Chapter 1: System and Network Administrator

Overview

A system and network administrator is responsible for managing and maintaining computer systems and networks in an organization. They ensure that all hardware and software components are functioning properly, and troubleshoot any issues that arise. They also manage user accounts, security settings, and backups.

The role of a system and network administrator requires a strong understanding of computer hardware and software, as well as network protocols and standards. They must be able to work with a variety of operating systems, such as Windows, Linux, and macOS, and have experience with virtualization technologies.

In addition to technical skills, a system and network administrator must also possess strong communication and problem-solving skills. They must be able to work with users to resolve issues and provide training when necessary. They must also be able to prioritize tasks and manage their time effectively.

The responsibilities of a system and network administrator may include installing and configuring hardware and software, monitoring network performance, managing backups and disaster recovery plans, and ensuring compliance with security policies and regulations.

Overall, the role of a system and network administrator is critical to the smooth operation of an organization's computer systems and networks. They play a key role in ensuring that users have access to the resources they need, and that the organization's data is secure and protected.

Objective

After going through this unit, you will be able to:

- Develop necessary skills and knowledge to effectively manage and maintain computer systems and networks
- learn about system administration tasks Design, implement, and manage computer networks, including configuring network devices, troubleshooting network issues, and ensuring network security.
- Describe the roles and responsibilities of a Network Administrator

1.1 Roles and Responsibilities of Network Administrator

What is Network and System Administration?

Network and system administration play crucial roles in the effective management and operation of an organization's IT infrastructure.

Network administration involves overseeing the implementation, maintenance, and troubleshooting of computer networks. Network administrators are responsible for designing network architecture, configuring network devices, and ensuring smooth connectivity among computers, servers, and other network components. They also handle tasks like IP address management, network security, and monitoring network performance. Network administrators play a vital role in optimizing network performance, resolving network issues, and ensuring data integrity and security.

On the other hand, system administration focuses on managing individual systems, such as servers and workstations, within the network. System administrators are responsible for tasks like installing and configuring operating systems, managing user accounts and access controls, and maintaining system updates and patches. They monitor system performance, diagnose and resolve system failures, and implement backup and disaster recovery solutions. System administrators work closely with network administrators to ensure seamless integration and proper functioning of systems within the network environment.

Both network and system administrators play critical roles in ensuring the reliable and secure operation of an organization's IT infrastructure. They need to possess technical expertise, problem-solving skills, and the ability to adapt to rapidly changing technologies. Effective network and system administration are essential for maintaining network uptime, optimizing performance, protecting against security threats, and providing reliable IT services to users within the organization.

Key differences between Sys. Admins and Network Admins

Differences	System administrator	Network administrator
Scope	They focus on operating systems, software platforms, and servers.	They take care of network maintenance.
Equipment	They use tower servers, mainframes, a variety of disc drives, memory cards, motherboards, and power supply cables.	They need network gateways, wireless access points, ethernet hubs, and repeaters.
Pay	An average of $83000 per year.	An average of $72000 per year.
Education	They tend to study topics like OS management, software development, virtualization, and device maintenance.	They tend to focus on cybersecurity, programming, and database administration.
Certifications	Programming certifications.	Security and cloud services certifications.
Experience	Start off at help desks or as server hardware technicians.	Start as IT communication specialists or database experts.

Fig. 1.1 Key differences between Sys. Admins and Network Admins

Dear learner, what are the roles and responsibilities of network administrator?

_____
_____
_____
_____

1.2 Designing the Network

The first phase in the life cycle of a network involves creating its design, a task not usually performed by new network administrators. Designing a network involves making decisions about the type of network that best suits the needs of your organization. In larger sites this task is performed by a senior network architect: an experienced network administrator familiar with both network software and hardware.

1.3 Setting up the Network

After the new network is designed, the second phase of network administration begins, which involves setting up and configuring the network. This consists of installing the hardware that makes up the physical part of the network, and configuring the files or databases, hosts, routers, and network configuration servers.

The tasks involved in this phase are a major responsibility for network administrators. You should expect to perform these tasks unless your organization is very large, with an adequate network structure already in place.

What are the steps involved in setting up a network? (Dear students you may need further reading to answer this question read the module 2)

_____
_____
_
_____
_____

1.4 Maintaining the Network

The third phase of network administration consists of ongoing tasks that typically constitute the bulk of your responsibilities and activities required to ensure that the network operates smoothly, securely, and efficiently.

Here are some key aspects of maintaining a computer network:

1. Monitoring network performance: Regularly monitor network traffic, bandwidth usage, latency, and other performance metrics to identify any issues or bottlenecks that may affect network performance.
2. Troubleshooting network issues: Diagnose and resolve network problems such as connectivity issues, slow performance, configuration errors, hardware failures, and software conflicts.
3. Updating software and firmware: Keep network devices such as routers, switches, access points, and servers up to date with the latest software patches, firmware updates, and security fixes to prevent vulnerabilities and ensure compatibility.
4. Managing network security: Implement security measures such as firewalls, antivirus software, intrusion detection systems, access controls, encryption, and regular security audits to protect the network from cyber threats and unauthorized access.
5. Optimizing network resources: Monitor and manage network resources such as bandwidth, storage capacity, and CPU usage to ensure optimal performance and efficient use of resources.
6. Backing up data: Regularly back up critical data and configuration settings to prevent data loss in case of hardware failures, natural disasters, or cyber-attacks.
7. Documenting network configurations: Maintain detailed documentation of network configurations, IP addresses, device settings, security policies, and troubleshooting procedures for reference and future maintenance.
8. Performing regular maintenance tasks: Conduct routine maintenance tasks such as cleaning dust from networking equipment, replacing outdated hardware components, updating software licenses, and optimizing network settings.

9. Training IT staff and end-users: Provide training to IT staff and end-users on network maintenance best practices, security protocols, troubleshooting techniques, and how to use network resources effectively.

- Administering network security
- Administering network services, such as AD services, name services, and electronic mail
- Troubleshooting network problems

1.5 Expanding the Network

Expanding a computer network involves adding new devices, upgrading existing components, and reconfiguring network settings to accommodate the increased capacity and connectivity requirements. Here are the necessary procedures to expand a computer network:

1. Assess current network infrastructure: Evaluate the existing network topology, devices, bandwidth usage, and performance metrics to identify areas that need expansion or improvement.

2. Define expansion goals: Determine the specific objectives of the network expansion, such as increasing bandwidth, adding new users or devices, enhancing security measures, or improving network reliability.

3. Plan network expansion: Develop a detailed expansion plan that includes the following steps:

- Identify the type and number of new devices to be added (e.g., routers, switches, access points).
- Determine the location of new network devices and cabling requirements.
- Calculate the additional bandwidth and capacity needed to support the expanded network.
- Consider scalability, future growth, and compatibility with existing network components.
- Create a timeline for implementing the network expansion.

4. Procure necessary equipment: Purchase new networking devices, cables, connectors, and other hardware components required for the expansion based on the planned network design.

5. Install new network devices: Physically install the new networking equipment in the designated locations according to the expansion plan. Ensure proper cabling, power connections, and mounting of devices.

6. Configure network settings: Configure the new network devices with appropriate IP addresses, subnet masks, gateway settings, VLANs, security protocols, and other parameters to integrate them into the existing network infrastructure.

7. Test network connectivity: Verify connectivity between new and existing devices by conducting network tests, ping tests, traceroute tests, and bandwidth tests to ensure proper communication and functionality.
8. Update network documentation: Update network diagrams, IP address assignments, device configurations, security policies, and other documentation to reflect the changes made during the network expansion.
9. Implement security measures: Apply security measures such as firewalls, access controls, encryption, and intrusion detection systems to protect the expanded network from cyber threats and unauthorized access.
10. Monitor network performance: Continuously monitor network performance, traffic patterns, bandwidth usage, and latency after expanding the network to identify any issues and optimize performance.

What is involved in expanding a network?

_____

_____

_____

_____

How to be a Sys/Net Admin?

Learn Operating System basics e.g. UNIX
Learn shell utilities and script programming
Learn how to install and Configure OS and network tools
Learn DNS, DHCP, Samba, Proxy servers
Learn TCP/IP networking protocol, remote traffic monitoring tool

- ✓ Learn NFS and NIS- Network Information Service (DB) (or equivalent…)
- ✓ Learn about system tuning and accounting

A network administrator is an IT expert who manages an organization's network and responsible for configuring and maintenance of network infrastructure and services. It also includes configuring and commissioning of various network services/protocols AD, DC, DHCP, DNS, FTP, HTTP, NFS, etc.

1.6 Applying technology in an environment

A key task of network and system administration is to build hardware configurations, another is to configure software systems. Both of these tasks are performed for users. Each of these tasks presents its own challenges, but neither can be viewed in isolation.
Hardware has to conform to the constraints of the physical world; it requires power, a temperate (usually indoor) climate, and a conformance to basic standards in order to work systematically. The type of hardware limits the kind of software that can run on it. Software requires hardware, a basic operating system infrastructure and a conformance to certain standards, but is not necessarily limited by physical concerns as long as it has hardware to run on. Modern software, in the context of a global network, needs to inter-operate and survive the possible hostilities of incompatible or inhospitable competitors. Today the complexity of multiple software systems sharing a common Internet space reaches almost the level of the biological. In older days, it was normal to find proprietary solutions, whose strategy was to lock users into one company's products. Today that strategy is less dominant, and even untenable, thanks to networking.

Dear learner, can you state what network hardware mean? What software means and inter-operate?

_____

_____

_____

Network hardware is defined as a set of physical or network devices that are essential for interaction and communication between hardware units operational on a computer network.
A software or computer software essentially a type of programs which enable the users to perform some particular specific tasks or actually used to operate their computer. It essentially directs all of the peripheral devices on the entire computer system- what exactly to do and how exactly to perform a task. A software plays a key role of a mediator between the user and the computer hardware. In the absence of software, a user essentially can't perform any task on a computer
Interoperability is a characteristic of a product, software or system to work with other products, software or systems

1.7 Ethical issues

Ethics are a set of moral principles and behaviors that control the individual or a group which is used to decide what is good and bad. Ethics talks about the human behavior's consequences such as being honest, fair and keep on the integrity and reliability and confidentiality among each other.
Basically, ethical decisions depend on the mindset and personal beliefs of the people. In additions to that, they can follow up some code of ethical principles, standards and the agreements such as non-disclosure agreement.
A network administrator of a company or organization generally has an ability and enough privileges to access all information and sensitive data of an existing network. This much of wide access means that network administrator should have a variety of ethical issues and regulations to be considered while doing his job.

Dear learner, what ethical issues should network and system administrators be aware of?

_____

_____

_____

_____

1.8 The challenges of system administration

System administration is not just about installing operating systems. It is about planning and designing an efficient community of computers so that real users will be able to get their jobs done. That means:

- Designing a network which is logical and efficient.
- Deploying large numbers of machines which can be easily upgraded later.
- Deciding what services are needed.
- Planning and implementing adequate security.
- Providing a comfortable environment for users.
- Developing ways of fixing errors and problems which occur.
- Keeping track of and understanding how to use the enormous amount of knowledge which increases every year.

Some system administrators are responsible for both the hardware of the network and the computers which it connects, i.e. the cables as well as the computers.

Dear learner Write the challenges of system administration

_____

_____

_____

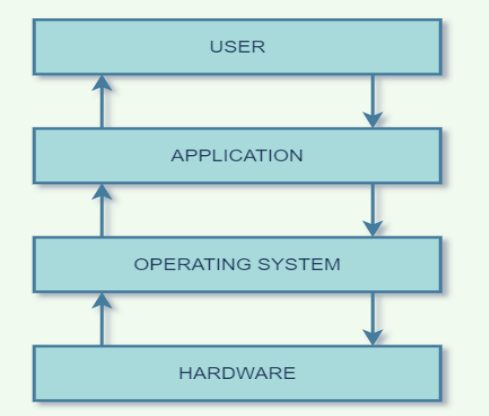
These are some of the challenges of system administration

- Designing a network which is logical and efficient.
- Deploying large numbers of machines which can be easily upgraded later.
- Deciding what services are needed.
- Planning and implementing adequate security.
- Providing a comfortable environment for users.
- Developing ways of fixing errors and problems which occur.
- Keeping track of and understanding how to use the enormous amount of knowledge which increases every year

Read more at: https://yourstory.com/mystory/what-software-types-examples

Overview of an operating systems

An operating system (OS) manages all other applications and programs in a computer, and it is loaded into the computer by a boot program. It enables applications to interact with a computer's hardware. Through a designated application programme interface, the application programmes request services from the operating system (API). The kernel is the software that contains the operating system's core components. To run other programmes, every computer has to have at least one operating system installed.



Operating Systems

Windows, Linux, and Android are examples of operating systems that enable the user to use programs like MS Office, Notepad, and games on the computer or mobile phone. It is necessary to have at least one operating system installed in the computer to run basic programs like browsers.

The core functions of an operating system include:

- Managing hardware resources: An operating system manages resources such as CPU, memory, and disk space, and assigns these resources to running applications.
- Running applications: An operating system provides an environment in which applications can run and interact with the user.
- Providing a user interface: An operating system provides a graphical user interface (GUI) that allows users to interact with the computer.
- Examples of popular operating systems include:
    - Windows: Microsoft Windows is the most popular desktop operating system, used by over 1 billion users worldwide. It has a wide range of features and applications, including the Office suite, gaming, and productivity tools.
    - macOS: macOS is the desktop operating system used by Apple Mac computers. It is known for its clean, user-friendly interface, and is popular among creative professionals.
    - Linux: Linux is an open-source operating system that is available for free and can be customized to meet specific needs. It is used by developers and

businesses, as well as individuals who prefer an open-source, customizable operating system.

Operating System Functions

- **Process Management:** An operating system manages the processor's work by allocating various jobs to it and ensuring that each process receives enough time from the processor to function properly.
- **Memory Management:** An operating system manages the allocation and de-allocation of the memory to various processes and ensures that the other process does not consume the memory allocated to one process.
- **Device Management:** There are various input and output devices. An OS controls the working of these input-output devices. It receives the requests from these devices, performs a specific task, and communicates back to the requesting process.
- **File Management:** An operating system keeps track of information regarding the creation, deletion, transfer, copy, and storage of files in an organized way. It also maintains the integrity of the data stored in these files, including the file directory structure, by protecting against unauthorized access.
- **Security:** The operating system provides various techniques which assure the integrity and confidentiality of user data. Following security measures are used to protect user data:
    - Protection against unauthorized access through login.
    - Protection against intrusion by keeping Firewall active.
    - Protecting the system memory against malicious access.
    - Displaying messages related to system vulnerabilities.
- **Error Detection:** From time to time, the operating system checks the system for any external threat or malicious software activity. It also checks the hardware for any type of damage. This process displays several alerts to the user so that the appropriate action can be taken against any damage caused to the system.
- **Job Scheduling**: In a multitasking OS where multiple programs run simultaneously, the operating system determines which applications should run in which order and how time should be allocated to each application.

Features of Operating Systems

- Here is a list of some important features of an operating system:
    1. Provides a platform for running applications
    2. Handles memory management and CPU scheduling

3. Provides file system abstraction
4. Provides networking support
5. Provides security features
6. Provides user interface
7. Provides utilities and system services
8. Supports application development

Unix-like systems Vs Windows systems

Operating systems serve as the foundation for computer systems, enabling users to interact with hardware, run applications, and manage files.

What Is UNIX?

UNIX is a multitasking, multi-user operating system developed for workstations, servers, and other devices. Its numerous applications include database management, software development, and networked applications.

Systems equipped with UNIX are preferred for their security, flexibility, and stability. This operating system is the basis for numerous others, including macOS and Linux.

UNIX was initially developed in the 1970s and was one of the first operating systems written using the C programming language. It is one of the most effective operating systems ever created, with numerous offshoots and a wide-reaching effect on the computing and electronics industries.

Its main advantages include stability, interoperability, and portability across multiple heterogeneous environments and devices.

Today, UNIX and its variants are leveraged for multiple IT systems, including servers, workstations, mobile devices, embedded systems, and supercomputers. However, while Linux, macOS, and Android–arguably the most popular UNIX spin-offs–continue to maintain a strong presence in their respective markets, the demand for UNIX itself has declined.

What Is Linux?

The **Linux Operating System** is a type of operating system that is similar to Unix, and it is built upon the Linux Kernel.

Linux is an open-source operating system available free of cost and based on UNIX. It is widely leveraged across various devices for stability, flexibility, and security

The Linux Kernel is like the brain of the operating system because it manages how the computer interacts with its hardware and resources. It makes sure everything works smoothly and efficiently. But the Linux Kernel alone is not enough to make a complete operating system

To create a full and functional system, the Linux Kernel is combined with a collection of software packages and utilities, which are together called Linux distributions.

These distributions make the Linux Operating System ready for users to run their applications and perform tasks on their computers securely and effectively. Linux distributions come in different flavors, each tailored to suit the specific needs and preferences of users.

What Is Windows?

Microsoft Windows is an operating system that features a graphical user interface and compatibility with a wide range of hardware and software, primarily for personal computers.

Every computer user has heard of Microsoft Windows, with over 75% of desktop and laptop computers worldwide using this operating system. Available in 32- and 64-bit versions, Windows is a user-friendly operating system that features a graphical user interface (GUI), virtual memory management, support for multiple peripherals, and multitasking functionalities.

Windows provides both server and client versions. Popular client versions include Windows 98, ME, XP, Vista, 7, 8, and 10. Windows 11 is the newest version of this operating system, released in 2021. Server versions include NT Server, 2000 Server, 2003 Server, Server 2008 R2, Server 2016, and Server 2022 (latest version).

Linux distributions and UIs

A Linux distribution, commonly known as a Linux distro, refers to a complete operating system that is based on the Linux kernel and bundled with various software packages. Moreover, a Linux distro can be explained as follows:

A Linux distribution is a complete operating system that includes the Linux kernel, system libraries, utilities, application software, and a package management system. It is created by assembling various software components from different sources and packaging them together to provide a cohesive and user-friendly computing environment.

Main Characteristics and Components of a Linux Distribution

1. **Linux Kernel**: The Linux kernel serves as the core component of the operating system, providing low-level functionality, hardware abstraction, and device drivers.
2. **System Libraries**: Linux distributions include a set of system libraries, such as the GNU C Library (glibc), which provide essential functions and interfaces for applications to interact with the underlying operating system.
3. **User Interface**: Linux distributions offer different user interfaces, including graphical user interfaces (GUIs) like GNOME, KDE, or Xfce, as well as command-line interfaces (CLIs). These interfaces allow users to interact with the system and run applications.
4. **Software Packages**: Linux distributions come with a collection of software packages, including productivity tools, web browsers, email clients, media players, development tools, and more. These packages are typically managed and installed through a package management system.
5. **Package Management System**: Linux distributions utilize a package management system to install, update, and remove software packages.
6. **Configuration and Customization**: Linux distributions provide tools and utilities for configuring various aspects of the system, such as network settings, display preferences, user accounts, and security options. Users can customize the operating system to suit their specific needs and preferences.

7. **Support and Community**: Linux distributions are often backed by vibrant communities and support forums where users can seek assistance, share knowledge, and contribute to the development and improvement of the distribution.

What is a "distribution?"

Linux distribution is an operating system that is made up of a collection of software based on Linux kernel or you can say distribution contains the Linux kernel and supporting libraries and software. And you can get Linux-based operating system by downloading one of the Linux distributions and these distributions are available for different types of devices like embedded devices, personal computers, etc.

Linux distributions provide to a wide range of users and purposes, from general-purpose desktop distributions like Ubuntu and Fedora to specialized distributions focused on specific tasks or environments, such as Kali Linux for penetration testing or CentOS for server deployments. Each distribution may have its own goals, target audience, default software selection, and release cycle, allowing users to choose the distribution that best fits their requirements and preferences.

- MX Linux
- Manjaro
- Linux Mint
- elementary
- Ubuntu
- Debian
- Solus
- Fedora
- openSUSE
- Deepin

Overall, Linux distributions offer a flexible, customizable, and open-source alternative to proprietary operating systems, empowering users with the ability to tailor their computing environment according to their specific needs.

File system hierarchy and standard

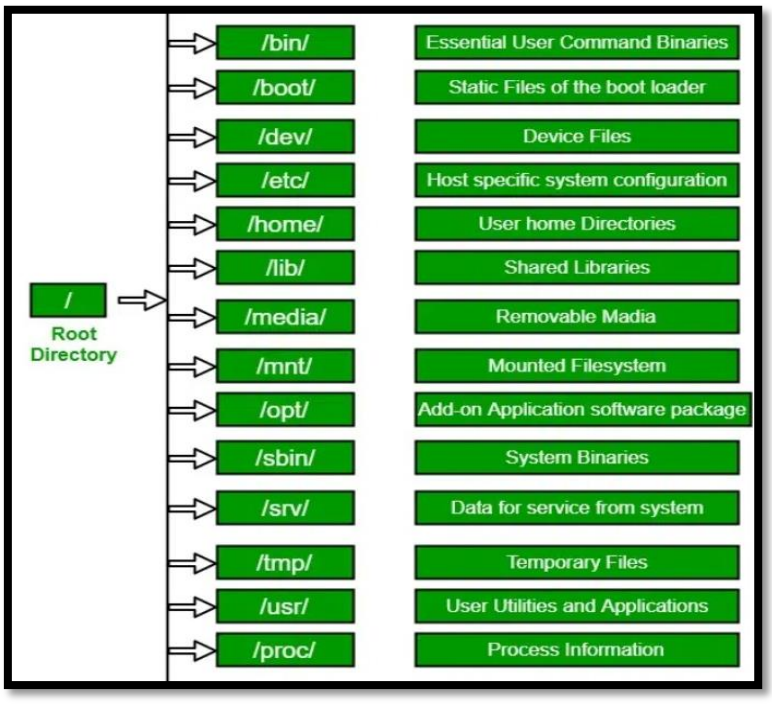
The Linux File Hierarchy Structure or the ***File system Hierarchy Standard*** (FHS) defines the directory structure and directory contents in Unix-like operating systems. It is maintained by the Linux Foundation.

1.1.1.1 Single-rooted hierarchy, seamless and extensible file systems

The root file system is the top of the hierarchical file tree. It contains the files and directories critical for system operation, including the device directory and programs for booting the system. The root file system also contains mount points where file systems can be mounted to connect to the root file system hierarchy.

In the FHS, all files and directories appear under the **root directory /**, even if they are stored on different physical or virtual devices.

Most of these directories exist in all UNIX operating systems and are generally used in much the same way; however, the descriptions here are those used specifically for the FHS and are not considered authoritative for platforms other than Linux.

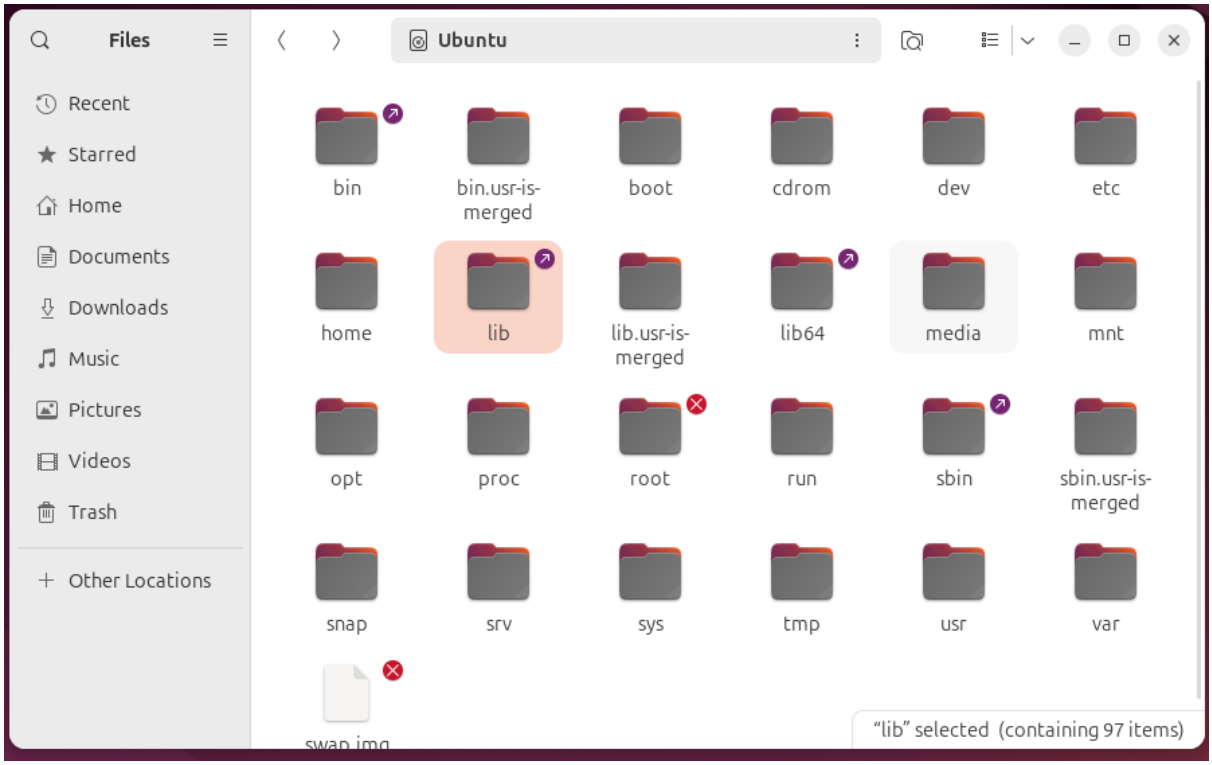


Linux-directory

The Filesystem Hierarchy Standard (FHS) defines the structure of file systems on Linux and other UNIX-like operating systems. However, Linux file systems also contain some directories that aren't yet defined by the standard.

/ – The Root Directory

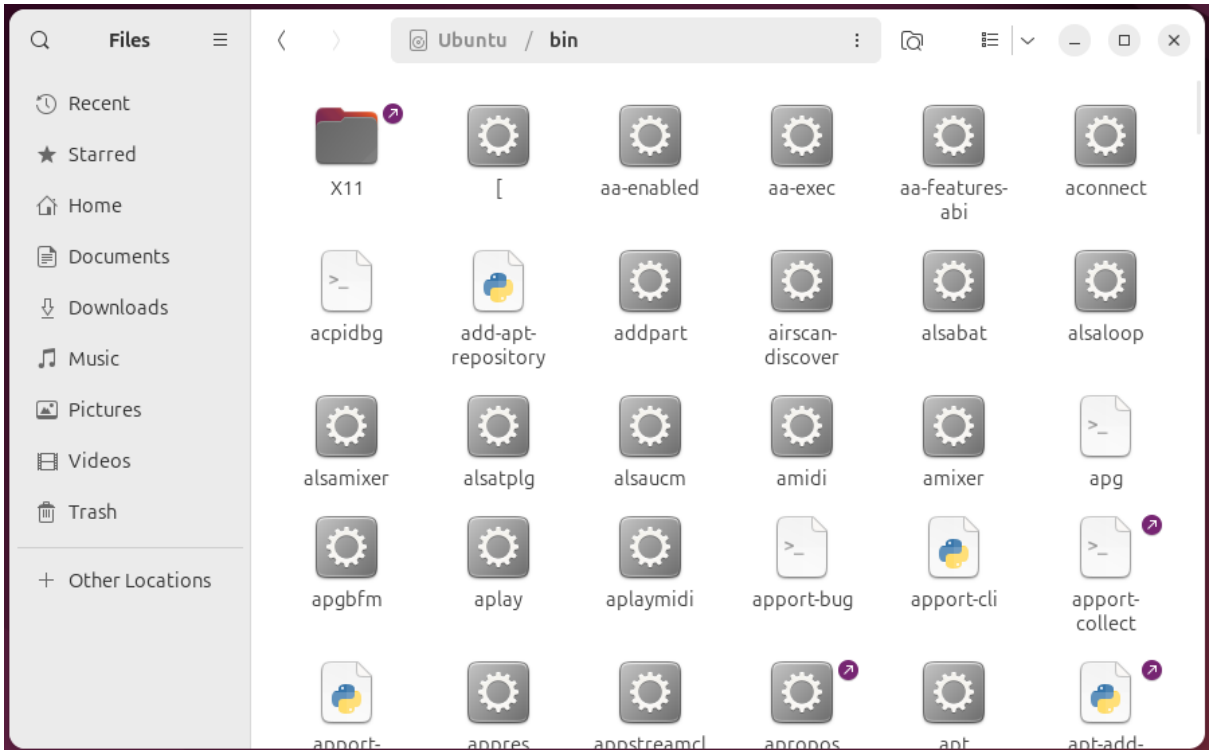
Everything on your Linux system is located under the / directory, known as the root directory. You can think of the / directory as being similar to the C:\ directory on Windows—but this isn't strictly true, as Linux doesn't have drive letters. While another partition would be located at D:\ on Windows, this other partition would appear in another folder under / on Linux.



/bin – Essential User Binaries

The /bin directory contains the essential user binaries (programs) that must be present when the system is mounted in single-user mode. Applications such as Firefox, if they aren't installed as Snaps, are stored in /usr/bin, while important system programs and utilities such as the bash shell are located in /bin. The /usr directory may be stored on another partition. Placing these files in the /bin directory ensures the system will have

these important utilities even if no other file systems are mounted. The /sbin directory is similar: it contains essential system administration binaries.



/boot – Static Boot Files

The /boot directory contains the files needed to boot the system. For example, the GRUB boot loader's files and your Linux kernels are stored here. The boot loader's configuration files aren't located here, though; they're in /etc with the other configuration files.

/cdrom – Historical Mount Point for CD-ROMs

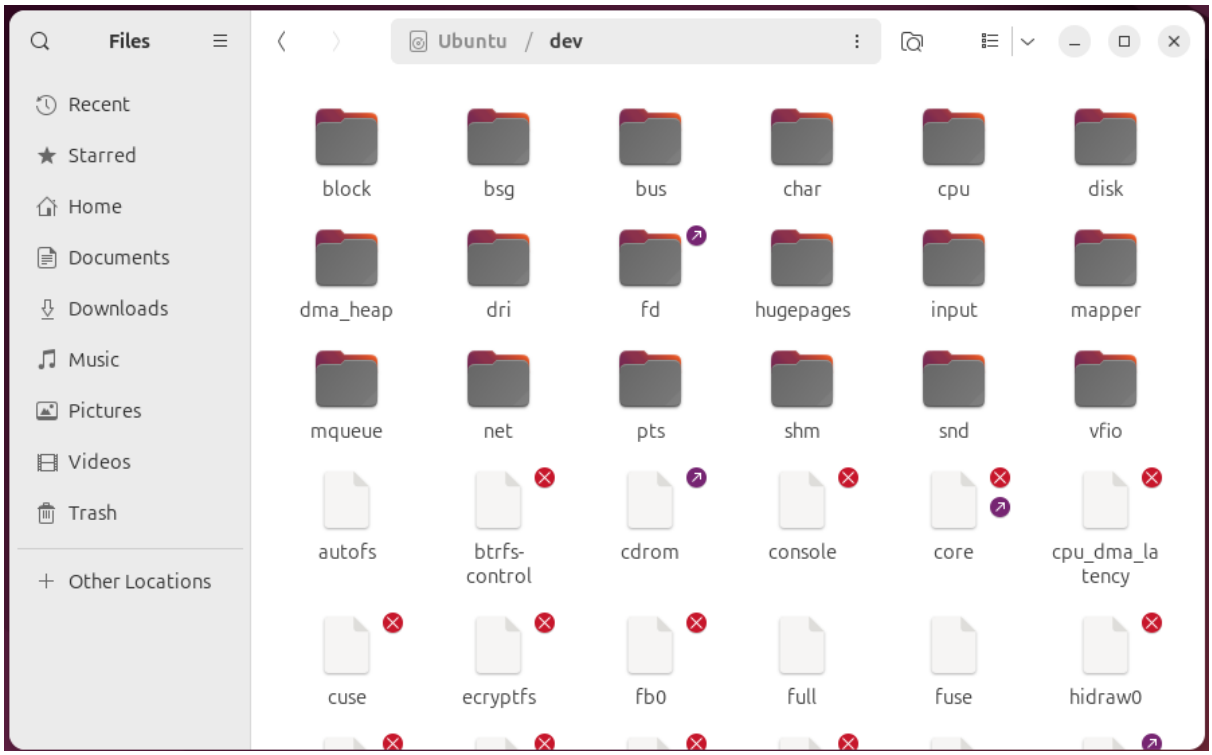
The /cdrom directory isn't part of the FHS standard, but you'll still find it on Ubuntu and other operating systems. It's a temporary location for CD-ROMs inserted in the system. However, the standard location for temporary media is inside the /media directory.

/dev – Device Files

Linux exposes devices as files, and the /dev directory contains a number of special files that represent devices. These are not actual files as we know them, but they appear as files. For example, /dev/sda represents the first SATA drive in the system. If you wanted to partition it, you could start a partition editor and tell it to edit /dev/sda.

This directory also contains pseudo-devices, which are virtual devices that don't actually

correspond to hardware. For example, /dev/random produces random numbers. /dev/null is a special device that produces no output and automatically discards all input; when you pipe the output of a command to /dev/null, you discard it.

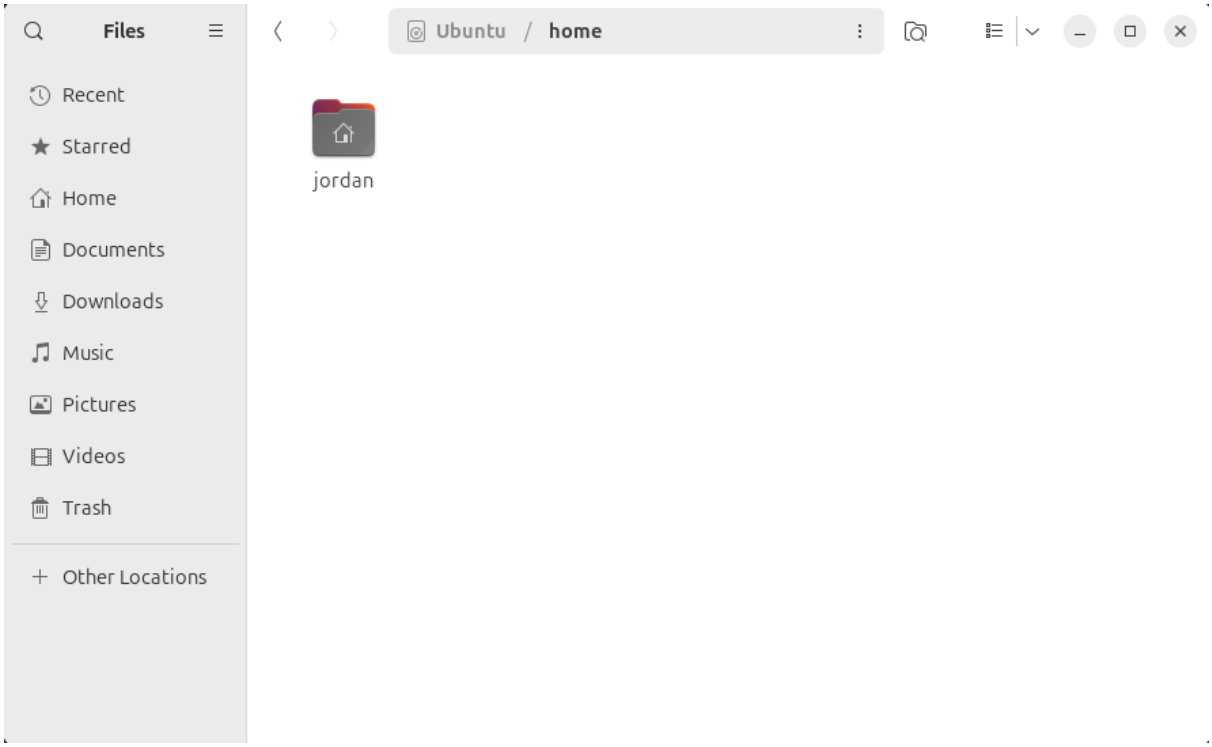


/etc – Configuration Files

The /etc directory contains configuration files, which can generally be edited by hand in a text editor. Note that the /etc/ directory contains system-wide configuration files. User-specific configuration files are located in each user's home directory.

/home – Home Folders

The /home directory contains a home folder for each user. For example, if your user name is bob, you have a home folder located at /home/bob. This home folder contains the user's data files and user-specific configuration files. Each user only has write access to their own home folder and must obtain elevated permissions (become the root user) to modify other files on the system.



/lib – Essential Shared Libraries

The /lib directory contains libraries needed by the essential binaries in the /bin and /sbin folder. Libraries needed by the binaries in the /usr/bin folder are located in /usr/lib. You'll also see a counterpart /lib64 folder on 64-bit systems.

/lost+found – Recovered Files

Each Linux file system has a lost+found directory. If the file system crashes, a file system check will be performed at next boot. Any corrupted files found will be placed in the lost+found directory, so you can attempt to recover as much data as possible.

/media – Removable Media

The /media directory contains subdirectories where removable media devices inserted into the computer are mounted. For example, when you insert a CD into your Linux system, a directory will automatically be created inside the /media directory. You can access the contents of the CD inside this directory.

/mnt – Temporary Mount Points

Historically speaking, the /mnt directory is where system administrators mounted temporary file systems while using them. For example, if you're mounting a Windows

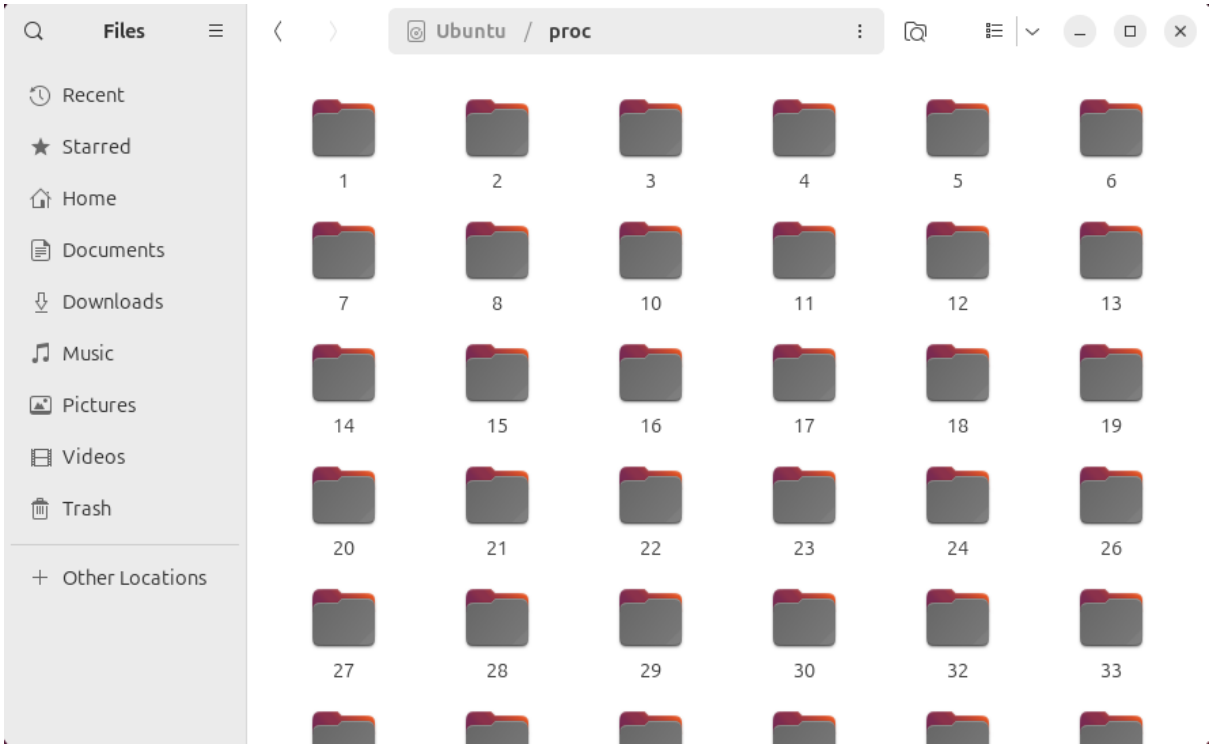
partition to perform some file recovery operations, you might mount it at /mnt/windows. However, you can mount other file systems anywhere on the system.

/opt – Optional Packages

The /opt directory contains subdirectories for optional software packages. It's commonly used by proprietary software that doesn't obey the standard file system hierarchy. For example, a proprietary program might dump its files in /opt/application when you install it.

/proc – Kernel and Process Files

The /proc directory similar to the /dev directory because it doesn't contain standard files. It contains special files that represent system and process information.



/root – Root Home Directory

The /root directory is the home directory of the root user. Instead of being located at /home/root, it's located at /root. This is distinct from /, which is the system root directory.

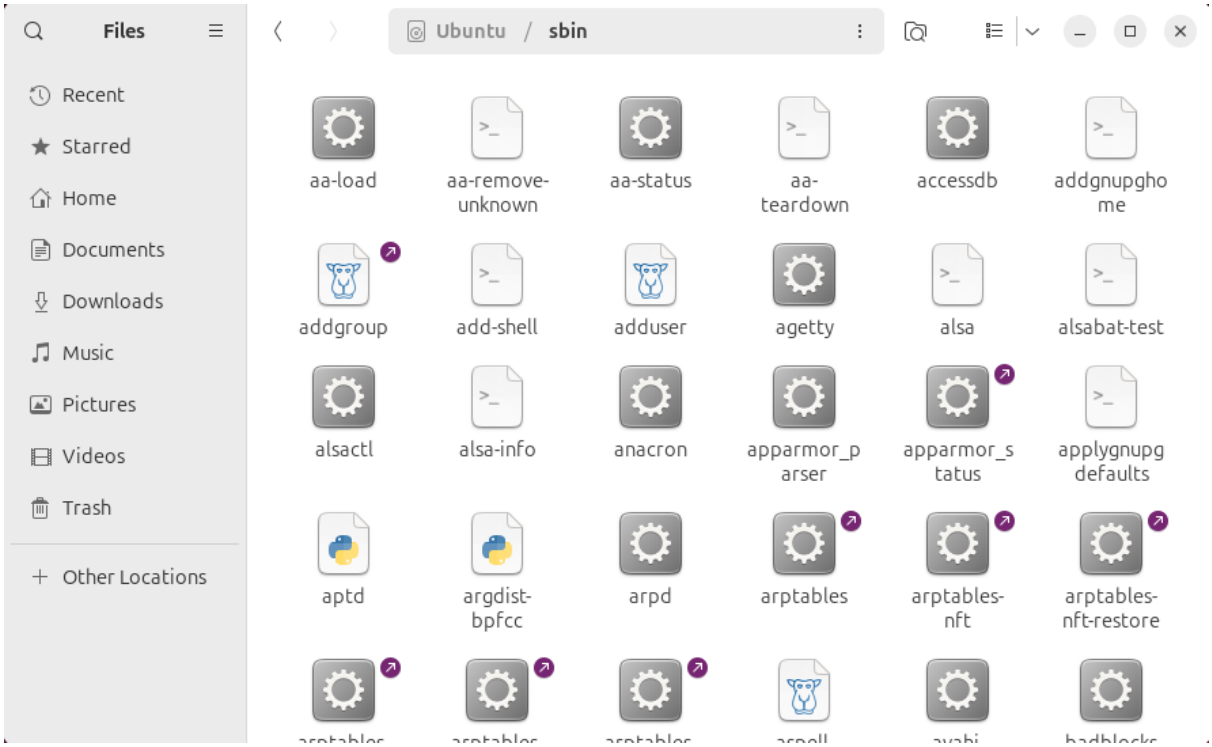
/run – Application State Files

The /run directory gives applications a standard place to store transient files they require like sockets and process IDs. These files can't be stored in /tmp because files in /tmp may

be deleted.

/sbin – System Administration Binaries

The /sbin directory is similar to the /bin directory. It contains essential binaries that are generally intended to be run by the root user for system administration.



/snap – Storage for Snap Packages

Another directory that isn't part of the FHS but is common to see these days is /snap. It holds installed Snap packages and other files associated with Snap. Ubuntu now uses Snaps by default, but if you're using a different distro that doesn't, you won't see this directory.

/srv – Service Data

The /srv directory contains "data for services provided by the system." If you were using the Apache HTTP server to serve a website, you'd likely store your website's files in a directory inside the /srv directory.

/tmp – Temporary Files

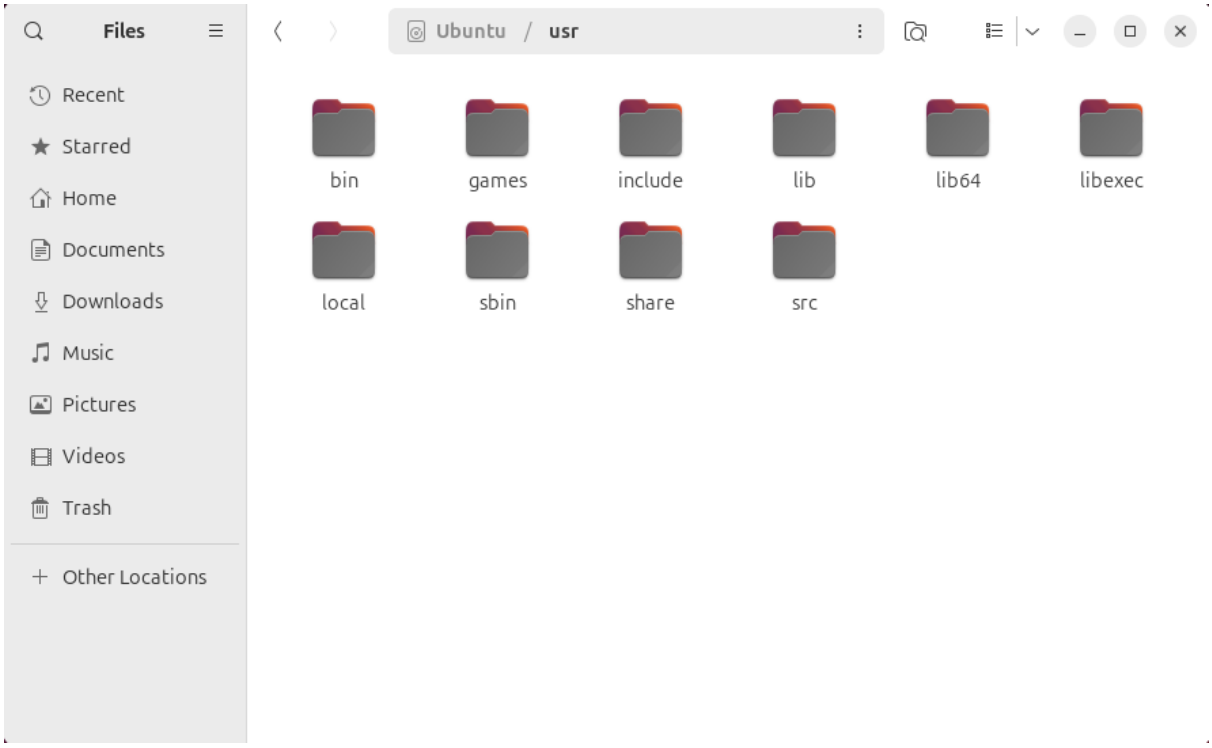
Applications store temporary files in the /tmp directory. These files are generally deleted whenever your system is restarted and may be deleted at any time by utilities such as

systemd-tmpfiles.

/usr – User Binaries & Read-Only Data

The /usr directory contains applications and files used by users, as opposed to applications and files used by the system. For example, non-essential applications are located inside the /usr/bin directory instead of the /bin directory and non-essential system administration binaries are located in the /usr/sbin directory instead of the /sbin directory. Libraries for each are located inside the /usr/lib directory. The /usr directory also contains other directories. For example, architecture-independent files like graphics are located in /usr/share.

The /usr/local directory is where locally compiled applications install to by default. This prevents them from mucking up the rest of the system.



/var – Variable Data Files

The /var directory is the writable counterpart to the /usr directory, which must be read-only in normal operation. Log files and everything else that would normally be written to /usr during normal operation are written to the /var directory. For example, you'll find log files in /var/log.

Mounting additional file systems

Before you access the files on a file system, you need to mount the file system. Mounting a file system attaches that file system to a directory (mount point) and makes it available to the system. The root (/) file system is always mounted. Any other file system can be connected or disconnected from the root (/) file system.

When you mount a file system, any files or directories in the underlying mount point directory are unavailable as long as the file system is mounted. These files are not permanently affected by the mounting process, and they become available again when the file system is unmounted. However, mount directories are typically empty, because you usually do not want to unclear existing files.

Advantages of Linux

- The main advantage of Linux is it is an open-source operating system. This means the source code is easily available for everyone and you are allowed to contribute, modify and distribute the code to anyone without any permissions.
- In terms of security, Linux is more secure than any other operating system. It does not mean that Linux is 100 percent secure, it has some malware for it but is less vulnerable than any other operating system. So, it does not require any anti-virus software.
- The software updates in Linux are easy and frequent.
- Various Linux distributions are available so that you can use them according to your requirements or according to your taste.
- Linux is freely available to use on the internet.
- It has large community support.
- It provides high stability. It rarely slows down or freezes and there is no need to reboot it after a short time.
- It maintains the privacy of the user.
- The performance of the Linux system is much higher than other operating systems. It allows a large number of people to work at the same time and it handles them efficiently.
- It is network friendly.

- The flexibility of Linux is high. There is no need to install a complete Linux suite; you are allowed to install only the required components.
- Linux is compatible with a large number of file formats.
- It is fast and easy to install from the web. It can also install it on any hardware even on your old computer system.
- It performs all tasks properly even if it has limited space on the hard disk.

Disadvantages of Linux

- It is not very user-friendly. So, it may be confusing for beginners.
- It has small peripheral hardware drivers as compared to windows.

File system object-oriented design and file system standard

A file is a collection of related information. The file system resides on secondary storage and provides efficient and convenient access to the disk by allowing data to be stored, located, and retrieved.

File system implementation in an operating system refers to how the file system manages the storage and retrieval of data on a physical storage device such as a hard drive, solid-state drive, or flash drive.

The file system implementation includes several components, including:

1. **File System Structure:** The file system structure refers to how the files and directories are organized and stored on the physical storage device.
2. **File Allocation:** The file allocation mechanism determines how files are allocated on the storage device. This can include allocation techniques such as contiguous allocation, linked allocation, indexed allocation, or a combination of these techniques.
3. **Data Retrieval:** The file system implementation determines how the data is read from and written to the physical storage device. This includes strategies such as buffering and caching to optimize file I/O performance.
4. **Security and Permissions:** The file system implementation includes features for managing file security and permissions. This includes access control lists (ACLs), file permissions, and ownership management.

5. **Recovery and Fault Tolerance:** The file system implementation includes features for recovering from system failures and maintaining data integrity. This includes techniques such as journaling and file system snapshots.

File system implementation is a critical aspect of an operating system as it directly impacts the performance, reliability, and security of the system. Different operating systems use different file system implementations based on the specific needs of the system and the intended use cases. Some common file systems used in operating systems include **NTFS** and **FAT** in Windows, and **ext4**, **HFS** and **ZFS** in Linux.

Key Steps Involved In File System Implementation

File system implementation is a crucial component of an operating system, as it provides an interface between the user and the physical storage device. Here are the key steps involved in file system implementation:

1. **Partitioning the storage device:** The first step in file system implementation is to partition the physical storage device into one or more logical partitions. Each partition is formatted with a specific file system that defines the way files and directories are organized and stored.
2. **File system structures:** File system structures are the data structures used by the operating system to manage files and directories. Some of the key file system structures include the superblock, inode table, directory structure, and file allocation table.
3. **Allocation of storage space:** The file system must allocate storage space for each file and directory on the storage device. There are several methods for allocating storage space, including contiguous, linked, and indexed allocation.
4. **File operations:** The file system provides a set of operations that can be performed on files and directories, including create, delete, read, write, open, close, and seek. These operations are implemented using the file system structures and the storage allocation methods.
5. **File system security:** The file system must provide security mechanisms to protect files and directories from unauthorized access or modification. This can be done by setting file permissions, access control lists, or encryption.

6. **File system maintenance:** The file system must be maintained to ensure efficient and reliable operation. This includes tasks such as disk defragmentation, disk checking, and backup and recovery.

Overall, file system implementation is a complex and critical component of an operating system. The efficiency and reliability of the file system have a significant impact on the performance and stability of the entire system.

Unix file and directory permissions

Unix-like operating systems, such as Linux, running on shared high performance computers use settings called permissions to determine who can access and modify the files and directories stored in their file systems. Each file and directory in a file system is assigned "**owner**" and "**group**" attributes.

Most commonly, by default, the user who ***creates a file or directory*** is set as **owner** of that file or directory. When needed the system's root administrator can change the user attribute for files and directories.

The **group** designation can be used to grant **teammates** and/or **collaborators** shared access to an owner's files and directories, and provide a convenient way to grant access to multiple users.

File and directory permissions

In the output example below, the first character in each line indicates whether the listed object is a file or a directory. Directories are indicated by a (d); the absence of a d at the beginning of the first line indicates that myfile.txt is a regular file.

Example; - rw - r- - r - - abc

drwxr - xr –x bag

The letters **rwx** represent different permission levels:

Permission	Files	Directories
r	can read the file	can `ls` the directory
w	can write the file	can modify the directory's contents
x	can execute the file	can `cd` to the directory

Note the multiple instances of r, w, and x. These are grouped into three sets that represent different levels of ownership:

- **Owner or user permissions:** After the directory (d) slot, the first set of three characters indicate permission settings for the **owner** (also known as the user).

  In the example -rw-r--r--, the owner permissions are rw-, indicating that the owner can read and write to the file but can't execute it as a program.

  In the example drwxr-xr-x, the owner permissions are rwx, indicating that the owner can view, modify, and enter the directory.

- **Group permissions:** The second rwx set indicates the **group** permissions.

  In the example -rw-r--r--, group members can only read the file.

  In the example drwxr-xr-x, group members can view as well as enter the directory.

- **Other permissions:** The final rwx set is for "**other**". This is anyone outside the group. In both examples above, these are set to the same permissions as the group.

Changing file permissions

To change file and directory permissions, use the command chmod (change mode). The owner of a file can change the permissions for user (u), group (g), or others (o) by adding (+) or subtracting (-) the read, write, and execute permissions.

There are two basic ways of using chmod to change file permissions: The **symbolic** method and the **absolute** form.

Symbolic method

The first and probably easiest way is the relative (or symbolic) method, which lets you specify permissions with single letter abbreviations. A chmod command using this method consists of at least three parts from the following lists:

Access class	Operator	Access Type
u (user)	+ (add access)	r (read)
g (group)	- (remove access)	w (write)
o (other)	= (set exact access)	x (execute)
a (all: u, g, and o)		

For example, To add permission for everyone to read a file in the current directory named `myfile`, is ***chmod a+r myfile***

You can also specify multiple classes and types with a single command. For example, to remove read and write permission for group and other users (leaving only yourself with read and write permission) on a file named myfile, is ***chmod go-rw myfile***

You can also specify that different permissions be added and removed in the same command. For example, to remove write permission and add execute for all users on myfile, is ***chmod a –w+x myfile***

Absolute form

The other way to use the `chmod` command is the absolute form, in which you specify a set of three numbers that together determine all the access classes and types. Rather than being able to change only particular attributes, you must specify the entire state of the file's permissions.

The three numbers are specified in the order: user (or owner), group, and other. Each number is the sum of values that specify read, write, and execute access:

Permission	Number
Read (r)	4
Write (w)	2
Execute (x)	1

Add the numbers of the permissions you want to give; for example:

- For file myfile, to grant read, write, and execute permissions to yourself (4+2+1=7), read and execute permissions to users in your group (4+0+1=5), and only execute permission to others (0+0+1=1), you would use: ***chmod 751 myfile.***
- To grant read, write, and execute permissions on the current directory to yourself only, you would use: ***chmod 700 myfile***

The three digit sequence as the sum of attributes you select from the following table:

Read by owner	400
Write by owner	200
Execute by owner	100
Read by group	040
Write by group	020
Execute by group	010
Read by others	004
Write by others	002
Execute by others	001

Sum all the accesses you wish to permit. For example, to give write and execute privileges to the owner of myfile (200+100=300), and give read privileges to all (400+040+004=444), you would enter: ***chmod 744 myfile***

Some other examples are:

777	anyone can do anything (read, write, or execute)
755	you can do anything; others can only read and execute
711	you can do anything; others can only execute
644	you can read and write; others can only read

Essential shell commands

A shell is a special user program that provides an interface to the user to use operating system services. Shell accepts human-readable commands from the user and converts them into something which the kernel can understand. It is a command language interpreter that executes commands read from input devices such as keyboards or from files.

The shell gets started when the user logs in or starts the terminal.

Displaying the file contents on the terminal:

- **cat**: It is generally used to concatenate the files. It gives the output on the standard output.
- **more**: It is a filter for paging through text one screen full at a time.
- **less**: It is used to viewing the files instead of opening the file. Similar to *more* command but it allows backward as well as forward movement.
- **head** : Used to print the first N lines of a file. It accepts N as input and the default value of N is 10.

- **tail** : Used to print the last N-1 lines of a file. It accepts N as input and the default value of N is 10. etc.

Basic file manipulation commands and directory navigation commands

Here, we want to look at the commands used to manipulate files. We are going to copy, move, and rename files, plus a few commands for reading files in different ways.

1. **cp** : cp file1to file2
   This command will copy the files and directories from the source path to the destination path. It can copy a file/directory with the new name to the destination path. It accepts the source file/directory and destination file/directory.
2. **mv** : mv file1to file2
   Used to move files or directories. This command's working is almost similar to *cp* command but it deletes a copy of the file or directory from the source path.
3. **rm** : rm [option] filename, Used to remove files or directories.
4. **touch** : touch filename, Used to create or update a file. etc.

Advanced file manipulation commands (Init, processes, and threads)

The term file Manipulation refers to the process of creating, modifying, and deleting files and directories. In Linux, not only files but directories, devices, processes, sockets, and everything else is a file. All these files and directories are organized in a hierarchical tree-like structure, with the root directory ("/") being the topmost level.

This makes understanding basic file manipulation commands important for Linux users since it covers the way through which we can navigate our Linux systems from the command line.

Advanced commands provide greater customization and controls in more specialized situations once you become familiar with basic commands. Like:

Init is parent of all Linux processes with PID or process ID of 1. It is the first process to start when a computer boots up and runs until the

system shuts down. init stands for initialization. In simple words the role of init is to create processes from script stored in the file /etc/inittab which is a configuration file which is to be used by initialization system. It is the last step of the kernel boot sequence.

/etc/init tab Specifies the **init** command control file.

- init script initializes the service. So, it responsible for initializing the system.
- Init scripts are also called rc scripts (run command scripts)
- Init script is also used in UNIX.

A **process** in Linux is nothing but a program in execution. It's a running instance of a program. Any command that you execute starts a process.

In Linux processes can be of two types:

- **Foreground Processes**
  depend on the user for input
  also referred to as interactive processes
- **Background Processes**
  run independently of the user
  referred to as non-interactive or automatic processes

A **thread** is a lightweight unit of execution within a process that can operate independently. It allows for concurrent execution of tasks, enabling parallel processing and efficient resource utilization. Threads share the same memory space and resources with other threads in the process, making data sharing and communication between threads seamless.

Advanced shell features

A shell is a program that acts as an interface to users, for the operating system. That is, it exposes the operating system to human users. These shells could either be command-line interfaces (CLI) or graphical user interfaces (GUI).
The Linux shell is a command-line interface that allows users to interact with the operating system and execute various built-in commands. It provides an interface between the user and the kernel to execute programs known as commands. In other words, the shell is the primary interface that provides users

with a way to communicate with the Linux operating system at a more fundamental level than a GUI.

Some basic commands in the Linux shell include- ls, cd, mkdir, touch, cp, mv, rm, etc. With these basic commands, the Linux shell provides a wide range of advanced commands and features that allow users to perform more complex tasks. These include pipes, which allow users to connect the output of one command to the input of another command. And also redirection, which allows users to redirect input or output to a file.

There are many commands available in the Linux shell. Each command has its own set of options and arguments. Some of the popularly used lists of commands are as follows:

1. **cd: Change directory. This command is used to change the current working directory.**
2. **ls: List directory contents. This command is used to list the files and directories in the current working directory.**
3. **mkdir: Make directory. This command is used to create a new directory.**
4. **rm: Remove file or directory. This command is used to delete files or current directories.**
5. **cp: Copy file or directory. This command is used to copy files or current directories.**
6. **mv: Move or rename a file or current directory. This command prompt is used to move files or directories to a new location or rename them.**
7. **cat: Concatenate and display files. This command is used to display the contents of an executable file.**
8. **grep: Search for a pattern in a file extension. This command is used to search for a specific string or pattern in a file type at the current time.**
9. **pwd: Print working directory. This simple command is used to display the current working directory.**
10. **man: Display manual pages. This command is used to display the manual pages for a specific command.**

Overall, the Linux shell is a powerful tool that provides a text-based interface for users to interact with the operating system and has a complete control structure and environment variables. It allows users to perform various tasks more quickly

and efficiently, provides greater flexibility and control, and can be used remotely. The Linux shell is based on a scripting language called Bash and provides a wide range of commands and features.

In Linux, a single user account generally belongs to a single user. The permissions for this user account determine the level of access the user will have while using the system.

Types of user accounts

1. **Super User**: This account is also known as the root account. It has all the permissions and can run all the commands without any restriction. The keyword `sudo` is often used to run a command as the root user.
2. **Normal User**: The general users of the system are assigned this account. This account has restricted access to the commands and files of the system. The root user can create such an account, modify its permissions or delete it.
3. **System User**: This is the type of account that is created only for a specific purpose or software. For example, a mail account.

2.1 Account and security administration

Account and security administration involves managing user accounts, groups, and implementing security measures to protect the system.

2.1.1 User and group concepts, and user private group scheme

User and Group concept

In a computer system, users are individuals who interact with the system, while groups are collections of users with similar access requirements. Here are some important concepts related to users and groups:

- ✓ User accounts: User accounts are created for individual users to access the system. Each user is assigned a unique username and password, which they use to authenticate themselves.
- ✓ Group accounts: Group accounts are created to manage access permissions collectively for a group of users. Users who are part of a group inherit the access rights and permissions associated with that group.

User identification (UID) and group identification (GID): Each user and group is assigned a unique identification number (UID and GID, respectively) by the system. These IDs are used internally by the system to identify and manage users and groups.

User Private Group Scheme:

The User Private Group (UPG) scheme is a common approach in Unix-like operating systems (such as Linux) to manage user and group administration. In this scheme:

- ✓ Each user is associated with a primary group, which has the same name as the username and the same GID.
- ✓ A private group is created for each user, with the same name as the primary group and a unique GID.
- ✓ The private group serves as a secondary group for the user and allows for better file and directory permissions management.

The UPG scheme helps enhance security by isolating each user's files and directories within their private group, preventing unauthorized access by other users.

2.1.2 User administration, modifying accounts and group administration

User administration involves creating, modifying, and managing user accounts. Some common tasks in user administration include:

- ✓ Creating user accounts: Administrators can create new user accounts by specifying a username, password, and other relevant details.
- ✓ Modifying user accounts: Administrators can modify user account settings such as the account name, password, home directory, shell, and resource limits.
- ✓ Disabling or deleting user accounts: In certain cases, user accounts may need to be disabled or deleted to revoke access.

Administrators typically use command-line tools or graphical interfaces provided by the operating system to perform user administration tasks.

Group Administration

Group administration involves creating and managing groups to simplify access control. Key tasks in group administration include:

- ✓ Creating groups: Administrators can create new groups and assign them a unique GID.
- ✓ Modifying groups: Admins can modify group settings such as the group name or GID.
- ✓ Adding or removing users from groups: Users can be added or removed from groups to manage their access permissions.

Group administration allows for the efficient management of access rights by assigning permissions to groups rather than individual users, simplifying the overall security management process.

2.1.3 Password aging and default user files

To enhance security, systems often implement password aging policies and provide default user files. Here's what these concepts involve:

- ✓ Password aging: Password aging refers to the practice of enforcing periodic password changes. It involves setting a maximum password age, after which users are prompted to change their passwords.
- ✓ Default user files: Default user files are preconfigured files or directories provided to users when their accounts are created. These files often include initial configuration settings, default preferences, and skeleton files for user-specific customization.

Password aging helps to ease the risk of compromised passwords by ensuring that users regularly update their passwords. Default user files provide a consistent starting point for user accounts, ensuring that users have necessary files and configurations readily available when they log in for the first time.
Overall, effective account and security administration, including user and group management, password aging, and default user files, contribute to maintaining the integrity, confidentiality, and availability of computer systems and protecting against unauthorized access.

2.2 Managing files and folder permission

Managing files and folder permissions, controlling access to files, and managing disk quotas are important aspects of system administration that help ensure proper data security, access control, and resource management.

2.2.1 Managing file ownership

File and folder permissions determine who can access, modify, and execute files and folders on a system. These permissions are typically represented by a combination of read (r), write (w), and execute (x) permissions, which can be assigned to three different groups: the owner of the file, the group associated with the file, and others (everyone else).

To manage file and folder permissions, administrators can use commands like chmod (change mode) in Unix-like systems or the Security tab in Windows systems. Key tasks include:

- ✓ Setting permissions: Administrators can assign appropriate permissions to files and folders, specifying which users or groups can read, write, or execute them.
- ✓ Modifying permissions: Administrators can modify existing permissions to grant or revoke access rights as needed.
- ✓ Viewing permissions: Administrators can view the current permissions assigned to files and folders to ensure proper access control.

2.2.2 Controlling access to files

File ownership refers to determining the user and group associated with a file or folder. Administrators can manage file ownership to control access and permissions. Key tasks include:

- ✓ Changing file ownership: Administrators can change the owner and group associated with a file using commands like chown (change owner) or chgrp (change group) in Unix-like systems.
- ✓ Transferring ownership: Administrators can transfer file ownership from one user to another, ensuring that the appropriate user has control over the file and its permissions.

Controlling access to files involves setting up access control lists (ACLs) or managing file permissions to restrict or grant access to specific users or groups. Key tasks include:

- ✓ Setting up ACLs: ACLs allow administrators to define fine-grained access control, specifying access permissions for individual users or groups on a file or folder basis.
- ✓ Assigning special permissions: Administrators can assign special permissions such as setuid, setgid, or sticky bits to files or folders to control access and execution behavior.

Controlling access to files ensures that only authorized users or groups can access sensitive data, protecting against unauthorized disclosure or modification.

2.2.3 Managing disk quotas

Disk quotas enable administrators to limit the amount of disk space individual users or groups can consume on a file system. This helps in managing disk usage and preventing any individual or group from monopolizing system resources. Key tasks include:

- ✓ Enabling disk quotas: Administrators can enable disk quotas on specific file systems to enforce usage limits.
- ✓ Setting quota limits: Administrators can set quota limits for individual users or groups, specifying the maximum amount of disk space they can consume.
- ✓ Monitoring disk usage: Administrators can monitor disk usage by users or groups to ensure compliance with the specified quotas.

By implementing disk quotas, administrators can prevent excessive disk usage, ensure fair resource allocation, and avoid potential system performance issues.

Proper management of files and folder permissions, file ownership, access control, and disk quotas is crucial for maintaining data security, controlling access, and effectively managing system resources.

Chapter 3

File Systems and Management of Data Storages

In a computer, a file system is the way in which files are named and where they are placed logically for storage and retrieval. Without a file system, stored information wouldn't be isolated into individual files and would be difficult to identify and retrieve. As data capacities increase, the organization and accessibility of individual files are becoming even more important in data storage.

Digital file systems and files are named for and modeled after paper-based filing systems using the same logic-based method of storing and retrieving documents.

File systems can differ between operating systems (OS), such as Microsoft Windows, macOS and Linux-based systems. Some file systems are designed for specific applications. Major types of file systems include ***distributed file systems***, ***disk-based file systems*** and ***special purpose file systems***.

3.1 File system administration

A file system stores and organizes data and can be thought of as a type of index for all the data contained in a storage device. These devices can include hard drives, optical drives and flash drives.

File systems specify conventions for naming files, including the maximum number of characters in a name, which characters can be used and, in some systems, how long the file name suffix can be. In many file systems, file names are not case sensitive.

Along with the file itself, file systems contain information such as the size of the file, as well as its attributes, location and hierarchy in the directory in the metadata. ***Metadata*** can also identify free blocks of available storage on the drive and how much space is available.

A file system also includes a format to specify the path to a file through the structure of directories. A file is placed in a directory -- or a folder in Windows OS -- or subdirectory at the desired place in the tree structure.

PC and mobile OSes have file systems in which files are placed somewhere in a hierarchical tree structure.

3.1.1 Partitioning disks with fdisk and parted command

Creating disk partitions enables you to split your hard drive into multiple sections that act independently.

In Linux, users must structure storage devices (USB and hard drives) before using them. Partitioning is also useful when you are installing multiple operating systems on a single machine.

Before files and directories are created on the storage medium, partitions should be put into place. A ***partition*** is a region of the hard disk or other storage that the OS manages separately. One file system is contained in the primary partition, and some OSes allow for multiple partitions on one disk. In this situation, if one file system gets corrupted, the data in a different partition will be safe.

The **fdisk** and **parted** commands are used to manipulate storage devices. The fdisk and parted commands perform the same function, but parted has more options.

The *fdisk* Command

***fdisk* is a menu-guided program for creating and manipulating partition tables. It understands MS-DOS-type partition tables and other disk labels.** Initially, *fdisk* wasn't designed for large partitions because it couldn't accept GUID (globally unique identifiers) partition tables. But, the recent version supports GPT (GUID partition table) though using MBR (Master Boot Record) (for drives less than 2TB) is recommended.

GPT support is currently new and, therefore, it's deemed an experimental phase. Further, the latest versions of *fdisk* criticized the use of cylinders as the default display unit and MS-DOS compatibility by default. *fdisk* has an interactive

command-line menu that allows us to use letters to specify actions:

```
$ sudo fdisk /dev/sda
```

When using *fdisk*, the sectors must be specified in absolute terms using sector numbers. If we're creating a sector, the size must be set in absolute terms to specify the beginning of that sector. For instance, using 80M as the first sector of a disk indicates that the sector is 80 MiB from the beginning of the disk.

When we're using the absolute position in sector numbers or as positions, the measuring unit (its size) should be in kibibytes (K), mebibytes (M), gibibytes (G), and tebibytes (T). Also, it's possible to express relative terms by preceding the size with either a +size or a -size. For example, a +1G indicates a point 1 GiB after the start sector, while a -100M indicates a point 100 MiB before the last available sector.

fdisk performs partition alignment automatically on the one-megabyte boundary if DOS compatibility mode has been disabled, making it compatible with various Advanced Format HDDs (hard disk drives) and SSDs (Solid-state drives). The default settings should give us accurate alignment.

The *parted* Command

parted is an alternative command to *fdisk*. It's a utility we use to manipulate disk partitions and **helps us to create space for new operating systems, reorganize disk usage, and copy data to new hard disks.** *parted* comes with better functionality than *fdisk*, such as resizing (shrinking and extending partitions). In addition, it supports multiple partition table formats, including MS-DOS and GPT.

We can install it by installing the *parted* package. **It has two modes: command-line and interactive. The interactive mode**

makes partitioning simple as it reduces unnecessary repetition by automatically applying all partitioning commands to the selected device. If we're using the interactive mode, the prompt changes from a dollar sign ($) to (parted):

Fdisk	*parted*

```
$ sudo parted /dev/sda
```

On the other hand, if we're using the command-line mode, we run the *parted* command together with the right options. If the command is successful, it doesn't throw an error and the cursor moves to a new line. We must specify the partition/drive we want to run the command on: the syntax for creating a new partition:

```
$ sudo parted mkpart part-type name fs-type start end
```

Let's have a look at the significant differences between these commands:

When we run *fdisk*, the output is displayed using the partition type.	Uses the file system to display partitions available in a storage device.
Performs partition alignment automatically on the one-megabyte boundary, but MS-DOS compatibility mode has to be disabled.	When creating a partition, might warn about improper partition alignment but doesn't alert when the alignment is proper, and it only aligns the partition start, not the size.
Doesn't give us the option to shrink or extend the partition.	We can extend a partition to utilize the unused disk space or shrink a partition to use its capacity for other purposes.
Only has an interactive mode in which we use letters to specify actions.	Has both interactive and command-line modes. In both of these modes, we have to specify the right commands with the required options.

We've looked at both fdisk and parted commands. We've mentioned the modes each of these commands has and the kind of file systems they can support. The interactive mode should be the most preferred when using parted as it simplifies the process of creating partitions.

fdisk supports automatic alignment only if DOS is turned off while parted only aligns the partition start. fdisk is present in most distros while parted is missing in some distributions and must be installed to be used. Lastly, we should be mindful when using either fdisk or parted as misuse can result in the loss of important data.

3.1.2 Creating, mounting and maintaining file systems

Creating a file system writes information to the device and creates order of the empty space. This file system–related data consumes a small percentage of the space. The remaining space on the disk drive is split into small, consistently sized segments

called blocks. Linux supports a number of file system types, some of which are described as follows.

File system	**Description**
ext2	High performance for fixed disk and removable media
ext3	Journaling version of ext2
ext4	Supports larger files and file system sizes
vfat	MS-DOS file system useful when sharing files between Windows and Linux
XFS	High-performance journaling file system
Btrfs	Addresses scalability requirements of large storage systems

Creating Filesystems

The command to build a Linux file system on a device, or hard disk partition, is ***mkfs***. The syntax for the command is:

```
# mkfs [options] device
```

The mkfs command is actually a front end for the different file system builder utilities such as mkfs.ext2 and mkfs.ext4. These utilities are executable directly from the command line. When using the mkfs wrapper, include the -t fs type option to specify the type of file system to be built. If not specified, the default file system type, ext2, is created.

Mounting File Systems

File systems on different partitions and removable devices, such as CDs, DVDs, or USB flash drives, must be attached to the directory hierarchy to be accessed. To attach a partition or device, a mount point must be created. A mount point is simply a directory created with the mkdir command. After a directory, or mount point, is created, attach the partition by using the mount command. Syntax for the mount command is:

```
# mount [options] device_file mount_point
```

The following example creates a mount point (/test) and attaches the partition:

```
# mkdir /test
# mount /dev/xvdf1 /test
```

To specify mount options, use the –o flag followed by a comma-separated string of options. The following are some of the available options for the mount command:

- auto: Allows the file system to be mounted automatically by using the mount –a command
- loop: Mounts the image as a loop device

- noauto: Disallows the automatic mount of the file system by using the mount –a command
- noexec: Disallows the execution of binary files on the file system
- nouser: Disallows an ordinary user (other than root) to mount and unmount the file system
- remount: Remounts the file system in case it is already mounted
- ro: Mounts the file system for reading only
- rw: Mounts the file system for both reading and writing
- user: Allows an ordinary user (other than root) to mount and unmount the file system

For example, to mount the /dev/xvdf1 partition on the /test mount point as read-only with only the root user able to mount and unmount the file system, enter:

```
# mount –o nouser,ro /dev/xvdf1 /test
```

Unmounting File Systems

To unmount a file system, use the **umount** command. The partition name, the device name, or the mount point is used as an argument.

Example: # umount /dev/xvdd1

umount /test

3.1.3 Swap

Linux divides its physical RAM (random access memory) into chunks of memory called pages. Swapping is the process whereby a page of memory is copied to the preconfigured space on the hard disk, called swap space, to free up that page of memory. The combined size of the physical memory and the swap space is the amount of virtual memory available.

What Is Swap Space?

Swap space is space on a hard drive (HDD or SSD) that represents a substitute for physical (RAM) memory. This feature allows an operating system to temporarily move inactive or less frequently used memory pages from RAM to a designated area on the hard drive and it allows processes to continue running when RAM is fully used and prevents memory errors.

Swap frees up RAM for more important tasks that require more processing power by transferring data to and from a designated disk space. The data interchange is called swapping, while the designated space is called swap space.

Swap space also enables hibernation and safeguards critical processes by temporarily offloading data. However, it should only be a complement to RAM because a system that relies on swap would suffer significant performance degradation.

Operating systems like Windows or Linux provide a certain amount of swap space by default, which users can later change in accordance with

their requirements. Users can also disable swap space, but that means the kernel must kill some processes to create enough free RAM for new processes.

Types of Swap Space

There are two types of swap space in Linux:

1. Traditional Swap Space

   The classic form of swap space that has been in use for decades is traditional swap space. It involves the designation and use of a dedicated partition on a hard drive. A swap partition is formatted specifically for this purpose and is separate from the main system partitions.

   Traditional swap space is suitable for scenarios where you have a dedicated server with specific disk partitions and need a fixed amount of swap space. Using swap space is common in server environments where a portion of the storage device is allocated for swap and isolated from the rest of the file system.

   Traditional swap spaces are also useful in performance-critical systems because they offer better performance than a swap file. Since swap space is fixed, it also provides predictable behavior, which is an advantage when you must ensure a specific amount of swap is always available.

2. Swap File

   A swap file is a file on the file system that the OS uses as swap space. Swap files offer more flexibility because users can create, resize, or remove the file without having to perform disk repartitioning.

   In Linux, there are two types of swap files:

   - **Temporary swap file**. It typically uses fragmented disk space and doesn't reserve a part of the hard drive, making it suitable for limited disk space.
   - **Permanent swap file**. It takes up a contiguous section of the hard drive, meaning it needs more disk space than a temporary swap

file. The advantage of using a permanent swap file is that it requires fewer I/O operations, which makes it less resource-intensive than a temporary swap file.

Swap files are often easier to manage than traditional swap spaces, and they can even be placed on different storage devices, which provide greater control over swap space management. Their flexibility makes them great for scenarios with limited disk space.

Lastly, swap files are particularly advantageous for virtualized environments. Virtual machines often have dynamic memory requirements, and swap files allow users to easily adjust swap space without altering the underlying disk setup.

Benefits of Using Swap Space

Using swap space provides multiple advantages. Some of them are:

- Preventing Memory-Related Crashes.
- Memory Over commitment.
- Large Programs and Multitasking Support.
- Improving Stability.
- Flexibility.
- Virtual Memory Management.
- Resource Allocation.

Although swap space offers many advantages, relying heavily on it leads to degraded performance, as accessing data stored on a hard drive is slower compared to RAM. Therefore, balancing swap space with an appropriate amount of physical RAM is the best way to ensure optimal system performance.

3.1.4 Determining disk usage with df and du

Determining disk usage is crucial for managing storage effectively and preventing disk space from running out. Two essential tools for this task are df and du.

df Command

The df command stands for "disk free" and displays information about the file system's disk space usage.

The df command provides an overview of disk usage for each mounted file system. It displays information such as:

- File system: The name of the mounted filesystem
- Total: The total disk space available on the filesystem
- Used: The amount of disk space currently used on the filesystem
- Available: The amount of disk space that is still available
- Use%: The percentage of disk space that is being used

The df command has several options that can be used to customize the output. Some of the most commonly used options include:

- -h: displays the output in a human-readable format
- -T: displays the file system type
- -i: displays the inode information

du Command

The du command stands for "disk usage" and displays the disk space a file or directory uses. The command displays the total amount of disk space used by the specified file or directory and its subdirectories.

The du command has several options that can be used to customize the output. Some of the most commonly used options include:

- -h: displays the output in a human-readable format
- -s: displays only the total disk space used

❖ df (disk free) and du (disk usage) are both commands used to check disk usage on Linux systems. However, they differ in their scope and purpose.

df provides an overview of disk usage for each mounted file system. It displays the total, used, available, and percentage of used space for each file system.

The output of df looks something like this:

```
Filesystem      Total       Used       Free       %
/dev/sda1       100G        80G        20G        80%
/dev/sda2        50G        30G        20G        60%
```

du provides a more detailed breakdown of disk usage for a specific directory or file. It displays the total size of the specified directory or file, along with the size of each file and subdirectory within it.

The output of du looks something like this:

```
/home/user
  100M total
    20M .bash_history
    10M .bashrc
     5M .vimrc
    75M Documents
     50M documents
     25M images
```

3.1.5 Configuring disk quotas

Disk quotas allow system administrators to control and limit the amount of disk space that users or groups can utilize on a file system. By configuring disk quotas, you can prevent users from consuming excessive disk space and ensure fair resource allocation.

3.2 Logical volume management (LVM) and RAID

Logical Volume Management (LVM) is a method of managing disk storage that provides a flexible and extensible way to store and manage data. LVM allows you to create logical volumes, which are virtual partitions that can span multiple physical disks. This means that you can use LVM to pool together multiple disks to create a single, large logical volume.

LVM provides several advantages over traditional partitioning, including:

- Flexibility: LVM allows you to easily resize logical volumes, even while they are in use. This means that you can add or remove disks from a logical volume without having to repartition your disks.
- Extensibility: LVM can span multiple physical disks, which means that you can create logical volumes that are larger than the size of any single disk.

- Fault tolerance: LVM can be used to create RAID arrays, which provide data redundancy in case of a disk failure.

3.2.1 Implementing LVM, creating logical volumes (LVs), manipulating VGs& LVs

Logical volumes and logical volume management make it easier to manage disk space. If a file system that hosts a logical volume needs more space, it can be allocated to its logical volume from the free space in its volume group and the file system can be resized. If a disk starts to fail, a replacement disk can be registered as a physical volume with the volume group and the logical volume's extents can be migrated to the new disk.

Physical devices

Physical devices are the storage devices used to save data stored in a logical volume. These are block devices and could be disk partitions, whole disks, RAID arrays, or SAN disks. A device must be initialized as an LVM physical volume in order to be used with LVM. The entire device will be used as a physical volume.

Physical volumes (PVs)

You must initialize a device as a physical volume before using it in an LVM system. LVM tools segment physical volumes into physical extents (PEs), which are small chunks of data that act as the smallest storage block on a physical volume.

Volume groups (VGs)

Volume groups are storage pools made up of one or more physical volumes. This is the functional equivalent of a whole disk in basic storage. A PV can only be allocated to a single VG. A VG can consist of unused space and any number of logical volumes.

Logical volumes (LVs)

Logical volumes are created from free physical extents in a volume group and provide the "storage" device used by applications, users, and the operating system. LVs are a collection of logical extents (LEs), which map to physical extents, the smallest storage chunk of a PV. By default, each LE maps to one PE. Setting specific LV options changes this mapping; for example, mirroring causes each LE to map to two PE's.

The basic steps for adding a new logical volume are as follows.

1. If necessary, install a new hard drive.
2. Optional: Create a partition on the hard drive.
3. Create a physical volume (PV) of the complete hard drive or a partition on the hard drive.
4. Assign the new physical volume to an existing volume group (VG) or create a new volume group.
5. Create a new logical volume (LV) from the space in the volume group.
6. Create a filesystem on the new logical volume.
7. Add appropriate entries to /etc/fstab for mounting the filesystem.
8. Mount the filesystem.

3.2.2 Advanced LVM concepts (i.e. system-config-lvm)

LVM stands for *Logical Volume Management*. It is a system of managing logical volumes, or file systems, that is much more advanced and flexible than the traditional method of partitioning a disk into one or more segments and formatting that partition with a file system.

One of the biggest advantages LVM has is that most operations can be done on the fly, while the system is running. Most operations that you can do with gparted require that the partitions you are trying to manipulate are not in use at the time, so you have to boot from the live cd to perform them. You also often run into the limits of the ms-dos partition table format with gparted, including only 4 primary partitions, and all logical partitions must be contained within one contiguous extended partition.

3.2.3 RAID concepts (Creating and managing a RAID-5 Array)

RAID (Redundant Array of Independent Disks) is a data storage technology that combines multiple disks to provide data redundancy and improve performance. RAID can be used to protect data from disk failures, and it can also be used to improve the performance of disk I/O.

There are several different levels of RAID, each of which offers different levels of data redundancy and performance. The most common levels of RAID are RAID 0, RAID 1, RAID 5, and RAID 6.

- RAID 0: RAID 0 does not provide any data redundancy. However, it can improve the performance of disk I/O by striping data across multiple disks.
- RAID 1: RAID 1 mirrors data across two disks. This means that if one disk fails, the data can still be read from the other disk.
- RAID 5: RAID 5 is a striped RAID array that also uses parity to protect data from disk failures. RAID 5 can tolerate the failure of one disk.
- RAID 6: RAID 6 is a striped RAID array that uses double parity to protect data from disk failures. RAID 6 can tolerate the failure of two disks.

Choosing between LVM and RAID

- ✓ LVM and RAID are both useful technologies for managing disk storage. The best choice for you will depend on your specific needs.
- ✓ If you need a flexible and extensible way to manage disk storage, then LVM is a good choice. LVM is also a good choice if you need to create RAID arrays.
- ✓ If you need to protect data from disk failures, then RAID is a good choice. RAID can also be used to improve the performance of disk I/O.

LVM is a good choice for flexible and extensible disk storage management for the following reasons:

1. Dynamic Storage Management: LVM allows for dynamic allocation and resizing of logical volumes, volume groups, and file systems. This means you can easily add or remove storage devices, expand or shrink volumes, and redistribute storage resources without interrupting ongoing operations. It provides a high degree of flexibility in managing storage capacity.
2. Logical Volume Abstraction: LVM abstracts the underlying physical storage devices and presents logical volumes to the operating system. This abstraction allows for easier management of storage resources, as you can treat logical volumes as independent entities rather than dealing with individual disks or partitions. It simplifies tasks such as resizing, moving, or creating snapshots of volumes.
3. Pooling of Storage Resources: LVM allows you to create volume groups that consist of multiple physical disks or partitions. This pooling of storage resources enables you to aggregate the capacity of multiple

devices into a single logical volume. It provides increased flexibility in managing storage and allows for efficient allocation of space across multiple disks.

On the other hand, RAID is a good choice for protecting data from disk failures for the following reasons:

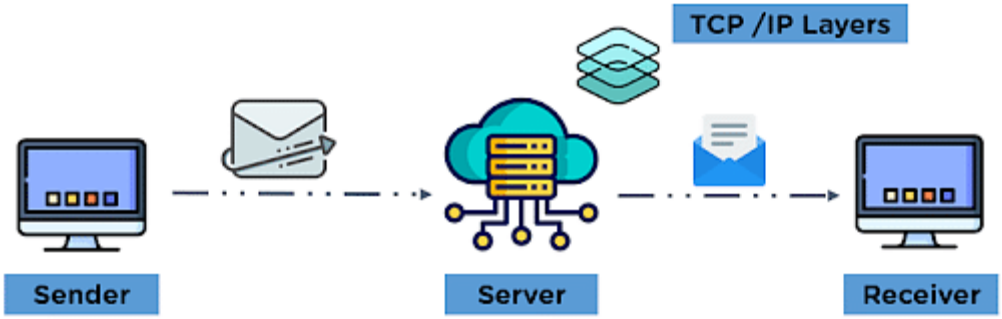
1. Data Redundancy: RAID provides redundancy by distributing data across multiple disks in different ways, such as mirroring or parity. This redundancy ensures that if one disk fails, the data can still be accessed from the remaining disks. RAID protects against data loss due to disk failures, enhancing data availability and reliability.
2. Improved Fault Tolerance: With RAID, the failure of a single disk does not result in immediate data loss. The redundant data on other disks can be used to reconstruct the missing information. Depending on the RAID level used, RAID can tolerate the failure of one or more disks, providing varying degrees of fault tolerance.
3. Performance Enhancement: Some RAID levels, such as RAID 0, offer performance benefits by striping data across multiple disks. This allows for parallel read and write operations, improving overall disk performance. While performance is not the primary focus of RAID for data protection, certain RAID configurations can deliver performance gains.

In general, LVM is a good choice for flexible and extensible disk storage management, while RAID is a good choice for protecting data from disk failures and improving performance. The best choice for you will depend on your specific needs and requirements.

Chapter 4: Network Management (6 hrs)

4.1 TCP/IP Networking

What is TCP/IP Model?



The TCP/IP model refers to the Transmission Control Protocol/Internet Protocol Model. This model is a part of the network domain designed specifically for overseeing efficient and error-free transmission of data.

The model works on a four-layered architecture model, where each layer implicit the required network protocols on the data to be transmitted, which remodels the data to the most optimum structure for efficient transmission over the network.

The History of TCP/IP

The Internet Protocol Suite, or TCP/IP for short, is the set of protocols that make up the network layer of the Internet.

- TCP/IP was developed during the Cold War as a way for the U.S. Department of Defense to connect computers within their networks and with each other across national boundaries. It's been used since the late 1960s when it was formalized by DARPA and later adopted by government agencies and universities worldwide as a common networking standard.
- The first version of TCP/IP was ARPANET (1975), which stands for Advanced Research Projects Administration Network. The name changed to TCP/IP in 1983, when it became an open standard that could be used on any network.
- To give researchers access to each other's equipment, they needed to send messages quickly over long distances without having them re-transmitted by any intermediate nodes along the way. This necessity led to the development of the Transmission Control Protocol (TCP) and Internet Protocol (IP). These protocols were intended for machine-to-machine connections, such as

between computers over local area networks or wide-area networks.

Prerequisite Layers of OSI Model

To understand the OSI model, it is first necessary to understand the concept of layering. Layering is a technique used in computer programming to divide a complex problem or system into smaller, more manageable parts. The OSI model is a seven-layer model that divides the complexity of network communications into seven smaller, more manageable parts, each responsible for a different aspect of the communication process.

The seven layers of the OSI model are:

- Physical layer,
- Data link layer,
- Network layer,
- Transport layer.
- Session layer,
- Presentation layer,
- Application layer.

Each OSI model layer is responsible for a different aspect of the communication process.

Features of the TCP/IP Model

Below mentioned are some of the features that make the TCP/IP model stand out in the network concepts:

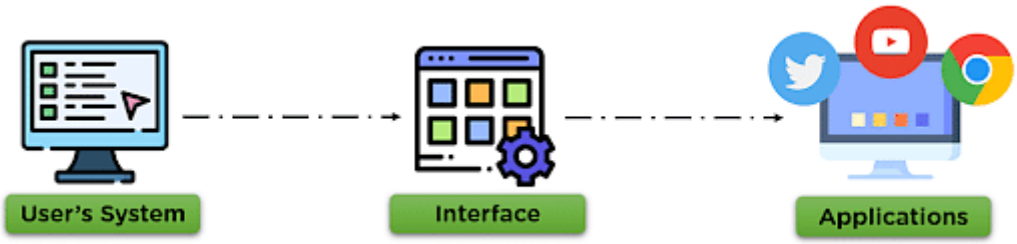
- The TCP/IP model is among one of the most important network concepts that contributed to the working of ARPANET.
- The TCP/IP model comprises four layers: the network access layer, the internet layer, the transport layer, and the application layer (going from bottom to top).
- The network model is implemented during network and communication-related issues.
- Communication between different modes of network devices is possible through the application of various layers.
- The layers in the model provide maintenance of communication channels, flow control, and reliability check format, among other applications in the form of protocols.

The TCP/IP model is divided into four different layers:

- Application layer
- Transport layer
- Internet layer
- Network Access layer

Each layer performs a specific task on the data that is being transmitted over the network channel, and data moves from one layer to another in a preset pattern.

Application Layer



This is the topmost layer which indicates the applications and programs that utilize the TCP/IP model for communicating with the user through applications and various tasks performed by the layer, including data representation for the applications executed by the user and forward it to the transport layer.

The application layer maintains a smooth connection between the application and user for data exchange and offers various features as remote handling of the system, e-mail services, etc.

Some of the protocols used in this layer are:

- HTTP: Hypertext transfer protocol is used for accessing the information available on the internet.
- SMTP: Simple mail transfer protocol, assigned the task of handling e-mail-related steps and issues.
- FTP: This is the standard protocol that oversees the transfer of files over the network channel.

Transport Layer

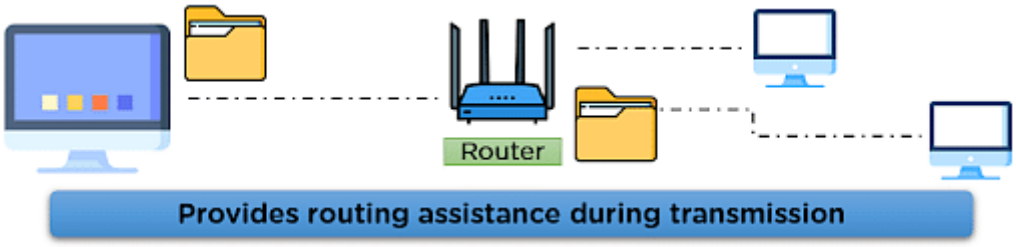
This layer is responsible for establishing the connection between the sender and the receiver device and also performs the task of dividing the data from the application layer into packets, which are then used to create sequences.

It also performs the task of maintaining the data, i.e., to be transmitted without error and controls the data flow rate over the communication channel for smooth transmission of data.

The protocols used in this layer are:

- TCP: Transmission Control Protocol is responsible for the proper transmission of segments over the communication channel. It also establishes a network connection between the source and destination system.
- UDP: User Datagram Protocol is responsible for identifying errors, and other tasks during the transmission of information. UDP maintains various fields for data transmission such as:
- Source Port Address: This port is responsible for designing the application that makes up the message to be transmitted.
- Destination Port Address: This port receives the message sent from the sender side.
- Total Length: The total number of bytes of the user datagram.
- Checksum: Used for error detection of the message at the destination side.

Internet Layer



The Internet layer performs the task of controlling the transmission of the data over the network modes and passes protocols related to the various steps related to the transmission of data over the channel, which is in the form of packets sent by the previous layer.

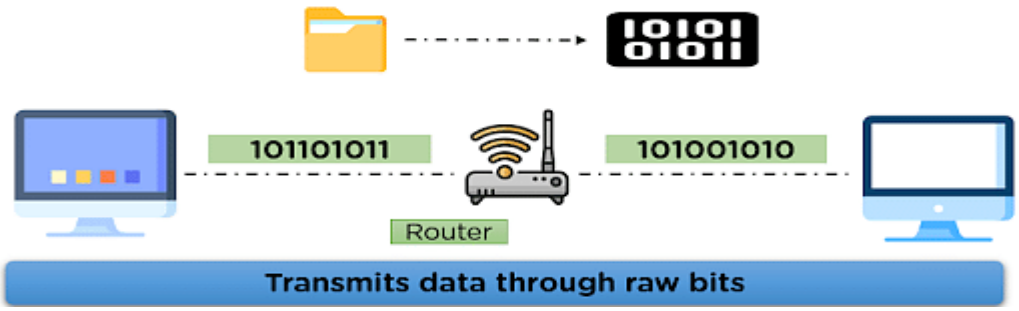
This layer performs many important functions in the TCP/IP model, some of which are:

1. It is responsible for specifying the path that the data packets will use for transmission.
2. This layer is responsible for providing IP addresses to the system for the identification matters over the network channel.

Some of the protocols applied in this layer are:

- IP: This protocol assigns your device with a unique address; the IP address is also responsible for routing the data over the communication channel.
- ARP: This protocol refers to the Address Resolution Protocol that is responsible for finding the physical address using the IP address.

Network Access Layer



This layer is the combination of data-link and physical layer, where it is responsible for maintaining the task of sending and receiving data in raw bits, i.e., in binary format over the physical communication modes in the network channel.

- It uses the physical address of the system for mapping the path of transmission over the network channel.

How Does TCP/IP Work?

The TCP/IP protocol suite is the set of communication protocols used to connect hosts on the Internet. TCP/IP allows computers on the same network to identify and communicate with each other. TCP/IP is a two-layer protocol, with the transport layer (TCP) responsible for reliable end-to-end communication and the Internet layer (IP) accountable for routing packets from the host to the host.

- At the transport layer, TCP provides a reliable byte-stream service to applications. TCP guarantees the delivery of data and that data will be delivered in the same order in which it was sent. TCP uses several mechanisms to provide this service, including sequence numbers, acknowledgments, and timeouts.
- At the Internet layer, IP is responsible for routing datagrams (packets) from host to host. IP does not guarantee the delivery of datagrams, but it tries to deliver them as best. If a datagram

cannot be delivered, IP will return an error message to the source host.

The TCP/IP protocol suite is the most commonly used protocol suite on the Internet today, and it is also the protocol suite used by most LANs and WANs.

Functions of TCP/IP Layers

The TCP/IP model is a four-layer model that divides network communications into four distinct categories or layers. The model is often referred to as the TCP/IP stack. The four important layers are the application layer, the transport layer, the network layer, and the link layer.

- The Application Layer: The application layer is closest to the end user. And this is the layer that users interact with directly, including protocols such as HTTP, FTP, and SSH. This layer is responsible for providing applications with access to the network.
- The Transport Layer: The transport layer ensures that data is delivered reliably and efficiently from one point to another. This layer handles data transmission between hosts, including protocols like TCP and UDP.
- The Internet Layer: The network layer is responsible for routing data through the web. This layer delivers data packets from one host to another, including the IP protocol.
- The Link Layer: The link layer provides reliable data links between the two nodes — for example, protocols like ethernet and Wi-Fi.

Advantages of TCP/IP:

- Scalability: The TCP/IP model is highly scalable and can accommodate small and large networks.
- Reliability: The model is robust and reliable, making it suitable for mission-critical applications.
- Flexibility: It is very flexible, allowing for interoperability between different types of networks.
- Security: The various protocols in the model provide robust security measures.
- Cost-effectiveness: TCP/IP is relatively inexpensive to implement and maintain.

Disadvantages of TCP/IP:

- Complexity: The model is quite complex and requires a certain degree of expertise to configure and maintain.

- Vulnerability: Because of its complexity, it is vulnerable to attack.
- Performance: Performance can be degraded due to network congestion and latency.

Here are some of the most valuable uses of TCP/IP models:

- World Wide Web: TCP/IP transfers data between web browsers and servers.
- Email: Applications such as Outlook, Thunderbird, and Gmail use TCP/IP protocols to send and receive emails.
- File Transfer: FTP, SFTP, and other file transfer services rely on TCP/IP to move files from one computer to another.
- Networking: TCP/IP links computers together in a network.
- Virtual Private Networks: VPNs use TCP/IP to encrypt data before it travels across a public or private network.
- Internet of Things: Many smart home devices use TCP/IP to communicate and transfer data.
- Voice Over Internet Protocol: VOIP services such as Skype and Google Voice use TCP/IP to transmit calls over the internet.

Active directory

Active Directory (AD) is Microsoft's proprietary directory service. It runs on Windows Server and enables administrators to manage permissions and access to network resources.

Active Directory stores data as objects. An object is a single element, such as a user, group, application or device such as a printer. Objects are normally defined as either resources, such as printers or computers, or security principals, such as users or groups.

Active Directory categorizes directory objects by name and attributes. For example, the name of a user might include the name string, along with information associated with the user, such as passwords and Secure Shell keys.

Active Directory's services

The main service in Active Directory is Domain Services (AD DS), which stores directory information and handles the interaction of the user with the domain. AD DS verifies access when a user signs into a device or attempts to connect to a server over a network. AD DS controls which users have access to each resource, as well as group policies.

Other Microsoft and Windows operating system (OS) products, such as Exchange Server and SharePoint Server, rely on AD DS to provide resource access. The server that hosts AD DS is the domain controller.

DNS server

The Domain Name System (DNS) is the phonebook of the Internet. Humans access information online through domain names, like mytimes.com or espn.com. Web browsers interact through Internet Protocol (IP) addresses. DNS translates domain names to IP addresses so browsers can load Internet resources.

Each device connected to the Internet has a unique IP address which other machines use to find the device. DNS servers eliminate the need for humans to memorize IP addresses such as 192.168.1.1

DHCP server

Dynamic Host Configuration Protocol (DHCP) is a client/server protocol that automatically provides an Internet Protocol (IP) host with its IP address and other related configuration information such as the subnet mask and default gateway.

Windows Server 2016 includes DHCP Server, which is an optional networking server role that you can deploy on your network to lease IP addresses and other information to DHCP clients. All Windows-based client operating systems include the DHCP client as part of TCP/IP, and DHCP client is enabled by default.

Every device on a TCP/IP-based network must have a unique unicast IP address to access the network and its resources. Without DHCP, IP addresses for new computers or computers that are moved from one subnet to another must be configured manually; IP addresses for computers that are removed from the network must be manually reclaimed.

With DHCP, this entire process is automated and managed centrally. The DHCP server maintains a pool of IP addresses and leases an address to any DHCP-enabled client when it starts up on the network. Because the IP addresses are dynamic (leased) rather than static (permanently assigned), addresses no longer in use are automatically returned to the pool for reallocation.

Web server

A web server is a computer system capable of delivering web content to end users over the internet via a web browser.

The end user processes a request via a web browser installed on a web server. The communication between a web server or browser and the end user takes place using Hypertext Transfer Protocol (HTTP). The primary role of a web server is to store, process, and deliver requested information or webpages to end users. It uses:

Physical Storage: All website data is stored on a physical web server to ensure its safety. When an end user enters the URL of your website or searches it using a keyword on a browser, a request is generated and sent to the web server to process the data.

Web browser: The role of web browsers such as Firefox, Chrome, or Internet Explorer is to find the web server on which your website data is located. Once the browser finds your server, it reads the request and processes the information.