DAYANANDA SAGAR UNIVERSITY
LINUX PROGRAMMING

ASSIGNMENTS-1

NAME : AKHIL VENKAT D

USN : ENG24CY0079

ROLL NO :46

SEACTION : A

1. What is Linux Operating System (OS)? List three pros and cons of it?
The Linux Operating System is an open-source, Unix-like operating system that is widely used for servers, desktops, mobile devices (like Android), and embedded systems
Three Pros of Linux:
1. Open Source & Free
○ Linux is open-source, meaning its source code is available to the public. Anyone can modify, distribute, and improve it. This also means that Linux is usually free to download and use, with many distributions (distros) like Ubuntu, Debian, Fedora, and CentOS available at no cost.
2. Security
○ Linux is considered one of the most secure operating systems available. It is less prone to viruses, malware, and other security vulnerabilities than other operating systems, such as Windows. Its permission-based security model also ensures that users can control who accesses their system.
3. Stability & Performance
○ Linux is known for its stability and performance. It doesn't require frequent restarts and can run for extended periods without crashing. It's also lightweight, which allows it to run efficiently on a variety of hardware, including older systems.
Three Cons of Linux:
1. Compatibility Issues
○ While Linux is compatible with many software programs, some commercial software (like Adobe Creative Suite, Microsoft Office) and games are not natively supported. Users may need to rely on alternatives or use tools like Wine to run Windows applications.

2. Steeper Learning Curve
○ Linux can be intimidating for beginners, especially those who are used to the graphical user interfaces (GUIs) of Windows or macOS. Many tasks require using the command line, which can be challenging for users without technical experience.
3. Hardware Compatibility
○ Some hardware devices, especially new or proprietary ones, may not have proper drivers for Linux, causing issues with functionality. While Linux support for hardware has improved over time, it may still be a concern for users with specific needs.
2. Differentiate between Linux, Mac, Android, and Windows OS with at least six unique features?
1. Linux OS
Linux is an open-source, Unix-like operating system used for desktops, servers, and embedded systems.
• Open Source: The source code is publicly available, allowing users to modify and distribute it. It's free to use.
• Customizability: Highly customizable; users can modify the kernel and system components to fit their needs. This flexibility is ideal for advanced users and developers.
• Security: Known for strong security with fewer malware attacks. It uses a permission-based system that restricts unauthorized access.
• Lightweight: Linux can run on low-resource hardware, making it great for older systems or embedded devices.
• Variety of Distributions: There are many Linux distributions (Ubuntu, Fedora, Debian, CentOS, etc.), allowing users to choose one based on their needs.
• Package Management: Uses package managers (like APT, YUM, or Pacman) to easily install and manage software.
2. Mac OS (Apple's macOS)
macOS is a Unix-based operating system designed by Apple, primarily for Mac computers.
• User-Friendly Interface: Known for its polished and intuitive graphical user interface (GUI). It's designed to be simple and easy to use.
• Exclusive Ecosystem: Seamlessly integrates with other Apple products (iPhones, iPads, Apple Watch, etc.), allowing features like Handoff, AirDrop, and Continuity.
• Hardware-Software Integration: Apple controls both hardware and software, which results in highly optimized performance and fewer compatibility issues.
• Security: Has a strong emphasis on security with features like Gatekeeper (which verifies downloaded software), FileVault (full disk encryption), and sandboxing.
• Proprietary Software: Comes with high-quality proprietary software like Final Cut Pro, Logic Pro, and Safari, making it a favorite for creative professionals.
• macOS Terminal: Like Linux, macOS provides a powerful command-line interface, though it’s less reliant on it than Linux.
3. Android OS
Android is a mobile operating system based on the Linux kernel, widely used on smartphones, tablets, and other mobile devices.
• Open Source: Android is open-source (though with some proprietary components) and customizable by manufacturers. Developers can modify it freely.

• Google Integration: Strong integration with Google services (Google Play, Gmail, Google Maps, etc.), making it the go-to OS for users deeply embedded in the Google ecosystem.
• App Flexibility: Android allows users to sideload apps (install apps from third-party sources), offering more flexibility than closed systems like iOS.
• Customizability: Users can modify almost every aspect of the user interface (UI), including installing custom launchers, widgets, and themes.
• Wide Device Support: Android runs on a wide range of devices, from budget smartphones to high-end flagship devices from various manufacturers like Samsung, Xiaomi, and OnePlus.
• Google Play Store: The Android platform is home to the largest app store, offering millions of apps across categories like productivity, gaming, social media, and more.
4. Windows OS
Windows is the most widely used operating system for desktops and laptops, developed by Microsoft.
• Wide Software Compatibility: Windows supports a vast array of software, including many commercial and enterprise applications, which makes it the primary choice for businesses and gamers.
• User Base: With a massive user base, Windows has extensive community support, software compatibility, and peripherals support (printers, scanners, etc.).
• Gaming: Windows is the most popular operating system for gaming, supporting DirectX, which allows for high-performance gaming and compatibility with a broad range of game titles.
• Backward Compatibility: Windows supports older applications and hardware better than many other OSes, allowing users to run legacy software on modern machines.
• Microsoft Ecosystem: Deep integration with Microsoft's services like OneDrive, Microsoft Office, and Teams, making it highly suitable for business and productivity tasks.
• User Interface (UI): Windows 10/11 features a familiar and flexible GUI, which has evolved over time but still retains classic elements like the Start Menu and taskbar.
3. Why is Linux preferred for Mainframe Servers for legacy application? Give three out-of-the-box technical reasons?
1. Cost Efficiency and Open Source Flexibility
• Cost-Effective: Mainframe environments traditionally rely on proprietary operating systems like IBM's z/OS, which can be costly in terms of licensing, support, and hardware. Linux, being open-source, eliminates these costs. Mainframe operators can use Linux on mainframe hardware (IBM Z or other mainframes) at a significantly lower cost since there are no licensing fees and the OS itself is free.
• Flexibility: Since Linux is open-source, organizations can customize the OS to suit their specific needs, especially for legacy applications. They have the flexibility to tweak the kernel, install different distributions, and use specialized tools without the constraints of a proprietary OS.
2. Performance and Scalability
• Mainframe-Specific Optimizations: Linux is highly optimized for running on powerful mainframe hardware, including IBM Z systems. It can leverage the massive parallel processing capabilities of mainframe hardware to run multiple legacy applications simultaneously without performance degradation.

• Scalability: Linux supports scalability on mainframe systems, allowing for the efficient management of huge amounts of data and workloads. It can handle thousands of transactions per second, which is critical for legacy applications that need to process a large volume of data efficiently on mainframes.
Linux has excellent support for large memory configurations, multi-core processors, and high-performance networking, making it capable of maintaining optimal performance as workloads scale.
3. Compatibility with Legacy Applications and Middleware
• Legacy Application Support: Linux provides support for a wide range of legacy applications and middleware that were originally designed to run on mainframes. The Linux kernel and various distributions have matured to the point where they are highly compatible with legacy software, allowing older programs (often written in COBOL or similar languages) to run on modern Linux-based mainframes with little or no modification.
• Emulation and Compatibility Layers: For legacy applications that rely on older mainframe OS environments, Linux can use emulation or compatibility layers (e.g., COBOL compilers, Z Open Development, or IBM's z/VM virtualization) to allow applications to run seamlessly. This helps organizations to continue using their legacy applications without significant rewrites or migrations.
4. Explain the structure of the Linux File System with proper diagram. Note: you can use the tree command to find it out?
Structure of the Linux File System
The Linux file system is hierarchical and organized in a tree-like structure. It is rooted at the top with the / (root) directory, and all other files and directories branch out from it.
Here's an explanation of the key directories and their roles in the Linux file system hierarchy:
Key Linux File System Directories:
1. / (Root Directory)
○ The topmost directory. All other files and directories are located beneath it. It serves as the starting point of the file system.
2. /bin
○ Contains essential command binaries (programs) that are required for the system to function in single-user mode and for basic operations in multi-user mode. Examples: ls, cp, mv, etc.
3. /boot
○ Holds boot-related files like the kernel image (vmlinuz) and bootloader configuration files.
4. /dev
○ Contains device files, representing hardware devices (e.g., /dev/sda for a hard drive, /dev/tty for terminal devices).
5. /etc
○ Houses configuration files for the system and installed software. For example, /etc/passwd for user information, /etc/hostname for the system name.
6. /home
○ This is where user directories are stored. Each user typically has their own directory within /home, e.g., /home/john for user "john".
7. /lib
○ Contains shared libraries (similar to DLLs in Windows) needed to boot the system and run programs in /bin and /sbin.

8. /media
○ A mount point for removable media like USB drives, CD/DVDs, and other external storage devices. It may contain directories like /media/usb or /media/cdrom.
9. /mnt
○ Traditionally used as a temporary mount point for mounting file systems manually (e.g., mounting a network share or external disk). It's less commonly used today, with mount points often being placed in /media.
10. /opt
• Typically used for optional software packages. Software installed here is often self-contained and doesn't require installation within system directories like /bin or /lib.
11. /proc
• A virtual filesystem that provides process and kernel information as files. For example, /proc/cpuinfo contains details about the CPU, and /proc/meminfo shows memory usage.
12. /root
• The home directory for the root user (superuser). Not to be confused with /, which is the root of the file system.
13. /run
• A temporary filesystem that contains system information that’s available to programs while the system is running (e.g., process IDs, system state).
14. /sbin
• Contains system binaries that are essential for system maintenance and for administrators (e.g., shutdown, mount, fsck).
15. /srv
• Stores data for services provided by the system, such as web servers or FTP servers. For example, /srv/www might contain the web server's data.
16. /sys
• Another virtual filesystem that contains information and configuration options related to the system’s kernel and hardware.
17. /tmp
• A directory for temporary files. These files are usually deleted when the system reboots or after a specified period.
18. /usr
• Contains user-related programs and data, including applications, libraries, and documentation. It is split into subdirectories like /usr/bin (user binaries), /usr/lib (user libraries), and /usr/share (shared data).
19. /var
• Contains variable data like logs, spool files, and cache files. For example, /var/log holds log files, and /var/spool contains mail and print jobs.
Linux File System Tree Example
Here's an example representation of the Linux file system structure using the tree command:

```
/
├── bin/
├── boot/
│   ├── grub/
│   └── vmlinuz
├── dev/
│   ├── sda
│   └── tty
├── etc/
│   ├── passwd
```

```
│   └── hostname
├── home/
│   └── john/
├── lib/
├── media/
│   └── usb/
├── mnt/
├── opt/
│   └── software/
├── proc/
│   ├── cpuinfo
│   └── meminfo
├── root/
│   └── .bashrc
├── run/
├── sbin/
├── srv/
│   └── www/
├── sys/
├── tmp/
├── usr/
│   ├── bin/
│   ├── lib/
│   └── share/
└── var/
    ├── log/
    └── spool/
```

5. If Linux OS is open-source, how do companies like Red Hat still making money from it? Do a market study and answer properly?

How Red Hat Generates Revenue — Their Business Model

Red Hat's success (now part of IBM) comes from combining several revenue streams built around open source:

Revenue stream	What is sold / delivered	Why customers pay for it
Subscriptions	For enterprise Linux. distributions (e.g. RHEL), cloud software, middleware etc. These include updates, security patches, long-term release lifecycles.	Enterprises need predictable, stable, secure OSes/services. They can't afford risks of unsupported, unpatched systems. When something breaks, or when custom

Support & Services.	Technical support contracts, service level agreements. (SLAs), consulting.	integration is needed, they want reliable help from guarantees (response

Revenue stream

What is sold / delivered	Why customers pay for it (what value)
Ensuring hardware, software, cloud providers are certified to run RHEL etc.; creating a partner ecosystem so customers know all components will work together.	Reduces risk, reduces “vendor-lock-in” fears, ensures compatibility. For an enterprise this avoids nasty surprises.

Training courses, certifications, best practice guides, documentation, workshops.	Enterprises have teams; they need to build skills, ensure staff are trained to deploy/manage systems properly.
E.g. orchestration tools, container platforms (OpenShift), cloud-native tools, monitoring, etc. These are often open source upstream but packaged and supported in enterprise form.	Enterprises want tools that interoperate, with good UX, security, support, integrations.

providing value-added services around it: support, reliability, guarantees, ecosystems, management, and integration.

6. Which command is used to check how long the system has been running?
To display today’s date and current system time in Linux, use the following command in the terminal:
date
Example Output:
Mon Sep 22 14:35:12 IST 2025
This command works in most Linux distributions and outputs the system’s current date and time based on the configured timezone.
7. Which command is used to check how long the system has been running?
To check how long the system has been running, use the following command:
uptime
Example Output:
14:36:52 up 3 days, 4:12, 2 users, load average: 0.15, 0.10, 0.08
Explanation:

• up 3 days, 4:12: The system has been running for 3 days and 4 hours 12 minutes.
• 2 users: Number of logged-in users.
• load average: CPU load over the last 1, 5, and 15 minutes.
Alternative (More Detailed) Command:
who -b
This shows the last boot time of the system:
system boot 2025-09-19 10:24
8. What is the difference between shutdown -h now and halt?
The commands shutdown -h now and halt both stop the system, but they do so in slightly different ways and are used in different contexts. Here's a comparison:
shutdown -h now
Full Command:
shutdown -h now
Meaning:
• shutdown: Initiates a system shutdown process.
• -h: Halt the system after shutting down (i.e., power off).
• now: Perform the action immediately.
halt
Full Command:
halt
Meaning:
• Immediately halts the system without going through the full shutdown process, depending on configuration.
9. Compare init 0 and shutdown -h. Which is safer? Why?
? init 0
This command tells the system to switch to runlevel 0, which is the halt state. Essentially, it instructs the system to stop all processes and power off.
? shutdown -h
The shutdown command schedules a system shutdown and halts the machine. The -h flag means halt after shutdown. It also sends notifications to logged-in users and prevents new logins.
? init 0
• Immediately tells init to bring the system to runlevel 0.
• Runs shutdown scripts associated with runlevel 0.
• Does not notify users by default.
• Does not delay or schedule shutdown.
? shutdown -h
• Sends a warning message to all logged-in users.
• Prevents new logins.
• Runs shutdown scripts cleanly.
• Can be scheduled (e.g., shutdown -h +5 to halt after 5 minutes).
• Halts the system safely.

shutdown -h is safer.
Why?
• User Notification: It warns all users currently logged in about the impending shutdown, giving them time to save work and log out cleanly.
• Prevention of New Logins: It blocks new logins, avoiding inconsistent states or lost work.
• Graceful Handling: It usually runs all necessary shutdown scripts properly to close services cleanly.
• Scheduling: You can schedule it to allow orderly preparation for shutdown.
init 0 doesn't notify users and immediately transitions runlevels, which can abruptly interrupt users or processes.
10. A system administrator accidentally powers off a Server machine without shutting it down properly. What problems can occur to the said Server?
Powering off a server abruptly without a proper shutdown can lead to several serious problems. Here's a rundown of the main issues that can happen:
1. File System Corruption
• Unflushed Data: When a system shuts down properly, it flushes all cached data to disk. An abrupt power off can leave data in memory buffers that never get written, causing inconsistencies.
• Journal Corruption: For journaling file systems (like ext4, XFS), an improper shutdown can corrupt the journal, leading to longer fsck (file system check) times or even data loss.
• Damaged Files: Files that were being written or modified might become corrupted or lost.
2. Data Loss
• Open Files: Any files open for writing may lose data or become corrupted.
• Database Inconsistencies: If the server runs databases (MySQL, PostgreSQL, etc.), abrupt power off can cause transaction logs to become inconsistent, risking data integrity.
3. Hardware Damage
• Disk Drive Damage: Especially with older spinning hard drives, sudden power loss can cause mechanical parts to stop abruptly, potentially leading to physical damage.
• Power Surges: In some cases, sudden power off and back on can cause voltage spikes, harming sensitive components.
4. Longer Boot Times / Recovery Time
• On reboot, the system might need to perform extensive file system checks (fsck) or repairs, delaying availability.
• Services may fail to start properly if config or data files are corrupted.
5. Service/Application Failures
• Applications might not have saved their state correctly.
• Configuration files could be partially written and corrupted.

• This can lead to crashes or unstable behavior until fixed.
6. Potential Security Risks
• If logs are corrupted or lost, important security audit trails may be missing.
• Unexpected downtime can be exploited by attackers if the system is part of a larger network.
7. Loss of Unsaved User Data
• Any users connected via SSH or applications might lose unsaved work, leading to frustration and possible data re-entry.