename.txt to find files in the current directory. **find**

./ -type d -name directoryname to look for directories.

➢ grep command

Another basic Linux command on the list is grep or global regular expression print. It lets you find a word by searching through all the texts in a specific file.

Once the grep command finds a match, it prints all lines that contain the specific pattern. This command helps filter through large log files.

For example, you want to search for the word blue in the notepad.txt file:

grep blue notepad.txt

The command's output will display lines that contain blue.

- **Displaying Output Commands:**

➢ head command

The head command allows you to view the first ten lines of a text. Adding an option lets you change the number of lines shown. The head command is also used to output piped data to the CLI.

Here's the general syntax:

head [option] [file]

For instance, you want to view the first ten lines of note.txt, located in the current directory:

head note.txt

Below are some options you can add:

-n or –lines prints the first customized number of lines. For example, enter head -n 5 filename.txt to show the first five lines of filename.txt.

-c or –bytes prints the first customized number of bytes of each file.

-q or –quiet will not print headers specifying the file name.

➢ tail command

The tail command displays the last ten lines of a file. It allows users to check whether a file has new data or to read error messages.

Here's the general format:

tail [option] [file]

For example, you want to show the last ten lines of the colors.txt file:

tail -n colors.txt

➢ cat command

Concatenate, or cat, is one of the most frequently used Linux commands. It lists, combines, and writes file content to the standard output. To run the cat command, type cat followed by the file name and its extension. For instance: **cat filename.txt.**

Here are other ways to use the cat command: **cat**

> filename.txt creates a new file.

cat filename1.txt filename2.txt > filename3.txt merges filename1.txt and filename2.txt and stores the output in filename3.txt.

tac filename.txt displays content in reverse order.

File Directory Permissions & Ownership Commands:

- **chmod command**

chmod is a common command that modifies a file or directory's read, write, and execute permissions. In Linux, each file is associated with three user classes – owner, group member, and others.

Here's the basic syntax:

chmod [option] [permission] [file_name]

For example, the owner is currently the only one with full permissions to change note.txt. To allow group members and others to read, write, and execute the file, change it to the -rwxrwxrwx permission type, whose numeric value is 777:

chmod 777 note.txt

This command supports many options, including:

-c or –changes displays information when a change is made.

-f or –silent suppresses the error messages.

-v or –verbose displays a diagnostic for each processed file.

- **chown command**

The chown command lets you change the ownership of a file, directory, or symbolic link to a specified username.

Here's the basic format: **chown**

[option] owner[:group] file(s)

For example, you want to make linuxuser2 the owner of filename.txt:

chown linuxuser2 filename.txt

5. Wildcards:

Following are the wildcards used in Linux shell terminal.

1. ***** - will match against none or one or a string of more than a character
2. **?** - can be used to match one character
3. **[]** - matches one specified character out of a group of characters

WildCard '*'

• '**ls file***' - list all the files in current directory starting with filename 'file'.

• '**ls *2.txt'** - list all the files in current directory ending with '2.txt'

WildCard '?'

• '**ls file.tx?**' - list all the files that begins with 'file.tx'

WildCard '[]'

• '**ls rmt[12345]**' - list all the file that begins with 'rmt' and has a 1,2,3,4 or 5 after it.

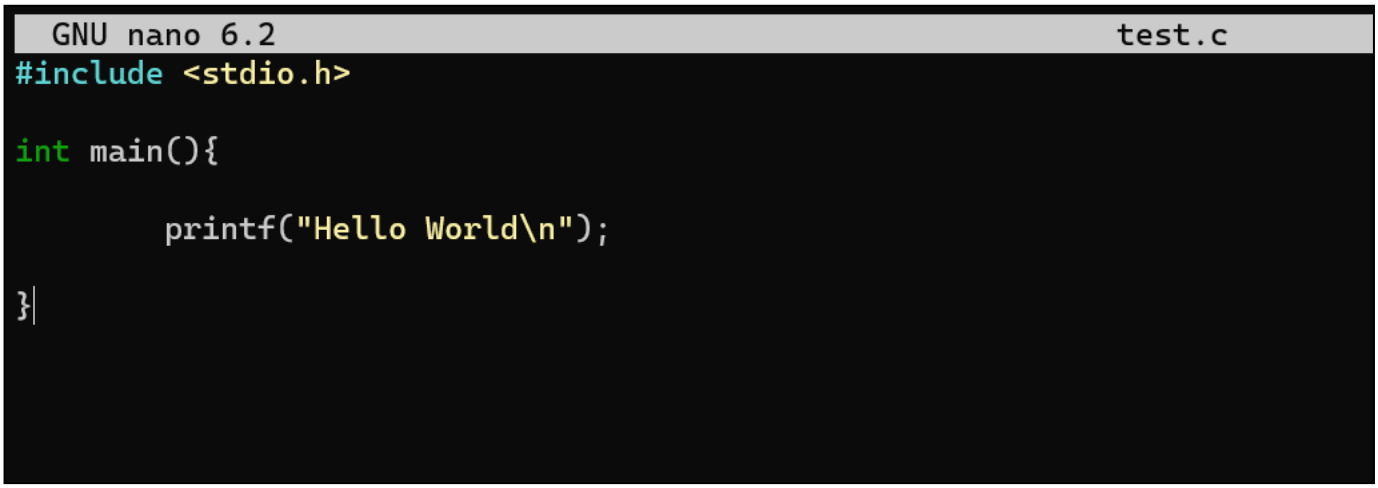
C- Code Creation and Compilation

You can create C files with the same method as any other file creation.

To get started use the following command

Nano test.c (Here I am using the nano editor although you can use any editor of your choice i.e. gedit)

Now write some C-Code in the file as an example I am writing a simple Hello World line in it.



Save the file and exit out of the editor.

Compilation:

Normally in a OS environment you would have an editor that would do the compilation of the code automatically but since we are using the terminal mainly, we would have to compile it ourselves. There are two popular ways to compile the above file.

gcc test.c

GCC is the GNU compiler collection which contains multiple compilers ranging from C to C++, Objective -C, Fortran, Ada, Go and D as well as the libraries for these programming languages. The above command will read the code and compile the code into an output file know as **a.out.**

```
monis@LAPTOP-CMQQNPQF:~$ ls
a.out  test.c  test.txt
```

The a.out is the output file which contains the output of the C-code that is compiled. Note if your code has I/O dependencies such as taking input from the user or writing to the file, It will not be completed until the output file is executed.

To run the output file use the following command

```
monis@LAPTOP-CMQQNPQF:~$ ./a.out
Hello World
monis@LAPTOP-CMQQNPQF:~$
```

This will output the contents of the executable to the terminal.

But having multiple C/C++ files with the same a.out will be a problem as the OS will keep overwriting the contents of the output file to the new file that it just compiled. To get around this we can rename this output file via the following command.

gcc test.c -o output_file

```
monis@LAPTOP-CMQQNPQF:~$ gcc test.c -o output_file
monis@LAPTOP-CMQQNPQF:~$ ls
a.out  output_file  test.c  test.txt
monis@LAPTOP-CMQQNPQF:~$ ./output_file
Hello World
monis@LAPTOP-CMQQNPQF:~$
```

Here you can see we compiled the output file to a new file which we named as output_file. Notice it has no extension (like we say .out with the old output file). That's because the operating system interprets it as an executable and it does not require an extension to be mentioned with the file. In GUI based Linux distribution this would be labelled with a green text to signify that this file is an executable file. The -o is a flag to forward the output to the file we want to create.