**WEEK-1: INTRODUCTION TO LINUX OPERATING SYSTEM AND COMMANDS**

**Introduction to the Linux Operating System**

Linux is a popular open-source operating system that is widely used for servers, desktops, and various other devices. It is known for its stability, security, and flexibility, and is based on the Unix operating system. Linux was first released in 1991 by Linus Torvalds and has since become a cornerstone of the open-source software movement.

Linux is essential for anyone working in the field. Here's how the Linux environment relates to cloud computing:

**1. Dominant Operating System in Cloud**

Linux is the most commonly used operating system in cloud environments, including major cloud service providers like AWS, Google Cloud Platform, and Microsoft Azure. This dominance is due to:

**Open Source Nature:** Linux is open source, which allows for extensive customization and cost savings. Cloud providers and users can modify the operating system to fit specific needs without licensing costs.

**Performance and Stability:** Linux is known for its stability and performance, making it ideal for high-demand cloud services.

**2. Development and Deployment Environment**

Many cloud-native applications are developed and deployed in Linux environments:

**Development Tools:** Linux supports a vast array of development tools and languages, making it a versatile environment for building applications. Tools like GCC, Make, Git, Docker, and others are commonly used in Linux.

**Containerization and Virtualization:** Technologies like Docker and Kubernetes, which are central to cloud-native application development, are primarily designed to run on Linux. Understanding Linux is crucial for managing containers and orchestrating workloads in the cloud.

**3. Server Management**

Linux is the backbone of many servers, including those used in cloud infrastructure:

**Command Line Interface (CLI):** Linux environments are often managed through the command line, which provides powerful and flexible tools for managing servers. Knowledge of commands like ssh, scp, ps, top, and others is essential for managing cloud instances.

**Scripting and Automation:** Shell scripting in Linux is a powerful way to automate tasks, including deployment, monitoring, and scaling of applications in the cloud. Tools like Bash scripts, cron jobs, and configuration management tools (e.g., Ansible, Puppet) are commonly used.

**4. Security and Networking**

Linux offers robust security and networking features, which are critical in cloud computing:

**Security:** Understanding Linux file permissions, user management, and security tools (e.g., iptables, SELinux) is essential for securing cloud environments.

**Networking:** Linux supports a wide range of networking tools and protocols, which are vital for managing cloud networks. Understanding tools like ifconfig, netstat, iptables, and networking configurations is crucial for setting up and maintaining cloud infrastructure.

**5. Cost Efficiency**

Linux's open-source nature allows cloud providers to offer cost-effective solutions:

**Lower Costs:** Since Linux is free, it reduces the cost for cloud providers and users. Many cloud providers offer Linux-based virtual machines (VMs) at a lower cost compared to proprietary operating systems.

6. Scalability and Flexibility

Linux provides the scalability and flexibility required for cloud computing:

**Scalable Architectures:** Linux supports scalable architectures, including clustered and distributed systems. It is often used in high-availability and load-balanced setups.

**Flexibility:** The ability to customize Linux distributions allows cloud providers and users to tailor the operating system to specific workloads and applications.

**Basic Linux Commands**

1. **ls** - Lists the contents of a directory.

Example: ‘ls’

1204 499 Downloads eceb\_22251A0499 Public

1209 a.out ece eceb\_499 SampleTextFile1.txt

1.c b ece499 eceb499 SampleTextFile2.txt

23255A1205 Desktop eceb Music Templates

2.c Documents eceb\_\_22251A0499 Pictures Videos

2. **mkdir** - Creates a new directory.

Syntax: `mkdir newdirectory`

Example: mkdir 23255A1205

2. **cd** - Changes the current directory.

Syntax: `cd directoryname`

Example: cd 23255A1205

3. **pwd** - Prints the current working directory.

Example: `pwd`

4. **help** - Displays help information about built-in commands.

Syntax: help

Example: help

GNU bash, version 5.0.17(1)-release (x86\_64-pc-linux-gnu)

These shell commands are defined internally. Type `help' to see this list.

Type `help name' to find out more about the function `name'.

Use `info bash' to find out more about the shell in general.

Use `man -k' or `info' to find out more about commands not in this list.

A star (\*) next to a name means that the command is disabled.

job\_spec [&] history [-c] [-d offset] [n] or his>

(( expression )) if COMMANDS; then COMMANDS; [ elif >

. filename [arguments] jobs [-lnprs] [jobspec ...] or jobs>

: kill [-s sigspec | -n signum | -sig>

[ arg... ] let arg [arg ...]

[[ expression ]] local [option] name[=value] ...

alias [-p] [name[=value] ... ] logout [n]

bg [job\_spec ...] mapfile [-d delim] [-n count] [-O o>

bind [-lpsvPSVX] [-m keymap] [-f fil> popd [-n] [+N | -N]

break [n] printf [-v var] format [arguments]

builtin [shell-builtin [arg ...]] pushd [-n] [+N | -N | dir]

caller [expr] pwd [-LP]

case WORD in [PATTERN [| PATTERN]...> read [-ers] [-a array] [-d delim] [>

cd [-L|[-P [-e]] [-@]] [dir] readarray [-d delim] [-n count] [-O>

command [-pVv] command [arg ...] readonly [-aAf] [name[=value] ...] >

compgen [-abcdefgjksuv] [-o option] > return [n]

complete [-abcdefgjksuv] [-pr] [-DEI> select NAME [in WORDS ... ;] do COM>

compopt [-o|+o option] [-DEI] [name > set [-abefhkmnptuvxBCHP] [-o option>

continue [n] shift [n]

coproc [NAME] command [redirections> shopt [-pqsu] [-o] [optname ...]

declare [-aAfFgilnrtux] [-p] [name[=> source filename [arguments]

dirs [-clpv] [+N] [-N] suspend [-f]

disown [-h] [-ar] [jobspec ... | pid> test [expr]

echo [-neE] [arg ...] time [-p] pipeline

enable [-a] [-dnps] [-f filename] [n> times

eval [arg ...] trap [-lp] [[arg] signal\_spec ...]

exec [-cl] [-a name] [command [argum> true

exit [n] type [-afptP] name [name ...]

export [-fn] [name[=value] ...] or e> typeset [-aAfFgilnrtux] [-p] name[=>

false ulimit [-SHabcdefiklmnpqrstuvxPT] [>

fc [-e ename] [-lnr] [first] [last] > umask [-p] [-S] [mode]

fg [job\_spec] unalias [-a] name [name ...]

for NAME [in WORDS ... ] ; do COMMAN> unset [-f] [-v] [-n] [name ...]

for (( exp1; exp2; exp3 )); do COMMA> until COMMANDS; do COMMANDS; done

function name { COMMANDS ; } or name> variables - Names and meanings of s>

getopts optstring name [arg] wait [-fn] [id ...]

hash [-lr] [-p pathname] [-dt] [name> while COMMANDS; do COMMANDS; done

help [-dms] [pattern ...] { COMMANDS ; }

5. **gedit** - A text editor in GNOME that opens a GUI window for editing text files.

Example: gedit p1.txt

6. **gcc** - The GNU Compiler Collection, used for compiling C programs.

Example: gcc program.c

7. **./a.out** - Executes the compiled output file

Example: ./a.out (Runs the executable a.out)

8. **touch** - Creates an empty file or updates the timestamp of an existing file.

Example: touch p1.txt

9. **cp** - Copies files or directories.

Example: cp p1.txt p2.txt

10. **mv** - Moves or renames files or directories.

Example: mv p1.txt prgm1.txt

11. **cat** - Concatenates and displays the content of files.

- Example: `cat file1.txt file2.txt`

12. **rm** - Removes files or directories.

Example: `rm file.txt` (To remove a directory, use `rm -r directory\_name`)

13. **df** - Reports disk space usage.

Example: `df -h`

14. **du** - Reports the size of directories and files.

Example: `du -sh /home/user`

15. **ps** - Displays information about running processes.

Example: `ps aux`

16. **kill** - Terminates processes.

Example: `kill 1234` (where 1234 is the process ID)

17. **sudo** - Executes a command as the superuser.

Example: `sudo apt-get update`

18. **cal** - Displays a calendar for the current month.

Example: cal

August 2024

Su Mo Tu We Th Fr Sa

1 2 3

4 5 6 7 8 9 10

11 12 13 14 15 16 17

18 19 20 21 22 23 24

25 26 27 28 29 30 31

19. **echo** - Outputs the text provided as an argument.

Example: echo "Hello, World!"

20. **date** - Displays or sets the system date and time.

Example: date

Wednesday 07 August 2024 09:59:10 AM IST

21. **whoami** - Prints the current logged-in username.

Example: whoami

lsroot123

22. **rmdir** - Removes an empty directory.

Example: `rmdir emptydirectory`

23. **clear** - Clears the terminal screen.

Example: clear