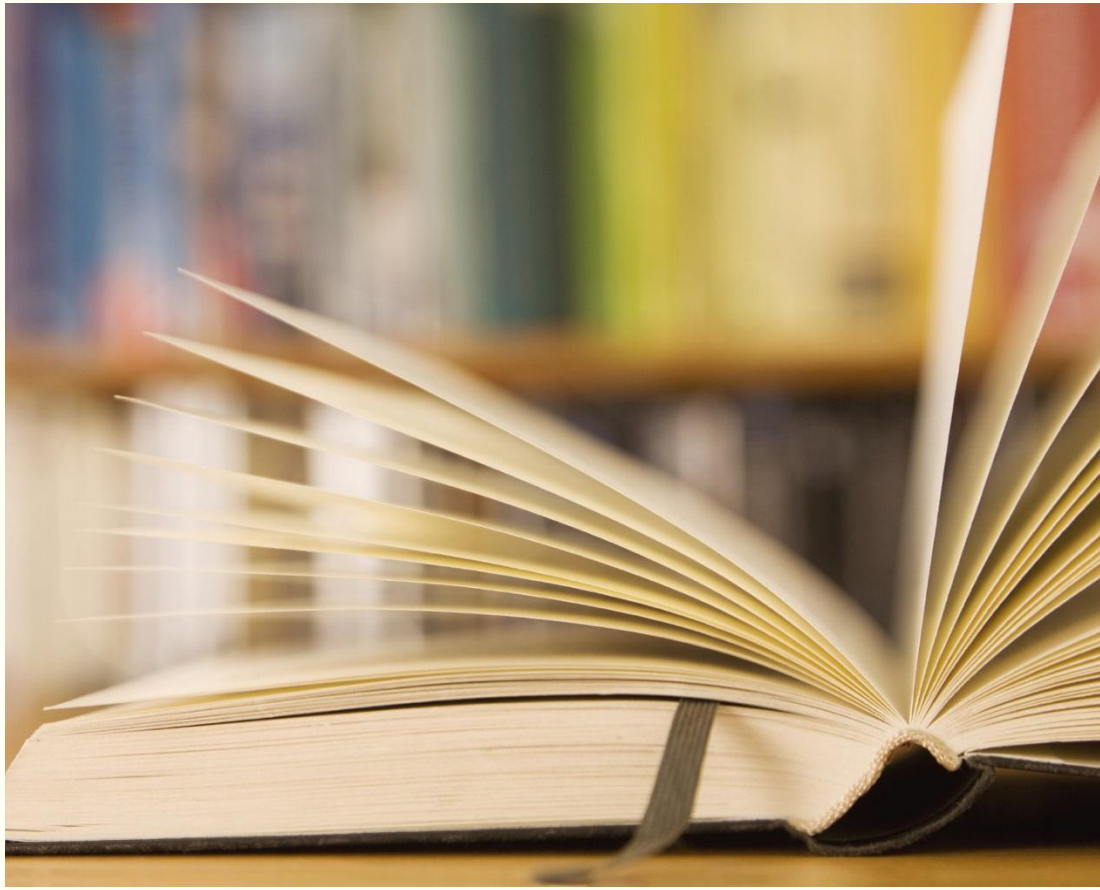
NETWORK & SYSTEM ADMINISTRATION



Unit 1: Networking Overview – 4Hrs

- Overview of Reference Model (OSI, TCP/IP)
- Overview of IPV4 & IPV6 addressing
- Windows & Linux Networking basics
- Switching & Routing basics
- Overview of SDN & OpenFlow

Network Overview

What is a Network?

A network consists of two or more computers that are linked in order to share resources (such as printers and CDs), exchange files, or allow electronic communications. The computers on a network may be linked through cables, telephone lines, radio waves, satellites, or infrared light beams. Two very common types of networks include:

1. **Local Area Network (LAN)**
   A Local Area Network (LAN) is a network that is confined to a relatively small area. It is generally limited to a geographic area such as a writing lab, school, or building.

2. **Metropolitan Area Networks (MAN)**
   A Metropolitan Area Network (MAN) is a network that spans a larger geographic area than a Local Area Network (LAN) but is smaller than a Wide Area Network (WAN). MANs typically cover a city or a large campus and are used to connect multiple LANs within this area

Network Overview

3. Wide Area Network (WAN)

A **Wide Area Network (WAN)** is a telecommunications network that extends or connects over a large geographical area, such as Florida, the United States, or the world. WANs are used to connect different smaller networks, such as Local Area Networks (LANs) and Metropolitan Area Networks (MANs),

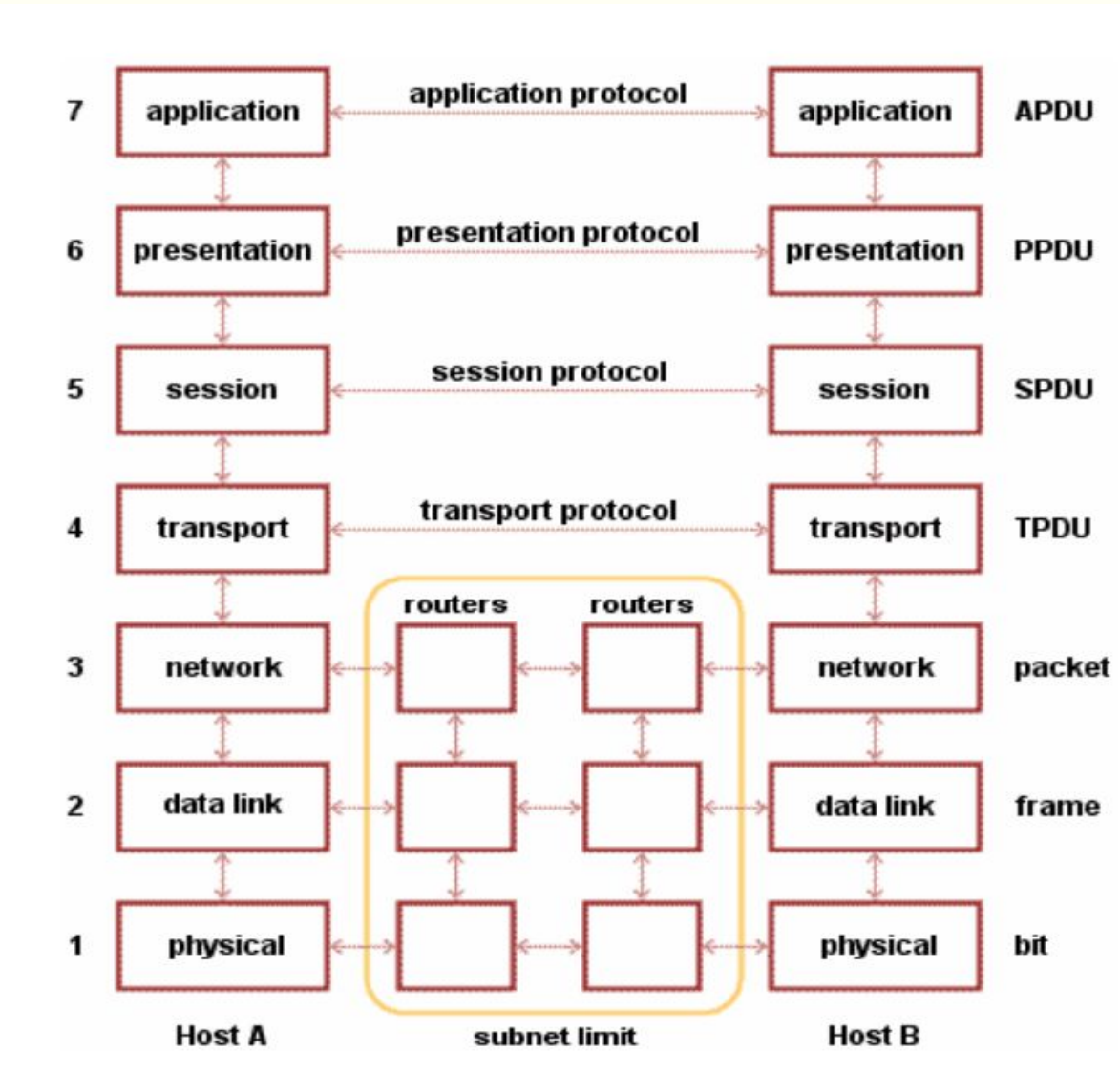
1.1 Overview of Reference Model [OSI, TCP/IP]

OSI (Open Systems Interconnection)

The OSI model, and any other network communication model, provides only a conceptual framework developed by the International Organization for Standardization (ISO). for communication between computers, but the model itself does not provide specific methods of communication. Actual communication is defined by various communication protocols.

OSI standardizes the function of communication into seven different layers. The layers are divided into two groups. The upper four layers are used whenever a message passes from or to a user. The lower three layers are used when any message passes through the host computer.

1.1 Overview of Reference Model (OSI, TCP/IP)



OSI (Open Systems Interconnection)

1. Physical Layer:

- Deals with the transmission and reception of raw bitstreams over a physical medium.
- Defines hardware specifications, such as cables, connectors, and data rates.
- Examples: Ethernet cables, fiber optics.
- Protocols/Technologies: RS-232, IEEE 802.3.

OSI (Open Systems Interconnection)

2. Data Link Layer:

- Ensures error-free data transfer between adjacent nodes by organizing bits into frames.
- Functions include framing, error detection, and flow control.
- Sub-layers:
  - Media Access Control (MAC): Manages access to the physical transmission medium.
  - Logical Link Control (LLC): Provides error detection and frame synchronization.
- Examples: Ethernet, Wi-Fi (802.11), PPP.

OSI (Open Systems Interconnection)

3. Network Layer:

- Handles routing, addressing, and forwarding of data packets across networks.
- Provides logical addressing (e.g., IP addresses).
- Supports internetworking between different networks.
- Protocols: IPv4, IPv6, ICMP, ARP.

OSI (Open Systems Interconnection)

4. Transport Layer:

- Ensures reliable end-to-end communication and error recovery.
- Provides segmentation, reassembly, and flow control.
- Protocols:
  - TCP: Reliable, connection-oriented communication.
  - UDP: Faster, connectionless communication.
- Examples: Streaming applications (UDP), and web browsing (TCP).

OSI (Open Systems Interconnection)

5. Session Layer:

- Manages and maintains connections between applications.
- Functions include session establishment, maintenance, and termination.
- Examples: NetBIOS, Remote Procedure Call (RPC).

OSI (Open Systems Interconnection)

6. Presentation Layer:

- Translates data into a format usable by the application layer.
- Handles data encryption, compression, and translation.
- Examples: SSL/TLS, JPEG, ASCII.

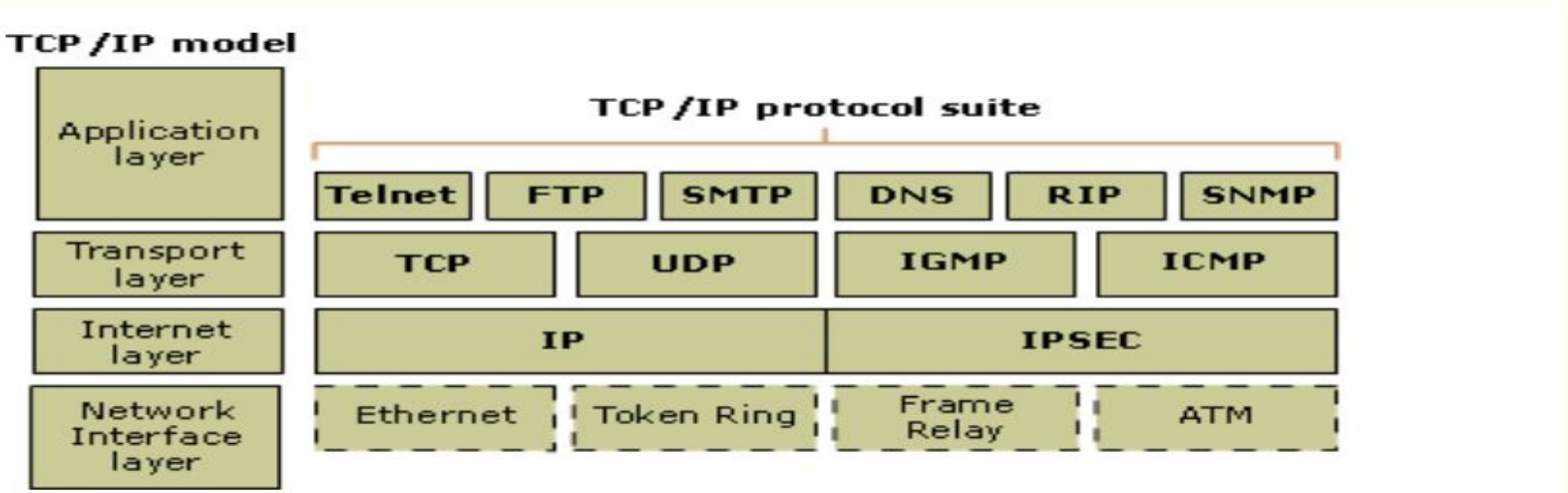
OSI (Open Systems Interconnection)

7. Application Layer:

- Provides end-user services and network applications.
- Interfaces directly with software applications.
- Protocols: HTTP, FTP, SMTP, DNS.

TCP/IP (Transmission Control Protocol/Internet Protocol)

The TCP/IP model is a practical framework developed by DARPA to facilitate internet communication. Unlike the OSI model, it combines some functions into fewer layers. There are four layers on the TCP/IP reference model

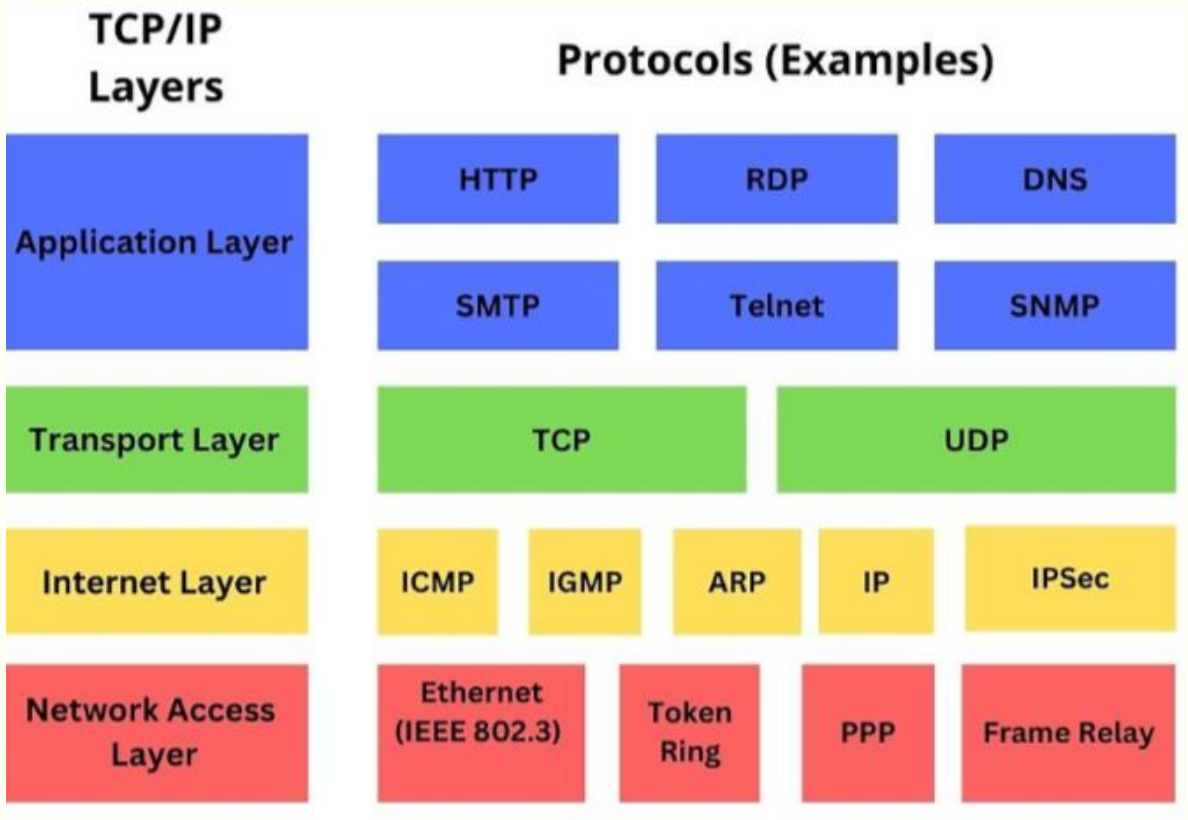


Layers:

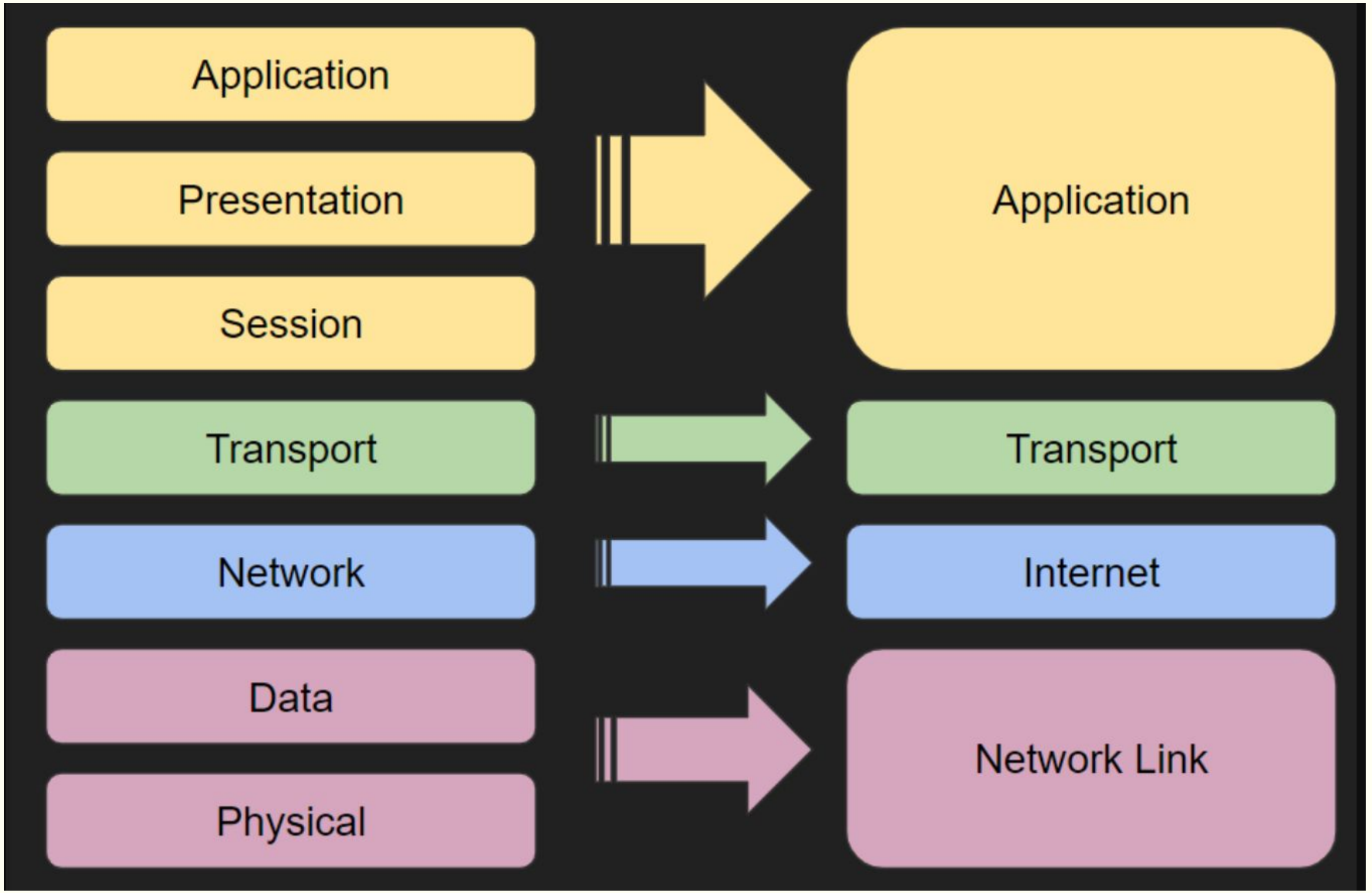
- Layer 1: The Network Layer
- Layer 2: The Internet Layer
- Layer 3: The Transport Layer
- Layer 4: The Application Layer

TCP/IP (Transmission Control Protocol/Internet Protocol)

The TCP/IP model is a practical framework developed by DARPA to facilitate internet communication. Unlike the OSI model, it combines some functions into fewer layers. There are four layers on the TCP/IP reference model



TCP/IP (Transmission Control Protocol/Internet Protocol)



TCP/IP (Transmission Control Protocol/Internet Protocol)

1. Network Access/Link Layer:

Handles physical data transmission between devices on the same network.

Includes hardware specifications and link protocols.

Examples: Ethernet, Wi-Fi, DSL.

TCP/IP (Transmission Control Protocol/Internet Protocol)

2. Internet Layer:

- Provides logical addressing and determines the best path for data delivery.
- Protocols:
  - IP: Provides addressing and routing.
  - ARP: Resolves IP addresses to MAC addresses.
  - ICMP: Diagnoses network issues (e.g., ping)..

TCP/IP (Transmission Control Protocol/Internet Protocol)

3. Transport Layer:

- Ensures reliable communication and data integrity.
- Protocols:
  - TCP: Reliable, connection-oriented.
  - UDP: Unreliable, connectionless.
- Examples: File downloads (TCP), VoIP calls (UDP).

TCP/IP (Transmission Control Protocol/Internet Protocol)

4. Application Layer:

- Interfaces directly with user applications.
- Provides high-level protocols for data exchange.
- Examples: DNS, HTTP, FTP, SNMP.

OSI VS TCP/IP

S.N	OSI	TCP/IP
1	It is developed by ISO (International Standard Organization)	It is developed by ARPANET (Advanced Research Project Agency Network).
2	It is defined before the advent of the Internet.	It is defined after the advent of the internet.
3	The minimum size of the OSI header is 5 bytes.	Minimum header size is 20 bytes.
4	OSI layers have seven layers.	TCP/IP has four layers.
5	OSI follows a vertical approach.	TCP/IP follows a horizontal approach.
6	OSI is tightly layered	TCP/IP is loosely layered
7	OSI is Conceptual Framework	TCP/IP is Practical Implementation

1.2 Overview of IPV4 & IPV6 addressing

IPv4 Addressing

- **Format**: 32-bit address written in dotted decimal notation (e.g., 192.168.1.1).
- **Structure**:
  - Divided into Network and Host portions.
  - Example: In 192.168.1.1 with a subnet mask of 255.255.255.0 (/24), the network portion is 192.168.1, and the host portion is 1.
- **Address Classes**:
  - Class A: 0.0.0.0 to 127.255.255.255 (Large networks).
  - Class B: 128.0.0.0 to 191.255.255.255 (Medium-sized networks).
  - Class C: 192.0.0.0 to 223.255.255.255 (Small networks).
  - Class D: 224.0.0.0 to 239.255.255.255 (Multicast).
  - Class E: 240.0.0.0 to 255.255.255.255 (Reserved for experimental use).
- **Subnetting**:
  - Divides a network into smaller subnets for efficient IP address utilization.
  - CIDR (Classless Inter-Domain Routing): Uses prefix notation (e.g., 192.168.1.0/24).
- **Limitations**:
  - Limited address space (approximately 4.3 billion addresses).
  - Lacks support for mobility and native encryption.

1.2 Overview of IPV4 & IPV6 addressing

Class A:

- **Private IP Range**: 10.0.0.0 to 10.255.255.255
- **Public IP Range**: 1.0.0.0 to 9.255.255.255 and 11.0.0.0 to 126.255.255.255
- **Subnet Mask**: 255.0.0.0 (or /8)

Class B:

- **Private IP Range**: 172.16.0.0 to 172.31.255.255
- **Public IP Range**: 128.0.0.0 to 172.15.255.255 and 172.32.0.0 to 191.255.255.255
- **Subnet Mask**: 255.255.0.0 (or /16)

Class C:

- **Private IP Range**: 192.168.0.0 to 192.168.255.255
- **Public IP Range**: 192.0.0.0 to 192.167.255.255 and 192.169.0.0 to 223.255.255.255
- **Subnet Mask**: 255.255.255.0 (or /24)

1.2 Overview of IPV4 & IPV6 addressing

IPv6 Addressing

- **Format**: 128-bit address written in hexadecimal, separated by colons (e.g., 2001:0db8:85a3:0000:0000:8a2e:0370:7334).
- **Key Features**:
  - Larger address space: 340 undecillion unique addresses.
  - Stateless auto-configuration.
  - Integrated support for IPSec for secure communication.
  - Simplified header structure for efficient processing.
- **Address Types**:
  - **Unicast**: One-to-one communication.
  - **Multicast**: One-to-many communication.
  - **Anycast**: One-to-one-of-many communication, delivering to the nearest node.
- **Advantages over IPv4**:
  - No need for NAT (Network Address Translation).
  - Better support for mobile devices.
  - Built-in quality of service (QoS).

1.2 Overview of IPV4 & IPV6 addressing

Comparison of IPv4 and IPv6

Feature	IPv4	IPv6
Address Size	32-bit	128-bit
Notation	Dotted decimal	Hexadecimal
Header Size	20 bytes	40 bytes
Address Space	~4.3 billion	Virtually unlimited
Security	Optional (IPSec)	Mandatory (IPSec)
Configuration	Manual/DHCP	Stateless/DHCPv6

1.3 Windows & Linux Networking Basics

Networking is fundamental to both Windows and Linux operating systems. While their implementation differs slightly, the concepts are quite similar. Below is an overview of the basics of networking in both environments:

1. IP Addressing

- **Windows:**
  - Use ipconfig in the Command Prompt to view IP configurations.
  - Static IP can be configured via **Control Panel > Network and Sharing Center** or netsh commands.
- **Linux:**
  - Use ifconfig or ip addr to view IP configurations.
  - Static IP configuration depends on the distribution (e.g., in Debian-based systems, it's in /etc/network/interfaces or /etc/netplan).

Key Points:

- **IP Address:** Identifies devices on a network.
- **Subnet Mask:** Defines the network portion of the IP.
- **Gateway:** Routes traffic to other networks.

1.3 Windows & Linux Networking Basics

2. DNS (Domain Name System)

- **Windows:**
  - DNS can be configured in the network adapter settings or using netsh commands.
- **Linux:**
  - Edit /etc/resolv.conf to set DNS servers manually.

Key Points:

- Resolves domain names (e.g., google.com) to IP addresses.
- Can use public DNS servers like Google's (8.8.8.8).

3. Network Interfaces

- **Windows:**
  - Manage interfaces through **Control Panel > Network and Sharing Center** or ncpa.cpl.
- **Linux:**
  - Interfaces are usually managed via /etc/network/interfaces, /etc/netplan/, or Network Manager GUI.
  - Commands: ifconfig, ip link.

Key Points:

An interface can be physical (Ethernet/Wi-Fi) or virtual (Loopback).

1.3 Windows & Linux Networking Basics

4. Routing

- **Windows:**
  - View routing table with route print.
  - Add routes using route add.
- **Linux:**
  - View routing table with route or ip route.
  - Add routes using ip route add.

Key Points:

- Routing ensures packets reach their destinations.
- Default routes are used when no specific route exists for a destination.

1.3 Windows & Linux Networking Basics

5. Firewall

- **Windows:**
  - Managed via **Windows Defender Firewall** or netsh advfirewall.
- **Linux:**
  - Tools: iptables, firewalld, or ufw (Uncomplicated Firewall).
  - Example: ufw allow 22 to allow SSH.

Key Points:

- Firewalls control incoming and outgoing traffic.
- Rules can allow or deny traffic based on ports, protocols, and IPs.

6. File Sharing and Services

- **Windows:**
  - Use SMB (Server Message Block) for file sharing.
  - Share files via **Advanced Sharing** or net share.
- **Linux:**
  - Use NFS (Network File System) or Samba (SMB for Linux).
  - Configure Samba in /etc/samba/smb.conf.

Key Points:

- File sharing allows access to files over a network.
- Services like SSH, HTTP, or FTP facilitate remote management or data transfer.

1.3 Windows & Linux Networking Basics

7. Troubleshooting Tools

- **Ping:**
  - ping [IP/hostname] in both OSes checks connectivity.
- **Traceroute:**
  - **Windows:** tracert [destination]
  - **Linux:** traceroute [destination]
- **DNS Lookup:**
  - **Windows:** nslookup
  - **Linux:** dig or nslookup
- **Packet Capture:**
  - Use **Wireshark** or tcpdump (Linux).

8. SSH and Remote Access

- **Windows:**
  - Use Remote Desktop Protocol (RDP) or enable OpenSSH in Windows.
- **Linux:**
  - Use SSH (ssh user@hostname) for secure remote access.

Key Points:

- Secure remote management is critical.
- Windows uses RDP natively, while Linux prefers SSH.

1.3 Windows & Linux Networking Basics

9. Host Files

- **Windows:**
  - Located in C:\Windows\System32\drivers\etc\hosts.
- **Linux:**
  - Located in /etc/hosts.

Key Points:

- Maps domain names to IP addresses locally.
- Useful for testing or overriding DNS.

10. Network Services

- **Windows:**
  - Managed via **Services.msc** or sc commands.
- **Linux:**
  - Use systemctl or service commands.
  - Example: systemctl start apache2.

Key Points:

- Services include web servers, FTP servers, and more.
- Properly configure and secure services to avoid vulnerabilities.

1.4 Switching & Routing Basics

Routing and Switching are fundamental concepts in networking that enable communication between devices within the same network (switching) or between different networks (routing). Let's break down these concepts in detail:

Switching:

Switching refers to the process of directing data packets within a **local area network (LAN)**. Switches are Layer 2 devices (operating at the Data Link Layer of the OSI model) that use MAC addresses to forward data frames to the correct device.

Circuit Switching:

- **Definition**: A dedicated communication path is established between two nodes for the duration of the connection.
- **Example**: Traditional telephone networks use circuit switching.

Packet Switching:

- **Definition**: Data is broken into packets, which are sent independently over the network. Each packet may take a different path to the destination.
- **Example**: The Internet uses packet switching.

Message Switching:

- **Definition**: Entire messages are sent from the source to the destination, one hop at a time. Each intermediate node stores the message until the next node is ready to receive it.
- **Example**: Older telegraph networks used message switching.

1.4 Switching & Routing Basics

Switching Technologies

- **VLAN (Virtual LANs)**: Allow segmentation of a LAN into multiple logical networks for better management and security.
- **Trunking**: Carries VLAN traffic between switches using protocols like 802.1Q.
- **EtherChannel**: Combines multiple links between switches or devices for redundancy and increased bandwidth.

Functions of a Switch

- **MAC Address Learning**: Switches learn the MAC addresses of devices connected to their ports and store them in the MAC address table.
- **Forwarding/Filtering**: Frames are forwarded only to the destination port based on the MAC address.
- **Loop Avoidance**: Protocols like Spanning Tree Protocol (STP) prevent loops in the network.
- **Segmentation**: Switches create multiple collision domains, reducing network congestion

Assignment

1. Explain the OSI & TCP/IP reference Model. Difference between OSI & TCP/IP model.
2. Difference between IPV$ & IPV6. Why do we need IPV6 over IPV4?
3. Explain Routing & Switching. What are their applications?
4. How do you use the Network troubleshooting command? Explain.
5. What is SDN & OpenFlow? What are the primary differences between traditional networking and SDN? Provide examples of scenarios where SDN would be advantageous.

Unit 2: Server Administration Basics

2.1 Open-Source Server and Client Installation
2.2 Linux installation, disk partitioning, logical volume manager
2.3 Boot Process and Startup Services: Xinetd/Inetd
2.4 Managing accounts: users, groups and other privileges
2.5 File Systems and Quota Management
2.6 Job Scheduling with cron, crontab, anacron and system log analysis
2.7 Process controlling and management
2.8 Online Server upgrade/update process
2.9 Administering Database, web, and proxy server
2.10 Shell programming fundamentals

Client

Client applications are installed on a user's computer or workstation, and interact with data and programs on a server. Client applications are not the same as desktop applications because client applications must interact with a server for full functionality. A common example of a client application is the video game World of Warcraft. Users install a client application on their computers that allows them to log into a server containing the game programming. Businesses can use client server applications to cut down on overhead requirements for work stations. Instead of installing hundreds of copies of a particular program, users log into the application server.

In client- server architecture, the client acts a smaller computer that is used by the employees of the organization in order to perform their day to day activities. The employee uses the client computer in order to access the data files or applications stored on the server machine. The rights authorized to the client machine can be different. Some employees have the access to data files of the organization while other may only access the applications present on the server.

Unit 2.1: Open-Source Server and Client Installation

Server

In client-server environment, the server computer acts as the “brains” of the business. A very large capacity computer is used as a server. There can be a mainframe also as it stores a wide variety of functionalities and data. Generally, applications and data files are stored on the server computer. Employee computers or workstations access these applications and files across the network. For example, an employee can access company’s data files stored on the server, from his/her client computer. In some cases, employees may access only specific applications from their client machine. Application server is the name given to this type of server. The client-server architecture is fully utilized in this type of environment as employees have to login from their client machine in order to access the application stored on the server. For example, these kinds of applications include graphic design programs, spreadsheets and word processors. The client- server architecture is illustrated in each case.

Apart from the storage medium, the server also acts as a processing power source. The client machines get their processing power from this server source. By doing so, no extra hardware for the client is needed and it utilizes greater processing power of the server.

Unit 2.1: Open-Source Server and Client Installation

Server

In client-server environment, the server computer acts as the “brains” of the business. A very large capacity computer is used as a server. There can be a mainframe also as it stores a wide variety of functionalities and data. Generally, applications and data files are stored on the server computer. Employee computers or workstations access these applications and files across the network. For example, an employee can access company’s data files stored on the server, from his/her client computer. In some cases, employees may access only specific applications from their client machine. Application server is the name given to this type of server. The client-server architecture is fully utilized in this type of environment as employees have to login from their client machine in order to access the application stored on the server. For example, these kinds of applications include graphic design programs, spreadsheets and word processors. The client- server architecture is illustrated in each case.

Apart from the storage medium, the server also acts as a processing power source. The client machines get their processing power from this server source. By doing so, no extra hardware for the client is needed and it utilizes greater processing power of the server.

Unit 2.1: Open-Source Server and Client Installation

Open-source server software: Software that provides services (web, file, proxy, etc.) to clients and is freely available with source code (e.g., Apache, Squid, Samba).

Open-source client software: Software used to request services from servers (e.g., Firefox, FileZilla, Thunderbird).

Unit 3: Network Configuration Basics

- **Network Interface Configuration**
- **Diagnosing Network startup issues**
- **Linux and Windows Firewall configuration**
- **Network troubleshooting commands**
- **Introduction to network programming with Mininet**
- **SDN controller and data plane communication**
- **Routing configuration in SDN**
- **Open source networking monitoring (e.g. Nagios)**

3.1 Network Interface Configuration

A network interface allows a computer to connect to a network. Interfaces can be physical (e.g., Ethernet card) or virtual (e.g., loopback). Each interface is assigned a unique IP address used for communication on a network.

Key components:

- **IP Address**: Uniquely identifies a host on a network.
- **Subnet Mask**: Distinguishes network and host portions of the address.
- **Gateway**: Router that allows traffic to exit the local network.
- **DNS Server**: Resolves human-readable domain names to IP addresses.

Types of IP Addressing:

- **Static**: Manually configured IP address.
- **Dynamic**: Assigned automatically via DHCP.
- **Loopback**: Internal virtual interface (127.0.0.1) for self-communication.

3.1 Network Interface Configuration

Linux Configuration

To view all interfaces:

- **ip a**

To configure a static IP (Netplan on Ubuntu):

1. Edit the Netplan configuration file:
   - **sudo nano /etc/netplan/01-netcfg.yaml**

Example:

```
network:
  version: 2
  ethernets:
    enp0s3:
      dhcp4: no
      addresses: [192.168.1.100/24]
      gateway4: 192.168.1.1
      nameservers:
        addresses: [8.8.8.8, 8.8.4.4]
```

2. Apply the configuration:
   - **sudo netplan apply**

3.1 Network Interface Configuration

Windows Configuration:
To view Windows Configuraton:

- **ipconfig /all**

To set static IP Using Powershell:

- **netsh interface ip set address name="Ethernet" static 192.168.1.100 255.255.255.0 192.168.1.1**

Explanation:

name="Ethernet": Interface name

static: Specifies static assignment

192.168.1.100: IP address

255.255.255.0: Subnet mask

192.168.1.1: Gateway

3.2 Diagnosing Network Startup Issues

When a system boots, it attempts to initialize network interfaces and establish connections.

Startup issues can prevent connectivity due to:

- Missing or misconfigured interface files
- DHCP server issues (no IP assigned)
- Hardware not detected or drivers missing
- Incorrect DNS or routing setup
- Firewall rules blocking connections

Common Symptoms:

- No IP assigned
- Cannot reach default gateway or DNS
- "Network unreachable" or "destination host unreachable"
- Long delay at boot time due to failed service

3.2 Diagnosing Network Startup Issues

Diagnosis on Linux System

1. Check Interface & IP Status
- **ip a** (Shows interface names, Ips & state (up/Down))

2. Check network Service
- **Systemctl status NetworkManager**
- **Systemctl status networking** (older system)

Check whether the network manager is active or not

3. Check recent logs
- **Journalctl –u NetworkManager** (shows service logs useful for DHCP Failure)
- **Dmesg | grep eth** (shows kernal messages useful for hardware issues)

4. Try to get DHCP manually
- **Sudo dhclient eth0**

3.2 Diagnosing Network Startup Issues

Diagnosis on Windows System

1. View Current IP Config
- **ipconfig /all** (Shows interface details)

2. Release & Renew DHCP lease
- **ipconfig /release**
- **ipconfig /renew**

3. Reset TCP/IP stack
- **Netsh int ip reset**

4. Check network service status
- Use **Event Viewer –** system logs
- Look for **Dhcp-Client** or **Tcpip** errors

3.3 Linux & Windows Firewall Configuration

A firewall is a security system that monitors and controls incoming and outgoing network traffic based on predefined rules. Firewalls protect systems by blocking unwanted or harmful traffic.

Types of Firewalls:

- Host-based: Installed on individual devices (e.g., ufw, Windows Firewall)
- Network-based: Deployed on the perimeter of the network

Firewall Rules:

- **Allow**: Permit traffic
- **Deny/Drop**: Block traffic silently
- **Reject**: Block traffic and notify the source

3.3 Linux & Windows Firewall Configuration

Linux Firewall

UFW (Uncomplicated Firewall):
sudo ufw status (Check Current status)
sudo ufw enable (Enable the firewall)
sudo ufw allow 22/tcp (Allow SSH)
sudo ufw deny 23 (Block Telnet)

Windows Firewall

Use Windows Defender Firewall from Control Panel (GUI) or via Command Line:

```
netsh advfirewall set allprofiles state on
netsh advfirewall firewall add rule name="Allow_HTTP" dir=in action=allow protocol=TCP localport=80
netsh advfirewall firewall add rule name="Block_Telnet" dir=in action=block protocol=TCP localport=23
```

- **set allprofiles state on:** Enables firewall for domain/private/public profiles
- **add rule**: Adds a new firewall rule
- **action=allow/block:** Permit or deny traffic
- **protocol=TCP localport=80:** Match HTTP traffic

3.4 Network Troubleshooting Commands

Network troubleshooting is the process of identifying, diagnosing, and resolving problems affecting network connectivity and performance. Common issues include DNS failures, unreachable hosts, or packet loss.

Basic Troubleshooting Steps:

- Check IP configuration.
- Test physical connection (e.g., cable/Wi-Fi).
- Test local and external connectivity.
- Identify DNS or gateway issues.
- Use command-line tools to isolate the issue.

3.4 Network Troubleshooting Commands

Basics Troubleshooting Commands

ping 8.8.8.8 (Ping google's dns)
Ping google.com (ping using domain name)

Display Ip address & Interface Details
ipconfig /all (windows)
ip a (linux)

Display the path taken by packets to reach destination
traceroute (linux)
tracert (windows)

Perform DNS queries
nslookup google.com (windows)
dig google.com (linux)

Displays active connection & listening ports
netstat -tuln (linux tcp/udp ports)
netstat -ano (windows all connection with PID)

3.5 Introduction to Network Programming with Mininet

Mininet:

- Mininet is a powerful open-source network emulator designed for Software-Defined Networking
- It creates virtual networks using OpenFlow (a protocol that enables communication between the control and data planes of a network.), allowing users to simulate SDN controllers, switches, hosts, and links.
- Mininet is highly customizable and supports scripting with Python for creating complex network topologies.
- It is more focused on emulating real-world network behavior and performance.

Key Features of Mininet:

1. **Realistic Emulation:** It runs real kernel, switch, and application code, making it highly realistic for testing.
2. **Customizable Topologies:** You can create custom network topologies using Python scripts or predefined templates.
3. **Interactive CLI:** Mininet provides a command-line interface to interact with and manage the virtual network.
4. **Rapid Prototyping:** It allows quick testing of network designs before deploying them on physical hardware.

3.5 Network Troubleshooting Commands

Example:

Install on Linux:

- sudo apt-get install mininet

Create Basic Network using cmd

- sudo mn --topo single,2 --mac --switch ovsk --controller remote

(This script creates a network with a single switch and two hosts, then tests connectivity between them.)

Test connectivity:

- mininet> h1 ping h2

3.5 Network Troubleshooting Commands

Customize with python:

```
from mininet.net import Mininet
from mininet.topo import Topo

class MyTopo(Topo):
    def build(self):
        switch = self.addSwitch('s1')
        host1 = self.addHost('h1')
        host2 = self.addHost('h2')
        self.addLink(host1, switch)
        self.addLink(host2, switch)

topo = MyTopo()
net = Mininet(topo=topo)
net.start()
net.pingAll()
net.stop()
```

To Run: sudo python3 mytopo.py

This script creates a network with a single switch and two hosts, then tests connectivity between them.

3.5 Network Troubleshooting Commands

- **Mininet:** This module is the core of Mininet and allows you to create and manage virtual networks.
- **Topo:** This module helps in defining custom network topologies.

- **MyTopo(Topo):** This defines a new class MyTopo that inherits from the Topo class.
- **build() Method:** This is a special method in Mininet's Topo class where you define the topology structure.
- **addSwitch('s1'):** Adds a virtual switch named s1 to the topology.
- **addHost('h1') and addHost('h2'):** Adds two virtual hosts named h1 and h2.
- **addLink(host1, switch) and addLink(host2, switch):** Connects the hosts (h1 and h2) to the switch (s1) using virtual links.
- **MyTopo():** Creates an instance of the custom topology defined in the MyTopo class.
- **Mininet(topo=topo):** Creates a Mininet object using the custom topology (topo).

- **net.start():** This initializes and activates the virtual network.
- **pingAll() :** This function sends a ping from every host to every other host in the network. In this case, it will check the connectivity between h1 and h2.
- **net.stop():** This command shuts down the virtual network and releases resources.

3.6 SDN Controller and Data Plane Communication

In Software-Defined Networking (SDN), the controller and data plane work together to manage and forward network traffic:

How Does Software-Defined Networking (SDN) Works?

In Software-Defined Networking (SDN), the software that controls the network is separated from the hardware. SDN moves the part that decides where to send data (control plane) to software, while the part that actually forwards the data (data plane) stays in the hardware.

.

3.6 SDN Controller and Data Plane Communication

In Software-Defined Networking (SDN), the controller and data plane work together to manage and forward network traffic:

SDN Controller:-

- Acts as the brain of the network.
- Operates in the control plane, where decisions about routing and forwarding are made.
- Communicates with the data plane using southbound APIs like OpenFlow.
- Examples: OpenDaylight, Ryu, ONOS.

Data Plane:-

- Responsible for forwarding packets based on instructions from the controller.
- Includes switches and routers that execute the controller's commands.
- Operates at high speed to ensure efficient packet forwarding.

Communication:

- The controller sends flow rules to the data plane devices via protocols like OpenFlow.
- Data plane devices report statistics and events back to the controller, enabling dynamic adjustments.

3.7 Routing Configuration in SDN

Routing Configuration in SDN

Instead of using distributed protocols (like OSPF), an SDN controller centrally computes the best route and programs switches accordingly.

Routing in SDN is centralized and programmable, unlike traditional networks:

Centralized Control:-

- The SDN controller manages routing decisions for the entire network.
- It uses algorithms like Dijkstra's Shortest Path to calculate optimal paths for data packets.

Dynamic Routing:-

- Routes can be adjusted in real-time based on network conditions (e.g., congestion, failures).
- This is achieved through APIs and protocols like REST or OpenFlow.

Example:-

- A controller can implement routing protocols like OSPF or BGP virtually, without requiring physical routers.
- It can also define custom routing policies, such as prioritizing certain types of traffic.

3.8 Open-Source Network Monitoring - Nagios

Nagios is a popular open-source tool for monitoring IT infrastructure:

- Features:- Monitors servers, switches, routers, applications, and services.
- Provides alerts for issues like downtime, performance degradation, or resource exhaustion.
- Supports plugins for extended functionality.

How It Works:-

- Nagios Core acts as the monitoring engine.
- Plugins collect data from devices and services.
- Alerts are sent via email, SMS, or other methods when issues are detected.

Example Use Case:-

- Monitor a network switch for uptime and bandwidth usage.
- Set thresholds for CPU or memory usage on servers, triggering alerts when exceeded.

Assignment:

- Explain the process to use network troubleshooting commands.
- How can you configure the windows & Linux firewalls system?
- Explain the steps required to configure a network interface in Linux and Windows.
- Describe common network startup issues and outline a step-by-step diagnostic process to identify and resolve such problems.
- Develop a simple network topology using Mininet.
- Discuss the advantages of centralized routing in software-defined networks.

Unit- 4 DHCP (Dynamic Host Configuration Protocol) – 3 hrs

- **DHCP Principle**
- **DHCP Options, Scope, Reservation & Relaying**
- **DHCP Troubleshooting**

DHCP

Introduction to DHCP

The **Dynamic Host Configuration Protocol (DHCP)** is a **network management protocol** used to automatically assign IP addresses and other network configuration details (such as subnet mask, gateway, and DNS servers) to devices on a network. DHCP eliminates the need for manual IP configuration, making network administration more efficient and less error-prone.

DHCP simplifies network management by automating IP address allocation. It is an essential protocol for both small and large networks, reducing administrative effort and improving efficiency. However, proper security measures should be implemented to prevent misuse or attacks.

DHCP

- **How DHCP Works**

The DHCP process consists of four primary steps, often referred to as **DORA**:

(a) Discovery (D)

1. When a client (such as a computer or mobile device) connects to a network, it does not have an IP address.
2. The client sends a **DHCPDISCOVER** message as a **broadcast** (sent to **255.255.255.255**) to locate available DHCP servers.
3. The message includes:
    1. Client's MAC address
    2. Request for IP address
    3. Optional requested parameters (such as gateway, DNS, etc.)

DHCP

(b) Offer (O)

1. A DHCP server on the network receives the **DHCPDISCOVER** message and checks its pool of available IP addresses.
2. The server sends a **DHCPOFFER** message back to the client, offering an available IP address along with other network parameters.
3. This offer is also broadcasted so that multiple DHCP servers can compete to offer their service.

DHCP

(c) Request (R)

1. The client receives one or more **DHCPOFFER** messages and selects the most suitable offer (usually the first one received).
2. The client sends a **DHCPREQUEST** message to the selected server, requesting to lease the offered IP address.
3. This message is broadcasted to inform other DHCP servers that it has accepted an offer from a specific server.

DHCP

- **(d) Acknowledgment (A)**

1. The DHCP server confirms the lease by sending a **DHCPACK** (Acknowledgment) message to the client.
2. The **DHCPACK** message contains:
    1. The leased IP address
    2. Subnet mask
    3. Default gateway
    4. DNS servers
    5. Lease duration (how long the IP is valid)
3. The client configures its network settings using the provided details and begins communication on the network.

DHCP

- **(d) Acknowledgment (A)**

1. The DHCP server confirms the lease by sending a **DHCPACK** (Acknowledgment) message to the client.
2. The **DHCPACK** message contains:
    1. The leased IP address
    2. Subnet mask
    3. Default gateway
    4. DNS servers
    5. Lease duration (how long the IP is valid)
3. The client configures its network settings using the provided details and begins communication on the network.

DHCP

- **DHCP Lease Process**
- The IP address assigned by DHCP is temporary and is known as a **lease**.
- The lease duration determines how long the client can use the assigned IP before it needs renewal.
- If the lease expires, the client must request a renewal or get a new IP address.
- **Lease Renewal Process**

1. **T1 Timer (Renewal Request)**:
    1. When 50% of the lease time has passed, the client sends a **DHCPREQUEST** directly to the DHCP server to renew the lease.
    2. If the server responds with **DHCPACK**, the lease is extended.
2. **T2 Timer (Rebinding Phase)**:
    1. If the client does not receive a renewal response by 87.5% of the lease time, it sends a **broadcast DHCPREQUEST** to any available DHCP server.
    2. If a DHCP server responds, the lease is extended.
3. **Lease Expiry**:
    1. If the lease expires and no renewal is received, the client must restart the DORA process to obtain a new IP.

DHCP

DHCP Components

(a) DHCP Server

- A server that assigns IP addresses and other network configurations to clients.
- Can be a dedicated server, a router, or even a network switch.

(b) DHCP Client

- Any device (computer, smartphone, printer, IoT device) that requests an IP address from the DHCP server.

(c) DHCP Relay Agent

- Used when DHCP clients and servers are on different networks.
- The relay agent forwards DHCP messages between clients and the DHCP server.

DHCP

(d) DHCP Scope

- A range of IP addresses that a DHCP server can assign.
- Example: 192.168.1.100 – 192.168.1.200

(e) DHCP Exclusions and Reservations

- **Exclusions:** Specific IP addresses that the DHCP server will not assign.
- **Reservations:** Assigns a **specific IP** to a particular MAC address.

DHCP

Message Type	Description
DHCPDISCOVER	Client requests an IP address.
DHCPOFFER	Server offers an IP address.
DHCPREQUEST	Client accepts the offer.
DHCPACK	Server confirms IP lease.
DHCPNAK	Server denies request (e.g., lease expired).
DHCPDECLINE	Client refuses the offer (e.g., IP conflict).
DHCPRELEASE	Client releases the IP address.
DHCPINFORM	Client requests additional information (e.g., DNS, gateway).

DHCP

DHCP Deployment Modes

(a) Automatic Allocation

- Assigns an IP address permanently to a device.
- Useful for devices that rarely change networks.

(b) Dynamic Allocation

- Assigns an IP for a limited lease time.
- Most common in large networks.

(c) Manual (Static) Allocation

- Administrator assigns fixed IP addresses using DHCP reservations.

DHCP

DHCP Advantages

- **Reduces manual configuration** – No need to assign IPs manually.
- **Minimizes IP conflicts** – DHCP ensures that no two devices get the same IP.
- **Centralized management** – IT admins can manage network settings from a single point.
- **Supports mobility** – Devices can move between networks and obtain IP addresses dynamically.
- **Improves scalability** – Ideal for large networks with many devices.

DHCP

DHCP Disadvantages

- **Single point of failure** – If the DHCP server is down, new clients cannot obtain IPs.
- **Security risks** – Rogue DHCP servers can assign malicious IP configurations.
- **IP lease delays** – Devices may temporarily lose network access if the lease expires.
- **Broadcast traffic** – DHCP relies on broadcasts, which may increase network congestion.

DHCP Options, Scope, Reservation & Relaying

DHCP Options

DHCP options are basic settings that a client needs for proper network communication. These options include an IP address, a subnet mask, a default gatewar, primary and secondary DNS servers, primary and secondary Windows Internet Name Service (WINS) if applicable, and DHCP lease expiration. You can define these options when creating the scope or change them later.

DHCP options define additional settings assigned to clients, such as:

- **Option 1** – Subnet Mask
- **Option 3** – Default Gateway
- **Option 6** – DNS Servers
- **Option 15** – Domain Name
- **Option 66 & 67** – PXE Boot Settings for network booting

DHCP Options, Scope, Reservation & Relaying

DHCP Scope

A DHCP Scope is a range of IP addresses and related configuration information available by request from a DHCP client. These scopes usually represent a single subnet, or segment of a network. Each scope is a continuous range of IP addresses defined by a beginning IP address and an ending IP address. If you need to exclude IP addresses, you must create exclusions for those addresses. One reason for creating these addresses might be hardware with static IP addresses, like printers.

In Short:

- A scope is a range of IP addresses that a DHCP server can allocate.
- Typically defined for each subnet.

DHCP Options, Scope, Reservation & Relaying

DHCP Reservation

- IP Reservation: Ensures specific devices always get the same IP.
- Uses MAC address binding.

When would you reserve an IP address? Well, in some cases, a network device needs to have a static IP address. An example would be a server, a router, or a network printer.

DHCP Options, Scope, Reservation & Relaying

DHCP Relaying

- Used when a DHCP server is on a different network.
- Relay Agents (IP Helpers) forward DHCP requests across subnets.

DHCP Troubleshooting

Common DHCP Issues and Solutions

Issue	Possible Cause	Solution
Clients not receiving IPs	DHCP server down, scope exhausted, network issues	Check DHCP server status, ensure sufficient IPs, verify network connectivity
IP Conflict	Duplicate static IP assignment	Avoid static IPs within DHCP range
Clients receiving wrong subnet IPs	Incorrect scope settings	Verify and adjust scope options
DHCP Relay not working	Incorrect relay configuration	Ensure relay agent is enabled and correctly configured
Slow DHCP response	Network congestion or server overload	Optimize network performance and check server capacity

DHCP Troubleshooting

- The most common problems with DHCP usually aren't related to the server; after the server is configured correctly there is no need to change any settings and it therefore runs reliably. The problems usually occur at the DHCP client's end for a variety of reasons.
- Whenever Microsoft DHCP clients are unable to contact their DHCP server they default to selecting their own IP address from the 169.254.0.0 network until the DHCP server becomes available again. This is frequently referred to as Automatic Private IP Addressing (APIPA). Here are some steps you can go through to resolve the problem:
  - Ensure that your DHCP server is configured correctly and use the pgrep command discussed earlier to make sure the DHCP process is running. Pay special attention to your 255.255.255.255 route, especially if your DHCP server has multiple interfaces.
  - Give your DHCP client a static IP address from the same range that the DHCP server is supposed to provide. See whether you can ping the DHCP server. If you cannot, double-check your cabling and your NIC cards.

DHCP Troubleshooting

- DHCP uses the BOOTP protocol for its communication between the client and server. Make sure there are no firewalls blocking this traffic. DHCP servers expect requests on UDP port 67 and the DHCP clients expect responses on UDP port 68. Use tcpdump on the server's NIC to verify the correct traffic flows.

- Most problems with an initial setup are often due to:
  - Incorrect settings in the /etc/dhcpd.conf file such as not defining the networks for which the DHCP server is responsible;
  - Firewall rules that block the DHCP bootp protocol on UDP ports 67 and 68;
  - Routers failing to forward the bootp packets to the DHCP server when the clients reside on a separate network.

Basic Troubleshooting Commands:

- **ipconfig /release & ipconfig /renew** (Windows)
- **dhclient -r && dhclient** (Linux)
- **show ip dhcp binding** (Cisco routers)
- **show dhcp leases** (Linux DHCP server)

Assignement-2

- Explain with principle the DHCP server and client communication process.
- Explain the DHCP principle and applications.
- How does of client machine get IP address via DHCP lease? Explain.
- What do you mean by DHCP lease? How DHCP relay works? Explain.
- How can you configure DHCP server? Explain the process.

Name Server & Configuration

- DNS Principles & Operations
- Basic Name Server & Client Configuration
- Caching only Name Server
- Primary & Slave Name Server
- DNS Zone Transfers
- DNS Dynamic Updates
- DNS Delegation
- DNS Server Security
- Troubleshooting

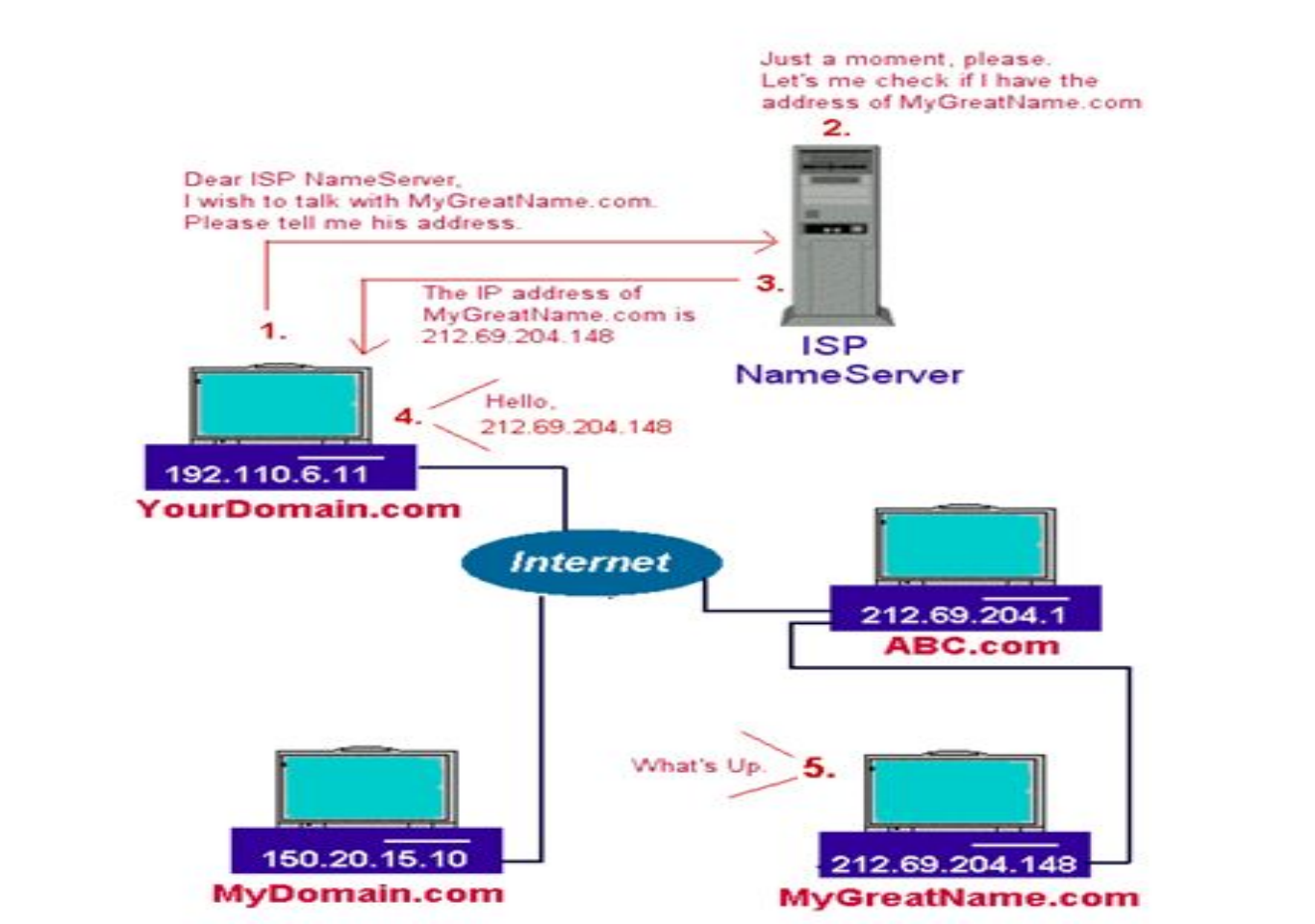
DNS Principles & Operations

- What Is DNS?
  Domain Name System (DNS) provides name resolution for all Internet communications and many private networks, including all Windows Active Directory domains. This means that, whenever you type a website's address into your Web browser, DNS will go out and fetch the IP address associated with that address. This beats having to remember the IP address 69.147.114.224 every time you want to go to Yahoo.com. DNS also assists the delivery of e-mail by providing mail exchanger records that tell SMTP servers where to send e-mail messages.

- Basic Concept of Domain Name System (DNS)
  When the computers (hosts) on the Internet is getting more and more, and thus the growing list of HOSTS.txt. The solutions used in the early stage of Internet was not more suitable. In view of this, Paul Mockapertris designed a system to manage the domain names on the Internet. The system is called Domain Name System, or DNS in short, in 1984

DNS Principles & Operations

- Working Procedure of DNS



DNS Principles & Operations

- Working Procedures of DNS:

1. The information (mainly domain names and their corresponding IP Addresses) of hosts (computers) on the Internet is saved in the Domain Name Servers. The Domain Name Servers are distributed widely on the Internet.

2. When your computer need to connect with a host on the Internet (e.g. MyGreatName.com), you only need to enter the Domain Name (e.g. MyGreatName.com) in the URL of browser. Your computer will then contact the configured or default Name Servers (usually your ISP Name Server), asking for the IP Address of the host (e.g. MyGreatName.com).

3. The Name Server will then tell your computer the IP Address of the query host.

DNS Principles & Operations

4. Once your computer get the IP Address of the host, your computer can then communicate with the host.
From the above working procedures of DNS, you should notice that there are a lot of disadvantages.
For example:

- Each Name Server has to save the information of ALL hosts on the Internet.
- If a Name Server forgets to update the information, many new domain names will not be found!
- To guarantee that all new domains are activated on the Internet, the information of all Name Servers must be updated. This may take 2 - 3 months!
- How to guarantee that all Name Servers are updated on schedule?

The above procedures only show the basic concept of DNS.

DNS Operation

DNS Servers

On the client side, a DNS resolver is used to send queries to DNS servers. The resolver is normally part of a library routine or it is built into the application. DNS uses zone files to keep name and IP address database information for the internet domain or hierarchical set of domains. Zones are storage of information in a file for a DNS domain or DNS subdomains (DNS domains are not the same as Windows domains). DNS does not yet support dynamic configuration but has been modified for Windows systems to do so. Different aliases may be created by the administrator for the same host. Three types of name servers as defined by how it relates to the zone information:

DNS Principles & Operations

1. **Primary** - Locally stored files exist on the name server data base. The master zone file copy is stored here.
2. **Secondary** - Gets data called a zone transfer from another server that is the zone authority.
3. **Caching Only** - Caches name server information and does not contain its own files.

A primary and secondary name server should be used on a network. When a zone is defined, some ser ver must be configured to be a master name server for the zone. There can be different master name servers for different zones. The master server provides copies of the zone information to the secondary DNS server. Name servers can be configured to get information from other name servers when the information is not found in the local database. These types are forwarders and slaves.

DNS Principles & Operations

Name servers as categorized by function:

1. Master - The zone authority that contains the master zone files.
2. Forwarders - A name server that passes name resolution requests to other name servers. This configuration is done on a per server basis.
3. Slaves - Slave name servers are configured to use forwarders.

DNS Principles & Operations

Queries

Query types are:

1. Inverse - Getting the name from the IP address. These are used by servers as a security check.
2. Iterative - Server gives its best answer. This type of inquiry is sent from one server to another.
3. Recursive - Cannot refer the query to another name server.

DNS Principles & Operations

Zone Transfers

The DNS zone file serial number is used to track DNS changes. The notify function is used to initiate zone transfers. Zone transfer types are:

1. Full - AXFR Query - Secondary server refresh interval expires and it sends an AXFR query.
2. Incremental - IXFR query - Only new or updated entries are copied.

DNS Principles & Operations

Zone Transfers

The DNS zone file serial number is used to track DNS changes. The notify function is used to initiate zone transfers. Zone transfer types are:

1. Full - AXFR Query - Secondary server refresh interval expires and it sends an AXFR query.
2. Incremental - IXFR query - Only new or updated entries are copied.

DNS Principles & Operations

DNS Zones

- **Possible zones include:**
- **Forward lookup zone** - Name to IP address map.
- **Reverse lookup zone** - IP address to name map.
- **Standard primary zone (primary zone)** - A master copy of a forward or reverse lookup zone.
- **Active Directory integrated zone** - A copy of a standard primary or Active Directory integrated zone. The IP address and computer name is stored in Active Directory and replicated to all local domain controllers. DNS information is not replicated to domain controllers outside the domain.
- **Standard secondary zone (secondary zone)**

DNS Principles & Operations

Name Server

A name server translates domain names into IP addresses. This makes it possible for a user to access a website by typing in the domain name instead of the website's actual IP address. For example, when you type in "www.microsoft.com," the request gets sent to Microsoft's name server which returns the IP address of the Microsoft website. Each domain name must have at least two name servers listed when the domain is registered. These name servers are commonly named ns1.servername.com and ns2.servername.com, where "servername" is the name of the server. The first server listed is the primary server, while the second is used as a backup server if the first server is not responding. Name servers are a fundamental part of the Domain Name System (DNS). They allow websites to use domain names instead of IP addresses, which would be much harder to remember. In order to find out what a certain domain name's name servers are, you can use a WHOIS lookup tool.

DNS Principles & Operations

DNS Name Server Type:

Primary: The primary nameserver is the authoritative source for all information about a specific domain. It loads the domain information from a locally maintained disk file that is built by the domain administrator. This file (the zone file) contains the most accurate information about a piece of the domain hierarchy over which this server has authority. The primary server is a master server, because it can answer any query about its domain with full authority. The terms master server and authoritative server are used interchangeably. Configuring a primary server requires creating a complete set of configuration files: zone files for the regular domain and the reverse domain, the boot file, the cache file, and the loopback file. No other configuration requires creating this complete set of files.

Secondary(replicating): A secondary server transfers a complete set of domain information from the primary server. The zone file is transferred from the primary server and stored on the secondary server as a local disk file. This transfer is called a zone file transfer. A secondary server keeps a complete copy of all domain information, and can answer queries about that domain with authority. Therefore, a secondary server is also considered a master server. Configuring a secondary server does not require creating local zone files, because the zone files are downloaded from the primary server. However, the other files (a boot file, a cache file, and a loopback file) are required.

DNS Principles & Operations

DNS Name Server Type:

Caching-only: A caching-only server runs the nameserver software, but keeps no nameserver database files. It learns the answer to every nameserver query from some remote server. Once it learns an answer, the server caches the answer and uses it to answer future queries for the same information. All nameservers use cached information in this manner, but a caching-only server depends on this technique for all of its nameserver information. It is not considered an authoritative (or master) server, because all of the information it provides is secondhand. Only a boot file and a cache file are required for a caching-only configuration. But the most common configuration also includes a loopback file. This is probably the most common nameserver configuration, and apart from the resolver-only configuration, it is the easiest to configure.

DNS Principles & Operations

DNS Name Server Type:

Caching-only: A caching-only server runs the nameserver software, but keeps no nameserver database files. It learns the answer to every nameserver query from some remote server. Once it learns an answer, the server caches the answer and uses it to answer future queries for the same information. All nameservers use cached information in this manner, but a caching-only server depends on this technique for all of its nameserver information. It is not considered an authoritative (or master) server, because all of the information it provides is secondhand. Only a boot file and a cache file are required for a caching-only configuration. But the most common configuration also includes a loopback file. This is probably the most common nameserver configuration, and apart from the resolver-only configuration, it is the easiest to configure.

DNS Principles & Operations

DNS Cache

DNS is a caching protocol. When a client queries its local DNS server, and the local DNS server is not authoritative for the query, then this server will go looking for an authoritative name server in the DNS tree. The local name server will first query a root server, then a TLD server and then a domain server. When the local name server resolves the query, then it will relay this information to the client that submitted the query, and it will also keep a copy of these queries in its cache. So when a(nother) client submits the same query to this name server, then it will retrieve this information form its cache.

DNS Principles & Operations

For example,
a client queries for the A record on www.linux-training.be to its local server. This is the first query ever received by this local server. The local server checks that it is not authoritative for the linux-training.be domain, nor for the .be TLD, and it is also not a root server. So the local server will use the root hints to send an iterative query to a root server. The root server will reply with a reference to the server that is authoritative for the .be domain (root DNS servers do not resolve fqdn's, and root servers do not respond to recursive queries). The local server will then sent an iterative query to the authoritative server for the .be TLD. This server will respond with a reference to the name server that is authoritative for the linux-training.be domain. The local server will then sent the query for www.linux training.be to the authoritative server (or one of its slave servers) for the linux-training.be domain. When the local server receives the ip-address for www.linux-training.be, then it will provide this information to the client that submitted this query. Besides caching the A record for www.linux-training.be, the local server will also cache the NS and A record for the linux training.be name server and the .be name server.

DNS zone transfer

DNS zone transfer, also sometimes known by the inducing DNS query type AXFR, is a type of DNS transaction. It is one of the many mechanisms available for administrators to replicate DNS databases across a set of DNS servers. Zone transfer comes in two flavors, full (AXFRRFC 1035) and incremental (IXFRRFC 1995). Nearly universal at one time, it is now becoming less popular in favor of the use of other database replication mechanisms that modern DNS server packages provide.

Operation

Zone transfer operates on top of the Transmission Control Protocol (TCP), and takes the form of a client–server transaction. The parties involved in a zone transfer are a client (the "slave" requesting the data from a portion of the database to be transferred to it) and a server (the "master" supplying those data from its database). Some sources refer to the slave as a "secondary" server and the master as a "primary" server. The portion of the database that is replicated is a "zone". Zone transfer comprises a preamble followed by the actual data transfer. The preamble comprises a lookup of the SOA (Start of Authority) resource record for the "zone apex", the node of the DNS namespace that is at the top of the "zone". The fields of this SOA resource

record, in particular the "serial number", determine whether the actual data transfer need occur at all. The client compares the serial number of the SOA resource record with the serial number in the last copy of that resource record that it has. If the serial number of the record being transferred is greater, the data in the zone are deemed to have "changed" (in some fashion) and the slave proceeds to request the actual zone data transfer. If the serial numbers are identical, the data in the zone are deemed not to have "changed", and the client may continue to use the copy of the database that it already has, if it has one.

DNS Principles & Operations

The actual data transfer proper begins by the client sending a query (opcode 0) with the special QTYPE (query type) AXFR (value 252) over the TCP connection to the server. The server responds with a series of response messages, comprising all of the resource records for every domain name in the "zone". The first response comprises the SOA resource record for the zone apex. The other data follow in no specified order. The end of the data is signalled by the server repeating the response containing the SOA resource record for the zone apex.

Some zone transfer clients perform the SOA lookup of the preamble using their system's normal DNS query resolution mechanism. These clients do not open a TCP connection to the server until they have determined that they need to perform the actual data transfer. However, since TCP can be used for normal DNS transactions, as well as for zone transfer, other zone transfer clients perform the SOA lookup preamble over the same TCP connection as they then (may) perform the actual data transfer. These clients open the TCP connection to the server before they even perform the preamble.

DNS Principles & Operations

The preceding describes full zone transfer. Incremental zone transfer differs from full zone transfer in the following respects:

- The client uses the special QTYPE IXFR (value 251) instead of the AXFR QTYPE.
- The client sends the SOA resource record for the zone apex that it currently has, if any, in the IXFR message, letting the server know which version of the "zone" it believes to be current.
- Though the server may respond in the normal AXFR manner with the full data for the zone, it may also instead respond with an "incremental" data transfer. This latter comprises the list of changes to the zone data, in zone serial number order, between the version of the zone that the client reported to the server as having and the version of the zone that is current at the server. The changes comprise two lists, one of resource records that are deleted and one of resource records that are inserted. (A modification to a resource record is represented as a deletion followed by an insertion.)

DNS Principles & Operations

Zone transfer is entirely client-initiated. Though servers can send a NOTIFY message to clients (that they have been informed about) whenever a change to the zone data has been made, the scheduling of zone transfers is entirely under the control of the clients. Clients schedule zone transfers initially, when their databases are empty, and thereafter at regular intervals, in a pattern controlled by the values in the "refresh", "retry", and "expire" fields in the SOA resource record of the zone apex.

Limitations

Though it is standardized, full-zone transfer being described as one of the possible database replication mechanisms in RFC 1034 (incremental zone transfer described in RFC 1995), zone transfer is the most limited of those database replication mechanisms. Zone transfer operates in terms of "wire format" resource records, i.e. resource records as they are transferred using the DNS protocol. However, the schema of wire format resource records may not match the database schema used by the back ends of the DNS servers themselves.

Dynamic update

Dynamic update enables DNS client computers to register and dynamically update their resource records with a DNS server whenever changes occur. This reduces the need for manual administration of zone records, especially for clients that frequently move or change locations and use DHCP to obtain an IP address. The DNS Client and Server services support the use of dynamic updates, as described in Request for Comments (RFC) 2136, "Dynamic Updates in the Domain Name System." The DNS Server service allows dynamic update to be enabled or disabled on a per-zone basis at each server configured to load either a standard primary or directory-integrated zone. By default, the DNS Client service will dynamically update host (A) resource records (RRs) in DNS when configured for TCP/IP.

How client and server computers update their DNS names?

By default, computers that are statically configured for TCP/IP attempt to dynamically register host (A) and pointer (PTR) resource records (RRs) for IP addresses configured and used by their installed network connections. By default, all computers register records based on their fully qualified domain name (FQDN). The primary full computer name, a FQDN, is based on the primary DNS suffix of a computer appended to its Computer name. Both of these settings are displayed or configured from the Computer Name tab in System properties.

DNS Principles & Operations

Dynamic updates can be sent for any of the following reasons or events:

- An IP address is added, removed, or modified in the TCP/IP properties configuration for any one of the installed network connections.
- An IP address lease changes or renews with the DHCP server any one of the installed network connections. For example, when the computer is started or if the ipconfig /renew command is used.
- The ipconfig /registerdns command is used to manually force a refresh of the client name registration in DNS.
- At startup time, when the computer is turned on.
- A member server is promoted to a domain controller.

DNS Principles & Operations

When one of the previous events triggers a dynamic update, the DHCP Client service (not the DNS Client service) sends updates. This is designed so that if a change to the IP address information occurs because of DHCP, corresponding updates in DNS are performed to synchronize name-to-address mappings for the computer. The DHCP Client service performs this function for all network connections used on the system, including connections not configured to use DHCP.

How dynamic update works?

Dynamic updates are typically requested when either a DNS name or IP address changes on the computer. For example, suppose a client named "oldhost" is first configured in System properties with the following names:

Computer name = oldhost
DNS domain name of computer = example.microsoft.com
Full computer name = oldhost.example.microsoft.com

In this example, no connection-specific DNS domain names are configured for the computer. Later, the computer is renamed from "oldhost" to "newhost", resulting in the following name changes on the system:

Computer name = newhost
DNS domain name of computer = example.microsoft.com
Full computer name = newhost.example.microsoft.com

DNS Principles & Operations

Once the name change is applied in System properties, you are prompted to restart the computer. When the computer restarts Windows, the DHCP Client service performs the following sequence to update DNS:

1. The DHCP Client service sends a start of authority (SOA) type query using the DNS domain name of the computer. The client computer uses the currently configured FQDN of the computer (such as "newhost.example.microsoft.com") as the name specified in this query.

2. The authoritative DNS server for the zone containing the client FQDN responds to the SOA-type query.
   For standard primary zones, the primary server (owner) returned in the SOA query response is fixed and static. It always matches the exact DNS name as it appears in the SOA RR stored with the zone. If, however, the zone being updated is directory integrated, any DNS server loading the zone can respond and dynamically insert its own name as the primary server (owner) of the zone in the SOA query response.

DNS Principles & Operations

3. The DHCP Client service then attempts to contact the primary DNS server. The client processes the SOA query response for its name to determine the IP address of the DNS server authorized as the primary server for accepting its name. It then proceeds to perform the following sequence of steps as needed to contact and dynamically update its primary server:
    1. It sends a dynamic update request to the primary server determined in the SOA query response. If the update succeeds, no further action is taken.
    2. If this update fails, the client next sends an NS-type query for the zone name specified in the SOA record.
    3. When it receives a response to this query, it sends an SOA query to the first DNS server listed in the response.
    4. After the SOA query is resolved, the client sends a dynamic update to the server specified in the returned SOA record. If the update succeeds, no further action is taken.
    5. If this update fails, then the client repeats the SOA query process by sending to the next DNS server listed in the response.

DNS Principles & Operations

4. Once the primary server is contacted that can perform the update, the client sends the update request and the server processes it. The contents of the update request include instructions to add A (and possibly PTR) RRs for "newhost.example.microsoft.com" and remove these same record types for "oldhost.example.microsoft.com", the name that was previously registered. The server also checks to ensure that updates are permitted for the client request. For standard primary zones, dynamic updates are not secured, so any client attempt to update succeeds. For Active Directory-integrated zones, updates are secured and performed using directory-based security settings.

DNS Principles & Operations

Dynamic updates are sent or refreshed periodically. By default, computers send a refresh once every 7 days. If the update results in no changes to zone data, the zone remains at its current version and no changes are written. Updates result in actual zone changes or increased zone transfer only if names or addresses actually change. Note that names are not removed from DNS zones if they become inactive or are not updated within the refresh interval (7 days). DNS does not use a mechanism to release or tombstone names, although DNS clients do attempt to delete or update old name records when a new name or address change is applied. When the DHCP Client service registers A and PTR resource records for a computer, it uses a default caching Time to Live (TTL) of 15 minutes for host records. This determines how long other DNS servers and clients cache a computer's records when they are included in a query response.

DNS Principles & Operations

IN Simpler Way:

Before the Name Change:

- The computer originally has the name **"oldhost"**.
- It does not have any specific DNS domain name set.
- After renaming, it becomes **"newhost"** with the DNS domain **"example.microsoft.com"**.
- The **Full Computer Name (FQDN)** becomes **"newhost.example.microsoft.com"**.
- The system prompts for a **restart** to apply the changes.

DNS Principles & Operations

After the Computer Restarts:

The **DHCP Client service** updates the DNS records in the following steps:

1. Checking the DNS Authority (SOA Query)

- The client (computer) asks the DNS server, **"Who is responsible for my domain (example.microsoft.com)?"**
- This is done by sending a **Start of Authority (SOA) query** to the DNS server.

2. DNS Server Response

- The **DNS server** responds with the **primary DNS server** responsible for handling updates for the **example.microsoft.com** zone.
- If the DNS is **Active Directory-integrated**, any DNS server in the domain can respond.

DNS Principles & Operations

3. Contacting the Primary DNS Server

- The client tries to **contact the primary DNS server** provided in the SOA response.
- It sends a **dynamic update request** to register its new name (**newhost.example.microsoft.com**) and remove the old one (**oldhost.example.microsoft.com**).
- If the update **succeeds**, nothing more is needed.
- If the update **fails**, the client follows backup steps:
  - It asks the DNS server for a list of **other DNS servers** handling the zone.
  - It then contacts another server and repeats the update process.

DNS Principles & Operations

4. Updating the DNS Records

- Once a primary server **accepts the update**, it updates:
  - **A record (Address Record)** → Links the new hostname (**newhost.example.microsoft.com**) to its IP address.
  - **PTR record (Pointer Record)** → Updates reverse DNS lookup for the new hostname.
  - The old hostname (**oldhost.example.microsoft.com**) is removed from DNS.

5. DNS Security & Permissions

- If the DNS zone is a **standard primary zone**, any client can update it freely.
- If it is an **Active Directory-integrated zone**, updates follow **security policies** (i.e., only authorized devices can update records).

DNS Principles & Operations

6. Periodic Refresh of DNS Records

- The computer **refreshes** its DNS registration **every 7 days**.
- If nothing changes, the DNS records remain the same.
- DNS **does not** automatically remove inactive records unless a client explicitly deletes them.

7. Caching and Time to Live (TTL)

- When the DHCP Client service updates the DNS, the records are cached with a **TTL (Time To Live) of 15 minutes**.
- This means other computers will hold the record in memory for **15 minutes** before checking again.

DNS Principles & Operations

Summary:

1. Computer is renamed → Restart required.
2. Computer queries DNS to find the authoritative server for updates.
3. DNS server responds with primary DNS information.
4. Computer contacts the primary DNS server and requests a **name update**.
5. Old DNS records are removed, new ones are added.
6. If the update fails, the client tries another DNS server.
7. The client refreshes its registration **every 7 days**.
8. DNS records are cached for **15 minutes**.

DNS Principles & Operations

Delegating Domain Names

Domain delegation means placement of DNS servers list with the domain technical records (zone-file) to DNS servers enabling top level domains functioning. Delegation is a prerequisite for site and mail operation on the domain.

DNS delegation is when the authority for a subdomain is handed over to another set of name servers. This means:

- If a top-level domain (TLD) DNS server (like .com) gets a request for a domain (like google.com), it does not store all the details.
- Instead, it redirects the request to Google's name servers that hold the details for google.com.

DNS Principles & Operations

What is Delegation Used For?

- DNS delegation is required for:
  - Domain Delegation to Hosting – Connecting a domain to a website hosting service.
  - Domain and Mail Redirection Services – Redirecting web traffic or emails to a different address.
  - Primary and Secondary DNS Services – Managing domain name resolution with multiple DNS servers for reliability.

DNS Principles & Operations

What is dns delegation?

When the authoritative name server for a domain receives a request for a subdomain's records and responds with NS records for other name servers, that is DNS delegation. Essentially it is saying "I am passing on authority for this subdomain to another collection of name servers, go ask them for the details." For example, "com" is a domain that delegates all (or perhaps almost all) of its subdomains to other name servers. When a request is received at a "com" name server for "google.com", the "com" name server responds with NS records pointing to Google's name servers.

DNS Principles & Operations

The dig command with the +trace option will demonstrate. Try `dig +trace google.com` and part of it will look like this:

google.com. 172800 IN NS ns2.google.com.

google.com. 172800 IN NS ns1.google.com.

google.com. 172800 IN NS ns3.google.com.

google.com. 172800 IN NS ns4.google.com.

;; Received 164 bytes from 192.31.80.30#53(d.gtld-servers.net) in 150 ms

Those NS records indicate that those 4 Google name servers are authoritative for the google.com domain.

Delegation Periods, Suspending/Unsuspending

Within the whole domain registration period the Registrant may suspend delegation in "Manage your account" → My domains“

(for gTLDs and ccTLDs the domain should be delegated with an empty DNS servers list).

Earlier suspended domain will be unsuspended subject to domains delegation periods.

Domains delegation periods are subject to the moment the modifications are made:

- for .RU domain - up to 9 hours;
- for .РФ and .SU domains - up to 2 hours;
- for third-level domains - up to 1 hour;
- delegation of gTLDs and ccTLDs takes several minutes.

DNS Principles & Operations

DNS Servers Testing

During domain delegation the specified DNS servers may be checked for operation. If testing proves successful the domain will be delegated with the new DNS servers.

How to Complete DNS Servers List

- specify at least two DNS servers;
- if you specify DNS servers, containing the name of the IDN (Internationalized Domain Names), the name of this IDN should be entered with "XN--" prefix, but not in national script;
  - Some domains use non-Latin characters (e.g., Arabic, Chinese, or Cyrillic).
  - The DNS system does not process directly written non-Latin characters.
  - Instead, you must convert the name into Punycode, which starts with "XN--"
  - Example: пример.рф (Cyrillic) → XN--e1afmkfd.xn--p1ai (Punycode)

DNS Principles & Operations

Irregular Cases

Case: Child DNS Servers Need an IP Address

What is a Child DNS Server?

If your DNS server name contains your own domain name, it is called a child DNS server (or "glue record").

Example:

You own **example.com**

You want to use **ns1.example.com** as your DNS server

Why Do You Need to Provide an IP Address?

Normally, DNS servers find the IP address by looking up another DNS server.

But in this case, it cannot look itself up (it creates a loop).

To fix this, you must manually provide the IP address for ns1.example.com.

DNS Principles & Operations

Case: .PRO Domains Only Support IPv4

What is .PRO?

- .PRO is a special domain extension (TLD) for professionals (like doctors, lawyers, etc.).
- .PRO only allows IPv4 addresses for DNS servers.
- IPv6 addresses are NOT supported.
- Eg: ns1.example.pro → 192.168.1.1 (IPv4)

DNS Principles & Operations

Case: .ORG Domains Need Different IPs for Child DNS Servers

If you are using child DNS servers (like ns1.example.org, ns2.example.org), they must have different IP addresses.

Why?

This ensures better redundancy—if one server goes down, the other still works.

Eg: ns1.example.org → 192.168.1.1

ns2.example.org → 192.168.1.2

Not Allowed:

ns1.example.org → 192.168.1.1

ns2.example.org → 192.168.1.1 (same IP, which is not allowed)

DNS Principles & Operations

Assignment:

- Explain about DNS principles and operations.
- What do you mean by caching only name server?
- What are the difference types of DNS queries? How does DNS zone transfer occur?
- What do you mean by DNS delegation? Explain.
- Why do we need slave name server? How does primary and slave server communicate with each other?
- What is FQDN? How does DNS delegation work? Explain.
- Explain how cashing-only name server works? What is the difference between iterative and recursive approach of DNS queries.

Unit 6: Web & Proxy Server Configuration

6.1 HTTP Server Configuration Basis

6.2 Virtual Hosting

6.3 HTTP Caching

6.4 Proxy Caching Server Configuration

6.5 Proxy ACL

6.6 Proxy Authentication Basics

6.7 Troubleshooting

6.1: HTTP Server Configuration Basis

Configuring an HTTP server involves setting up the server to handle requests and responses over the Hypertext Transfer Protocol (HTTP)

What is an HTTP Server?

An HTTP server is software that listens for HTTP requests from clients (like web browsers) and responds with the appropriate resources (like HTML pages, images, or data). Examples include Apache HTTP Server, Nginx, Microsoft IIS, and others.

Apache is probably the most popular Linux-based Web server application in use. Once you have DNS correctly setup and your server has access to the Internet, you'll need to configure Apache to accept surfers wanting to access your Web site.

6.1: HTTP Server Configuration Basis

Managing the Apache Server

Managing Apache's httpd daemon is easy to do, but the procedure differs between Linux distributions. Here are some things to keep in mind.

1. Firstly, different Linux distributions use different daemon management systems. Each system has its own set of commands to do similar operations. The most commonly used daemon management systems are SysV and Systemd.

2. Secondly, the daemon name needs to be known. Like httpd, apache2.

Armed with this information you can know how to:

1. Start your daemons automatically on booting

2. Stop, start and restart them later on during troubleshooting or when a configuration file change needs to be applied.

6.1: HTTP Server Configuration Basis

Configuration Files

HTTP servers use configuration files to define how they operate.

For example:- Apache: httpd.conf or apache2.conf

Nginx: nginx.conf

These files control server behavior, such as:-

Port numbers (default is 80 for HTTP, 443 for HTTPS).

Document root (the directory where website files are stored).

Logging (error logs and access logs).

Virtual hosts (to host multiple websites on the same server).

6.1: HTTP Server Configuration Basis

Install HTTP Server:

On Ubuntu

1. Install Apache

 sudo apt install apache2

2. Once Installed Start the Server

sudo systemctl start apache2

3. Configure the Server

sudo nano /etc/apache2/apache2.conf

6.1: HTTP Server Configuration Basis

Configuration File’s key Section:

Document root

- Defines the directory where the server looks for files to serve.
- Eg: DocumentRoot "/var/www/html“
- Files placed in /var/www/html will be accessible via the browser.

Listening Port

- Specifies the port the server listens on.
- Eg: Listen 80
- Port 80 is the default for HTTP. For HTTPS, use port 443.

6.1: HTTP Server Configuration Basis

Virtual Hosts

- Allows hosting multiple websites on the same server.
- Eg:

```
<VirtualHost *:80>
    ServerName example.com
    DocumentRoot /var/www/example
    ErrorLog ${APACHE_LOG_DIR}/example-error.log
    CustomLog ${APACHE_LOG_DIR}/example-access.log combined
</VirtualHost>
```

ServerName: The domain name of the website.

DocumentRoot: Directory for the website files.

ErrorLog and CustomLog: Paths for logging errors and access requests.

6.1: HTTP Server Configuration Basis

Directory Permission

- Controls access to directories.
- Eg:

```
<Directory "/var/www/html">
    Options Indexes FollowSymLinks
    AllowOverride None
    Require all granted
</Directory>
```

Options: Specifies features like directory listing.

Require all granted: Allows access to all users.

Logging

- Logs help monitor server activity.
- Eg:.

```
ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined
```

6.1: HTTP Server Configuration Basis

After configuring the server, restart it to apply changes

sudo systemctl restart apache2

Test The Server

Place the HTML file in document root

echo "<h1>Hello, World!</h1>" > /var/www/html/index.html

Open a browser and navigate to http://your-server-ip. You should see the "Hello, World!" message.

6.2: Virtual Hosting

Virtual hosting is a technique used to host multiple domain names (websites) on a single server. This allows the server to share resources like memory and processing power. Virtual hosting is commonly used for web servers but is applicable to other internet services as well. It is widely used in shared web hosting, where multiple customers are hosted on a single server, reducing costs.

There are three main types of virtual hosting:

1. **Name-Based Virtual Hosting**:
    1. Multiple domain names share the same IP address.
    2. The server differentiates sites based on the "Host" HTTP header.
    3. It's efficient but has challenges with SSL/TLS, though Server Name Indication (SNI) helps partially.
    4. DNS issues can make accessing sites tricky via direct IP.

6.2: Virtual Hosting

```
<VirtualHost *:80>
    ServerName example.com
    DocumentRoot /var/www/example
</VirtualHost>

<VirtualHost *:80>
    ServerName test.com
    DocumentRoot /var/www/test
</VirtualHost>
```

Note: Both **example.com** and **test.com** resolve to the same IP address, but the server serves different content based on the domain name.

6.2: Virtual Hosting

2. IP-Based Virtual Hosting:

- Each domain has a unique IP address.
- More reliable for SSL/TLS and compatibility.
- Consumes more IPs, contributing to IPv4 exhaustion.
- The server uses the IP address in the request to determine which website to serve.

6.2: Virtual Hosting

- <VirtualHost 192.168.1.10:80>
- ServerName example.com
- DocumentRoot /var/www/example
- </VirtualHost>

- <VirtualHost 192.168.1.20:80>
- ServerName test.com
- DocumentRoot /var/www/test
- </VirtualHost>

6.2: Virtual Hosting

3. Port-Based Virtual Hosting:

- Different sites run on different ports (e.g., example.com:8080).
- Rarely used because it's inconvenient and less user-friendly.

```
<VirtualHost *:8080>
    ServerName example.com
    DocumentRoot /var/www/example
</VirtualHost>

<VirtualHost *:8081>
    ServerName test.com
    DocumentRoot /var/www/test
</VirtualHost>
```

6.3: HTTP Caching

whenever a user, hardware device or software process requests a particular piece of data, there is a good chance it will ask for it again in the near future. Thus, by storing recently-retrieved items in a cache, we can eliminate duplicated effort. This is why caching plays an important role

HTTP caching is a technique used to improve the performance and efficiency of web applications by storing copies of responses to HTTP requests. This allows subsequent requests for the same resource to be served faster, reducing latency and conserving bandwidth.

Caching is important to HTTP because Web users tend to request the same documents over and over again. For example, in writing this section on HTTP, I made reference to RFC 2616 many, many times. Each time, I loaded it from a particular Web server. Since the document never changes, it would be more efficient to just load it from a local cache rather than having to retrieve it from the distant Web server each time.

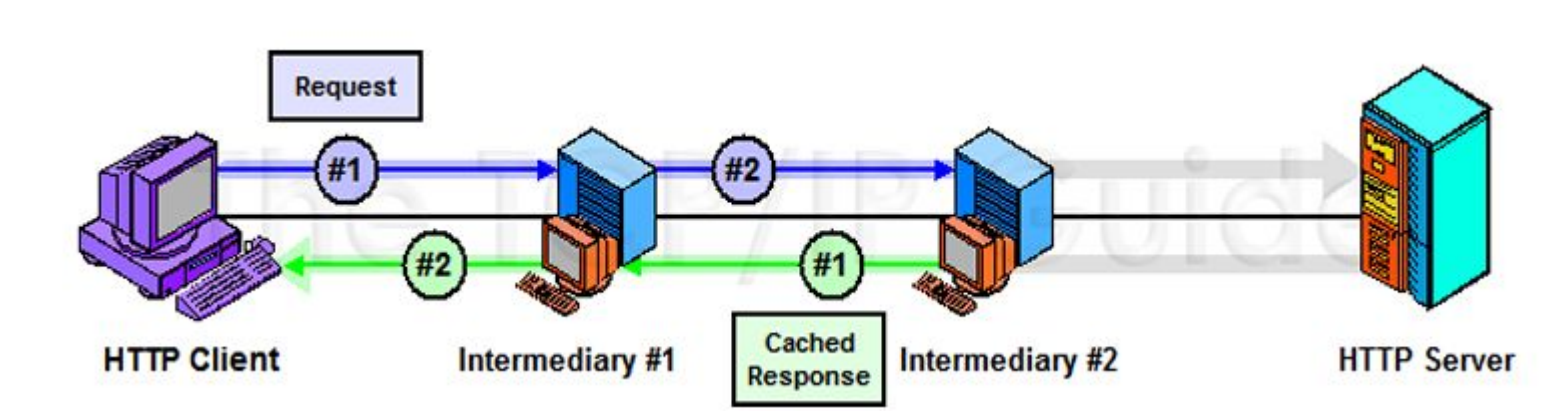
However, caching is even more essential to HTTP than to most other protocols or technologies where it used. The reason is that Web documents tend to be structured so that a request for one resource leads to a request for many others. Even if I load a number of different documents, they may each refer to common elements that do not change between user requests. Thus, caching can be of benefit in HTTP even if a user never asks for the same document twice, or if a single document changes over time so that caching the document itself would be of little value.

6.3: HTTP Caching

For example, suppose that each morning I load up CNN's Web site to see what is going on in the world. Obviously, the headlines will be different every day, so caching of the main CCN.com Web home page won't be of much value. However, many of the graphical elements on the page (CNN's logo, dividing bars, perhaps a "breaking news" graphic) will be the same, every day, and these can be cached. Another example would be a set of discussion forums on a Web site. As I load up different topics to read, each one is different, but they have common elements (such as icons and other images) that would be wasteful to have to retrieve over and over again.

Caching in HTTP yields two main benefits. The first is reduced bandwidth use, by eliminating unneeded transfers of requests and responses. The second, equally important, is faster response time for the user loading a resource. Consider that on many Web pages today, the image files are much larger than the HTML page that references them. Caching these graphics will allow the entire page to load far more quickly.

6.3: HTTP Caching



This diagram illustrates the impact of caching on the request/response chain of Figure 316. In this example, intermediary #2 is able to satisfy the client's request from its cache. This "short-circuits" the communication chain after two transfers, which means the client gets its resource more quickly, and the HTTP server is spared the need to process the client's request.

6.3: HTTP Caching

How HTTP Caching Works

- **Caching Responses:**
  - When a client (like a browser) makes an HTTP request, the server can store the response in a cache.
  - If the same resource is requested again, the cached response is reused instead of fetching it from the server.
- **Freshness and Validation:**
  - Cached responses are considered "fresh" if they are still valid based on caching rules.
  - If a cached response is stale, the client may validate it with the server to check if it can still be used.
- **Cache-Control Headers:**
  - HTTP headers like Cache-Control and Expires are used to define caching policies.
  - For example, Cache-Control: max-age=3600 specifies that the response can be cached for 3600 seconds.

6.3: HTTP Caching

How HTTP Caching Works

- **Caching Responses:**
  - When a client (like a browser) makes an HTTP request, the server can store the response in a cache.
  - If the same resource is requested again, the cached response is reused instead of fetching it from the server.
- **Freshness and Validation:**
  - Cached responses are considered "fresh" if they are still valid based on caching rules.
  - If a cached response is stale, the client may validate it with the server to check if it can still be used.
- **Cache-Control Headers:**
  - HTTP headers like Cache-Control and Expires are used to define caching policies.
  - For example, Cache-Control: max-age=3600 specifies that the response can be cached for 3600 seconds.

6.3: HTTP Caching

Types of Caches

- **Private Cache:**
  - Stored on the client side (e.g., browser cache).
  - Used only by the client that made the request.
- **Shared Cache:**
  - Located between the client and server (e.g., proxy servers or CDNs).
  - Can serve multiple clients with the same cached response.

Benefits of HTTP Caching

- **Reduced Latency:** Cached responses are delivered faster than fetching from the server.
- **Lower Bandwidth Usage:** Reduces the amount of data transferred over the network.
- **Decreased Server Load:** Fewer requests reach the origin server, improving scalability.

6.4: Proxy Caching Server Configuration

Configuring a proxy caching server involves setting up a system that stores frequently accessed web content to improve performance and reduce bandwidth usage.

Here's a general guide to configuring a proxy caching server, using **Squid** as an example:

squid is a high-performance proxy caching server for web clients, supporting FTP, gopher, ICAP, ICP, HTCP and HTTP data objects. Unlike traditional caching software, squid handles all requests in a single, non-blocking process. squid keeps meta data and especially hot objects cached in RAM, caches DNS lookups, supports non-blocking DNS lookups, and implements negative caching of failed requests. squid supports SSL, extensive access controls, and full request logging. By using the lightweight Internet Cache Protocols ICP, HTCP or CARP, squid caches can be arranged in a hierarchy or mesh for additional bandwidth savings.

squid consists of a main server program squid, some optional programs for custom processing and authentication, and some management and client tools. When squid starts up, it spawns a configurable number of helper processes, each of which can perform parallel lookups. This reduces the amount of time the cache waits for results.

6.4: Proxy Caching Server Configuration

Steps to Configure a Proxy Caching Server (Squid)

Install Squid:

On Linux:
- **sudo yum install squid** # For CentOS/RHEL

- **sudo apt-get install squid** # For Ubuntu/Debian

Start the Squid service:
- **sudo systemctl start squid**
- **sudo systemctl enable squid**

Edit the Configuration File:

The main configuration file is located at /etc/squid/squid.conf.

Open it in a text editor:
- **sudo nano /etc/squid/squid.conf**

6.4: Proxy Caching Server Configuration

Setup Access Control:

- Define access control lists (ACLs) to specify which IP ranges can use the proxy.
- Remove or modify default ACLs to match your environment.
- **acl localnet src 192.168.1.0/24**
  http_access allow localnet

Configure Cache Settings:

- Define the cache directory, size, and structure:
- **cache_dir ufs /var/spool/squid 10000 16 256**

ufs: Cache type.

/var/spool/squid: Cache directory.

10000: Cache size in MB.

16 and 256: Directory structure.

6.4: Proxy Caching Server Configuration

Set Refresh pattern:

- Define how often Squid checks for updates to cached content:

refresh_pattern ^ftp: 1440 20% 10080

refresh_pattern ^gopher: 1440 0% 1440

refresh_pattern . 0 20% 4320

Restart Squid:

- Apply the changes by restarting the Squid service:
- **sudo systemctl restart squid**

6.5: Proxy ACL

A Proxy ACL (Access Control List) is a set of rules that define which users, IP addresses, or domains can access the proxy server. ACLs help enforce security policies by restricting access to certain websites, protocols, or services.

Access Control Lists (ACLs) in proxy servers like Squid are used to control access to network resources based on specific conditions. These rules define who can access what, when, and how.

ACLs helps to:

- Allow or deny access based on IP addresses or subnets.
- Restrict access to certain websites or domains.
- Control access based on request methods like GET or POST.
- Allow specific time-based rules (e.g., internet access only during office hours).

6.5: Proxy ACL

Types of ACL in Squid:

ACL Type	Description	Example
drc	Source IP address or range	acl office src 192.168.1.0/24
dst	Destination IP or domain	acl blocked dst 203.0.113.0/24
dstdomain	Destination domain name	acl badsite dstdomain .facebook.com
url_regrex	URL matching using regres	acl streaming url_regrex –I youtube
time	Match request based on time	acl workhours time MTWHF 09:00-17:00
method	HTTP request method	acl POST method POST
port	Destination port	acl ssl_pport port 443

6.5: Proxy ACL

Example: Allow Access to internal user & block facebook

Define ACLs

acl office_network src 192.168.1.0/24

acl block_facebook dstdomain .facebook.com

Access control rules

http_access deny block_facebook

http_access allow office_network

http_access deny all

TO apply the config restart the Squid

sudo systemctl restart squid

6.6: Proxy Authentication Basic

Proxy authentication allows only authorized users to access internet services through the proxy. It ensures controlled access, logging, and accountability.

Benefits:

- Prevents unauthorized access.
- Helps track user activity.
- Enforces usage policies.

Common Authentication Methods

Method	Description
Basic Authentication	Uses a local username & password (e.g NCSA)
LDAP/Active Directory	Authenticates against a centralized directory service
NTLM/Kerberos	Integrated Windows-based authentication for domain environments

6.6: Proxy Authentication Basic

Basic Authentication Using NCSA in Squid:

Step-by-Step Configuration:

Install required package:

sudo apt install apache2-utils

Create password file and user:

sudo htpasswd -c /etc/squid/passwd user1

You will be prompted to enter a password.

6.6: Proxy Authentication Basic

Edit squid.conf:

Add the following lines:

Authentication settings

auth_param basic program /usr/lib/squid/basic_ncsa_auth /etc/squid/passwd

auth_param basic realm Squid Proxy Authentication

Define authenticated users

acl authenticated proxy_auth REQUIRED

Access control

http_access allow authenticated

http_access deny all

Restart Squid:

sudo systemctl restart squid

Client Behavior:

When users try to access the web, they will be prompted for a username and password.

Assignments

1. What is HTTP? Explain the steps to configure HTTP-APACHE server for Linux virtual hosting of the website www.tu.edu.np
2. Explain the process of proxy caching server configuration.
3. What is proxy? How does HTTP proxy work?
4. Explain about Virtual Hosting Methods.

Unit:3 Network Configuration Basics

- Network Interface Configuration
- Diagnosing Network Startup Issues
- Linux & Windows Firewall Configuration
- Network Troubleshooting Commands
- Introduction to Network Programming with Mininet
- SDN Controller & Dataplance Communication
- Routing Configuration in SDN
- Open Source Network Monitoring - Nagios

Unit:7 FTP, File & Print Server

7.1 General Samba Configuration

7.2 CUPS Configuration Basics

7.3 FTP Principles

7.4 Anonymous FTP Server

7.5 Troubleshooting

General Samba Configuration

Samba is an open-source software implementation of the Server Message Block/Common Internet File System (SMB/CIFS) protocol. It allows users on non-Windows platforms to access network resources shared by Windows computers and servers. Without Samba, Windows and non-Windows computers would be isolated from one another, even while connected to the same LAN. Samba is available for free under the GNU General Public License, and most Unix and Linux distributions include it to support cross-platform file sharing.

Samba implements the SMB protocol for both sides of the client-server model. It allows non-Windows clients to access network shares created on Windows servers while also allowing non-Windows servers to create SMB network shares. Unix and Linux users can use Samba to mount network shares as part of the computer's file structure, or they may also use software utilities to access network shares like they would access an FTP server.

Non-Windows computers can also use Samba to access other shared network resources, like network printers. Samba also supports Active Directory, which allows Unix and Linux systems to participate in Microsoft Active Directory domains.

General Samba Configuration

It allows interoperability between Unix/Linux and Windows systems by providing services like:

- File sharing
- Printer sharing
- Authentication and authorizationIntegration with Windows domains (including Active Directory)

Key Services:

smbd: Manages SMB file-sharing protocol, handles authentication, permissions.

nmbd: Handles NetBIOS name resolution and browsing.

winbindd: Used for integration with Windows AD domains.

General Samba Configuration

Installation:

On Debian/Ubuntu: **sudo apt install samba**

On Centos/RHEL: **sudo dnf install samba samba-common samba-client**

Basic Configuration:

The main configuration file: **/etc/samba/smb.conf**

Example:

Create a directory to share:
- sudo mkdir -p /srv/samba/shareddirectory

Change Ownership of the folder (owner:group)

- sudo chown nobody:nogroup /srv/samba/shareddirectory

Change Permission of the directory:

- sudo chmod 0775 /srv/samba/shareddirectory

General Samba Configuration

Edit Samba Configuration:

- sudo nano /etc/samba/smb.conf

Add the following on the buttom:

[Shareddirectory]

path = /srv/samba/shareddirectory

browseable = yes

read only = no

guest ok = yes

General Samba Configuration

Restart Samba Service:

- sudo systemctl restart smbd
- sudo systemctl enable smbd #make it automatically start whenever the system reboots

Allow through Firewall:

- sudo ufw allow 'Samba’
- sudo firewall-cmd --reload

Access the shared Directory:

- Open Run (Win + R), type:\\<your_server_IP>\shareddirectory

General Samba Configuration

Validate your Samba config file:

- testparm

Shows active Samba connections:

- smbstatus

List available Shares:

- smbclient -L //localhost

General logs:

- /var/log/samba/

General Samba Configuration

SWAT (Samba Web Administration Tool):

- Web interface to configure SambaURL: http://127.0.0.1:901
- Edits smb.conf but removes comments
- Recommendation: Back up config file:
  - cp /etc/samba/smb.conf /etc/samba/smb.conf.original
- SWAT login not encrypted
- Access control via **/etc/xinetd.d/swat**
- Ensure TCP port 901 is open

General Samba Configuration

Example SWAT xinetd Configuration

```
service swat
{
    port        = 901
    socket_type = stream
    protocol    = tcp
    wait        = no
    user        = root
    server      = /usr/sbin/swat
    log_on_failure += USERID
    disable     = no
    only_from   = localhost
}
```

NFS (Network File System)

NFS stands for **Network File System**, a protocol developed by Sun Microsystems in the 1980s. It allows users to access files over a network as if they were on their local machine.

Key Features of NFS

- **Remote File Access:** NFS enables users to mount remote directories on their local systems, making files accessible across different machines.
- **Transparency:** Files accessed via NFS appear as if they are stored locally, providing seamless integration.
- **Platform Independence:** NFS is designed to work across various operating systems, including UNIX, Linux, and Windows.
- **Client-Server Architecture:** The server hosts the files, while clients request access to them. Multiple clients can access the same files simultaneously.

NFS (Network File System)

How NFS Works ?

1. **Exporting Directories:** The server specifies which directories are shared (exported) and who can access them.
2. **Mounting:** Clients mount the exported directories to their local file system using the mount command.
3. **Communication:** NFS uses Remote Procedure Calls (RPCs) to communicate between the client and server.

NFS (Network File System)

Step by Step NFS Setup:

On the Server:

1. Install NFS Server:

 sudo apt install nfs-kernel-server

2. Create Shared Directory:

sudo mkdir -p /srv/nfs/shareddirectory

sudo chown nobody:nogroup /srv/nfs/shareddirectory

 sudo chmod 777 /srv/nfs/shareddirectory

3. Configure Exports File:

 /srv/nfs/shareddirectory 192.168.1.0/24(rw,sync,no_subtree_check)

NFS (Network File System)

4. Export the shared directory:

- sudo exportfs -a # Applies changes
- sudo systemctl restart nfs-kernel-server

5. Check Exported Shares:

- sudo exportfs –v

NFS (Network File System)

On the NFS Client:

Install NFS client:

- sudo apt install nfs-common

Create Mount Point:

- sudo mkdir -p /mnt/nfs/shareddirectory

Mount the NFS Share:

- sudo mount 192.168.1.10:/srv/nfs/shared /mnt/nfs/shared (#your server IP)

Permanent Mount: (/etc/fstab)

- 192.168.1.10:/srv/nfs/shared /mnt/nfs/shared nfs defaults 0 0

7.2 CUPS Configuration Basics

One of the latest Linux printing solutions is the Common UNIX Printing System (CUPS), it supports the Internet Printing Protocol (IPP) and provides a complete platform independent printing solution for most networking environments. The CUPS server is able to be administered remotely using a web browser, which makes it ideal for a ‘headless server’.

One of the major CUPS advantages is that it can be completely controlled remotely using a standard web browser, so really all we need to do is get it configured and then access it remotely. However this guide will provide all the steps necessary to configure from the command line (print drivers need to be updated manually).

To access the resources remotely via a web browser, we need to specify the access controls (deny/allow) which will be applied to each resource.

The “/” (root) resource may be provided to all users without any authentication so they may view which printers are available and the status of the queues.

7.2 CUPS Configuration Basics

Installation:

- sudo apt install cups

Start CUPS Service:

- sudo systemctl start cups

Web Interface:

Accessible at http://localhost:631

Admin functions (add/remove printers) require authentication.

7.2 CUPS Configuration Basics

Config File:

- /etc/cups/cupsd.conf

Important Configuration:

```
Listen localhost:631        # Specifies the port and address to listen on
Browsing On                 # Enables printer discovery
DefaultAuthType Basic       # Authentication type
<Location />
Order allow,deny
Allow @LOCAL                # Allows access from local network
</Location>
```

7.2 CUPS Configuration Basics

Add Printer Via CLI:

- sudo lpadmin -p HPPrinter -E -v usb://HP/DeskJet -m everywhere

7.3 FTP Principles

File Transfer Protocol (FTP) is a standard network protocol used to transfer files between a client and a server on a computer network. It operates over a TCP/IP network (like the internet) and is designed to facilitate file sharing, remote file access, and data exchange.

FTP uses a client-server approach. The user sends requests from his computer through a ftp client program to a remote computer which receives it through a ftp server program. Thus the communication is asymmetric. Assuming one is on line to machine A and that one wants to exchange files with a remote computer B, it means that:

- Computer A must run a client ftp
- Computer B must run a server ftp

However with the above programs running, a user on line to computer B, will not be able to exchange files with computer A since there is no client running on machine B and no server on machine A. To allow symmetric communications one needs both client and server programs running on the same machine.

7.3 FTP Principles

How does FTP work ?

The client will initiate a connection with the server, usually via port 21 - this connection does not pass any data, only control messages, hence it is called the Control Channel. This allows the connection to be established, and the login to take place (either anonymous or using a special username and password). The transport of data (in either direction, and this includes file/directory listings) requires the setting up of a separate data channel (initiated by the FTP client), which may be of two possible types, Active or Passive modes, and these are both usually supported by FTP clients.

7.3 FTP Principles

Active Data Connection

Via the control channel, the client sends a port number to the server (using the PORT command), then waits for the server to connect to it on this port, from port 20 on the server (the use of port 20 by the server is a convention, like the use of port 21 for the Control Channel). The data (in either direction) is transported, and the data channel closed. If more data is to be transported, the client will send another port number to the server, the server will connect through its port 20 with the specified port on the client, and that data transported. A different port is used because there is a delay between the instruction to close the original port and its actually closing, which mean it cannot be re-used immediately.

Passive Data Connection

Via the control channel, the client will issue an instruction to the server (using the PASV command) that the server should open a port to which the client can open a connection. The server will respond with a port number and await the client's initiating a connection to that port. Once established, this will be used for the transport of data, then closed again, just as with the Active method above.

7.3 FTP Principles

Ports:

Port 21: Control connection (commands)

Port 20: Data transfer (active mode)

Modes:

Active Mode:

- Client opens port and listens; server connects back for data.
- May be blocked by firewalls.

Passive Mode:

- Server opens a random port and tells the client to connect there.
- More firewall-friendly.

7.3 FTP Principles

Secure Enhancement:

Traditional FTP is unencrypted, transmitting files, usernames, and passwords in plain text. To address this, modern secure alternatives are used:

FTPS (FTP Secure):

- Leverages SSL/TLS (Secure Sockets Layer / Transport Layer Security) for encryption.
- Supports both implicit and explicit encryption modes.
  - Implicit Mode: Encryption starts immediately upon connection.
  - Explicit Mode: Encryption begins after a specific command (AUTH TLS) is issued.

SFTP (SSH File Transfer Protocol):

- Works over the Secure Shell (SSH) protocol to provide a fully encrypted channel.
- Operates as a subsystem of SSH and differs fundamentally from FTP despite its name.

7.4 Anonymous FTP

Anonymous FTP is a method of accessing files on an FTP server without needing a personal username and password. It is commonly used to allow public access to a repository of files while maintaining some level of security and administrative control.

Key Features of Anonymous FTP

Login Details:

- Instead of providing specific credentials, users log in with the username "anonymous".
- For the password, users often provide their email address, but it's not always strictly required—some servers accept any input or leave it blank.

Purpose:

- Anonymous FTP servers are typically set up to share public files like software distributions, documentation, or other non-sensitive resources.
- It facilitates ease of access without the need for managing individual accounts.

7.4 Anonymous FTP

Restricted Access:

While users can download files, upload permissions are usually disabled to prevent misuse (e.g., uploading malicious content).

Some servers allow uploads to specific directories with limited access for security purposes.

Common Use Cases:

- Distributing open-source software (e.g., Linux distributions).
- Sharing large data sets or research materials.
- Making public documentation available.

7.4 Anonymous FTP

Security Concerns:

Anonymous FTP can pose risks, such as unauthorized access or abuse of resources, so administrators often impose strict controls:-

- Limiting the directories accessible via anonymous login.
- Monitoring server activity for unusual patterns.
- Using firewalls and encryption for added protection.

Connection Process:

- Users connect to the server using an FTP client or web browser.
- Upon connection, they select anonymous as the username.
- After logging in, they can browse, download, or interact with permitted files.

Assignment

1. Explain Samba. What are the features of SAMBA-SWAT?
2. How can you configure NFS? Explain.
3. Explain the process of CUPS configuration process.
4. What is Anonymous FTP? Explain FTP Principle

Unit:8 Mail Server Basics

- **SMTP, POP & IMAP Principles**
- **SMTP Relaying Principles**
- **Mail Domain Administration**
- **Basic Mail Server Configuration**
- **SPAM Control & Filtering**
- **Troubleshooting**

Unit:8 Mail Server Basics

SMTP, POP & IMAP Principles

Simple Mail Transfer Protocol (SMTP) is an Internet standard for electronic mail (e-mail) transmission across Internet Protocol (IP) networks. SMTP was first defined by RFC 821 (1982, eventually declared STD 10), and last updated by RFC 5321 (2008) which includes the extended SMTP (ESMTP) additions, and is the protocol in widespread use today. SMTP uses TCP port 25. The protocol for new submissions mail submission agent (MSA) is effectively the same as SMTP, but it uses port 587 instead. To enhance security, SMTP connections can be encrypted using SSL/TLS. This is commonly called SMTPS, though SMTPS itself is not a separate protocol but just SMTP with encryption.

Unit:8 Mail Server Basics

SMTP, POP & IMAP Principles

Email Servers & SMTP

- Email servers (like Gmail, Yahoo Mail, or corporate mail servers) **use SMTP** to send and receive emails from other mail servers.
- When you send an email, your server forwards it to the recipient's mail server using SMTP.

Email Clients & Sending Emails

- Applications like Outlook, Thunderbird, or mobile email apps **only use SMTP for sending emails** to their mail server.
- The mail server then **relays** (forwards) the email to the recipient's server using SMTP.

Unit:8 Mail Server Basics

SMTP, POP & IMAP Principles

Email Clients & Receiving Emails

- Email clients **do not use SMTP to receive emails**. Instead, they retrieve emails from the mail server using one of these protocols:
  - **IMAP (Internet Message Access Protocol)**: Keeps emails on the server, allowing access from multiple devices.
  - **POP3 (Post Office Protocol v3)**: Downloads emails and removes them from the server (works best for single-device access).
  - **Proprietary Systems**: Some companies use specialized email systems, like **Microsoft Exchange or Lotus Notes/Domino**, which have their own ways of managing emails.

Unit:8 Mail Server Basics

Mail processing model

Email is submitted by a mail client (MUA, mail user agent) to a mail server (MSA, mail submission agent) using SMTP on TCP port 587. Most mailbox providers still allow submission on traditional port 25. From there, the MSA delivers the mail to its mail transfer agent (MTA, mail transfer agent). Often, these two agents are just different instances of the same software launched with different options on the same machine. Local processing can be done either on a single machine, or split among various appliances; in the former case, involved processes can share files; in the latter case, SMTP is used to transfer the message internally, with each host configured to use the next appliance as a smart host. Each process is an MTA in its own right; that is, an SMTP server.

Unit:8 Mail Server Basics

Mail processing model

The boundary MTA has to locate the target host. It uses the Domain name system (DNS) to look up the mail exchanger record (MX record) for the recipient's domain (the part of the address on the right of @). The returned MX record contains the name of the target host. The MTA next connects to the exchange server as an SMTP client.

Once the MX target accepts the incoming message, it hands it to a mail delivery agent (MDA) for local mail delivery. An MDA is able to save messages in the relevant mailbox format. Again, mail reception can be done using many computers or just one —the picture displays two nearby boxes in either case. An MDA may deliver messages directly to storage, or forward them over a network using SMTP, or any other means, including the Local Mail Transfer Protocol (LMTP), a derivative of SMTP designed for this purpose.

Unit:8 Mail Server Basics

Mail processing model

Once delivered to the local mail server, the mail is stored for batch retrieval by authenticated mail clients (MUAs). Mail is retrieved by end-user applications, called email clients, using Internet Message Access Protocol (IMAP), a protocol that both facilitates access to mail and manages stored mail, or the Post Office Protocol (POP) which typically uses the traditional mbox mail file format or a proprietary system such as Microsoft Exchange/Outlook or Lotus Notes/Domino. Webmail clients may use either method, but the retrieval protocol is often not a formal standard.

SMTP defines message transport, not the message content. Thus, it defines the mail envelope and its parameters, such as the envelope sender, but not the header or the body of the message itself. STD 10 and RFC 5321 define SMTP (the envelope), while STD 11 and RFC 5322 define the message (header and body), formally referred to as the Internet Message Format.

Unit:8 Mail Server Basics

IMAP & POP3

IMAP and POP3 POP3 and IMAP are two different protocols used to access e-mail. POP3 and IMAP function very differently from each other and each has their own advantages. POP3 is useful for checking your e-mail from one computer at a single location. IMAP is the better option when you need to check your e-mail from multiple computers at different locations such as at work, home, or on the road.

POP3

POP3 works by accessing the inbox on the mail server and then downloading the new messages to your computer. With this type of account you do not have to stay logged on to the Internet. You can log on when you want to receive and send new messages. Once your new messages have been downloaded to your computer you can log off to read them. This option is good when you connect to the internet through a dial up service and are charged for your connection time.

Unit:8 Mail Server Basics

IMAP

IMAP lets you view the headers of the new messages on the server and then retrieves the message you want to read when you click on them. When using IMAP, the mail is stored on the mail server. Unless you copy a message to a "Local Folder" the messages are never copied to your PC. Using this protocol, all your mail stays on the server in multiple folders, some of which you have created. This enables you to connect to any computer and see all your mail and mail folders. In general, IMAP is great if you have a dedicated connection to the internet or you like to check your mail from various locations. It is important to point out that POP3 can be set to leave your e-mail on the server instead of downloading to your computer. This feature enables you to check your e-mail from multiple locations just like IMAP. So why choose IMAP rather than POP3 with the leave mail on server setting? With the POP3 leave mail on server setting only your e-mail messages are on the server, but with IMAP your e-mail folders are also on the server.

Unit:8 Mail Server Basics

IMAP

IMAP lets you view the headers of the new messages on the server and then retrieves the message you want to read when you click on them. When using IMAP, the mail is stored on the mail server. Unless you copy a message to a "Local Folder" the messages are never copied to your PC. Using this protocol, all your mail stays on the server in multiple folders, some of which you have created. This enables you to connect to any computer and see all your mail and mail folders. In general, IMAP is great if you have a dedicated connection to the internet or you like to check your mail from various locations. It is important to point out that POP3 can be set to leave your e-mail on the server instead of downloading to your computer. This feature enables you to check your e-mail from multiple locations just like IMAP. So why choose IMAP rather than POP3 with the leave mail on server setting? With the POP3 leave mail on server setting only your e-mail messages are on the server, but with IMAP your e-mail folders are also on the server.

Unit:8 Mail Server Basics

When should I use POP3?

- When you only check e-mail from one computer at a single location. -You want to remove your e-mail from the mail server.
- You connect to the internet through dial up and are charged for connection time. When should I use IMAP?
- When you need to check e-mail from multiple computers at multiple locations.
- You use Web mail such as Gmail, Yahoo or Hotmail.
- You need to preserve the different e-mail folders you have created.

Unit:8 Mail Server Basics

When should I use POP3?

- When you only check e-mail from one computer at a single location. -You want to remove your e-mail from the mail server.
- You connect to the internet through dial up and are charged for connection time.

When should I use IMAP?

- When you need to check e-mail from multiple computers at multiple locations.
- You use Web mail such as Gmail, Yahoo or Hotmail.
- You need to preserve the different e-mail folders you have created.

Unit:8 Mail Server Basics

Comparison of POP and IMAP

	POP	IMAP
What does it stand for?	Post Office Protocol	Internet Message Access Protocol
Which protocol would suit me best?	If you access mail using only one computer e.g. your office PC or a laptop.	If you want to access your mail from multiple computers or locations.
Which mail programs can I use?	All mail programs or clients have POP capability	Most mail programs have IMAP capability and you will also be able to access your mail via a web page using any web browser.

Unit:8 Mail Server Basics

Comparison of POP and IMAP

	POP	IMAP
Check for incoming mail	By default, incoming messages are transferred to your local machine when you check your incoming mail. Only new messages are available if you connect to the server using a PC other than your normal one. You are connected to the server only for the transfer of messages.	By default, incoming messages stay on the server when you check your mail - only headers are transferred with full messages only downloaded when selected for reading. All your messages are always available no matter where or how you connect to the server. You remain connected to the server whilst you deal with mail but some clients allow for off-line working.

Unit:8 Mail Server Basics

Comparison of POP and IMAP

	POP	IMAP
Read and respond to mail	Reading and responding to messages is done on your local machine.	You can read and respond to messages directly on the server but you can also read and respond to messages on your local machine, after downloading for offline working (depending on client). When you reconnect, your mailboxes are resynchronised to reflect the changes you have made.

Unit:8 Mail Server Basics

Comparison of POP and IMAP

	POP	IMAP
Create mailboxes for storing messages	Creating mailboxes can be done only on your local machine.	You can create mailboxes directly on the server. By default, an Inbox is automatically created on the server when you begin using IMAP. The Inbox functions as the master mailbox (or folder) as well as the mailbox for incoming messages. All other mailboxes, including a trash box, need to be created within the Inbox.

Unit:8 Mail Server Basics

Comparison of POP and IMAP

	POP	IMAP
Move messages in and out of mailboxes	You can move messages in and out of mailboxes only on your local machine..	You can move messages in and out of mailboxes on the server and on your local machine.
Transfer messages from local machine to server and vice versa	You cannot transfer any messages from your local machine to the server. Messages are automatically transferred from the server to your local machine when you check your incoming mail.	You can transfer individual messages from mailboxes on your local machine into mailboxes on the server and vice versa.
Delete selected messages on the server	When using some clients (e.g. Eudora), if you specified to leave messages on the server, you can delete individual messages left there.	You can delete individual messages and groups of messages directly on the server as well as on your local machine.

Unit:8 Mail Server Basics

SMTP Relaying Principles

What is SMTP Relaying?

SMTP (Simple Mail Transfer Protocol) **relaying** is the process of transferring an email from one mail server to another until it reaches the recipient's mail server. It acts as a bridge between different email systems, ensuring that emails are delivered correctly across the Internet.

- For example, if you send an email from **you@example.com** to **friend@gmail.com**, your email server does not send the email directly to your friend's inbox. Instead, it **relays** it through different SMTP servers until it reaches Gmail’s mail server, which then delivers it to your friend.

Unit:8 Mail Server Basics

SMTP Relaying Principles

How SMTP Relaying Works

SMTP relaying follows a series of steps to ensure the email is properly transmitted:

Step 1: Sender Composes Email

- You use an email client (Outlook, Gmail, Thunderbird, etc.) to write an email.
- When you press **"Send"**, your email client sends the message to your **outgoing SMTP server**.
- SMTP typically uses **port 587** (for client-to-server submission).

Step 2: SMTP Server Processes the Email

- Your email client connects to the SMTP **Mail Submission Agent (MSA)**.
- The SMTP server authenticates you (if required) and checks if the email is valid.

Unit:8 Mail Server Basics

SMTP Relaying Principles

Step 3: SMTP Relaying Begins

- If the recipient's email is on the **same** domain (e.g., both sender and receiver are @example.com), the email is delivered directly to the mailbox.
- If the recipient is on a **different domain** (e.g., @gmail.com), the SMTP server looks up the **MX (Mail Exchange) records** of the recipient's domain using DNS (Domain Name System).

Step 4: Forwarding to the Recipient’s Mail Server

- The sending SMTP server contacts the recipient’s SMTP server (found via the MX record).
- The email is transferred using SMTP on **port 25** (server-to-server communication).

Step 5: Recipient’s Mail Server Accepts the Email

- The recipient's mail server stores the email in the recipient’s inbox.
- The recipient retrieves the email using **IMAP, POP3, or Exchange**.

Unit:8 Mail Server Basics

Example of SMTP Relaying in Action

Let's say Alice (alice@example.com) wants to send an email to Bob (bob@gmail.com):

1. **Alice** writes an email to **Bob** using Outlook.
2. Her email client connects to **Example.com's SMTP server** (mail.example.com) via port 587.
3. Example.com's SMTP server looks up **gmail.com's MX record**.
4. It finds that Gmail's mail server (e.g., smtp.gmail.com) is responsible for handling email.
5. The email is relayed from **mail.example.com → smtp.gmail.com** over **port 25**.
6. Gmail's server accepts the email and places it in **Bob's inbox**.
7. **Bob** retrieves the email using IMAP or POP3 on his email client.

Unit:8 Mail Server Basics

Open vs. Authenticated SMTP Relaying

Open SMTP Relaying (Not Secure)

- Before 2000, many SMTP servers allowed **"open relays,"** meaning anyone could send emails through them.
- Spammers exploited this to send bulk spam emails.
- Today, most SMTP servers block open relays to prevent spam.

Authenticated SMTP Relaying (Secure)

- Requires **username and password** authentication to relay emails.
- Prevents spammers from abusing the mail server.
- Uses encryption like **TLS/SSL** to secure emails.

Unit:8 Mail Server Basics

ISSUE	CAUSE	SOLUTION
Emails marked as spam	Poor sender reputation, missing SPF/DKIM/DMARC	Set up SPF, DKIM, and DMARC records
SMTP relay blocked	Open relay detected, blacklisted IP	Use authentication & secure relay
Emails not delivered	Incorrect MX records	Verify domain’s MX records
Slow email delivery	High server load	Use a dedicated SMTP relay service

Unit:8 Mail Server Basics

Mail Domain Administration

Mail Domain Administration involves managing email services for a domain, ensuring email delivery, security, and compliance. It includes setting up mail servers, configuring DNS records, enforcing security policies, and managing user accounts.

Key Responsibilities of Mail Administrator

Task	Purpose
Setting up Mail Servers	Configuring SMTP, IMAP, and POP3 servers
Managing DNS Records	Ensuring proper MX, SPF, DKIM, and DMARC configurations
User & Mailbox Management	Creating, deleting, and managing user email accounts
Email Security & Anti-Spam	Preventing spoofing, phishing, and spam attacks
Monitoring & Troubleshooting	Ensuring email uptime, monitoring logs, and fixing issues

Unit:8 Mail Server Basics

Setting Up Mail Server

Component	Role	Examples
SMTP Server	Sends emails to other mail servers	Postfix, Exim, Sendmail
IMAP/POP3 Server	Receives & stores emails for users	Dovecot, Courier
Webmail	Provides a browser-based email interface	Roundcube, RainLoop
Mail Relay	Routes emails through external mail servers	Gmail SMTP Relay, SendGrid

Unit:8 Mail Server Basics

Setting Up Mail Server

Common Mail Server Setups:

- **Linux Mail Servers**: Postfix + Dovecot (for self-hosted mail).
- **Enterprise Solutions**: Microsoft Exchange, Zimbra, Google Workspace.
- **Cloud-Based Mail Services**: Gmail, Outlook 365, Zoho Mail.

Unit:8 Mail Server Basics

Setting Up Mail Server

Common Mail Server Setups:

- **Linux Mail Servers**: Postfix + Dovecot (for self-hosted mail).
- **Enterprise Solutions**: Microsoft Exchange, Zimbra, Google Workspace.
- **Cloud-Based Mail Services**: Gmail, Outlook 365, Zoho Mail.

Unit:8 Mail Server Basics

DNS Records For Mail

DNS Record	Purpose
MX (Mail Exchange)	Specifies the mail server for a domain
SPF (Sender Policy Framework)	Defines authorized mail servers
DKIM (DomainKeys Identified Mail)	Uses cryptographic signatures to verify email authenticity
DMARC (Domain-based Message Authentication, Reporting & Conformance)	Protects against email spoofing & phishing
PTR (Reverse DNS)	Maps IP address to domain for email validation

Unit:8 Mail Server Basics

User & Mailbox Management:

Administrators need to **create, manage, and secure user mailboxes**:

Creating Email Accounts

Quota Management: Limit storage per user to prevent excessive usage.

Aliases & Forwarding: Set up email aliases (e.g., support@example.com → multiple users). Forward emails to external accounts if needed.

Group Emails & Distribution Lists: Configure distribution lists for departments (e.g., sales@example.com).

Mailbox Migration: Move emails from old servers to new ones using IMAP sync or Microsoft Exchange migration tools.

Unit:8 Mail Server Basics

Email Security & Anti-Spam Measures:

Key Security Risks:

- Spoofing (Fake sender email).
- Phishing (Tricking users into revealing credentials).
- Spam & Malware (Junk emails and viruses).

Unit:8 Mail Server Basics

How to Protect Mail Domain:

Security Measures	Protection Against	Implementation
SPF	Prevents unauthorized servers from sending emails	Configure TXT record
DKIM	Ensures email integrity with a cryptographic signature	Set up DKIM signing
DMARC	Prevents spoofing by enforcing SPF/DKIM rules	Policy: reject or quarantine
Greylisting	Blocks unknown senders temporarily	Enabled in Postfix or Exchange
Rate Limiting	Prevents email abuse by limiting sending speed	Configure SMTP server
RBL (Real-Time Blackhole Lists)	Blocks known spam sources	Use Spamhaus, Barracuda
Attachment Filtering	Blocks malware files in emails	Set rules for .exe, .bat

Use an email security gateway (e.g., Proofpoint, Barracuda, Mimecast) for additional protection.

Unit:8 Mail Server Basics

Monitoring Email Issues:

- **Mail Logs** (/var/log/mail.log, Exchange Event Viewer).
- **Mail Queue** Checks (postqueue -p, exim -bp).
- **MX & DNS Checks** (nslookup -type=MX example.com).
- **Email Headers Analysis** (Checking SPF, DKIM, DMARC pass/fail).

Unit:8 Mail Server Basics

Troubleshooting:

Common Email Issues & Fixes

Issues	Possible Cause	Solution
Emails going to spam	Missing SPF/DKIM/DMARC	Set up proper DNS records
Cannot send emails	SMTP server misconfigured	Check logs & firewall rules
Delayed email delivery	High mail queue or slow DNS lookup	Monitor mail queue & optimize DNS
Emails rejected by recipient servers	Blacklisted mail IP	Check **Spamhaus** & request delisting
Users can't receive emails	Incorrect MX records	Verify MX records with dig

Unit:8 Mail Server Basics

Assignment:

- Define the term MUA, MTA, MDA and MAA with protocol example. What do you mean by SMTP relay?
- What are the Properties and features of IMAP server.
- What do you mean by SPAM control and filtering? Explain.
- what are the different mail agents and their functions?
- How does SMTP relay work? Explain
- How can you manage the mail domain administration?
- Explain about POP and IMAP principle.

Unit:9 Remote Administration and Management

- **Router Configuration – RIP/OSPF Router**
- **Webmin/usermin**
- **Team Viewer**
- **Telnet**
- **SSH**
- **SCP, Rsync**

Unit:9 Remote Administration and Management

OSPF

OSPF is an interior gateway protocol that routes Internet Protocol (IP) packets solely within a single routing domain (autonomous system). It gathers link state information from available routers and constructs a topology map of the network. The topology determines the routing table presented to the Internet Layer which makes routing decisions based solely on the destination IP address found in IP packets. OSPF was designed to support variable-length subnet masking (VLSM) or Classless Inter-Domain Routing (CIDR) addressing models.

OSPF detects changes in the topology, such as link failures, very quickly and converges on a new loop-free routing structure within seconds. It computes the shortest path tree for each route using a method based on Dijkstra's algorithm, a shortest path first algorithm.

The link-state information is maintained on each router as a link-state database (LSDB) which is a tree-image of the entire network topology. Identical copies of the LSDB are periodically updated through flooding on all OSPF routers.

Unit:9 Remote Administration and Management

- **Router Configuration – RIP/OSPF Router**

The OSPF routing policies to construct a route table are governed by link cost factors (external metrics) associated with each routing interface. Cost factors may be the distance of a router (round-trip time), network throughput of a link, or link availability and reliability, expressed as simple unitless numbers. This provides a dynamic process of traffic load balancing between routes of equal cost.

An OSPF network may be structured, or subdivided, into routing areas to simplify administration and optimize traffic and resource utilization. Areas are identified by 32-bit numbers, expressed either simply in decimal, or often in octet-based dot-decimal notation, familiar from IPv4 address notation.

By convention, area 0 (zero) or 0.0.0.0 represents the core or backbone region of an OSPF network. The identifications of other areas may be chosen at will; often, administrators select the IP address of a main router in an area as the area's identification. Each additional area must have a direct or virtual connection to the backbone OSPF area. Such connections are maintained by an interconnecting router, known as area border router (ABR). An ABR maintains separate link state databases for each area it serves and maintains summarized routes for all areas in the network.

Unit:9 Remote Administration and Management

- **Router Configuration – RIP/OSPF Router**

OSPF does not use a TCP/IP transport protocol (UDP, TCP), but is encapsulated directly in IP datagrams with protocol number 89. This is in contrast to other routing protocols, such as the Routing Information Protocol (RIP), or the Border Gateway Protocol (BGP). OSPF handles its own error detection and correction functions.

OSPF uses multicast addressing for route flooding on a broadcast network link. For non broadcast networks special provisions for configuration facilitate neighbor discovery. OSPF multicast IP packets never traverse IP routers, they never travel more than one hop. OSPF reserves the multicast addresses224.0.0.5 for IPv4 or FF02::5 for IPv6 (all SPF/link state routers, also known as AllSPFRouters) and 224.0.0.6 for IPv4 or FF02::6 for IPv6 (all Designated Routers, AllDRouters)

Unit:9 Remote Administration and Management

OSPF router types

OSPF defines the following router types:

1. **Area border router (ABR)** : An area border router (ABR) is a router that connects one or more areas to the main backbone network. It is considered a member of all areas it is connected to. An ABR keeps multiple copies of the link-state database in memory, one for each area to which that router is connected.

2. Autonomous system boundary router (ASBR) : An autonomous system boundary router (ASBR) is a router that is connected to more than one Routing protocol and that exchanges routing information with routers in other protocols. ASBRs typically also run an exterior routing protocol (e.g., BGP), or use static routes, or both. An ASBR is used to distribute routes received from other, external ASs throughout its own autonomous system. (An interactive ASBR simulation shows how an ASBR creates External LSA (Link State Adversiment) for external addresses and floods them to all areas via ABR.) Routers in other areas use ABR as next hop to access external addresses. Then ABR forwards packets to the ASBR that announces the external addresses.

OSPF router types

OSPF defines the following router types:

3. **Internal router (IR)** : An internal router is a router that has OSPF neighbor relationships with interfaces in the same area. An internal router has all its interfaces in a single area.

4. **Backbone router (BR)** : Backbone routers are all routers that are connected to the OSPF backbone, irrespective of whether they are also area border routers or internal routers of the backbone area. An area border router is always a backbone router, since all areas must be either directly connected to the backbone or connected to the backbone via a virtual link (spanning across another area to get to the backbone).

Unit:9 Remote Administration and Management

Routing Information Protocol

The Routing Information Protocol (RIP) is a distance-vector routing protocol, which employs the hop count as a routing metric. RIP prevents routing loops by implementing a limit on the number of hops allowed in a path from the source to a destination. The maximum number of hops allowed for RIP is 15. This hop limit, however, also limits the size of networks that RIP can support. A hop count of 16 is considered an infinite distance and used to deprecate inaccessible, inoperable, or otherwise undesirable routes in the selection process.

RIP implements the split horizon, route poisoning and holddown mechanisms to prevent incorrect routing information from being propagated. These are some of the stability features of RIP. It is also possible to use the so called RMTI (Routing Information Protocol with Metric based Topology Investigation) algorithm to cope with the count-to-infinity problem. With its help, it is possible to detect every possible loop with a very small computation effort.

Unit:9 Remote Administration and Management

Routing Information Protocol

The Routing Information Protocol (RIP) is a distance-vector routing protocol, which employs the hop count as a routing metric. RIP prevents routing loops by implementing a limit on the number of hops allowed in a path from the source to a destination. The maximum number of hops allowed for RIP is 15. This hop limit, however, also limits the size of networks that RIP can support. A hop count of 16 is considered an infinite distance and used to deprecate inaccessible, inoperable, or otherwise undesirable routes in the selection process.

RIP implements the split horizon, route poisoning and hold down mechanisms to prevent incorrect routing information from being propagated. These are some of the stability features of RIP. It is also possible to use the so called RMTI (Routing Information Protocol with Metric based Topology Investigation) algorithm to cope with the count-to-infinity problem. With its help, it is possible to detect every possible loop with a very small computation effort.

Unit:9 Remote Administration and Management

Routing Information Protocol

Originally each RIP router transmitted full updates every 30 seconds. In the early deployments, routing tables were small enough that the traffic was not significant. As networks grew in size, however, it became evident there could be a massive traffic burst every 30 seconds, even if the routers had been initialized at random times. It was thought, as a result of random initialization, the routing updates would spread out in time, but this was not true in practice. Sally Floyd and Van Jacobson showed in 1994 that, without slight randomization of the update timer, the timers synchronized over time. In most current networking environments, RIP is not the preferred choice for routing as its time to converge and scalability are poor compared to EIGRP, OSPF, or IS-IS (the latter two being link-state routing protocols), and (without RMTI) a hop limit severely limits the size of network it can be used in. However, it is easy to configure, because RIP does not require any parameters on a router unlike other protocols (see here for an animation of basic RIP simulation visualizing RIP configuration and exchanging of Request and Response to discover new routes).

RIP uses the User Datagram Protocol (UDP) as its transport protocol, and is assigned the reserved port number 520.

Unit:9 Remote Administration and Management

Routing Information Protocol

Originally each RIP router transmitted full updates every 30 seconds. In the early deployments, routing tables were small enough that the traffic was not significant. As networks grew in size, however, it became evident there could be a massive traffic burst every 30 seconds, even if the routers had been initialized at random times. It was thought, as a result of random initialization, the routing updates would spread out in time, but this was not true in practice. Sally Floyd and Van Jacobson showed in 1994 that, without slight randomization of the update timer, the timers synchronized over time. In most current networking environments, RIP is not the preferred choice for routing as its time to converge and scalability are poor compared to EIGRP, OSPF, or IS-IS (the latter two being link-state routing protocols), and (without RMTI) a hop limit severely limits the size of network it can be used in. However, it is easy to configure, because RIP does not require any parameters on a router unlike other protocols (see here for an animation of basic RIP simulation visualizing RIP configuration and exchanging of Request and Response to discover new routes).

RIP uses the User Datagram Protocol (UDP) as its transport protocol, and is assigned the reserved port number 520.

Unit:9 Remote Administration and Management

Introduction To Webmin

Webmin is a web-based interface for system administration for Unix. Using any browser that supports tables and forms (and Java for the File Manager module), you can setup user accounts, Apache, DNS, file sharing and so on. Webmin consists of a simple web server, and a number of CGI (Common Gateway Interface) programs which directly update system files like /etc/inetd.conf and /etc/passwd. The web server and all CGI programs are written in Perl version 5, and use no non-standard Perl modules.

Unit:9 Remote Administration and Management

What is Usermin?

Usermin is a web-based interface for webmail, password changing, mail filters, fetch mail and much more. It is designed for use by regular non-root users on a Unix system, and limits them to tasks that they would be able to perform if logged in via SSH or at the console.

Who can use Usermin?

Most users of Usermin are sysadmins looking for a simple webmail interface to offer their customers. Unlike most other webmail solutions, it can be used to change passwords, read email with no additional servers installed (like IMAP or POP3), and setup users' Procmail configurations for forwarding, spam filtering and autoreponders. Usermin also provides web interfaces for viewing and managing data in MySQL and PostgreSQL databases, editing Apache .htaccess configuration files, and running commands on the server. The administrator has full control over which of these modules are available to users.

Unit:9 Remote Administration and Management

TeamViewer

TeamViewer is a proprietary computer software package for remote control, desktop sharing, online meetings, web conferencing and file transfer between computers. The software operates with the Microsoft Windows, Mac OS X, Linux, iOS, and Android operating systems. It is possible to access a machine running TeamViewer with a web browser. While the main focus of the application is remote control of computers, collaboration and presentation features are included.

TeamViewer may be installed with an installation procedure, although the 'Quick Support' version will run without installation. To connect to another computer, TeamViewer has to be running on both machines. To install TeamVieweradministratoraccess is required, but once installed it can be run by any user. When TeamViewer is started on a computer, it generates a partner ID and password (user-defined passwords are also supported). To establish a connection from a local client to a remote host machine, the local operator must communicate with the remote operator, request the ID and password, then enter these into the local TeamViewer.

Unit:9 Remote Administration and Management

TeamViewer

TeamViewer is a proprietary computer software package for remote control, desktop sharing, online meetings, web conferencing and file transfer between computers. The software operates with the Microsoft Windows, Mac OS X, Linux, iOS, and Android operating systems. It is possible to access a machine running TeamViewer with a web browser. While the main focus of the application is remote control of computers, collaboration and presentation features are included.

TeamViewer may be installed with an installation procedure, although the 'Quick Support' version will run without installation. To connect to another computer, TeamViewer has to be running on both machines. To install TeamVieweradministratoraccess is required, but once installed it can be run by any user. When TeamViewer is started on a computer, it generates a partner ID and password (user-defined passwords are also supported). To establish a connection from a local client to a remote host machine, the local operator must communicate with the remote operator, request the ID and password, then enter these into the local TeamViewer.

Unit:9 Remote Administration and Management

Telnet (Telecommunication Network)

Telnet is a network protocol used for remote login to a system over a TCP/IP network. It allows users to connect to a remote device and execute commands as if they were physically present.

Key Features

- Uses port 23.
- Unencrypted communication (not secure).
- Allows remote administration of network devices.
- Replaced by SSH due to security concerns.

Unit:9 Remote Administration and Management

How to Use Telnet?

1. Connecting to a Remote Host
   -> telnet <IP Address or Hostname>
   -> telnet 192.168.1.100

2. Testing Port Connectivity
   -> telnet <IP Address> <Port>
   -> telnet 192.168.1.100 80

Why Telnet is Insecure

- Sends data in plain text (including passwords).
- Susceptible to MITM (Man-in-the-Middle) attacks.
- No encryption or authentication mechanisms.

Unit:9 Remote Administration and Management

SSH (Secure Shell)

SSH is a secure protocol that allows secure connections between computers for remote login, command execution, and file transfers over a network.

Key Features

- Uses port 22 (default).
- Provides encrypted communication.
- Supports password & key-based authentication.
- Can be used for tunneling, port forwarding, and file transfer.

Unit:9 Remote Administration and Management

How to Use SSH ?

1. Connecting to a Remote Host
   -> ssh user@<IP Address or Hostname>
   -> ssh admin@192.168.1.100

2. Running a Remote Command Over SSH
   -> ssh user@192.168.1.100 "ls -l /var/www“

3. Tunneling with SSH
   SSH Port Forwarding:
   -> ssh -L 8080:localhost:80 user@192.168.1.100

Unit:9 Remote Administration and Management

SCP (Secure Copy Protocol)

SCP is a **secure file transfer protocol** that uses SSH to copy files between local and remote systems.

The scp command can be used in three* ways: to copy from a (remote) server to your computer, to copy from your computer to a (remote) server, and to copy from a (remote) server to another (remote) server. In the third case, the data is transferred directly between the servers; your own computer will only tell the servers what to do.

Key Features

- Uses SSH (port 22) for encryption.
- Supports password & key-based authentication.
- Faster than SFTP, but lacks interactive features.

Unit:9 Remote Administration and Management

How to Use SCP?

1. Copy a File from Local to Remote

    -> scp file.txt user@192.168.1.100:/home/user/

2. Copy a File from Remote to Local

    -> scp user@192.168.1.100:/home/user/file.txt /local/path/

3. Copy a File from Remote to Remote

    -> scp user1@192.168.1.10:/home/user1/file.txt user2@192.168.1.20:/home/user2/

Unit:9 Remote Administration and Management

rSync (Remote Synchronization)

rsync is a **fast, efficient file synchronization tool** that can copy files between local and remote systems while minimizing data transfer.

Key Features

- Uses SSH (port 22) for secure transfers.
- Supports incremental synchronization.
- Can copy only modified parts of files.
- Preserves file permissions, timestamps, and symbolic links.

Unit:9 Remote Administration and Management

rSync (Remote Synchronization)

rsync is a **fast, efficient file synchronization tool** that can copy files between local and remote systems while minimizing data transfer.

Key Features

- Uses SSH (port 22) for secure transfers.
- Supports incremental synchronization.
- Can copy only modified parts of files.
- Preserves file permissions, timestamps, and symbolic links.

Use Cases:

Syncing large directories across servers.
Backing up files efficiently.
Copying website files without overwriting unchanged files.

Unit:9 Remote Administration and Management

How to Use rsync?

1. Copy a File from Local to Remote

rsync -av file.txt user@192.168.1.100:/home/user/

-a: Archive mode (preserves metadata)

-v: Verbose mode (displays progress)

2. Copy a Directory Recursively

rsync -av /local/dir user@192.168.1.100:/remote/dir

3. Copy Only Modified Files

rsync -av --progress /local/dir user@192.168.1.100:/remote/dir

Unit:9 Remote Administration and Management

Why rsync is Better Than SCP?

- Syncs only changed files (SCP copies everything).
- Supports resuming interrupted transfers.
- Preserves file permissions, timestamps, and symlinks.
- Efficient bandwidth usage (compresses data before sending).

Unit:9 Remote Administration and Management

Comparison: SCP vs. rsync

Feature	SCP	rsync
Security	Encrypted via SSH	Encrypted via SSH
Efficiency	Transfers whole files	Transfers only changes
Resuming Transfers	No	Yes
Compression	No	Yes
Bandwidth Control	No	Yes
Best For	Simple file transfers	Large file syncs

Unit:9 Remote Administration and Management

Assignment: