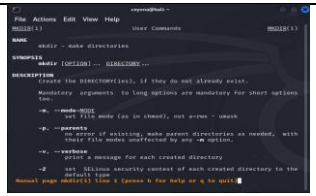
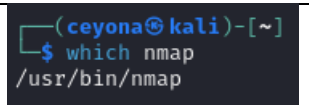
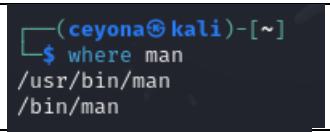
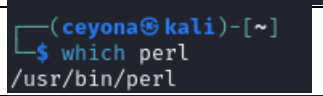
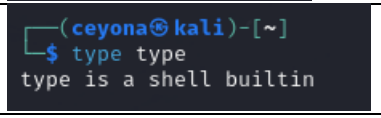
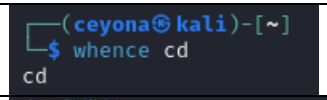
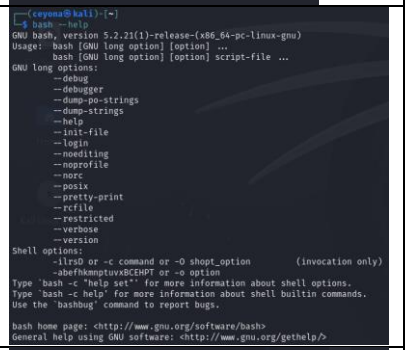
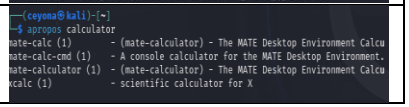
Topic Name:

The main aim of this lab session is to provide hands-on experience on

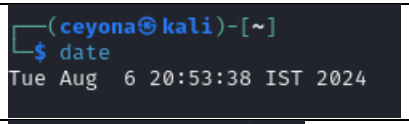
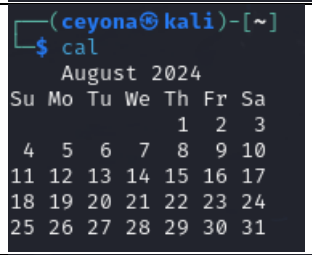
- Getting Help
- Basic Commands
- Navigation
- File System
- simple shell script

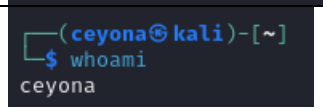
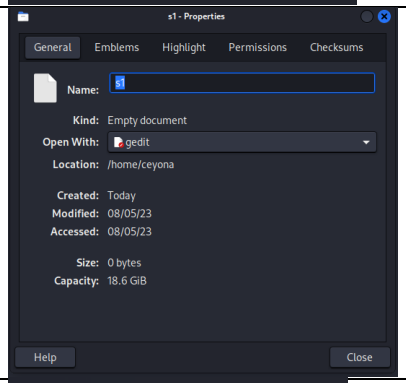
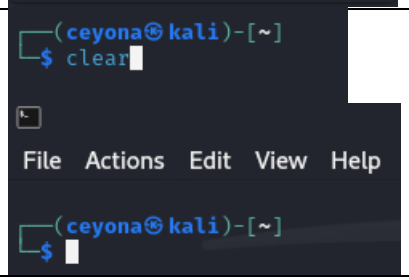
1. Getting Help

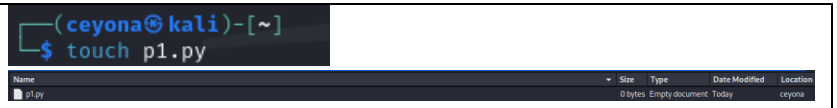
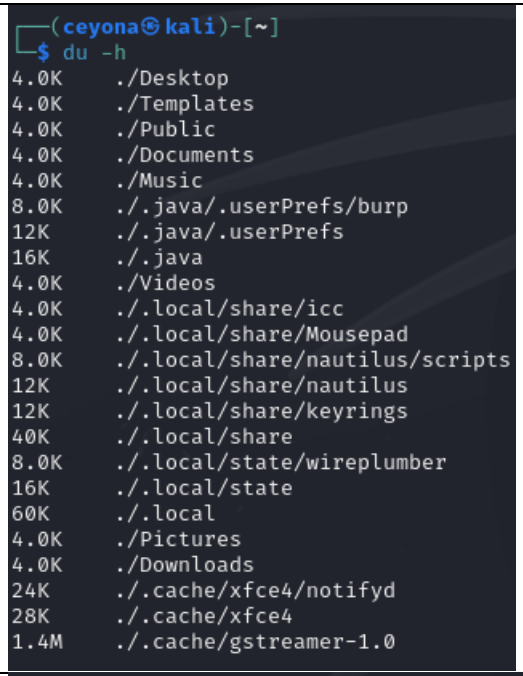
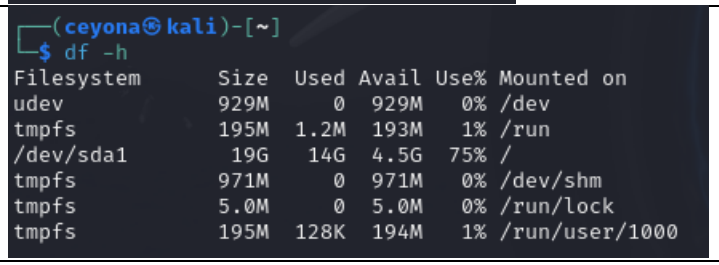
Task	Command Name	Syntax	Example	Screenshots
To get manual page for the known command	man	man command-name	man mkdir	
To get manual page for the unknown command	which	Which command-name	Which nmap	
To know the source file binary	where	where command-name	where man	
To know the path of the command	which	Which command-name	which perl	
To know the command is external or internal	type	type command-name	type type	

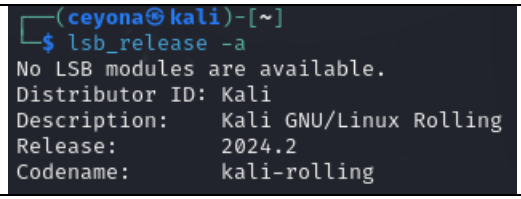
To get help for the internal command	whence	Whence command-name	Whence	
To list out bash commands	help	Bash --help	Bash --help	
To know the usage of the command	apropos	apropos command-name	Apropos calculator	

2. Basic Commands

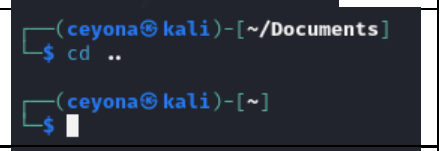
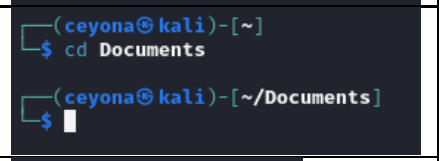
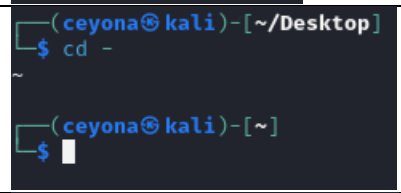
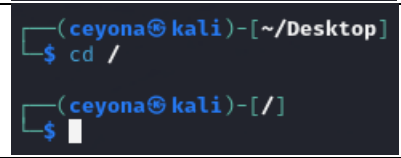
Task	Command Name	Syntax	Example	Screenshots
To know today's date	date	date	date	
To print calendar	cal	cal	cal	
To print kernel version	cat /proc/version	cat /proc/version	cat /proc/version	

To print default shell	echo $SHELL	echo $SHELL	echo $SHELL	
To print currently logged in user	whoami	whoami	whoami	
To create shortcut for command	alias	alias shortcut-name=command-name	alias wish='echo Good Morning!'	
To delete shortcut	unalias	unalias shortcut-name	unalias wish	
To change the timestamp of the file	touch	touch –t<yearmonthdaytime>	touch –t 202308281840	
To clear the screen	clear	clear	clear	

To create empty files	touch	touch.filename	touch p1.py	
To know disk usage	du	du[options][path]	du -h	
To know free space in the system	df	df[options]	df -h	

To know about the Linux release	lsb_release	lsb_release -a	lsb_release -a		

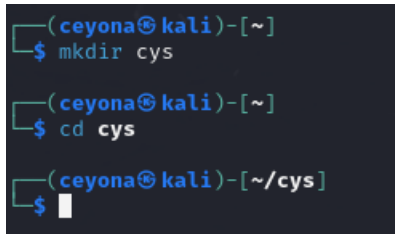
3. Navigation

Task	Command	Syntax	Screenshots
To navigate home directory	cd	cd	
To navigate to the parent directory	cd ..	cd ..	
To navigate to the child directory	cd <directory name>	cd <directory name>	
Alternate command to cd	pushd	pushd<directory name>	
To go back to the previous directory	cd -	cd -	
To go to the root directory	cd /	cd /	

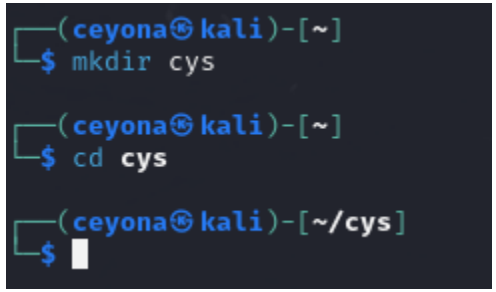
4. **File System**

Task	Syntax	Command
How to identify the file system	**lsblk -f**	**lsblk -f**

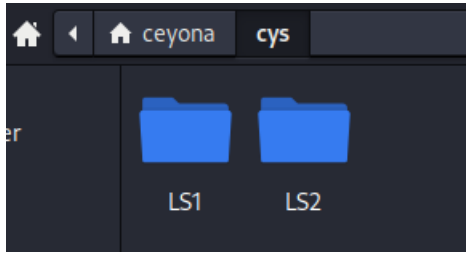
a. Create Folder “CYS”



b. Navigate to CYS



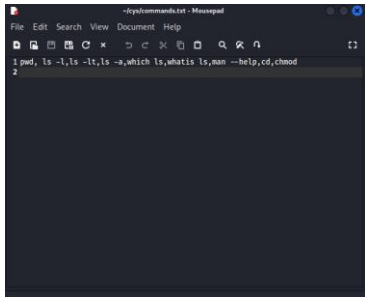
c. Create folder LS1 and LS2 under CYS



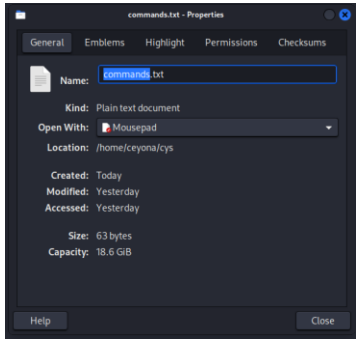
d. Go back to CYS

e. Working with Files

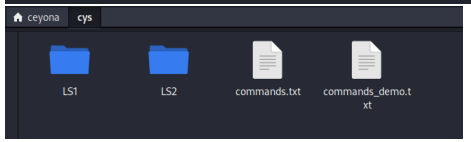
i. Add commands which you learnt during lab session in the file commands.txt



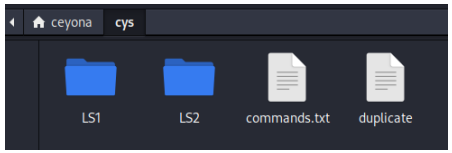
ii. Change the timestamp of the file to yesterday



iii. Copy the contents from the file commands.txt to commands_demo.txt



iv. Rename the file commands_demo.txt to duplicate



v. Delete the file duplicate

```
┌──(ceyona㉿kali)-[~/cys]
└─$ rm duplicate

┌──(ceyona㉿kali)-[~/cys]
└─$
```

vi. Copy the contents commands.txt to unit4 and unit5 (using relative path)

```
┌──(ceyona㉿kali)-[~/cys]
└─$ cp commands.txt ../LS2/unit4.txt
```

```
┌──(ceyona㉿kali)-[~/cys]
└─$ cp commands.txt ../LS2/unit5.txt
```

vii. Delete the contents from unit5 (using absolute path)

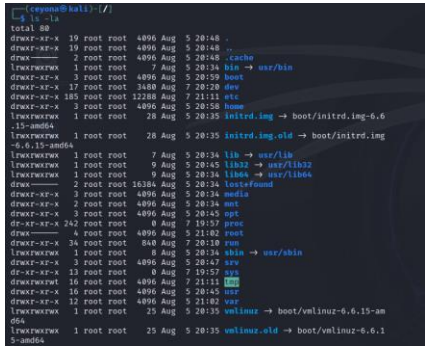
```
┌──(ceyona㉿kali)-[~/cys]
└─$ cp commands.txt ../LS2/unit5.txt
```

viii. Navigate to root

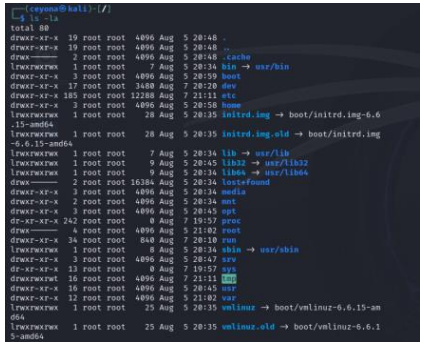
```
┌──(ceyona㉿kali)-[~/cys]
└─$ cd /

┌──(ceyona㉿kali)-[/]
└─$
```

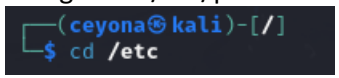
ix. List all the files under root



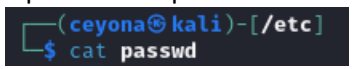
x. Explore all the folders (Do not delete any folder)



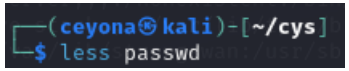
xi. Navigate to /etc/passwd



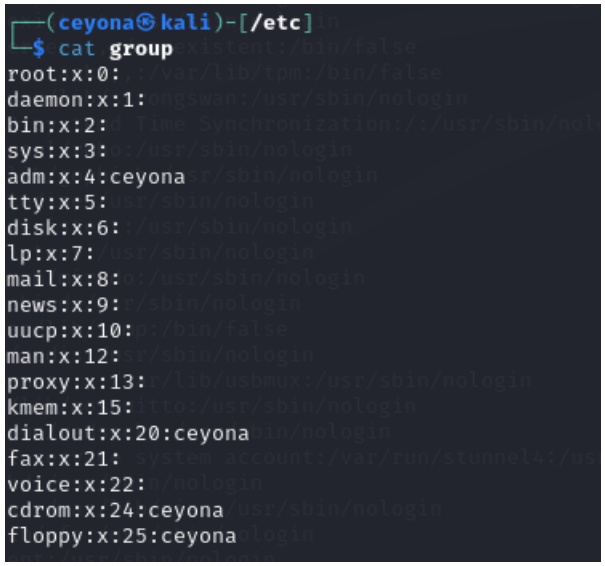
xii. Open the file passwd



xiii. Explore the file passwd



xiv. Navigate to /etc/group and explore



f. Difference between

i. GUI vs. CLI

GUI (Graphical User Interface):

- **Visual Interaction**: GUI relies on visual elements like windows, icons, buttons, and menus. Users interact with these elements using a mouse, touchpad, or touchscreen.
- **User-Friendly**: Generally considered more intuitive and easier for beginners because it provides a visual representation of options and tasks.
- **Learning Curve**: Easier to learn for most people due to its visual nature. Users can often discover functions by exploring the interface.
- **Multitasking**: Typically allows users to multitask more effectively with multiple windows and applications visible simultaneously.
- **Performance**: May use more system resources (like memory and processing power) due to the graphical elements.
- **Examples**: Windows operating system, macOS, most modern web applications, and graphical desktop applications.

CLI (Command-Line Interface):

- **Text-Based Interaction**: CLI relies on text commands entered via a keyboard. Users type specific commands to perform tasks or execute programs.
- **Efficient for Experts**: Preferred by more experienced users and system administrators for its speed and precision.
- **Learning Curve**: Can be more difficult to learn due to the need to memorize commands and syntax.
- **Scripting and Automation**: Highly effective for scripting and automating repetitive tasks. Users can write scripts to perform complex sequences of actions.
- **Performance**: Typically uses fewer system resources as it doesn't require graphical rendering.
- **Examples**: Terminal in Unix/Linux, Command Prompt and PowerShell in Windows, and the shell interfaces in various operating systems.

ii. man vs info

`man` (Manual Pages):

- **Purpose**: Provides access to the traditional Unix manual pages, which contain documentation about commands, system calls, library functions, and other aspects of the system.
- The documentation is divided into sections, with each section containing information about a specific type of content (e.g., commands, library functions). The layout is text-based and often includes sections like NAME, SYNOPSIS, DESCRIPTION, OPTIONS, and SEE ALSO.
- **Example**: To read the manual page for the `ls` command, you would use `man ls`.

Info:

- **Purpose**: Provides access to the GNU info system, which offers a more extensive and hierarchical documentation format, often used for GNU software and projects.
- **Format**: The documentation is organized in a tree-like structure with nodes and hyperlinks, allowing for more detailed and structured information. It supports rich formatting, such as tables of contents, cross-references, and embedded hyperlinks.
- **Example**: To read the info documentation for the `ls` command, you would use `info coreutils 'ls invocation'` (assuming you are looking for the GNU coreutils version of `ls`).

iii. which vs. whereis

Which:

- The `which` command is used to locate the executable file associated with a command by searching through directories listed in the PATH environment variable.
- It provides the full path to the executable file that will be run when the command is executed.
- The output is limited to the location of the command's executable and does not include other related files or documentation.
- If the command is not found in the PATH, `which` returns no result or an indication that the command is not available.
-

Whereis:

- The `whereis` command searches for the binary, source code, and manual page files related to a command in standard directories.
- It provides paths to multiple types of files, including the executable, source code, and documentation, if available.
- The search scope of `whereis` extends beyond the PATH environment variable to include system-wide directories and locations.
- The output is more comprehensive than `which`, showing various related files associated with the command, not just the executable.

iv. Terminal vs shell

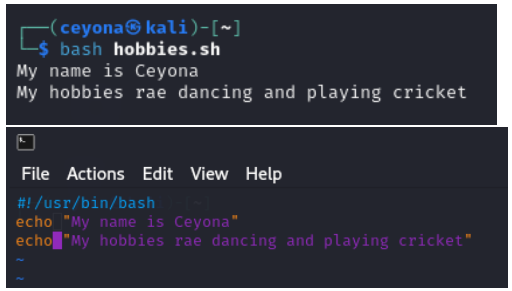
Terminal:

- A terminal is a software application or physical device that provides a text-based interface for user interaction with the operating system.
- It acts as a window or environment where users can input commands and view the output from the system.
- Terminal emulators like `gnome-terminal`, `xterm`, and `Terminal.app` facilitate access to the command-line interface in graphical environments.
- Essentially, the terminal is the medium through which users interact with the shell and the broader command-line environment.

Shell:

- A shell is a command-line interpreter that processes and executes user commands typed into a terminal.
- It provides features like scripting, variable management, and control flow to automate and manage tasks.
- Common shells include `bash`, `zsh`, and `fish`, each offering different features and functionalities.
- The shell operates within the terminal, interpreting commands and returning results to the user through the terminal interface.

g. Write a simple shell script to print your name and your hobbies!



Interesting commands to Explore

Banner

History

Note: Include your screenshots

Evaluation :

Marks : 10 (Deadline : 4 – Originality :3 – Completeness :3)

Deadline: 06.08.2024

“All our dreams can come true if we have the courage to pursue them.”

- Walt Disney