1. Transmission modes in computer networks

There are several transmission modes in computer networks, including:

Simplex: Data can only flow in one direction.

Half-duplex: Data can flow in both directions, but not simultaneously.

Full-duplex: Data can flow in both directions simultaneously.

Simplex mode is mainly used for one-way communication such as a keyboard or a monitor. Half-duplex is mainly used for communication between two devices, where one device can only transmit or receive at a time. Full-duplex is mainly used for communication between multiple devices where all devices can transmit and receive at the same time.

2 Basic communication model in computer networks

The basic communication model in computer networks is called the OSI (Open Systems Interconnection) reference model. It consists of seven layers, each with a specific role in the communication process:

Physical Layer: Responsible for transmitting raw bits over a physical medium, such as a cable.

Data Link Layer: Responsible for creating a reliable link between two devices on the same network segment.

Network Layer: Responsible for routing packets to their destination.

Transport Layer: Responsible for ensuring that data is delivered reliably and in order.

Session Layer: Responsible for establishing, maintaining, and terminating communication sessions between applications.

Presentation Layer: Responsible for formatting and encrypting data to be sent over the network.

Application Layer: Responsible for providing interfaces for networked applications to access the network's services.

Each layer communicates with the layer directly above and below it using a protocol specific to that layer. The communication starts with the application layer on the sender's device, and it goes down through the OSI layers. Then it goes through the communication channel and on the receiver side it goes up through the OSI layers, until it reaches the application layer.

3.Unique identifiers of network

Unique identifiers of a network can include IP addresses, Media Access Control (MAC) addresses, and hostnames. IP addresses are numerical labels assigned to devices connected to a computer network that use the Internet Protocol for communication. MAC addresses are unique hardware addresses assigned to network interfaces, such as Ethernet or Wi-Fi cards. Hostnames are human-friendly labels assigned to devices on a network, such as "computer1" or "printer2".

4.Types of computer networks

There are several types of computer networks, including:

Local Area Network (LAN): A network that connects devices in a small geographic area, such as a home, office, or building.

Wide Area Network (WAN): A network that spans a large geographic area, such as a city, country, or even the entire world.

Metropolitan Area Network (MAN): A network that spans a medium-sized geographic area, such as a city or campus.

Wireless LAN (WLAN): A wireless version of a LAN that uses radio waves to communicate between devices.

Wireless WAN (WWAN): A wireless version of a WAN that uses cellular technology to communicate between devices.

Virtual Private Network (VPN): A secure, private network that uses the internet to connect remote devices.

Peer-to-Peer (P2P) Network: A type of network in which each device can function as both a client and a server, allowing for direct communication between devices without the need for a central server.

Cloud Network: A network that consists of a collection of interconnected and virtualized computers that are delivered as a service over the internet.

Internet of Things (IoT) network: A network that connects a wide range of devices and objects, such as appliances, vehicles, and medical devices, using internet protocols.

5.-Different kinds of protocols

There are many different protocols used in computer networks, some examples include:

Transmission Control Protocol (TCP): A connection-oriented protocol that ensures the reliable delivery of data by establishing a reliable connection between devices and acknowledging receipt of data.

User Datagram Protocol (UDP): A connectionless protocol that allows devices to send data without establishing a connection, which makes it faster but less reliable than TCP.

Internet Protocol (IP): A network-layer protocol that routes data packets between devices on a network using IP addresses.

File Transfer Protocol (FTP): A standard protocol for transferring files over the internet.

Simple Mail Transfer Protocol (SMTP): A protocol for sending and receiving email messages.

Hypertext Transfer Protocol (HTTP): A protocol for sending and receiving web pages and other information over the internet.

Simple Network Management Protocol (SNMP): A protocol for managing and monitoring network devices.

Dynamic Host Configuration Protocol (DHCP): A protocol that automatically assigns IP addresses to devices on a network.

Routing Information Protocol (RIP) and Open Shortest Path First (OSPF) are examples of routing protocols, they are responsible for sharing information about the network topology among routers in a network.

Internet Group Management Protocol (IGMP) is used by IP hosts to report their multicast group memberships to any neighboring multicast routers.

These are just a few examples, there are many more protocols used in computer networks, each with its own specific purpose and function.

6.Network topology

Network topology refers to the physical and logical layout of a computer network, including the devices and connections that make up the network. There are several types of network topologies, including:

Bus topology: All devices are connected to a single cable or backbone, with data passing through each device in turn.

Star topology: All devices are connected to a central hub, with data passing through the hub to reach other devices on the network.

Ring topology: All devices are connected in a closed loop, with data passing through each device in turn.

Mesh topology: All devices are connected to multiple other devices, creating multiple paths for data to travel.

Tree topology: A hybrid of bus and star topologies, where a central backbone connects multiple star networks.

Hybrid topology: A combination of two or more of the above topologies.

The choice of topology for a network depends on factors such as the number of devices, the distance between devices, and the type of data being transmitted. Bus and ring topologies are generally less expensive to implement, but are more susceptible to errors or failures, while star and mesh topologies are more expensive but more resilient.

7-OSI model

A reference model for how communications should take place between two endpoints in a communication network is the OSI (Open Systems Interconnection) model. It separates the data transmission process into seven levels, each of which serves a distinct purpose. Physical, Data Link, Network, Transport, Session, Presentation, and Application are the layers in order from bottom to top. Each layer is in charge of a particular portion of the communication process and communicates with the layers immediately above and below it. This model is frequently used to explain how various technologies and protocols interact in a network.

8-Functions of different layers of osi model

The OSI (Open Systems Interconnection) model is a framework that describes the functions of different layers of a network protocol stack. The seven layers of the OSI model are:

Physical Layer: This layer is responsible for transmitting raw bits over a physical medium, such as a cable or wireless connection.

Data Link Layer: This layer is responsible for creating and maintaining a reliable link between two devices on a network. It performs tasks such as error checking, flow control, and addressing.

Network Layer: This layer is responsible for routing packets of data between different devices on a network. It performs tasks such as IP addressing and determining the best path for data to travel.

Transport Layer: This layer is responsible for ensuring that data is delivered reliably and in the correct order. It performs tasks such as segmentation, flow control, and error correction.

Session Layer: This layer is responsible for establishing, maintaining, and terminating connections between applications on different devices.

Presentation Layer: This layer is responsible for converting data into a format that can be understood by the application layer. It performs tasks such as data compression, encryption, and decryption.

Application Layer: This layer is responsible for providing services to the user, such as file transfer, email, and web browsing.

9-Advantages and disadvantages of osi model

advantages of the OSI model:

Modularity: The OSI model separates the functions of a network protocol stack into distinct layers, which allows for greater flexibility and ease of modification.

Standardization: The OSI model provides a standard framework for network communication, which allows for greater interoperability between different devices and systems.

Troubleshooting: The OSI model makes it easier to identify and troubleshoot problems in a network, as issues can often be isolated to a specific layer.

Educational: The OSI model provides a useful way to learn and understand the different aspects of network communication.

Disadvantages of the OSI model:

Complexity: The OSI model can be complex and difficult to understand, especially for those new to the field of networking.

Limited applicability: The OSI model is primarily a theoretical framework and may not always match the real-world implementation of network protocols.

Not all protocols fit into the OSI model: Some protocols may not fit neatly into one of the seven layers, which can make it difficult to understand how they work in a real-world context.

Some protocols span multiple layers: Some protocols may span multiple layers of the OSI model, which can make it difficult to understand where they fit in the overall scheme of things.

10-Origin of Version control system

When software was first being developed, programmers would manually keep track of changes to their code by making numerous copies or backups of their files. This is where version control systems got their start.

SCCS (Source Code Control System), which was created at Bell Labs in the 1970s, was one of the first version control programmes. SCCS was created to handle the source code for the UNIX operating system, and it tracked changes using a straightforward file-based system.

RCS (Revision Control System), which was created at the MIT Artificial Intelligence Laboratory

in the 1980s, was another early version control system. When compared to SCCS, RCS has characteristics including the capacity to combine modifications and support for many developers.

In the late 1980s and early 1990s, a number of distributed version control systems were developed, such as CVS (Concurrent Versions System) and Subversion. These systems allowed multiple developers to work on the same codebase simultaneously and provided a more robust approach to version control.

Today, the most popular version control systems are Git and Mercurial, both distributed version control systems that have become the de facto standard for software development.

11-Different kinds of version control system

There are several different types of version control systems, including:

Centralized Version Control Systems (CVCS): In this type of version control system, all developers check out code from a central repository. CVCSs are simple to use and are best suited for small teams and projects. Examples of CVCSs include Subversion, Perforce, and Visual SourceSafe.

Distributed Version Control Systems (DVCS): In this type of version control system, each developer has a local copy of the entire codebase and repository, and changes are shared between developers via peer-to-peer communication. DVCSs are more flexible and are best suited for larger teams and projects. Examples of DVCSs include Git, Mercurial, and Bazaar.

File-based Version Control Systems: In this type of version control system, the repository is a file-based system, where the files and directories are versioned. These systems are simple to use and are best suited for small teams and projects. Examples of file-based VCSs include RCS, SCCS and the various backup software.

Database-based Version Control Systems: In this type of version control system, the repository is stored in a database, which allows for more advanced features such as branching, merging, and atomic commits. These systems are more complex but are best suited for large teams and projects. Examples of database-based VCSs include Team Foundation Server, ClearCase, and Plastic SCM

Hybrid Version Control Systems: They are a combination of both centralized and distributed version control systems, which allows for a centralized management and control, but also allows for distributed development and offline use. An example of hybrid VCS is GitLab.

Cloud-based Version Control Systems: These are version control systems hosted on cloud platforms, allowing for easy collaboration and remote access to the codebase. Examples include GitHub, GitLab, and Bitbucket.

It's important to note that each version control system has its own unique features and capabilities, and the best system for a particular project will depend on the specific requirements and constraints of that project.

12-How to create a pull request in git

For creating the pull request we need to-

Fork the repository: First, you will need to create a copy of the repository you want to contribute to by clicking on the "Fork" button on the repository's page. This will create a copy of the repository under your own account.

Clone the forked repository: Next, you will need to clone the forked repository to your local machine. You can do this by running the command git clone <repository-url> in your terminal, where <repository-url> is the URL of the forked repository.

Create a new branch: Once you have the forked repository on your local machine, you will need to create a new branch where you will make your changes. You can do this by running the command git checkout -b <branch-name>, where <branch-name> is the name of the branch you want to create.

Make your changes: Now you can make the changes you want to submit as a pull request. Make sure to test and verify your changes before committing them.

Commit your changes: Once you are satisfied with your changes, you can commit them to the branch you created. You can do this by running the command git commit -am "<commit-message>", where <commit-message> is a brief description of the changes you made.

Push your changes: Next, you will need to push your changes to the forked repository on GitHub. You can do this by running the command git push origin <branch-name>, where <branch-name> is the name of the branch you created.

Open a pull request: Now that your changes are on the forked repository, you can open a pull request. Go to the original repository's page and click on the "New pull request" button. Select the branch you just pushed as the "compare" branch and then click on the "Create pull request" button.

Provide a brief description of your changes and request for a review.

Your pull request will be reviewed by the repository maintainers, and if it is approved, it will be merged into the main codebase.

13-Different states of a file in git

Modified: The file is being tracked by Git and is part of a commit, and it has been modified since the last commit.

Staged: The file is being tracked by Git and is part of a commit, and it has been modified since the last commit. The changes have been staged and are ready to be committed.

Committed: The file is being tracked by Git and is part of a commit. The changes have been committed to the repository.

14-Workflow of Git

The following phases make up Git's fundamental workflow:

Running the git init command will enable you to create a local repository on your computer.

The files in your project need be modified.

To stage the changes you've made, use the git add command. By doing this, you are getting ready to add the modifications to the repository.

To save the modifications to the repository, issue the git commit command. This updates the project with your modifications in a new version.

To send your changes to a remote repository, such as GitHub, type the command git push. This enables others to access your project and work with you on it.

Additionally, you may grab changes from a remote repository and merge them into your local repository using the git pull command.

Additionally, you may utilise Git Branching to work concurrently on several versions of your project without impacting the main branch.

15-Filesystem and how it works

In Git, a file system is used to store the entire history of a project in a special type of repository called a "git repository" or "repo" for short. Each git repo contains a complete copy of all the files in the project, along with the full history of all the changes that have been made to those files.

A git repository is organized into a series of commits, each of which represents a single set of changes to the project. Each commit is identified by a unique SHA-1 hash, and it contains the following information:

The author and committer of the commit

A message describing the changes that were made

The state of the project's files at the time the commit was made

When you make changes to a file in a git repo, you first stage the changes using the git add command. This tells git to track the changes you've made and prepare them for committing. Once you've staged your changes, you can commit them to the repository using the git commit command. This creates a new commit in the repo that contains your changes and adds it to the project's history.

Each git repo also has a special pointer called the "HEAD" that points to the most recent commit. When you make a new commit, the HEAD pointer is updated to point to the new commit.

When you clone a git repository, you are essentially creating a copy of the entire repo on your local machine, including all the commits and the entire project history. This allows you to make changes to the project, commit them, and push them back to the remote repository for other people to access.

Additionally, Git uses branches to create multiple working version of the project parallelly without affecting the main branch

16-Git filesystem

Git uses a specialized filesystem to store the entire history of a project in a git repository. The git filesystem is a key-value store that uses a unique SHA-1 hash as the key for each object in the repository. Each object in the git filesystem represents a specific piece of information related to the project, such as a file, a commit, or a tag.

The main types of objects in the git filesystem are:

Tree objects: These objects store the structure of the project's directories and subdirectories. Each tree object is identified by a unique SHA-1 hash, and it contains a list of references to other tree and blob objects.

Commit objects: These objects store metadata about a particular commit, such as the author, the committer, and the commit message. Each commit object is identified by a unique SHA-1 hash, and it contains a reference to the root tree object of the project at the time of the commit.

Tag objects: These objects store extra information about a specific commit, such as a release version number.

The git filesystem uses a technique called "content-addressable storage" which means that the contents of a file are used to generate a unique hash. This ensures that the contents of the file are always the same, regardless of the file name or location. This allows git to track the history of a file, even if it's been renamed or moved to a different location within the project.

17-Storage basics

The actual hardware or material used to store computer data is referred to as storage. Memory cards, solid-state drives, and hard drives are the most widely used types of storage.

Hard drives are mechanical devices that store data on rotating discs. Despite being very cheap, they can be slower than solid-state drives.

A more recent form of storage called solid-state drives (SSD) uses flash memory to store data. Although they cost more, they are more durable and swifter than hard discs.

One kind of solid-state storage that is frequently seen in portable electronics like cameras and cellphones is memory cards.

There are cloud storage services that let you save your data on distant servers in addition to these conventional methods of storage. These services include, among others, Dropbox, iCloud, and Google Drive.

On these devices, data may be stored in a variety of ways. Depending on the file system being utilised, data can either be kept in a single large file or divided into several smaller files.

Depending on the type of data and the application that will use it, data can be saved in a variety of formats. Text files, on the other hand, are often saved in formats like ASCII or UTF-8, whilst picture files are typically kept in formats like JPEG or PNG.

All things considered, storage is a crucial part of any computer system and is crucial to how we utilise and interact with our data.