CCNA 200-301

Category	Certificate
Created	@June 11, 2023 6:53 PM
Status	Open
URL	https://www.youtube.com/playlist?list=PLAqaqJU4wzYXBeFUFYs4qQ2qnWm_28xBV
Updated	@July 26, 2023 10:26 AM

@June 30, 2023

@July 10, 2023

```
%%{init: {'theme': 'default', 'themeVariables': { 'fontSize': '24px' }}}%%
graph TD;
A[CCNA Certification] --> B[Networking Fundamentals];
A --> C[Network Access];
A --> D[IP Connectivity];
A --> E[IP Services];
A --> F[Security Fundamentals];
B --> G[OSI Model];
B --> H[LANs and WANs];
C --> I[Physical Components];
C --> J[Network Topologies];
D --> K[IP Addressing and Subnetting];
D --> L[Routing Fundamentals];
E --> M[NAT and DHCP];
E --> N[DNS and SNMP];
F --> O[Security Layers];
F --> P[Security Threats];

style A fill:#4CAF50, color:#fff, stroke:#333;
style B fill:#FF9800, color:#fff, stroke:#333;
style C fill:#9C27B0, color:#fff, stroke:#333;
style D fill:#00BCD4, color:#fff, stroke:#333;
style E fill:#E91E63, color:#fff, stroke:#333;
style F fill:#2196F3, color:#fff, stroke:#333;
```

Table of Conetents

NETWORK FUNDAMENTALS:

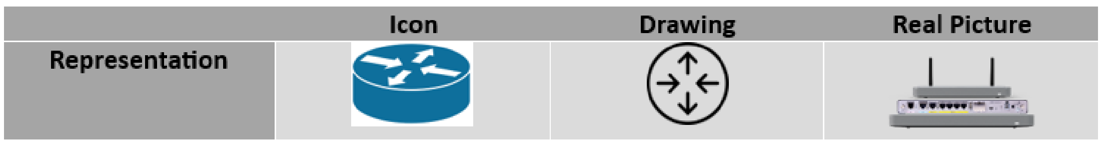
Network Fundamentals:

- A network is 2 or more devices sharing information using a common media.
- Network types include LAN, WAN.
  - **Local Area Network (LAN):**
    - Connects devices within the same room or department using a switch device.
    - Can also connect users in different rooms or departments using a router and some switches.
  - **Wide Area Network (WAN):**
    - Connects users globally through the Internet.
    - Requires service providers and a group of devices such as routers, switches, and other networking equipment.
  - **Difference between LAN and WAN:**
    - **Area:**

        - LAN connects devices within a small geographic area, typically a single building or campus.
        - WAN connects devices across a wide geographic area, typically different cities or countries.
    - **Management:**
        - LAN can be set up and managed by a single organization.
        - WAN typically requires service providers and a group of devices such as routers and switches to connect the devices.
    - **Data transfer rates and latency:**
        - LAN typically has higher data transfer rates and lower latency compared to WAN, which can have slower speeds due to the distance between the connected devices.

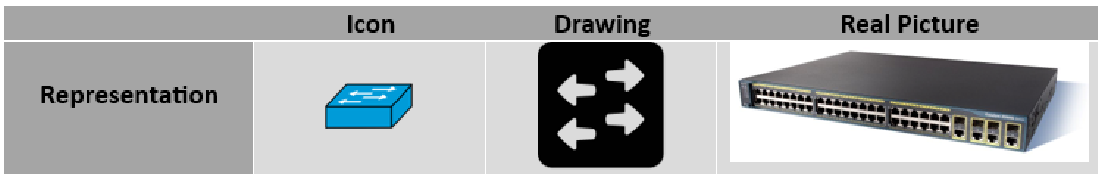
Network Components:

- **Routers:** connect different network domains and route IP packets.
    - Each **interface** on a router represents a separate broadcast domain
    - **port density:** refers to the number of physical ports available on a network device



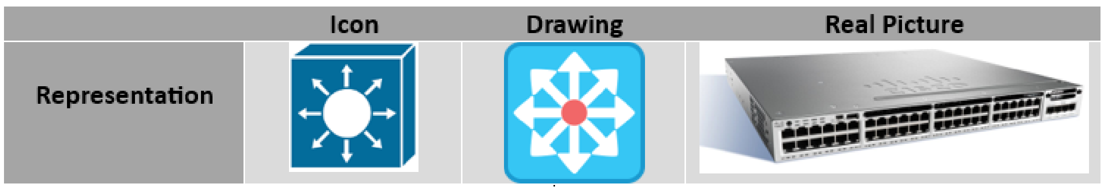
	Icon	Drawing	Real Picture
Representation			

- **Switches:** connect devices in one network domain.
    - **Bridges:** were one of the earliest technologies used to connect different network segments, and they were later replaced by switches
    - **L2 Switches:**
        - Operate at the Data Link layer (Layer 2) of the OSI model.
        - Forward data based on the MAC address of the destination device.



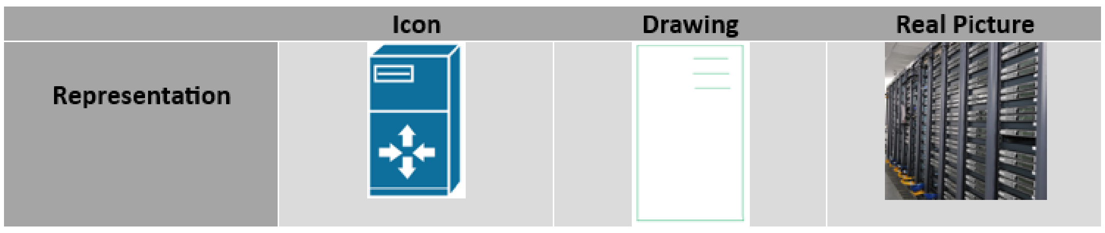
	Icon	Drawing	Real Picture
Representation			

    - **L3 Switches:**
        - Operate at the Network layer (Layer 3) of the OSI model.
        - Forward data based on the IP address of the destination network.
        - Used for connecting multiple network domains and performing routing functions.



	Icon	Drawing	Real Picture
Representation			

- **Firewalls:** protect against internet threats
    - **intrusion prevention systems:** do deep packet inspection to spot attacks.
        - Deep packet inspection (DPI) is a technique to analyze the contents of data packets as they travel across a network.
    - Next-generation firewalls (NGFW) combine firewall and IPS capabilities.
- **user devices**
    - **End devices:** devices that users interact with to access the network.
    - **host:** is any device connected to a network that participates in the communication process.
- **Access points:** are wireless destinations for hosts to communicate with each other.
- **Controllers:**
    - **Wireless controllers:** provide central management for multiple access points.
    - **Cisco DNA Center:** is a powerful management tool for analytics, automation, and configuration across all topologies
- **Servers:** powerful devices that are designed to store and manage data, applications, and services for multiple users or clients. Clients, on the other hand, are devices that consume or generate data.
    - **client-server model:** end devices requests a service or resource from a server. The client sends a request message to the server, which processes the request and sends a response message back to the client.

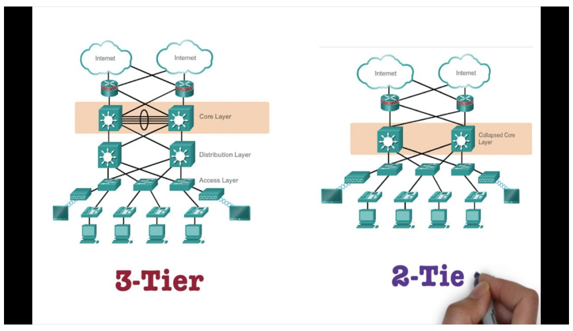


	Icon	Drawing	Real Picture
Representation			

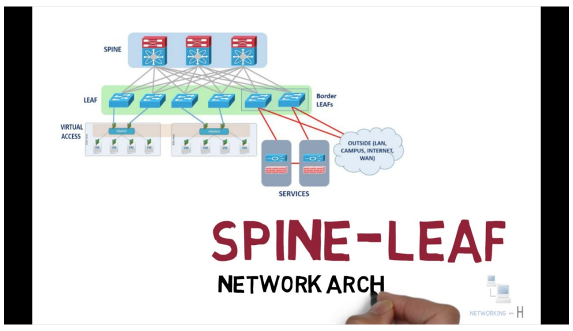
- **Virtual machines:** are software-based representations of hardware that can run multiple operating systems on one physical machine.

Network Topologies:

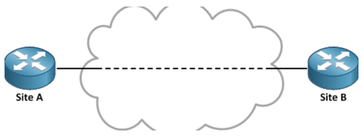
- **2Tier & 3Tier:**
  - Typical for Enterprise & Campus Networks
  - **3Tier topology** came first, and it consists of:
    - Core —→ Distribution —→ Access.
  - **In the 3-tier:** network topology, each Access layer switch should connect to each Distribution layer switch to ensure redundancy and high availability.
  - **2Tier topology** has only two layers:
    - Aggregation —→ Access .



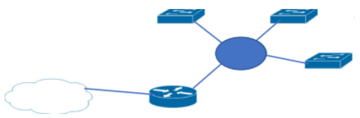
  - **Access layer:** The purpose of the Access layer is to connect end devices to the network.
  - **Distribution layer,:** the purpose of the Distribution layer is to provide routing and filtering services between different parts of the network.
  - **Core layer:** The Core layer is responsible for high-speed switching and routing between different distribution layer devices.
  - **Aggregation layer:** is responsible for aggregating traffic from multiple Access layer switches and providing connectivity to the Core layer or directly to the Internet.
  - **Difference**:
    - 3Tier topology is more popular in larger networks due to its scalability and flexibility, while 2Tier topology is more suitable for smaller networks with fewer devices.
    - The main difference between the two topologies is the number of layers and the functions they perform.
- **Spin & Leaf:**
  - Especially for **Data Centers**
  - Consists of **two** layers: **leaf layer** and **spine layer**.
    - **Leaf layer :**provides access to end devices.
    - **spine layer:** provides high-speed switching and routing between leaf switches..
  - **Fabric:** is the term used to describe the combination of the leaf and spine layers.



  - Provides high bandwidth, low latency, and scalability.
  - Uses Special Switches (**Nexus**)
    - A family of data center switches produced by Cisco Systems.
    - Designed for **high-performance**, high-density, and **low-latency** environments.
  - **Full Redundancy**
  - **NO Outage**
    - Refers to a network design or implementation that is designed to provide continuous operation **without** any **downtime**.
- **Wide Area Networks Topology (WAN):**
  - **3 types**: **Point to Point** (P2P), **Broadcast** (MetroE), **Non-Broadcast Multi-Access** (NBMA)
    - **Point to Point (P2P):**
      - A network topology in which two devices are connected directly to each other using a dedicated link.

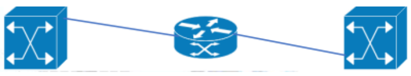


    - **Broadcast (MetroE):**
      - A network topology in which all devices on the network share a common communication channel.



      - Data is transmitted to all devices on the network, and each device filters out the data it needs.
    - **Non-Broadcast Multi-Access (NBMA):**

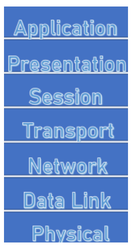
            - A network topology in which multiple devices are connected to a common network, but the network does not support broadcast communication.
            - Devices must use a specific addressing scheme to communicate with each other.



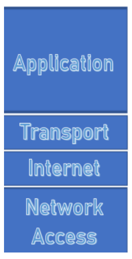
            - Requires additional configuration and management compared to broadcast networks.
- **Small Office / Home Office (SOHO):**
    - 2 Types: **Single** Router / Switch, **Few** Users, Less **Concern** about security
    - **SO: Small Office**
        - Operates out of a residential space and typically has a single individual or small team of employees.
        - Communication with the company is typically done through the company's router or VPN, providing a secure work environment at home.
        - Requires a router to provide connectivity to the Internet and other devices on the network.
    - **HO: Home Office**
        - Typically has a small LAN with fewer user links that connect to switches and routers to provide connectivity to the Internet and other devices on the network.
- **On-Premise & Cloud-Based Networks:**
    - **Difference**: **On-Premise** has everything in the office, Company, Data Center while **Cloud-Based** has everything at the Cloud Company
    - **Classic known network:** On-Premise

Network Architecture Models:

- The Open Systems Interconnection model **(OSI model)**:
    - It was developed by the International Organization for Standardization (**ISO**)
    - Provides a specific and detailed framework for network communication.
    - Consists of seven layers, each with a specific function and set of protocols.
    - The **seven** layers are:
        - **Physical layer:** transmits bits over a physical medium ,**Unit:** bit
        - **Data Link layer:** organizes bits into frames and provides error detection and correction. **Unit:** frame
        - **Network layer:** routes frames between networks using logical addresses. **Unit:** packet,datagram
        - **Transport layer:** provides reliable end-to-end delivery of packets. **Unit:** segment
        - **Session layer:** establishes, manages, and terminates sessions between applications.
        - **Presentation layer:** translates, compresses, and encrypts data for application compatibility.
        - **Application layer:** provides network services to applications.



    - **Encapsulation** and **decapsulation** occur at each layer, making troubleshooting easier.
        - **Encapsulation:** is the process of adding headers and trailers to data
        - **Decapsulation:** is the process of removing headers and trailers from data
- The Transmission Communication Protocol/Internet Protocol Model **(TCP/IP Model)**:
    - Provides a less specific framework for network communication.
    - Consists of four layers, each with a broader function compared to the OSI model.
    - The **four** layers are:
        - **Network Access layer:** handles transmission of data between a host and its network interface.
        - **Internet layer:** routes packets between networks using logical addresses.
        - **Transport layer:** provides reliable end-to-end delivery of packets.
        - **Application layer:** provides network services to applications.



    - **Encapsulation** and **decapsulation** still occur at each layer.
    - Widely used in the Internet and most modern networks.

Layer 1 Technologies(Physical layer):

Cables:

- Layer 1 technologies **include copper** and **optical fiber** connections, as well as **Power over Ethernet** (PoE).
- **Copper (Ethernet)**
    - **Copper** is the **oldest** Layer 1 technology, with a **variety** of speeds developed over time.
    - Speed measurement for **data transfer** is usually measured in **bits per second** (bps).
    - Data **storage capacity** is usually measured in **bytes**.
    - **Transfer:**
        - **copper cables** transmit data by using **electric** pulses that represent binary data in the form of 0s and 1s. These electric pulses are **modulated** by changing their **frequency** levels to differentiate between 0s and 1s **(ASK).**
        - For **transfer** to occur, the circuit must be **closed**, which requires using 2 pairs of copper wires.
    - **Twisted Pair:**
        - Twisted pair is a type of copper cable that consists of pairs of wires twisted together.
        - The twisting helps reduce **interference** from neighboring wires and external sources.
        - Benefits of twisted pairs include improved **signal quality**, reduced **crosstalk**, and increased transmission speeds.
        - **Electrical** current in a wire creates a magnetic field that can interfere with the magnetic fields of other nearby wires, causing electromagnetic interference or cross-talk. Twisted-pair cabling reduces this interference by twisting the wires together, which cancels out the magnetic fields to some extent. This is because the magnetic fields created by each wire are in opposite directions and tend to cancel each other out.
    - **Pairs**
        - More pairs of copper wires in a cable can increase the transfer units.
        - It uses 4 pairs of copper wires in a matter of electric circuit.
        - **Cat5:**
            - However, even with 4 pairs of copper wires, only 2 pairs are needed to transfer data, while the other 2 pairs can be used for other purposes such as Power over Ethernet (PoE).
            - 2 pairs are used for 100 Mbps
        - **Cat6:**
            - 4 pairs are used for 1000 Mbps.
            - Cat 6 cables can be used for PoE and data transmission.
    - **Shielded and Unshielded Copper Cables**
        - Copper connections can be shielded or unshielded, and use an RJ45 connector.
        - **Shielded Twisted Pair (STP):**
            - Shielded cables are wrapped in a protective layer to reduce interference.
            - more difficult to install than Unshielded Twisted Pair.
            - more resistant to interference and can support longer distances than UTP cabling.
        - **Unshielded Twisted Pair (UTP):**
            - Unshielded cables have no protective layer, but are cheaper and easier to install.
- **Optical Fibers**
    - Optical fiber is a group of fine fibers made of glass that uses the speed of light to transmit data.
    - Fiber is made of glass because it has a lower attenuation rate than other materials, allowing light to travel long distances without losing signal quality.
    - Optical fibers are a newer Layer 1 technology that can transmit data at even higher speeds than copper.
    - A single fiber is enough, and speeds can reach tens of Gbps.
    - Optical fibers use either light or laser for transmission.
    - Fiber requires more than one fiber to enable bi-directional communication.
    - Data is transmitted as light pulses that correspond to either a 0 or a 1. These light pulses are generated by a process called light modulation, where the light pulses are encoded with different characteristics, such as pulse duration, amplitude, or phase. For instance, a long pulse may represent a 1, while a short pulse may represent a 0.
    - **A Small Form-Factor Pluggable (SFP)**
        - chip is used to convert the light pulses to electric signals to be transmitted between devices.
        - The SFP determines the speed of data transmission between devices.
    - **types of transmission media for optical fiber:**
        - There are two types of transmission media for optical fiber: light for up to 700 meters and laser for up to 2000 meters.
        - **Single-mode:**
            - Single-mode fiber uses laser light to transfer data in a straight line, allowing it to travel long distances with minimal interference.
        - **Multimode fiber**
            - Multimode fiber uses light to transfer data, but the light pulses can collide with the cable's glass, limiting the distance it can travel.
    - Connectors for optical fibers include **LC, SC, FC, ST, and MTP/MPO.**
        - MT connectors are used for high-speed connections in optical fiber.
- **Point to Point & Shared Media**
    - Connections can be either point to point (P2P) or shared media, which involves a layer 2 device in the way.
    - P2P connections are directly connected without any intermediaries.
- **Power over Ethernet (PoE)**

- PoE is a technology for carrying power over 2 pairs of copper cables, which is enough to power some network devices.
- **AC adapters:** provide the electrical source.
- **Power Sourcing Equipment (PSE):** is a device that provides power to a Powered Device (PD) over Ethernet cables in Power over Ethernet (PoE) networks like switch .
- **Powered Device (PD)**:is a device that receives power over Ethernet cables in Power over Ethernet (PoE) networks like printer or pcs.
- PoE can replace an AC adapter and uses Power Sourcing Equipment (PSE).
- Negotiation occurs between the PSE and PD before/after power supply to agree on the specific wattage required for the device.
- PoE can supply power from 15 to 95 watts total, while **Universal PoE** uses all **4 pairs** to carry both data and power.

- **Collisions:**
  - Occur when more than one device (PC) transmits at the same time on a shared media.
  - Carrier sense multiple access/collision detection (CSMA/CD) is used to resolve collisions.
    1. When a device wants to transmit data, it first listens to the network to check if any other device is currently transmitting data.
    2. If the network is idle, the device starts transmitting data immediately.
    3. If the network is busy, the device waits for a random period of time before retrying the transmission.
    4. If two devices start transmitting data at the same time and a collision occurs, both devices stop transmitting and wait for a random period of time before retrying the transmission.
    5. During the collision, CSMA/CD uses a collision detection mechanism to detect the collision and stop the transmission of data.
    6. After a collision, the devices involved in the collision wait for a random time before attempting to re-transmit their data to avoid another collision.
- **Errors:**
  - Cabling issues and unsupported SFPs can cause errors in network communication.
  - **Duplex Mismatch:**
    - Occurs when the duplex mode (half or full) of two connected devices does not match.
    - Duplex mode must match for effective network communication.
- **Speed:**
  - The speed of data transmission (10/100/1000) must match between connected devices.
  - Mismatched speeds can cause network slowdowns or failures.

Networking Languages:

- **The Binary Language:**
  - Consists of only two digits: 0 and 1.
  - Used for all network communications.
  - Each digit is equivalent to 1 bit.
  - Zeros represent a low electric pulse or low frequency light wave, while ones represent the opposite.
  - each digit represents a power of 2 in decimal

$$\begin{gathered}2^0 = 1\\2^1 = 2\\2^2 = 4\\2^3 = 8\\2^4 = 16\\2^5 = 32\\2^6 = 64\\2^7 = 128\\2^8 = 256\\\text{and so on.}\end{gathered}$$

- **The Decimal Language:**
  - Consists of 10 digits: 0 to 9.
  - Used to represent values from 0 to 255 for protocols.
  - Used for ease of understanding by humans.
- **The Hexa-Decimal Language:**
  - Consists of 16 digits: 0 to 9 and A to F.
  - Used for representing binary data in a more compact and readable format.
  - 0 represents the smallest value, while F represents the largest value.

Layer 2 Technologies(Data-Link layer):

Media Access Control Address (MAC Address):

- Layer 2 Technology
- Hexa-Decimal Language
- **Physical Address:**
  - The MAC address is a physical address assigned to a network interface controller (NIC) by the manufacturer.
  - The **NIC** is responsible for sending and receiving data over the network and uses the MAC address to identify the device on the network.
  - Every port on a switch has a unique MAC address

- Constant and Unique
- 48 Bit length:
  - MAC addresses are 48 bits in length, represented as 12 hexadecimal digits.
  1. Ensure each character is a hexadecimal digit (0-9, A-F).
  2. Separate every two hexadecimal digits with a colon or a hyphen.
  3. Make sure the MAC address has 12 hexadecimal digits in total (6 groups of two digits separated by colons or hyphens).
  - **Examples:**

$$12:34:56:78:9A:BC \\ 00:11:22:33:44:55 \\ AB-CD-EF-01-23-45$$

- Half for the Organization, half for the product:
  - The first 24 bits of the MAC address represent the organization **OUI** that manufactured the NIC, while the remaining 24 bits represent the specific product.

Layer 3 Technologies(Network layer):

IPV4:

- Layer 3 Technology
- **Decimal Language (and Binary):**
  - IPv4 addresses are represented using decimal numbers (in dotted-decimal notation) and binary numbers.
- **Logical Address:**
  - The IPv4 address is a logical address assigned to a device by the network administrator.
- **Variable, based on the need:**
  - The number of IPv4 addresses required by a network can vary based on the network's size and complexity.
- **32 Bit length:**
  - Each octet in an IPv4 address is 8 bits or 1 byte in length.
  - IPv4 addresses are 32 bits in length, represented as 4 octets of decimal numbers or binary digits.
  - Examples

$$192.168.0.1 \\ 10.0.0.1 \\ 172.16.0.1$$

  - **Part for the Network, Part for the Hosts:**
    - The IPv4 address is divided into two parts, with a portion used to identify the network and the other portion used to identify the hosts on the network.
- **Addressing:**
  - **subnet mask:** A subnet mask is a 32-bit value used in IPv4 networking to divide an IP address into two parts: the network address and the host address. Here is an organized list of key points about subnet
    - The subnet mask is written in three forms: using the **"/ notation,"** in **binary form**, and in **decimal form.**
  - IPv4 addresses can be converted from binary to decimal and vice versa, and the network and host portions of the address are defined based on the subnet mask.
  - **The network portion** of an IP address is determined by performing a bitwise logical **AND** operation between the IP address and the subnet mask.
  - **The host portion** of an IP address is the remaining bits of the IP address after the network portion has been determined.
  - Each octet in an IP address is equal to 256 which 2^8, so the values range from 0 to 255.
  - **The first IP address** in a network range is reserved for the **network ID**, while the **last IP** address is **reserved** for the **network broadcast ID**.
  - The maximum number of hosts that can be assigned unique IP addresses in IPv4 is approximately 4.3 billion (2^{32})
  - **Examples:**

$$\text{IP address:}192.168.1.100 \\ \text{Subnet mask:}255.255.255.0 \\ \text{Binary representation of the subnet mask:}11111111.11111111.11111111.00000000 \\ \text{Network portion of the IP address:}192.168.1.0 \\ \text{Host portion of the IP address:}0.0.0.100$$

$$\text{Example 1:} \\ \text{IPv4:}172.16.0.0/16 \\ \text{The subnet mask in binary form is } 11111111.11111111.00000000.00000000. \\ \text{The network portion of the IP address is}172.16.0.0, \text{and the host portion is}0.0.0.0. \\ \text{The number of hosts in the network is } 2^{16}-2. \\ \text{The network ID is } 172.16.0.0, \text{and the broadcast ID is } 172.16.255.255.$$

$$\text{Example 2:} \\ \text{IPv4:}10.0.0.0/8 \\ \text{The subnet mask in binary form is } 11111111.00000000.00000000.00000000. \\ \text{The network portion of the IP address is}10.0.0.0, \text{and the host portion is } 0.0.0.0. \\ \text{The number of hosts in the network is}2^{24}-2. \\ \text{The network ID is } 10.0.0.0, \text{and the broadcast ID is } 10.255.255.255.$$

- **Subnetting:**
  - Subnetting is the process of dividing a larger network into smaller subnetworks, accomplished by manipulating the subnet mask.

- **Example:**

  Suppose you have been assigned the IP address block **192.168.0.0/24** for your network. However, you have two separate departments in your organization that require their own subnets. You can subnet the network into two subnets, each with its own unique network ID and broadcast ID.

  To create two subnets, you can use the subnet mask **255.255.255.128 (/25)** to divide the network into two **equal subnets** of **128** addresses each. This means that the **first subnet** will have the **network ID of 192.168.0.0 and the broadcast ID of 192.168.0.127**, while the **second subnet** will have the **network ID of 192.168.0.128 and the broadcast ID of 192.168.0.255.**

- **Variable-Length Subnet Mask (VLSM):**
  - VLSM is a technique used in subnetting to allocate IP addresses more efficiently by using different subnet masks for different subnetworks.
  - VLSM is much more economical for the use of subnetting because it allows for the creation of smaller subnets with more addresses, which can be assigned to larger departments or networks with more hosts.
- **IPv4 Classes:**
  - IPv4 addresses were originally divided into classes based on the size of the network, with the class defining the network and host portions of the address.
  - The IPv4 address space is divided into five classes, namely **A, B, C, D, and E.**
    - Class **A** addresses have the first octet in the range 1-126, with the remaining three octets used for host addresses.
    - Class A: /8 1.0.0.0 --- 126.255.255.255
    - Class **B** addresses have the first two octets in the range 128-191, with the remaining two octets used for host addresses.
    - Class B: /16 128.0.0.0 --- 191.255.255.255
    - Class **C** addresses have the first three octets in the range 192-223, with the last octet used for host addresses.
    - Class C: /24 192.0.0.0 --- 223.255.255.255
    - Class **D** addresses are used for multicasting, with the first four bits set to 1110.
    - Class D: /8 224.0.0.0--- 239.255.255.255
    - Class E addresses are reserved for experimental purposes, with the first four bits set to 1111.
    - Class E: /8 240.0.0.0--- 255.255.255.255
  - Some IP addresses are reserved for special purposes, such as **0.x.x.x**, which is reserved by **IANA**, and **127.x.x.x**, which is reserved by **Microsoft for loopback assignment.**
  - Another reserved address range is **169.254.0.0/16**, which is reserved for Automatic Private IP Addressing (**APIPA**).
- **Classless Inter-Domain Routing (CIDR):**
  - CIDR is a more flexible method of allocating IP addresses that does not rely on fixed address classes and can use VLSM.
  - CIDR notation is expressed as an IP address followed by a slash (/) and a number indicating the number of bits in the subnet mask.
  - To solve the problem of strict IP address classes, CIDR allows for the subdivision of IP address blocks into smaller subnets of varying sizes.
  - **Examples**:
    - Example 1: subnetting for **20** employees with IP address range **192.168.1.0/24:**
      - The required number of IP addresses is 20 + 2 (for network ID and broadcast ID) = 22.
      - To accommodate 22 IP addresses, the subnet mask needs to be /27 because $2^5 = 32$, which is the smallest power of 2 that is greater than or equal to 22.
      - The subnet range will be 192.168.1.0/27, with a network ID of 192.168.1.0, a broadcast ID of 192.168.1.31, and 30 usable IP addresses (from 192.168.1.1 to 192.168.1.30).
      - 10 IP addresses are not used and can be reserved for future use.
    - Example 2: subnetting for **120** employees with IP address range **10.0.0.0:**
      - The required number of IP addresses is 120 + 2 (for network ID and broadcast ID) = 122.
      - To accommodate 122 IP addresses, the subnet mask needs to be /25 because $2^7 = 128$, which is the smallest power of 2 that is greater than or equal to 122.
      - The subnet range will be 10.0.0.0/25, with a network ID of 10.0.0.0, a broadcast ID of 10.0.0.127, and 126 usable IP addresses (from 10.0.0.1 to 10.0.0.126).
      - 6 IP addresses are not used and can be reserved for future use.
- **Private vs. Public IPv4 Addresses:**
  - Private IPv4 addresses are reserved for internal networks and are not publicly routable, while public IPv4 addresses are assigned by regional Internet registries and are routable on the public Internet.
  - This is done to prevent IP address exhaustion and to avoid collisions or duplication of IP addresses.
  - Private IP addresses are often used in local area networks (LANs).
  - Network Address Translation (**NAT**) is a technique used to translate private IP addresses to public IP addresses, allowing devices on a private network to communicate with devices on the internet.
  - **Private Addresses:**
    - There are three ranges of private IPv4 addresses that can be used on internal networks: **10.0.0.0/8**, **172.16.0.0/12**, and **192.168.0.0/16.**

IPV6:

- IPv6 (Internet Protocol Version 6) is the **latest** version of the Internet Protocol, designed to **replace** IPv4 and address its **limitations**.
- IPv6 uses a **hexadecimal** language to represent IP addresses, which is different from the **decimal notation** used in IPv4.
- IPv6 addresses are **128** bits in length, which is four times the length of IPv4 addresses, allowing for a much larger address space.
- IPv6 addresses are divided into **eight** parts, separated by **colons**, with each part consisting of **four** hexadecimal digits.
  - The **classical way** of writing an IPv6 address is to include leading zeros in each part, for **example:**
    - $2001 : DB80 : 0000 : 0000 : 0000 : 0000 : 0000 : 0001/x.$

  - **Leading zeros can be omitted** in each part, **for example:**
    - $2001 : DB80 : 0000 : 0000 : 0000 : 0000 : 0000 : 1/x$ can be **written** as $2001 : DB80 :: 1/x$
    - $2001 : DB80 : 0000 : 0000 : 0000 : 0000 : 0000 : 1/x$ can be **written** as $2001 : DB80 : 0000 : 0000 : 0000 : 0000 : 0000 : 1/x$
    - $2001 : DB80 : 0000 : 0000 : 0000 : 0000 : 0000 : 1/x$ can be **written** as $2001 : DB80 :: 0001/x$
  - **Double colons** can be used **once** in an IPv6 address to represent multiple sets of consecutive zeros in a part
- The **total numbe**r of possible **IPv6** addresses is 2^{128}, which is approximately 340 undecillion, a number so large that it is difficult to comprehend.
- **There are several types of IPv6 addresses:**
  - **Global Unicast:** These are public IPv6 addresses that are globally routable on the internet. The range for global unicast addresses is $\mathbf{2000 :: /3.}$
  - **Unique Local:** These are private IPv6 addresses that are used within a private network and are not routable on the internet. The range for unique local addresses is $FC00 :: /7.$
  - **Link Local:** These are IPv6 addresses that are assigned to an interface on a device, usually based on its MAC address. The range for link local addresses is $FE80 :: /10.$
    - EUI-64 (Extended Unique Identifier-64) is a method used to automatically generate a unique interface identifier for an IPv6 link-local address from a device's MAC address.
    - **steps to create a link-local address using EUI-64:**
      1. Take the **MAC address**: $001C : D101 : E5C5.$
      2. Add the **fixed** prefix $\mathbf{FE80\ ::}$to the beginning of the IPv6 address.
      3. Divide the MAC address into two halves: $\mathbf{001C : D1}$and $\mathbf{01 : E5C5}$.
      4. Take the left half of the MAC address and insert it into the IPv6 address after the prefix, separated by colons: $FE80 :: \mathbf{001C : D101}$.
      5. Add the hexadecimal value $\mathbf{FF : FE}$ in between the two halves of the MAC address: $FE80 :: 001C : D1\mathbf{FF : FE}01 : E5C5$.
      6. The resulting address is the link-local address based on the device's MAC address.
    - Devices on the same network segment can communicate with each other using link-local addresses, even if they do not have global or unique local addresses assigned.
    - Link-local addresses are **automatically** assigned and do not require any configuration, making them a convenient way to enable communication between devices on a network segment.
  - **Anycast:** is an addressing and routing technique in IPv6 where a single IP address can be assigned to multiple nodes, but only the node that is nearest to the sender is selected to receive and process the data sent to that IP address. This technique is used for load balancing and redundancy purposes.
  - **Multicast:** These are IPv6 addresses that are used to send data to multiple hosts simultaneously. The range for multicast addresses is $FF00 :: /8.$
- IPv6 also includes several new features and improvements over IPv4, such as improved security, auto-configuration, and support for larger payload sizes.

Layer 4 Technologies(Transport layer):

Transport Layer:

- The transport layer uses ports to identify different applications and services on a device.
- A port number is a 16-bit unsigned integer that identifies a specific process on a device.
- Ports allow for multiple applications and services to run simultaneously on a device without interfering with each other.
- Ports are numbered between 0 and 65535 and are divided into two ranges: 0-1023 are reserved for well-known applications and services, and registered ports (1024-49151) while ports above 49151 are available for general use.
- TCP and UDP are the two most commonly used transport layer protocols.

Transmission Communication Protocol & User Datagram Protocol:

- **Transmission Control Protocol (TCP):**
  - TCP is a reliable transport protocol that provides a connection-oriented service for data transmission.
    - **TCP guarantees reliability by using several mechanisms:**
      - **Three-Way Handshake:(SYN, SYN-ACK, ACK)**
      - **Error Checking:** TCP uses a checksum to check for errors in the data being transmitted. If any errors are detected, the receiving device requests that the sender retransmit the data.
      - **Flow Control:** TCP uses a sliding window mechanism to control the flow of data between the sending and receiving devices. This ensures that the receiving device can handle the amount of data being sent and prevents data loss or congestion.
      - **Retransmission:** If data is lost during transmission, TCP will retransmit the lost data until it is received by the receiving device. This ensures that all data is transmitted correctly and efficiently.
  - TCP is slower than User Datagram Protocol (UDP) due to the overhead of establishing and maintaining a connection.
  - TCP uses a **three-way handshake** to establish a connection between two devices before data transmission can begin.
    - TCP uses a three-way handshake **(SYN, SYN-ACK, ACK)** to establish a connection between two devices before data transmission can begin.
    - The sending device (client) initiates the handshake by sending a SYN packet to the receiving device (server), indicating that it wants to establish a connection.
    - The server responds with a SYN-ACK packet, acknowledging the client's request and indicating that it is ready to establish a connection.
    - The client then sends an ACK packet to the server, confirming that it received the SYN-ACK packet and that the connection is established.
    - Once the connection is established, data can be transmitted between the two devices with built-in error-checking and flow control mechanisms to ensure that the data is transmitted reliably and efficiently.
  - TCP is used for applications that require reliable data transmission, **such as:**
    - **HTTP** (Hypertext Transfer Protocol) is the protocol used for transmitting web pages over the internet. HTTP uses TCP port **80** by default.

- **HTTPS** (Hypertext Transfer Protocol Secure) is a secure version of HTTP that uses **SSL/TLS** encryption to protect data transmitted over the internet. HTTPS uses TCP port **443** by default.
- **FTP** (File Transfer Protocol) is used for transferring files over the internet. FTP uses TCP ports **20** and **21** for data transfer and control, respectively.
- **SSH** (Secure Shell) is a secure protocol used for remote access and management of devices over the internet. SSH uses TCP port **22** by default.
- **SMTP** (Simple Mail Transfer Protocol) is used for sending and receiving email over the internet. SMTP uses TCP port **25** by default.
- **BGP** (Border Gateway Protocol) is used for exchanging routing information between devices on the internet. BGP uses TCP port **179** by default.

- **User Datagram Protocol (UDP):**
    - UDP is a not-reliable transport protocol that provides a connectionless service for data transmission.
    - UDP is faster than TCP because it does not perform any pre-steps before transmitting data.
    - UDP is used for applications that can tolerate some data loss, **such as:**
        - **SNMP** (Simple Network Management Protocol) is used for managing and monitoring network devices, such as routers and switches. SNMP uses UDP port 161 by default.
        - **TFTP** (Trivial File Transfer Protocol) is a simple protocol used for transferring files between devices on a network. TFTP uses UDP port **69** by default.
        - **DNS** (Domain Name System) is used for translating domain names into IP addresses. DNS uses UDP port **53** by default.
        - **SYSLOG** is a protocol used for transmitting system log messages between devices on a network. SYSLOG uses UDP port **514** by default.
    - UDP does not provide any error-checking or recovery mechanisms, meaning that any lost or corrupted data will not be retransmitted.

IP Parameters and Network Tools:

- IP parameters are settings used to configure the network interface on a client or end device operating system.
- Useful network tools for troubleshooting and managing network connections include Ping, Traceroute, FTP, SCP, Telnet, SSH, and Ipconfig.
    - **Ping:** is a Layer 3 tool used to **test** the availability of a device or network connection by sending ICMP echo message to a specific IP address and waits for an ICMP echo reply.
        - The **command** for Ping is **"ping x.x.x.x"** where x.x.x.x is the **IP address.**
    - **Traceroute:** is a Layer 3 tool that shows the path that packets take between two devices on a network.
        - The **command** for Traceroute is **"tracert x.x.x.x"** where x.x.x.x is the **IP address**. The "-d" tag can be used to **disable** DNS name resolution.
    - **FTP:** (File Transfer Protocol) and **SCP :**(Secure Copy Protocol) are used for **transferring** files between devices on a network.
    - **Telnet** and **SSH:** are Layer 4 tools used for **remote access** to devices on a network. **Telnet** is an **unsecured** protocol, while **SSH** is a **secure** protocol that uses **encryption**. The command for Telnet is **"telnet x.x.x.x"** where x.x.x.x is the I**P address.**
        - **SecureCRT** and **PuTTY** are software programs used for Telnet and SSH connections.
- **Ipconfig:** is a Layer 3 tool used to display the IP **address** assigned to a **network interface**. The command for Ipconfig is **"ipconfig"**. Tags such as **"/all"** can be used to display **additional** information about the network configuration.

Virtualization and Virtual Machines:

- **Network components** have physical components like CPUs, memory, and storage, but they use different operating systems than personal computers.
- **Virtualization** is the process of creating a virtualized environment that emulates multiple devices inside one physical device. This allows for ease of management and flexibility in resource allocation.
- **Virtualization** allows for multiple operating systems to be installed on one physical device, creating multiple virtual machines that are isolated and have access to a large pool of shared resources.
    - **Virtual machines (VMs)** are simulated computers that run on a physical computer or server, created through virtualization software. Each VM has its own virtual hardware, including a virtual CPU, memory, storage, and network interfaces.
- The **Hypervisor** is the mediator between the software and hardware in a virtualized environment. The Hypervisor schedules VM requests to the hardware and distributes hardware resources between VMs.
    - The **Hypervisor** is a layer of software that sits between the hardware and the operating systems installed on the virtual machines.
    - To use **virtualization**, the Hypervisor must be loaded onto the physical hardware first, and then the operating systems are installed on top of the Hypervisor.
- There are **two** types of Hypervisors: Type 1 **(native or bare metal)** and Type 2 **(hosted).**
    - **Type 1** Hypervisors run directly on the hardware resources, and the virtual machines are created on top of the Hypervisor. Examples include Citrix Xen, Oracle VM, Microsoft Hyper-V, and VMwares ESX/ESXi.
    - **Type 2** Hypervisors are hosted applications that run on top of an operating system, and the virtual machines are created on top of the Hypervisor. Examples include VirtualBox and VMwares Workstation.
- **Virtual switches:** are used to connect VMs together like a real switch. Each VM is assigned a virtual network interface card (V.NIC) and a separate V.NIC for internet access.
    - In **Type 1 Hypervisors**, the virtual switch is included by default and does not need to be created or configured separately.
    - **Port groups:** can be created to completely isolate VMs, similar to VLANs.
    - **Examples** of **Hypervisors** include Microsoft Hyper-V and ESXi VSwitch.

Introduction to Cisco IOS Systems:

Cisco IOS:

- Cisco IOS is the operating system that runs on Cisco network devices, including routers, switches, and other networking equipment.
- To configure a Cisco network component, you will need a PC, the network component, and a console cable.
- The console cable is a light blue cable with a connector on one side that is Ethernet and the other side is a DB-9 connector. Newer console cables have an Ethernet connector on one side and a USB connector on the other.

- To access the Cisco IOS CLI, you will need to use an application like PuTTY or Teraterm to connect to the device through the console port or COM port on your computer.
- The COM port is a physical interface on a computer for connecting serial devices, such as a console cable for a Cisco network device.

Practicing Cisco Network Component Configuration:

- **Packet Tracer** is a network simulation tool developed by Cisco that allows users to simulate network topologies and configure Cisco network components, including routers, switches, and other networking equipment.
- **Access commands**
  - To access the CLI in Packet Tracer, select the device and click the **"CLI"** tab.
  - The CLI has different modes, including **User Mode** (>), **Privilege Mode** (#), and **Global Configuration Mode** (Config) #, each with different levels of access and functionality.
  - To enter Privilege Mode, use the **"enable"** or **"en"** command.
  - Some common commands in Privilege Mode include **"show," "ping,"** and **"traceroute."**
  - To enter Global Configuration Mode, use the **"configure terminal"** command.
  - In Global Configuration Mode, you can execute any command to configure the device.
  - To return to Privilege Mode, use the **"exit"** command.
- **Changing Device Configuration:**
  - To change the hostname of a device, enter the **"hostname"** command followed by the desired name.
  - The changes made to the device configuration are stored in NVRAM and are temporary. If the device is turned off or restarted, the settings will be lost.
  - To save the running configuration to the startup configuration, use the **"copy running-config startup-config"** or **"write"**command.
  - To restart the device, use the **"reload"** command.
- **Common Cisco IOS Commands:**
  - **"enable":** Enters Privilege Mode, allowing access to advanced commands.
  - **"configure terminal":** Enters Global Configuration Mode, allowing configuration of the device.
  - **"interface fa0/0/1":** Selects the interface to configure. This example is for FastEthernet port 0/0/1.
  - **"ip address 192.168.1.1 255.255.255.0":** Assigns the specified IP address and subnet mask to the interface.
  - **"hostname Router1212":** Sets the hostname of the device to "Router1212".
  - **"reload"**: Reloads the device, causing it to restart.
  - "**copy running-config startup-config":** Saves the running configuration to the startup configuration, so that it persists after a restart.
  - **"write erase":** Erases the startup configuration, restoring the device to its default settings.
  - **"shutdown":** Disables the selected interface.
  - **"no shutdown":** Enables the selected interface.
- **Common Show Commands:**
  - **"show ip interface brief":** Displays a brief summary of the status and IP address configuration of all interfaces on the device.
  - **"show interface description":** Displays the description of each interface on the device.
  - **"show version":** Displays information about the device, including the IOS version and hardware configuration.
  - **"show running-config":** Displays the current running configuration of the device.
  - **"show mac address-table":** Displays the MAC address table, which shows the MAC addresses of devices connected to the device's interfaces.
  - **"show interface status":** Displays the status and current configuration of all interfaces on the device.

Lab 1:

To perform a simple lab exercise, you will need two PCs connected to the same network range (e.g., 192.168.1.0/24) and a switch to connect the PCs.

To verify connectivity between the two PCs, use the `"ping"` command in the CLI to send ICMP packets between the two devices.

- **steps to build this lab in Cisco:**
  1. Connect a switch to the network. Connect the two PCs to the switch.
  2. Configure the IP addresses on the PCs.
     - **PC1:**
       - **IP address:** $192.168.1.1.$
       - **subnet mask:** $255.255.255.0.$
     - **PC2:**
       - **IP address:** $192.168.1.2.$
       - **subnet mask:** $255.255.255.0.$
  3. Dont need to Configure the switch. By default its configured.
  4. Verify connectivity between the two PCs.

  ```
  Cisco Packet Tracer PC Command Line 1.0
  C:\>ping 192.168.1.1
  C:\>ping 192.168.1.2
  ```

  Syntax:

Layer 2 NETWORK ACCESS (Data Link layer):

Switch:

Switching Concepts:

- Switches were developed as an improvement over bridges, which had low port density and used bridge tables.
- Switches use MAC learning to build MAC tables, which are used to forward frames based on their destination MAC address.
    - **MAC learning:** is the process by which a switch learns the MAC addresses of devices connected to its ports.
    - **The MAC table:** is a database that a switch uses to store MAC addresses and their associated ports.
    - **scenario of how MAC learning happens in a switch:**
        - Let's say we have a switch with **three** ports, labeled **Port 1**, **Port 2**, and **Port 3**. We also have two devices connected to the switch: **Device A** and **Device B**.

```
graph LR
    DeviceA["Device A"]
    DeviceB["Device B"]
    Switch["Switch (Ports 1-3)"]
    DeviceA -- Port 1 --> Switch
    DeviceB -- Port 2 --> Switch
```

1. Device **A** sends a frame to Device **B**. The frame has Device **B's** MAC address as the destination and Device A's MAC address as the source.

Destination MAC Address	Source MAC Address
B MAC Address	A MAC Address

2. The switch receives the frame on **Port 1**.
3. The switch doesn't have any **entries** in its **MAC table yet**, so it doesn't know which port to use to forward the frame to **Device B**.
4. The switch adds a **dynamic** entry to its MAC table, associating Device **A's MAC** address with **Port 1**.

MAC Address	Port
A MAC Address	1

5. The switch **floods** the frame out to **all** ports **except** Port 1, since it doesn't know which port Device **B** is connected to.
6. The frame is **received** by Device **B**, which sends a response back to Device **A**.

Destination MAC Address	Source MAC Address
A MAC Address	B MAC Address

7. The **switch** receives the response on **Port 2**.
8. The switch **looks up** Device **B's** MAC address in its **MAC table** and finds the **entry** for **Port 1**.
9. The switch adds a **dynamic** entry to its MAC table, associating Device **B's** MAC address with **Port 2**.

MAC Address	Port
A MAC Address	1
B MAC Address	2

10. The switch **forwards** the **response** out **Port 1** to Device **A**.
11. From now on, when Device A sends frames to Device B, the switch will know to forward them out Port 2 based on the MAC table entries it has learned.

- The **structure** of **a frame** includes :

Field	Size	Description
Destination MAC Address	6 bytes	The MAC address of the intended recipient of the frame.
Source MAC Address	6 bytes	The MAC address of the device that sent the frame.
EtherType or Length	2 bytes	Indicates the type of payload carried in the frame, or the length of the payload if it is less than the minimum frame size. **Example:** 0x0800 → payload of the frame is an IP packet
Data Payload	46-1500 bytes	The payload of the frame, which can be any type of data.
Frame Check Sequence (FCS)	4 bytes	A checksum that is used to verify the integrity of the frame during transmission.

- When a frame is received on a switch port, the switch looks up the destination MAC address in its MAC table to determine which port to forward the frame to.
    - Switches have a lookup engine that can only process one frame at a time, but they use scheduling to ensure that frames are forwarded efficiently.
- MAC entries in the MAC table have an **aging time**, which determines how long they remain in the table before being removed. This prevents stale entries from filling up the table and causing performance issues.
    - For **example**, **Cisco** switches have a **default aging time** of **300** seconds (5 minutes)
- If the switch does not have an **entry** in its **MAC table for a destination MAC address**, it will flood the frame out to all ports except the port it was received on. This ensures the frame reaches its destination, but can cause unnecessary network traffic if the destination is not found quickly.
    - To **achieve** this, the switch sets the **destination MAC address** of the frame to the broadcast address **FFFF:FFFF:FFFF**. This ensures that the frame is **sent to all devices** on the network, without the switch having to know their individual MAC addresses.

Virtual Local Area Networks (VLANs):

- VLANs are used to logically separate hosts into **different** groups, even if they are **physically** connected to the same switch or network.
- A switch can be divided into multiple virtual switches, with each virtual switch acting as a separate switch with its own set of ports and VLAN configuration.
- Each group of hosts becomes its **own virtual LAN**, which can be assigned unique characteristics and policies.

- Each VLAN is identified by a **number** from **1 to 4096**.
- **Vlan Frame Structure :**

Preamble	Start Frame Delimiter	Destination MAC	Source MAC	VLAN Tag	EtherType/Length	Payload	FCS
7 bytes	1 byte	6 bytes	6 bytes	4 bytes	2 bytes	46-1500 bytes	4 bytes

  - **Preamble**:Preamble is to allow receiving devices to synchronize their clocks with the incoming data stream, and to detect the beginning of a new frame.
  - **Start Frame Delimiter**:The SFD is used by receiving devices to synchronize their clocks with the incoming data stream, and to detect the start of the Ethernet frame.
  - **VLAN Tag:**
    - **Priority**: A 3-bit field that specifies the priority of the frame, from 0 (lowest) to 7 (highest).
    - **CFI**: A 1-bit field that is reserved for compatibility with Token Ring networks, but is not used in Ethernet VLANs.
    - **VLAN ID**: A 12-bit field that specifies the VLAN to which the frame belongs, from 1 to 4094.
    - **Reserved**: A 16-bit field that is reserved for future use.
- Every single switch port must be assigned to either an **access port** or a **trunk port**.
- **Access port**
  - Access ports are used for connecting end devices such as computers, printers, and servers. Frames transmitted on access ports are not tagged with a VLAN ID.Because end devices do not understand VLANs.
- **Trunk port**
  - Trunk ports are used for connecting switches or other network devices that need to carry traffic for multiple VLANs. Frames transmitted on trunk ports are tagged with a VLAN ID, which allows the receiving switch to know which VLAN the frame belongs to.
- There are different types of VLANs, including
  - **Data VLANs**, which are used for ordinary traffic.Such as data packets sent between computers, servers, and other devices on the network.
  - **voice VLANs**, which are used for voice data traffic and have higher priority.Such as VoIP (Voice over IP) phone calls. These VLANs have higher priority than data VLANs
  - **default/native VLANs**, which are used for untagged traffic and are assigned to all ports on the switch by default.Native Vlan can be changed but default VLANs do not use tags, nor can they be removed or renamed.
- **Trunking** is used when more than one VLAN needs to cross a link between switches. This is done by using **encapsulation**, such as the **DOT1Q** protocol, which adds a **VLAN ID** tag to the frame header.
- **DTP** (Dynamic Trunking Protocol) is a Cisco proprietary protocol that is used to negotiate trunk links between switches.
  - DTP is used to automatically configure trunk links between switches without requiring manual configuration.
  - DTP messages are sent over the native VLAN of the link. The native VLAN is the VLAN that is untagged on a trunk link.

Lab2:

A company has two departments, each isolated on their own switch. The company wants to create VLAN 10 for Department 1 and VLAN 20 for Department 2 on each switch. There is also another building with the same structure, where there are two departments and one switch. The company wants to connect the two switches with a link so that Department 1 in the main building can communicate with Department 1 in the other building, and Department 2 can communicate with Department 2. The IP address range used is 192.168.1.0/24.

```
graph LR

subgraph Main_Building
    subgraph Vlan10_M
        PC1(PC1)
    end

    subgraph Vlan20_M
        PC2(PC2)
    end

    PC1 --- Switch1
    PC2 --- Switch1
end

subgraph Other_Building
    subgraph Vlan20_o
        PC3(PC3)
    end

    subgraph Vlan10_o
        PC4(PC4)
    end

    PC3 --- Switch2
    PC4 --- Switch2
end

Switch1 --- Switch2
```

- **Steps**
  1. Connect the switches within each building (PC's) using Ethernet cables. You can use the "Copper Straight-Through" cable type to connect the switches.
  2. Configure IP addresses for each PC . For example
     - **PC1**
       - **IP Adress:** $192.168.1.1$
       - **Subnet Mask:** $255.255.255.0$
     - **PC2**
       - **IP Adress:** $192.168.1.2$
       - **Subnet Mask:** $255.255.255.0$

- **PC3**
    - **IP Adress:** $192.168.1.3$
    - **Subnet Mask:** $255.255.255.0$
- **PC4**
    - **IP Adress:** $192.168.1.4$
    - **Subnet Mask:** $255.255.255.0$

3. Configure VLAN 10 for Department 1 and VLAN 20 for Department 2 on each switch.

```
Switch> enable
Switch# configure terminal
Switch(config)# vlan 10
Switch(config-vlan)# name Department1
Switch(config-vlan)# vlan 20
Switch(config-vlan)# name Department2
```

Syntax:

4. Assign the appropriate switch ports to each VLAN. For example, for Switch1 in the main building, you can assign ports 1 to VLAN 10 and ports 2 to VLAN 20

```
Switch> enable
Switch# configure terminal
Switch(config)# interface range fastethernet 0/1
Switch(config-if-range)# switchport mode access
Switch(config-if-range)# switchport access vlan 10
Switch(config-if-range)# exit
Switch(config)# interface range fastethernet 0/2
Switch(config-if-range)# switchport mode access
Switch(config-if-range)# switchport access vlan 20
Switch(config-if-range)# exit
```

Syntax:

5. Connect the two switches using a crossover Ethernet cable. You can use the "Copper Crossover" cable type to connect the switches.
6. Configure a trunk port on each switch to allow VLAN traffic to pass between the switches.

```
Switch> enable
Switch# configure terminal
Switch(config)# vlan 10
Switch(config-vlan)# name Department1
Switch(config-vlan)# vlan 20
Switch(config-vlan)# name Department2
```

Syntax:

7. Test communication between devices in each department and between departments in both buildings,Commands

```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.1.1
```

Syntax:

```
Switch> enable
Switch# show interface trunk
```

Syntax:

```
Switch> enable
Switch# show vlan brief
```

Syntax:

Spanning Tree Protocol(STP):

- **Spanning Tree Protocol** (STP) is used to prevent broadcast storms and loops in a network by creating a loop-free logical topology.
    - A broadcast storm, also known as a layer 2 loop, can occur when broadcast frames are forwarded in an endless loop between switches, consuming network bandwidth and causing network performance issues.
    - Broadcast storms can also result in high CPU and memory usage on switches, which can eventually lead to switch failure if left unaddressed.

```
graph LR
    Switch1 --> Switch2
    Switch2 --> Switch3
    Switch3 --> Switch1
```

- Spanning Tree Protocol (STP) is a protocol that is **enabled** by **default** on most switches
- By **default**, the bridge **priority** for Cisco switches is **32768**, but this can be changed to a value between **0 and 65535**.
- **The bridge ID** is a combination of the bridge **priority** and **MAC address** .
    - **The bridge ID=**$32768.0011 : FF01 : 02A4$
- STP works by **electing** a **root bridge** based on **bridge priority** and **MAC address**. The bridge with the **lowest priority** and **MAC address** becomes the **root bridge.**
- **Bridge Protocol Data Unit (BPDU):**
    - Messages exchanged between switches to elect a root bridge and determine the optimal path for data transmission in a network.
    - Sent out periodically by the root bridge and other switches to ensure that all devices have up-to-date information about the network topology
    - Typically sent out every 2 seconds by default, but can be adjusted to optimize network performance and stability
    - BPDU format typically includes 12 fields, divided into 3 sections: 4 general fields, 4 VLAN-specific fields, and 4 communication-specific fields

General Field	Size (bytes)	Description
VLAN ID	2	Indicates the VLAN ID of the BPDU. This field is used to identify the VLAN for which the BPDU is being sent.
Priority	2	Indicates the priority of the BPDU for the given VLAN. This field is used to determine the root bridge for the given VLAN.
Remaining Hops	1	Indicates the remaining number of hops to reach the root bridge. This field is used to determine the shortest path to the root bridge for the given VLAN.
Message Age	2	Indicates the age of the BPDU in tenths of a second for the VLAN. This field is used to determine the freshness of the BPDU for the given VLAN.

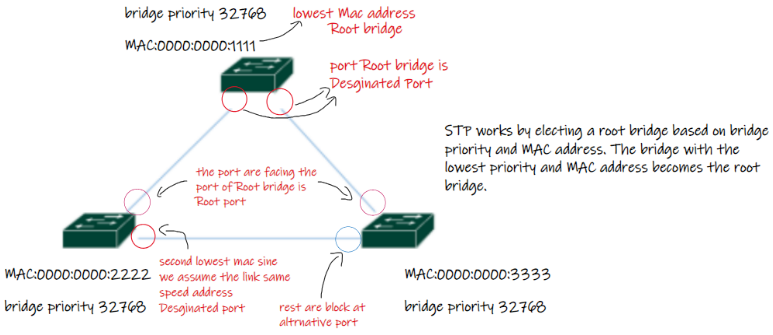
VLAN-specific Field	Size (bytes)	Description
VLAN ID	2	Indicates the VLAN ID of the BPDU. This field is used to identify the VLAN for which the BPDU is being sent.
Priority	2	Indicates the priority of the BPDU for the given VLAN. This field is used to determine the root bridge for the given VLAN.
Remaining Hops	1	Indicates the remaining number of hops to reach the root bridge. This field is used to determine the shortest path to the root bridge for the given VLAN.
Message Age	2	Indicates the age of the BPDU in tenths of a second for the VLAN. This field is used to determine the freshness of the BPDU for the given VLAN.

communication-specific Field	Size (bytes)	Description
Root ID	8	The ID of the root bridge in the network. This field is used to determine the root bridge for the STP instance.
Root Path Cost	4	The total cost of the path to the root bridge. This field is used to determine the shortest path to the root bridge for the STP instance.
Bridge ID	8	The ID of the sending bridge. This field is used to identify the sender of the BPDU.
Port ID	2	The ID of the sending bridge's port. This field is used to identify the port on the sending bridge that is being used for the STP instance.

- In STP, the "**cost**" is a metric used to determine the preferred path for data frames to travel between switches.
    - The cost is based on the **speed** of the link and is **calculated** as a function of the link's **bandwidth**. The higher the bandwidth, the lower the cost.

Link speed	Cost
10 Mbps	100
100 Mbps	19
1 Gbps	4
10 Gbps	2

- then **assigns port roles** and **states**:
- **Port:**
    - **Designated ports** are in forwarding state and are responsible for forwarding frames to non-root bridges.
        - The switch with the lowest bridge **priority** becomes the Root bridge switch for that segment, and the port on that switch becomes the **designated** port.
            - the lowest path cost to the root bridge ,if there are multiple ports with the same lowest path cost the switch with the lowest MAC address on that port is selected as the designated switch
            - OR The designated port is the port with the second lowest MAC address. if all link same speed
        - **Root ports** are in forwarding state and are the ports that provide the shortest path to the root bridge.
            - Each switch in the network determines the shortest path to the root switch by selecting the port with the lowest path cost to the root switch.
            - OR the port **facing** the **designated** port.
        - **Alternative ports** are in blocking state and are used as backups in case the designated or root ports fail.
            - The remaining ports are blocked ports and are named as alternative ports.



STP works by electing a root bridge based on bridge priority and MAC address. The bridge with the lowest priority and MAC address becomes the root bridge.

  - **States:**
    - States: **disabled** → **listening** → **learning** → **forwarding** $||$ **blocking**
      - **Disabled**: initial state, no frames are sent or received.
      - **Listening**: all devices or ports listen to the election process of the root bridge, devices are informed of the root bridge.
      - **Learning**: all ports learn their role as designated port, root port, or alternative port.
      - **Forwarding**: root and designated ports change their state to forwarding.
      - **Blocking**: ports that are not root or designated ports are disabled to prevent loops
    - Takes 50 seconds to complete
    - Learning requires 15 seconds, forwarding also requires 15 seconds, max age takes 20 seconds
      - **Maximum** age allowed for a BPDU (Bridge Protocol Data Unit) to be considered valid
      - If a BPDU is not received within the max age time period, the receiving device assumes that the link or switch has failed and takes appropriate action (e.g. transition to blocking state)
    - Standard is 802.1D
- The election process can take up to 30-50 seconds which is equal to $20 + (\text{Forwarding Delay} = 15) + (\text{Learning Delay} = 15) = 50\text{Seconds}$
- **Rapid STP (RSTP):**
  - Upgraded version of STP
  - States: **discard** → **learning** → **forwarding** → **discard (return)**
    - **discard**
      - In this state, the port does not forward any data frames and only listens to BPDU messages
      - The port does not learn any MAC addresses or participate in STP calculations
    - **Learning**: all ports learn their role as designated port, root port, or alternative port.
    - **Forwarding**: root and designated ports change their state to forwarding, and the other ports return to discard states.
  - Delay is 6 seconds
  - Two additional BPDU messages: **purpose** and **acknowledgement**
    - **Purpose message:** is used to inform neighboring devices of a port's role and status to improve convergence time
    - **Acknowledgement message:** is used to confirm that a purpose message has been received to improve reliability
  - Standard is 802.1W
- **Per-VLAN STP (PVST):** allows each VLAN to have its own root bridge and election process, improving network performance and stability.
  - Performance is slow
  - Election time is 30-50 seconds
- **Rapid PVST+ (RPVST+):** is an even faster version of PVST.
  - Faster election time, only 6 seconds
- **Multiple Spanning Tree Protocol (MST):** is another version of STP that allows multiple VLANs to be mapped to a single STP instance, reducing the number of STP instances required in a network.
  - Upgrade of PVST
  - Elects root bridge for group of VLANs
  - Standard is 802.1s
- **STP port types:**
  - **Edge port:** port for edge user such as a PC, not involved in STP
  - **Port fast:** assigns designated port automatically, not counted in election process
    - **Advantages of port fast:**
      - Faster connectivity without waiting for election process
      - Useful for customers with VIP needs and minimal latency
    - **Disadvantages of port fast:**
      - Potential for loops if not configured correctly
- The "**show spanning-tree**" command is a network command used to display the current Spanning Tree Protocol (STP) configuration and status on a switch.
  - including the **root bridge**, **root port**, **designated port**, and **blocked port**.

Lab3:

- In Lab3, three switches are connected to each other to create a closed loop, and STP is applied to prevent loops and ensure network stability.

```
graph LR
    S1 --- S2
    S2 --- S3
    S3 --- S1
```

- Drag and drop three switches from the "**Switches and Hubs**" section of the "**Devices**" panel onto the workspace.
- Connect the switches to each other to create a closed loop.
- STP is enabled by default, so there is no need to enable it manually.

  ```
  Switch> enable
  Switch# show spanning-tree
  ```

  Syntax:

Lab4 :

- In Lab4, two switches are connected to each other using two cables, and STP is applied to prevent loops and ensure network stability.

  ```
  graph LR
      S1 --- S2
      S1 --- S2
  ```

  1. Drag and drop two switches from the **"Switches and Hubs"** section of the **"Devices"** panel onto the workspace.
  2. Connect the switches to each other using **two cables**, one on each switch.
  3. STP is enabled by default, so there is no need to enable it manually.
  4. In a **point**-**to**-**point topology**, the higher-numbered port states as alternative port, and its election time is reduced because the connection is considered "point-to-point." This ensures quick availability of the alternative port in case the primary port fails, while preventing network instability and loops.

     ```
     Switch> enable
     Switch# show spanning-tree
     ```

     Syntax:

Cisco Discovery Protocol (CDP):

- **Cisco Discovery Protocol** (CDP) and **Link Layer Discovery Protocol** (LLDP) are used in networking to discover neighboring devices.
- **CDP** is a proprietary protocol used by Cisco devices, while **LLDP** is a vendor-neutral protocol (IEEE) used by many networking devices.
- Both CDP and LLDP allow **devices** to **exchange detailed** information about their neighbors, such as:
  - The device's identity and type
  - The interface or port that connects the devices
  - The IP address of the neighbor device
  - The MAC address of the neighbor device
  - A description of the neighbor device (if available)
- This information can be used to map out the topology of a network and troubleshoot connectivity issues.
- While CDP is only supported on Cisco devices, LLDP is supported by many vendors and can be used to discover neighboring non-Cisco devices as well.
- CDP and LLDP are both enabled by default on many networking devices, but can be disabled or configured as needed.

Lab5:

Create a network topology with three switches connected in a linear topology. Use Cisco Discovery Protocol (CDP) to discover neighboring switches, and set a description for one of the switches. Verify the results using appropriate commands in the command line interface.

```
graph LR
    S1 --- S2
    S2 --- S3
```

1. Drag and drop three switches from the **"Switches and Hubs"** section of the **"Devices"** panel onto the workspace.
2. Connect the switches to each other using cables in a linear topology (not a closed loop).
3. Select one of the switches and double-click it to open its configuration panel.
4. In the configuration panel, click on the **"CLI"** tab to access the switch's command line interface.
5. Type the following commands to set the description for the switch:

   ```
   Switch> enable
   Switch# configure terminal
   Switch(config)# interface fa0/1
   Switch(config-if)# description Connected_to_Switch0
   ```

   Syntax:

6. CDP is enabled by default, so there is no need to enable it manually.

   ```
   Switch> enable
   Switch# show cdp neighbors
   ```

Syntax:

```
Switch> enable
Switch# show cdp neighbors details
```

Syntax:

Lab6:

Create a network topology with three switches connected in a linear topology. Use Link Layer Discovery Protocol (LLDP) to discover neighboring switches, and set a description for one of the switches. Verify the results using appropriate commands in the command line interface.

```
graph LR
    S1 --- S2
    S2 --- S3
```

1. Drag and drop three switches from the "Switches and Hubs" section of the "Devices" panel onto the workspace.
2. Connect the switches to each other using cables in a linear topology (not a closed loop).
3. Select one of the switches and double-click it to open its configuration panel.
4. In the configuration panel, click on the "CLI" tab to access the switch's command line interface.
5. Type the following command to enable LLDP on the each switch:

   ```
   Switch> enable
   Switch# configure terminal
   Switch(config)# lldp run
   ```

   Syntax:

6. Type the following commands to set the description for the switch:

   ```
   Switch> enable
   Switch# configure terminal
   Switch(config)# interface fa0/1
   Switch(config-if)# description Connected_to_Switch0
   ```

   Syntax:

   ```
   Switch> enable
   Switch# show lldp neighbors
   ```

   Syntax:

   ```
   Switch> enable
   Switch# show lldp neighbors details
   ```

   Syntax:

Link Aggregation Control Protocol (LACP):

- In some cases, the bandwidth of a single network interface may not be sufficient for a particular network application or workload.
  - But Increasing the number of links between switches can increase the available bandwidth, but can also lead to problems such as broadcast storms. Spanning Tree Protocol (STP) can be used to prevent loops and ensure network stability, but it also disables all but one link between switches, which can limit available bandwidth.
- **Link Aggregation Control Protocol** (LACP) can be used to combine multiple physical interfaces into a single logical interface, providing greater bandwidth and redundancy for network traffic. and named as **port-Channel.**
- In Cisco switches, LACP can combine between **2** to **8** links.
- In some models of switches, the links to be combined using LACP must be **organized** in a **specific order** (e.g., 1,2,3,4,5), **while** in others the order **does not** matter.
- LACP works by **negotiating** between two devices and combining multiple interfaces into a single logical link, which appears as a single interface to the network.
- LACP supports **two** device roles:
  - **Active**: The device **actively** negotiates with the other device to form an LACP link.
  - **Passive**: The device **waits** for the other device to initiate the LACP negotiation before forming an LACP link.
- Both devices involved in the LACP negotiation must support LACP, and at least one of the devices must be configured as an **"active"** participant in the negotiation.
- LACP supports load balancing mechanisms such as "src-dst-mac" to distribute traffic across the bundled interfaces and improve network performance.
- LACP can be used in both Layer 2 (switches) and Layer 3 (routers) networks, but in Layer 3 networks, there is no need to negotiate between devices or assign device roles.

- **scenarios**:
  - In a scenario where **one** switch is **active** and the other is **passive**, the passive switch will only form an LACP link if the active switch initiates the negotiation.
  - In a scenario where **both** switches are active, **both** devices will **negotiate** to form an LACP link.
- LACP provides benefits such as increased bandwidth, redundancy, and improved network performance, making it a popular choice for high-performance network applications.

Lab7:

A network topology with three switches where Switch 1 is connected to Switch 2 with two links and Switch 3 is connected to Switch 1 with one link. You will then use LACP to bundle both links between Switch 1 and Switch 2 to increase the available bandwidth and improve network performance. Finally, you will verify the LACP configuration using appropriate commands to confirm that the LACP bundle is formed successfully.

```
graph LR
    S1 --- S2
    S1 --- S2
    S1 --- S3
```

1. Drag and drop three switches from the "Switches and Hubs" section onto the workspace.
2. Connect the switches as follows:
   - Connect Switch 1 to Switch 2 with two links.
   - Connect Switch 1 to Switch 3 with one link.
3. onfigure the switch ports as follows:
   - On Switch 1, configure both ports connected to Switch 2 as LACP active ports.

     ```
     Switch> enable
     Switch(config)# interface range GigabitEthernet1/0/1-2
     Switch(config-if-range)# channel-group 1 mode active
     ```

     Syntax:

   - On Switch 2, configure both ports connected to Switch 1 as LACP passive ports.

     ```
     Switch> enable
     Switch(config)# interface range GigabitEthernet1/0/1-2
     Switch(config-if-range)# channel-group 1 mode passive
     ```

     Syntax:

   - On Switch 3, configure the port connected to Switch 1 as a regular access port.
     - no need to configure since there is an exist default vlan as 1
4. Verify the LACP configuration using the appropriate commands in the command line interface:

   ```
   Switch> enable
   Switch# show interfaces port-channel
   ```

   Syntax:

   ```
   Switch> enable
   Switch# show etherchannel summary
   ```

   Syntax:

Layer 3 IP CONNECTIVITY (Network layer):

Routing:

- As a **router**, one of the main tasks is to **separate broadcast** domains.
- When a packet is received by a router, it stops at the interface where it was received.
- The forwarding **decision determines** how the packet will be forwarded or routed.
- The **routing table** is a key component of a router's operation. It is a database that contains information about the **best** path for **forwarding** packets to their **destination**.
  - The routing table is **organized** as a list of entries, with each entry containing the following **information**:
    - **Destination network or host**: This is the **IP address** of the **destination** network or host that the **entry applies** to.
    - **Subnet mask:** This is the **subnet mask** that defines the **range of IP addresses** covered by the entry.
    - **Next-hop IP address or egress interface:** This is the **IP address** of the **next-hop** router that should be used to **forward** packets to the **destination** network or host, or the **egress interface** that should be used to **directly reach** the **destination** network or host.

        - **The administrative distance** (AD) value is a way for routers to determine which routing protocol to trust more when they receive routing information from multiple sources. A lower AD value indicates a higher degree of trustworthiness, meaning that the information provided by the protocol with the lower AD value is considered more reliable and accurate.
        - **Metric**: This is a **measure** of the **cost** or **distance** associated with the path to the **destination** network or host. Entries with **lower** metrics are considered **better** paths.

Destination	Subnet Mask	Next Hop	Admin Distance	Metric
192.168.1.0	255.255.255.0	10.0.0.1	110	1
192.168.2.0	255.255.255.0	10.0.0.2	110	1
192.168.3.0	255.255.255.0	10.0.0.1	110	2
192.168.4.0	255.255.255.0	10.0.0.2	110	2
0.0.0.0	0.0.0.0	10.0.0.3	1	1

- The forwarding **decision** involves several steps, **including**:
    1. Checking the **longest match** for the **prefix** of the packet's destination address.
        - The longest match refers to the process of comparing a packet's destination IP address to the entries in a router's routing table to determine the best route for the packet.
        - meaning the route that shares the most significant bits with the destination IP address.
        - **Example**
            - Suppose a router receives a packet with a destination IP address of $192.168.1.50$. The router checks its routing table for a matching entry and finds two possible routes:
            - **Route 1:** $192.168.1.0/24$ via interface **1**
            - **Route 2:** $192.168.1.0/25$ via interface **2**
            - Both routes have the **same prefix** of $192.168.1.0$, but they have different subnet masks. **Route 1** has a subnet mask of $255.255.255.0(/24)$ which means that it **covers** a range of **IP** addresses from $192.168.1.0$ to $192.168.1.255$. Route 2 has a subnet mask of $255.255.255.128(/25)$ which means that it **covers** a **range** of IP addresses from $192.168.1.0$ to $192.168.1.127$.
            - To determine the **best** route for the packet, the router uses the longest match rule and selects the route with the longest matching prefix. In this case, **Route 2** has the longer prefix (**25** bits compared to **24** bits for Route 1), so the router selects Route 2 as the **best** route for the packet. The packet is then forwarded out of **interface 2** towards its destination at 192.168.1.50.
    2. Deciding which **routing protocol** should handle the forwarding of the packet.
        - In a network, a **protocol** is a set of rules and procedures that govern the exchange of data between devices.
            - A protocol **defines how** data is transmitted, received, and processed by devices on the network.
        - There are **two** types of routing: **static** and **dynamic**.
            - **Static** routing is a manual process in which the network administrator manually configures the routing table entries on each router.
            - **Dynamic** routing is an automated process in which routers exchange information with each other to dynamically update their routing tables.
                - Dynamic routing protocols include **distance vector** and **link state** protocols.
                    - **Distance vector** protocols use a simple algorithm to determine the best path for forwarding packets based on **distance** metrics such as hop count.
                        - **Examples** of distance vector protocols include **RIP** and **EIGRP**.
                    - **Link state** protocols use a more complex algorithm to determine the best path for forwarding packets based on a variety of factors, including the **bandwidth** and **reliability** of network links.
                    - **Examples** of link state protocols include **OSPF** and **IS-IS**.
    3. Submitting the packet to the desired routing protocol, which will apply its own rules (metrics) to determine the best path for forwarding the packet.
        - **The administrative distance** (AD) value is a way for routers to determine which routing protocol to trust more when they receive routing information from multiple sources. A lower AD value indicates a higher degree of trustworthiness, meaning that the information provided by the protocol with the lower AD value is considered more reliable and accurate.

Routing Protocol	Administrative Distance
Connected	0
Static	1
eBGP	20
EIGRP internal	90
IGRP	100
OSPF	110
RIP	120
EIGRP external	170
BGP	200

- The routing protocol used by the router depends on the network **environment** and the **preferences** of the network administrator.
- A **default gateway** is a network device that serves as the **default** path for network **traffic** that is destined for a location **outside** of the local network. In other words, it i**s the IP address of the router** on the **local** network that connects to the wider **internet** or **another** network.

Static Route:

- A **static** route is a manual method of routing a **specific** packet to a specific **route**.
- It is the only method of **manually** configuring a **routing table** in which the network administrator **manually** enters the path a packet should take to reach its **destination**.
- **Unlike** dynamic routing protocols, static routes **do not have metrics** associated with them. Instead, the administrator manually specifies the next-hop IP address or egress interface for a given destination network or host.

- A static route can be used to **route traffic** to a s**pecific next-hop IP** address or to a **specific egress interface** port ID.
- Static routes are available for both **IPv4** and **IPv6** networks.
- A static route can be used to route traffic to a **single host** or an **entire network**.
- There are **two flavors** of static routes:
  - **Default route:** A default route is used to route **all traffic** for which there is **no specific route** in the **routing table**. It is sometimes referred to as the "**gateway of last resort**". A default route can be used to route traffic to a **specific next-hop IP address** or to a **specific egress interface port ID**.
  - **Floating static route:** A floating static route is a **backup** route that is used when the **primary** route is **unavailable**. It has a **higher administrative distance** than the **primary** route, which means that it is **normally not** used, but is available as a **backup** if the primary route **fails**.

Lab8:

Configure static routes to enable communication between two PCs connected to different subnets that are separated by two routers.

There are two PCs and two routers in a network. PC1 is connected to Router1, which is then connected to Router2. PC2 is connected directly to Router2. The IP address range between PC1 and Router1 is 192.168.1.0/24, between Router1 and Router2 is 10.0.0.0/24, and between Router2 and PC2 is 172.16.0.0/24.

```
graph LR
    A[PC1]
    B[Router1]
    C[Router2]
    D[PC2]

    A -->|192.168.1.0/24| B
    B -->|10.0.0.0/24| C
    C -->|172.16.0.0/24| D
```

- **Steps:**
  - PC1 $(192.168.1.2/24)$is connected to Router1 $(192.168.1.1/24)$ on interface $\mathrm{GigabitEthernet}0/0$.
  - Router1 $(192.168.1.1/24)$ is connected to Router2 $(10.0.0.2/24)$ on interface $\mathrm{GigabitEthernet}0/1$.
  - Router2 $(10.0.0.1/24)$ is connected to PC2 $(172.16.0.2/24)$ on interface $\mathrm{GigabitEthernet}0/0$.

  1. Add the devices to the workspace: two **PCs**, two **routers**, and four **Ethernet cables**.
  2. Connect the devices according to the **topology** described above.
  3. Configure the IP addresses on each device as **follows:**
     - **PC1:**
       - **IP address:** $192.168.1.2$,
       - **subnet mask:** $255.255.255.0$
       - **default gateway:** $192.168.1.1$.
     - **Router1**:

       ```
       Router> enable
       Router# configure terminal
       Router(config)# interface GigabitEthernet0/0
       Router(config-if)# ip address 192.168.1.1 255.255.255.0
       Router(config-if)# no shutdown
       Router(config-if)# exit
       Router(config)# interface GigabitEthernet0/1
       Router(config-if)# ip address 10.0.0.1 255.255.255.0
       Router(config-if)# no shutdown
       Router(config-if)# end
       ```

       Syntax:

     - **Router2:**

       ```
       Router> enable
       Router# configure terminal
       Router(config)# interface GigabitEthernet0/0
       Router(config-if)# ip address 10.0.0.2 255.255.255.0
       Router(config-if)# no shutdown
       Router(config-if)# exit
       Router(config)# interface GigabitEthernet0/1
       Router(config-if)# ip address 172.16.0.1 255.255.255.0
       Router(config-if)# no shutdown
       Router(config-if)# end
       ```

       Syntax:

     - **PC2:**
       - **IP address:** $172.16.0.2$
       - **subnet mask:** $255.255.255.0$

    - **default gateway:** $172.16.0.1$.

4. Test connectivity by **pinging** from **PC1 to PC2**. The ping should **fail** since the routers do not have routes to each other's subnets.

   ```
   Cisco Packet Tracer PC Command Line 1.0
   C:\>ping 172.16.0.2
   ```

   Syntax:

5. Configure **static** routes on **Router1** and **Router2** to enable communication between the two subnets:

   - **On Router1:**

     ```
     Router> enable
     Router# configure terminal
     Router(config)# ip route 172.16.0.0 255.255.255.0 10.0.0.2
     ```

     Syntax:

   - **On Router2:**

     ```
     Router> enable
     Router# configure terminal
     Router(config)# ip route 192.168.1.0 255.255.255.0 10.0.0.1
     ```

     Syntax:

6. Test connectivity again by pinging from PC1 to PC2. The ping should succeed, indicating that the static routes are working correctly.

   ```
   Cisco Packet Tracer PC Command Line 1.0
   C:\>ping 172.16.0.2
   ```

   Syntax:

7. **verify** that the **routing** tables on both Router1 and Router2 have been **updated** with the correct routes. We can use the following **commands** to view the interface information and routing tables:

   ```
   Router> enable
   Router# show ip interface brief
   ```

   Syntax:

   ```
   Router> enable
   Router# show ip route
   ```

   Syntax:

Lab9:

```
graph LR
  PC1((PC1)) -- 1.1.1.0/24 --> MLS1((MLS1))
  MLS1 -- 2.2.2.0/30 --> MLS2((MLS2))
  MLS1 -- 3.3.3.0/30 --> MLS3((MLS3))
  MLS2 -- 4.4.4.0/30 --> MLS4((MLS4))
  MLS3 -- 5.5.5.0/30 --> MLS4
  MLS4 -- 6.6.6.0/24 --> PC2((PC2))
```

Lab Scenario:

- Two PCs: PC1 and PC2
- Four Multi-Layer Switches (MLS1, MLS2, MLS3, MLS4)
- PC1 is connected to MLS1 with IP address $1.1.1.1/24$
- MLS1 is connected to MLS2 with IP address $2.2.2.1/30$
- MLS1 is connected to MLS3 with IP address $3.3.3.1/30$
- MLS2 is connected to MLS4 with IP address $4.4.4.1/30$
- MLS3 is connected to MLS4 with IP address $5.5.5.1/30$
- MLS4 is connected to PC2 with IP address $6.6.6.1/24$
- Configure a static route on MLS1 to allow PC1 to connect to PC2

Configure the network so that PC1 can communicate with PC2 using a static route on MLS1, and test the connectivity between the two PCs.

Steps:

1. Add two PCs and four Multi-Layer Switches to the topology, and connect them as per the lab scenario.
2. Assign IP addresses to the devices as follows:
   - **PC1:**

  - **IP address:** 1.1.1.1
  - **subnet mask:** 255.255.255.0
  - **default gateway:** 1.1.1.2.
- **MLS1:**

```
Switch> enable
Switch# configure terminal
Switch(config)# interface GigabitEthernet0/1
Switch(config-if)# no switchport
Switch(config-if)# ip address 1.1.1.2 255.255.255.0
Switch(config-if)# exit
Switch(config)# interface GigabitEthernet0/2
Switch(config-if)# no switchport
Switch(config-if)# ip address 2.2.2.2 255.255.255.252
Switch(config-if)# exit
Switch(config)# interface GigabitEthernet0/3
Switch(config-if)# no switchport
Switch(config-if)# ip address 3.3.3.2 255.255.255.252
```

Syntax:

- **MLS2:**

```
Switch> enable
Switch# configure terminal
Switch(config)# interface GigabitEthernet0/1
Switch(config-if)# no switchport
Switch(config-if)# ip address 2.2.2.1 255.255.255.252
Switch(config-if)# exit
Switch(config)# interface GigabitEthernet0/2
Switch(config-if)# no switchport
Switch(config-if)# ip address 4.4.4.2 255.255.255.252
```

Syntax:

- **MLS3:**

```
Switch> enable
Switch# configure terminal
Switch(config)# interface GigabitEthernet0/1
Switch(config-if)# no switchport
Switch(config-if)# ip address 3.3.3.1 255.255.255.252
Switch(config-if)# exit
Switch(config)# interface GigabitEthernet0/2
Switch(config-if)# no switchport
Switch(config-if)# ip address 5.5.5.2 255.255.255.252
```

Syntax:

- **MLS4:**

```
Switch> enable
Switch# configure terminal
Switch(config)# interface GigabitEthernet0/1
Switch(config-if)# no switchport
Switch(config-if)# ip address 4.4.4.1 255.255.255.252
Switch(config-if)# exit
Switch(config)# interface GigabitEthernet0/2
Switch(config-if)# no switchport
Switch(config-if)# ip address 5.5.5.1 255.255.255.252
Switch(config-if)# exit
Switch(config)# interface GigabitEthernet0/3
Switch(config-if)# no switchport
Switch(config-if)# ip address 6.6.6.2 255.255.255.0
```

Syntax:

- **PC2:**
  - **IP address:** 6.6.6.6
  - **subnet mask:** 255.255.255.0
  - **default gateway:** 6.6.6.2.

3. Test the interface by

   ```
   Switch> enable
   Switch# show ip interface brief
   ```

   Syntax:

4. static routes

   - **Enable** IP routing on all the multilayer switches:

     ```
     Switch> enable
     Switch# configure terminal
     Switch(config)# ip routing
     ```

     Syntax:

   - **MLS1** to allow traffic from PC1 to reach PC2 via MLS2:

     ```
     Switch> enable
     Switch# configure terminal
     Switch(config)# ip route 6.6.6.0 255.255.255.0 2.2.2.1
     ```

     Syntax:

   - **MLS2**: one to forward traffic from PC1 to MLS4, and another to forward traffic from MLS1 to PC1:

     ```
     Switch> enable
     Switch# configure terminal
     Switch(config)# ip route 6.6.6.0 255.255.255.0 4.4.4.1
     Switch(config)# ip route 1.1.1.0 255.255.255.0 2.2.2.2
     ```

     Syntax:

   - **MLS4** to allow traffic from MLS2 to reach PC2:

     ```
     Switch> enable
     Switch# configure terminal
     Switch(config)# ip route 6.6.6.0 255.255.255.0 4.4.4.2
     ```

     Syntax:

   - Verify the static routes are working by checking the routing tables on each switch:

     ```
     Router> enable
     Router# show ip route
     ```

     Syntax:

5. If the switches support **floating static routes**, you can configure them to provide redundancy in case of a link failure. For example, on MLS1, you can configure a floating static route to forward traffic to PC2 via MLS3 in case the link between MLS1 and MLS2 fails:

   - **MLS1:**

     ```
     Switch> enable
     Switch# configure terminal
     Switch(config)# ip route 6.6.6.0 255.255.255.0 3.3.3.1 2
     ```

     Syntax:

   - **MLS3:**

     ```
     Switch> enable
     Switch# configure terminal
     Switch(config)# ip route 6.6.6.0 255.255.255.0 5.5.5.1 2
     Switch(config)# ip route 1.1.1.0 255.255.255.0 3.3.3.2
     ```

     Syntax:

   - **MLS4:**

     ```
     Switch> enable
     Switch# configure terminal
     Switch(config)# ip route 1.1.1.0 255.255.255.0 5.5.5.2
     ```

     Syntax:

   - **Verify** the floating static routes are working by checking the routing tables on each switch:

```
Router> enable
Router# show ip route
```

Syntax:

```
Router> enable
Router# show run
```

Syntax:

Lab10:

In this lab exercise, you have been provided with two PCs and two routers. Your goal is to configure the network so that PC1 can communicate with PC2. PC1 is connected to Router1 with the IPv6 address prefix $fec1 :: /64$, Router1 is connected to Router2 with the IPv6 address prefix $2001 :: /64$, and Router2 is connected to PC2 with the IPv6 address prefix $fec2 :: /64$.

You need to configure the IPv6 addresses of all devices, enable IPv6 routing on both routers, and configure static IPv6 routes on Router1 and Router2 to enable communication between PC1 and PC2. Finally, you need to test the connectivity between PC1 and PC2 to ensure that the network is working correctly.

```
graph LR

    PC1((PC1)) -- fec1::/64 --> router1((Router 1))
    router1((Router 1)) -- 2001::/64 --> router2((Router 2))
    router2((Router 2)) -- fec2::/64 --> PC2((PC2))
```

1. Drag and drop two routers and two PCs from the device list onto the workspace.
2. Connect PC1 to Router1 using a straight-through cable, and connect Router1 to Router2 using a crossover cable. Finally, connect Router2 to PC2 using a straight-through cable.
3. Enable **IPv6 routing** on both routers by running the following command in global configuration mode:

   ```
   Router> enable
   Router# configure terminal
   Router(config)# ipv6 unicast-routing
   ```

   Syntax:

4. Configure the IPv6 addresses on the interfaces of both routers by running the following commands in interface configuration mode:
   - **Router1:**

     ```
     Router> enable
     Router# configure terminal
     Router(config)# interface <interface connected to PC1>
     Router(config-if)# ipv6 address fec1::1/64
     Router(config-if)# exit
     Router(config)# interface <interface connected to Router2>
     Router(config-if)# ipv6 address 2001::1/64
     ```

     Syntax:

   - **Router2:**

     ```
     Router> enable
     Router# configure terminal
     Router(config)# interface <interface connected to PC1>
     Router(config-if)# ipv6 address 2001::2/64
     Router(config-if)# exit
     Router(config)# interface <interface connected to Router2>
     Router(config-if)# ipv6 address fec2::1/64
     ```

     Syntax:

5. **Verify** the IPv6 addresses have been **configured** correctly by checking the **status** of the **interfaces**

   ```
   Router> enable
   Router# show ipv6 interface brief
   ```

   Syntax:

6. **Gateway:**
   - **router 1 gateway:**$fec1 :: 1/64$.
   - **router 2 gateway:**$fec2 :: 1/64$**.**
7. **Test** the **connectivity** between PC1 and Router1 by **pinging** the IPv6 address of Router1's interface connected to PC1.

8. Configure **static IPv6 routes** on Router1 and Router2 to enable communication between PC1 and PC2 by running the following commands in global configuration mode:

   - **Router1:**

     ```
     Router> enable
     Router# configure terminal
     Router(config)#ipv6 route fec2::/64 2001::2
     ```

     Syntax:

   - **Router2**

     ```
     Router> enable
     Router# configure terminal
     Router(config)#ipv6 route fec1::/64 2001::1
     ```

     Syntax:

9. Verify the static IPv6 routes have been configured correctly by checking the routing table on each router.

   ```
   Router> enable
   Router# show ipv6 route
   ```

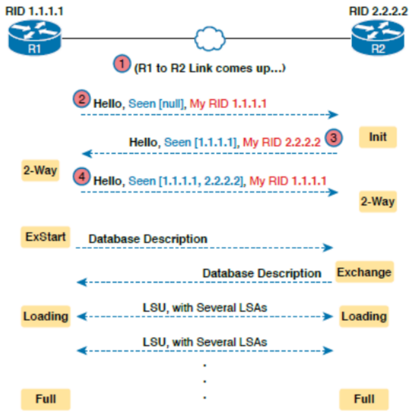
   Syntax:

10. **Test** the **connectivity** between PC1 and PC2 by **pinging** the IPv6 address of the other device.

Open Shortest-Path First (OSPF):

- **OSPF** is a **dynamic routing** protocol that uses the **Dijkstra** algorithm and **SPF** algorithm for **route** decision.
- OSPF uses the **metric** of **cost** to determine the best route, with lower **cost** being **preferred**.
  - **where cost** is calculated as 10^8 divided by the **bandwidth** of the link.
- OSPF has an **administrative distance** of **110**, which means it is preferred over **RIP** but less preferred than **EIGRP** and **BGP**.
- OSPF can have multiple instances, each with its own process ID.
  - The **process ID** is a unique identifier for each OSPF process, which allows multiple **instances** of **OSPF** to run on the **same** router.
    - Each OSPF **process** has its own **link-state database** (LSDB), which **contains** information about the **topology** of the **network**.
  - The **area ID** is a unique identifier for each area in an OSPF network, which allows for the **isolation** of **LSDBs** and the **reduction** of the **amount** of **information** that each router needs to process.
- **Rules:**
  - The **backbone area** (**area 0**) is a special area in OSPF that **connects** all other **areas** in the network.
  - Other areas are **typically** connected to the **backbone** area, either **directly** or **through** other areas.
- OSPF routers negotiate **link-state advertisements** (LSAs) to exchange information about their local network topology.
- The **LSDB** (**Link-State Database**) is a **database** maintained by **each** OSPF router in the network, which contains **information** about the network **topology**. It is built through the **exchange** of Link-State Advertisements (**LSAs**) between OSPF routers, and contains information about the **state** and **cost** of each link, as well as information about the routers and networks in the network.
  - Each router in an OSPF network has a copy of the LSDB for the areas to which it belongs.
  - Because LSDBs contain information about the entire network topology, each router has the same information about the network as every other router.
  - This allows routers to make consistent and accurate routing decisions, even if the topology changes.
- There are three types of LSAs: **LSRequest**, **LSUpdate**, and **LSAcknowledgement**.
  - **LSRequest** is sent by an OSPF router to request information about a specific LSA from another router. This is typically used when a router is missing information about a particular part of the network topology.
  - **LSUpdate** is sent by an OSPF router to advertise changes to the network topology. This can happen when a link goes down, a new link is added, or there are other changes to the network topology.
  - **LSAcknowledgement** is sent by an OSPF router to acknowledge receipt of an LSUpdate or LSRequest message. This is used to confirm that the information has been received and processed correctly.
- Each OSPF router has a **unique Router** ID (RID), which is a **32-bit identifier** used to **identify** the router in the network.
  - The RID can be assigned **manually**, or it can be chosen **automatically** based on the **highest** IP **address assigned interface** on the **router**.
- OSPF routers discover each other through a process called **neighbor discovery**.
  1. When a link between two OSPF routers **comes up**, the routers **send hello** messages to each other to **establish** a **neighbor** relationship.
     - **Hello messages** include information about the sending **router's RID**, **network mask**, and **other** OSPF-related information.
  2. If the receiving router **recognizes** the **RID** in the **hello** message, it will **reply** with its own **hello** message and enter the **Init state**.
     - In the **Init state**, the routers exchange **messages** to **establish** the **direction** of the link and to **agree** on **certain parameters**, such as the **dead** interval and the router **priority**.
  3. The next state is the **2-Way state**, where the routers **establish bidirectional** communication and **exchange** information about **their neighbors**.to establish a **full** neighbor **relationship**
  4. In the **Exstart state**, the routers **exchange Database Description** (DD) packets to compare their **LSDBs** and agree on a **sequence number** for future exchanges.
     - The **DD** packet contains a **summary** of the sender's **LSDB**, including a list of **all LSAs** that it has.

    - The **DD** packet also includes a **sequence number** that is used to keep **track** of the order in which LSAs will be **exchanged**.
    - When a router receives a **DD** packet, it compares the sequence number to its own and determines which router has the **more up-to-date** information.
    - The router with the **more up-to-date** information becomes the **master** and sends a **new DD** packet with a **higher sequence number** to the other router.
5. The **Exchange state** is where the routers actually exchange LSAs to update each other's LSDBs.
6. In the **Loading state**, the routers **send multiple LSAs** to each other to **update** their **LSDBs**.
7. Finally, in the **Full state**, the routers have **synchronized** their **LSDBs** and are ready to **exchange routing updates**.



- **Point-to-Point (P2P) network:**
    - In a **P2P** network, OSPF routers are connected to each other through a dedicated point-to-point link.
    - Because there is **only one** other router on the link, there is **no need for neighbor discovery**, and routers can immediately enter the **2-Way state** and **exchange LSAs**.
        - This is because the **only** router on the other **end** of the link is already **known**.
    - The router with the **highest priority** value (0-255) or **highest Router** ID is elected as the **DR**, and the other router becomes the **BDR**.
    - If there are **only two** routers on the P2P link, there is **no need** for a **DR** or **BDR**.
- **Network connected through a switch:**
    - In a network **connected** through a **switch**, OSPF routers are connected through a **shared** medium, and must perform **additional** steps to **establish neighbor relationships**.
    - When a **new** router is **added** to the **network** or a link between two routers comes up, the routers send **hello packets** to each other.
    - Because the network is **broadcast**, **all** routers on the network will **receive** the **hello** packets.
    - **Elections** must take place to **determine** which router will **become** the **Designated Router** (DR) and which router will become the **Backup Designated Router** (BDR).
    - The **DR** is **responsible** for **updating the** topology database and **forwarding** updates to all **other routers** on the network.
    - The BDR is a **backup** for the DR and takes **over** if the DR **fails**.
    - **Routers** with a **higher priority** value **(0-255)** or a **higher Router ID** have a better chance of becoming the DR or BDR with Second a **higher priority** value **(0-255)** or a **higher Router ID**.
    - If two routers have the same priority, the router with the higher RID will win the election.
    - The election **process takes 40 seconds** to complete.
    - Once the **DR** and **BDR** have been selected, only the DR and BDR will **reach Full state**.
    - Other routers on the network will **only reach the 2-Way state** and are referred to as **DRothers**.

Lab11:

You have two PCs and two routers. PC1 is connected to Router1 on the 10.10.10.0/24 network. Router1 is connected to Router2 on the 10.10.11.0/30 network, and Router2 is connected to PC2 on the 10.10.12.0/24 network. Configure OSPF routing protocol on both routers, and demonstrate connectivity between the two PCs.

```
graph LR

    PC1((PC1)) -- 10.10.10.0/24 --> router1((Router 1))
    router1((Router 1)) -- 10.10.11.0/30 --> router2((Router 2))
    router2((Router 2)) -- 10.10.12.0/24 --> PC2((PC2))
```

- **Steps:**

1. Connect the devices as per the topology shown in the question.
2. Configure the IP addresses and default gateways for PC1 and PC2 as follows:
    - **PC1:**
        - **IP address:** $10.10.10.10$
        - **subnet mask:** $255.255.255.0$
        - **default gateway:** $10.10.10.1.$
    - **PC2:**
        - **IP address:** $10.10.12.10$
        - **subnet mask:** $255.255.255.0$
        - **default gateway:** $10.10.12.1.$
3. Configure the IP addresses on the router interfaces as follows:
    - **Router1:**

```
Router> enable
Router# configure terminal
Router(config)# interface GigabitEthernet0/0
Router(config-if)# ip address 10.10.10.1 255.255.255.0
Router(config-if)# no shutdown
Router(config-if)# exit
Router(config)# interface Serial0/0/0
Router(config-if)# ip address 10.10.11.1 255.255.255.252
Router(config-if)# no shutdown
```

Syntax:

- **Router2:**

```
Router> enable
Router# configure terminal
Router(config)# interface Serial0/0/0
Router(config-if)# ip address 10.10.11.2 255.255.255.252
Router(config-if)# no shutdown
Router(config-if)# exit
Router(config)# interface GigabitEthernet0/0
Router(config-if)# ip address 10.10.12.1 255.255.255.0
Router(config-if)# no shutdown
```

Syntax:

4. Configure OSPF routing protocol on both routers using the following commands:

- **Router1:**

```
Router> enable
Router# configure terminal
Router(config)# router ospf 1
Router(config-router)# network 10.10.10.0 0.0.0.255 area 0
Router(config-router)# network 10.10.11.0 0.0.0.3 area 0
```

Syntax:

- **Router2:**

```
Router> enable
Router# configure terminal
Router(config)# router ospf 1
Router(config-router)# network 10.10.11.0 0.0.0.3 area 0
Router(config-router)# network 10.10.12.0 0.0.0.255 area 0
```

Syntax:

Note that the wildcard masks used in the network statements are 0.0.0.255 and 0.0.0.3, respectively.

Note advertises or distributes routing information between routers by exchanging LSAs.

- A **wildcard mask** is a **32**-bit pattern used in networking to specify a range of IP addresses or a subset of a network. It is the **opposite** of the **subnet mask** and is used with an IP address to **determine** which addresses are **included** or **excluded** in a specific task. The wildcard mask designates which bits in an IP address should be treated as "**don't care**" bits. 0s in the wildcard mask indicate that the corresponding bits in the IP address must match exactly, while 1s **indicate** that they can be either 0 or 1.

5. Verify OSPF neighbor relationships.

```
Router> enable
Router# show ip ospf neighbor
```

Syntax:

6. Verify OSPF routing table entries.

```
Router> enable
Router# show ip route
```

Syntax:

7. Verify OSPF database entries .

```
Router> enable
Router# show ip ospf database
```

Syntax:

8. Test connectivity between the two PCs by **pinging** the IP address of PC2 $(10.10.12.10)$ from PC1. The ping should be successful.

Lab12:

Configure a network topology consisting of two PCs, one router, and five multilayer switches such that PC1 with IP address 1.1.1.10/24 is connected to MLS1 with IP address 1.1.1.1/24, which is connected to Router1 with IP address 2.2.2.2/30. Router1 is connected to MLS5 with IP address 4.4.4.2/30, which is connected to PC2 with IP address 6.6.6.10/24. MLS1 is also connected to MLS2 with IP address 3.3.3.1/30, which is connected to MLS3 with IP address 5.5.5.2/30. MLS3 is connected to MLS4 with IP address 7.7.7.1/30, which is connected to MLS5 with IP address 9.9.9.2/30. Configure OSPF on all the devices so that PC1 can communicate with PC2, and show the commands used for the configuration.

```
graph LR

    PC1((PC1)) -- 1.1.1.0/24 --> mls1((mls1))
    mls1((mls1)) -- 2.2.2.0/30 -->  router1((Router 1))
    mls1((mls1)) -- 3.3.3.0/30 -->  mls2((mls2))
    mls2((mls2)) -- 5.5.5.0/30 -->  mls3((mls3))
    mls3((mls3)) -- 7.7.7.0/30 -->  mls4((mls4))
    mls4((mls4)) -- 9.9.9.0/30 -->  mls5((mls5))
    router1((Router 1)) -- 4.4.4.0/30 --> mls5((mls5))
    mls5((mls5)) -- 6.6.6.0/24 --> PC2((PC2))
```

Steps:

1. Add two PCs, five multilayer switches, and one router to the topology.
2. **Connecting**
    - Connect PC1 to MLS1 using a FastEthernet link.
    - Connect MLS1 to Router1 using another FastEthernet link.
    - Connect Router1 to MLS5 using another FastEthernet link.
    - Connect MLS5 to PC2 using a FastEthernet link.
    - Connect MLS1 to MLS2 using a GigabitEthernet link.
    - Connect MLS2 to MLS3 using another GigabitEthernet link.
    - Connect MLS3 to MLS4 using another GigabitEthernet link.
    - Connect MLS4 to MLS5 using another GigabitEthernet link.
3. Configure the IP addresses and default gateways for PC1 and PC2 using the following commands:
    - **For PC1:**
        - **ip address:**$1.1.1.10$
        - **Subnet Mask:** $255.255.255.0$
        - **default-gateway:**$1.1.1.1$
    - **For PC2:**
        - **ip address:** $6.6.6.10$
        - **Subnet Mask:** $255.255.255.0$
        - **default-gateway:** $6.6.6.1$
4. Configure the interfaces on MLS1, Router1, MLS5, MLS2, MLS3, and MLS4.
    - **On MLS1:**

```
Switch> enable
Switch# configure terminal
Switch(config)# interface FastEthernet0/1
Switch(config-if)# no switchport
Switch(config-if)# ip address 1.1.1.1 255.255.255.0
Switch(config-if)# exit
Switch(config)# interface FastEthernet0/2
Switch(config-if)# no switchport
Switch(config-if)# ip address 2.2.2.2 255.255.255.252
Switch(config-if)# exit
Switch(config)# interface GigabitEthernet0/1
Switch(config-if)# no switchport
Switch(config-if)# ip address 3.3.3.1 255.255.255.252
```

Syntax:

- **On Router 1:**

```
Router> enable
Router# configure terminal
Router(config)# interface FastEthernet0/0
Router(config-if)# ip address 2.2.2.1 255.255.255.252
Router(config-if)# exit
Router(config)# interface FastEthernet0/1
Router(config-if)# ip address 4.4.4.1 255.255.255.252
```

Syntax:

- **On MLS5:**

```
Switch> enable
Switch# configure terminal
Switch(config)# interface FastEthernet0/1
Switch(config-if)# no switchport
Switch(config-if)# ip address 4.4.4.2 255.255.255.252
Switch(config-if)# exit
Switch(config)# interface GigabitEthernet0/1
Switch(config-if)# no switchport
Switch(config-if)# ip address 9.9.9.2 255.255.255.252
Switch(config-if)# exit
Switch(config)# interface FastEthernet0/2
Switch(config-if)# no switchport
Switch(config-if)# ip address 6.6.6.1 255.255.255.0
```

Syntax:

- **On MLS2:**

```
Switch> enable
Switch# configure terminal
Switch(config)# interface GigabitEthernet0/1
Switch(config-if)# no switchport
Switch(config-if)# ip address 3.3.3.2 255.255.255.252
Switch(config-if)# exit
Switch(config)# interface GigabitEthernet0/2
Switch(config-if)# no switchport
Switch(config-if)# ip address 5.5.5.1 255.255.255.252
```

Syntax:

- **On MLS3:**

```
Switch> enable
Switch# configure terminal
Switch(config)# interface GigabitEthernet0/1
Switch(config-if)# no switchport
Switch(config-if)# ip address 5.5.5.2 255.255.255.252
Switch(config-if)# exit
Switch(config)# interface GigabitEthernet0/2
Switch(config-if)# no switchport
Switch(config-if)# ip address 7.7.7.1 255.255.255.252
```

Syntax:

- **On MLS4:**

```
Switch> enable
Switch# configure terminal
Switch(config)# interface GigabitEthernet0/1
Switch(config-if)# no switchport
Switch(config-if)# ip address 7.7.7.2 255.255.255.252
Switch(config-if)# exit
Switch(config)# interface GigabitEthernet0/2
Switch(config-if)# no switchport
Switch(config-if)# ip address 9.9.9.1 255.255.255.252
```

Syntax:

5. Enable IP routing on **MLS1**, **MLS5**, **MLS2**, **MLS3**, and **MLS4 .**

```
Switch> enable
Switch# configure terminal
Switch(config)# ip routing
```

Syntax:

6. Configure OSPF on each device using the following commands:
   - **For MLS1:**

```
Switch> enable
Switch# configure terminal
Switch(config)# router ospf 1
Switch(config-router)# network 1.1.1.0 0.0.0.255 area 0
Switch(config-router)# network 2.2.2.0 0.0.0.3 area 0
Switch(config-router)# network 3.3.3.0 0.0.0.3 area 0
```

Syntax:

- **For Router1:**

```
Router> enable
Router# configure terminal
Router(config)# router ospf 1
Router(config-router)# network 2.2.2.0 0.0.0.3 area 0
Router(config-router)# network 4.4.4.0 0.0.0.3 area 0
```

Syntax:

- **For MLS5:**

```
Switch> enable
Switch# configure terminal
Switch(config)# router ospf 1
Switch(config-router)# network 4.4.4.0 0.0.0.3 area 0
Switch(config-router)# network 6.6.6.0 0.0.0.255 area 0
Switch(config-router)# network 9.9.9.0 0.0.0.3 area 0
```

Syntax:

- **For MLS2:**

```
Switch> enable
Switch# configure terminal
Switch(config)# router ospf 1
Switch(config-router)# network 3.3.3.0 0.0.0.3 area 0
Switch(config-router)# network 5.5.5.0 0.0.0.3 area 0
```

Syntax:

- **For MLS3:**

```
Switch> enable
Switch# configure terminal
Switch(config)# router ospf 1
Switch(config-router)# network 5.5.5.0 0.0.0.3 area 0
Switch(config-router)# network 7.7.7.0 0.0.0.3 area 0
```

Syntax:

- **For MLS4:**

```
Switch> enable
Switch# configure terminal
Switch(config)# router ospf 1
Switch(config-router)# network 7.7.7.0 0.0.0.3 area 0
Switch(config-router)# network 9.9.9.0 0.0.0.3 area 0
```

Syntax:

7. **Verify** the OSPF configuration using the following commands:

- For each device, use the `show ip ospf neighbor` command to check if the device is able to **establish** OSPF neighbor relationships with its neighboring routers.

```
Switch> enable
Switch# show ip ospf neighbor
```

Syntax:

- From PC1, use the `ping 6.6.6.10` command to check if PC1 is **able** to **communicate** with PC2.

```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 6.6.6.10
```

Syntax:

- From any device, use the `show ip route` command to **check** the OSPF **routing table** and confirm that all the networks are reachable.

  ```
  Switch> enable
  Switch# show ip route
  ```

  Syntax:

IP SERVICES:

DHCP (Dynamic Host Configuration Protocol):

- **Provides** a **dynamic** and automatic method to **assign IP addresses**, **subnet masks**, **gateways (**if there is router), and **DNS** information to **devices** on a network.
- Devices are assigned information for a specific amount of time, **typically 24** hours by default, and will **renew** their assignment **when 50**% and **87.5**% of the assignment time has **elapsed**.
  - **DHCP lease time** determines how **long** a client can use an assigned IP address **before** it must be **renewed**.
- DHCP can be **configured** on **routers** or **multilayer switches** that act as DHCP servers.
- DHCP **servers** can be configured with a **pool** of **available** IP addresses, and the server will **assign** an IP address from the **pool** to each client that **requests** one.
- **Advantages of DHCP:**
  - Simplifies network administration
  - Reduces IP address conflicts
  - Supports mobility
  - Facilitates centralized management
  - Enables dynamic IP address allocation
- **Unicast Message:**
  - A message that is sent from one **sender** to one **receiver (point-to-point)**. In the context of DHCP, the DHCP **offer** and **ack** messages are sent as unicast messages from the DHCP server to the client.
- **DHCP Process:**
  - When a client connects to the network, it sends a **DHCP discover messag**e as a **broadcast** to look for a DHCP server.
  - The DHCP server **responds** with a **DHCP offer message**, which is sent as a **unicast** to the client. The offer includes the **assigned IP address**, **subnet** mask, **gateway**, and **DNS** information.
  - The client sends a **DHCP request message** to the **DHCP** server to **accept** the offer.
  - The DHCP server sends a **DHCP ack message** to the client to **confirm** the **request** and **assign** the IP address. The ack message is also sent as a **unicast**.
- If the first router/gateway is not a DHCP server, a DHCP relay (also known as a helper address) can be used to redirect the broadcast message to the correct DHCP server.
  - **DHCP relay** is used when a network has **multiple subnets** and the DHCP server is **located** on a **different** subnet than the clients.
  - When a client sends a DHCP request, it sends a **broadcast** message to the **local** network (subnet) asking for an **IP** address.
  - If the DHCP server is not located on the same subnet, the broadcast message will not **reach** the server.
  - This is where DHCP relay comes in. A DHCP relay agent (which is typically a router) is configured with an IP address for the DHCP server, which is known as the `"helper address"`.
  - When the DHCP relay agent **receives** a broadcast message from a client, it **forwards** the message to the DHCP server using the **helper address**.
  - The DHCP server receives the message and **responds** with an IP address that is **valid** for the subnet where the client is **located**.
  - The DHCP relay agent then **forwards** the **response** back to the client.
  - DHCP relay is useful because it allows a **single** DHCP server to **serve multiple** subnets, which can simplify network management and **reduce** the number of DHCP servers needed.

Lab13:

In Lab13, you are required to configure DHCP for a network consisting of one router, one switch, and four PCs. The PCs are connected to the switch, which in turn is connected to the router with an IP address of 192.168.10.0/24. You need to configure the router to provide DHCP service to the PCs so that they can obtain IP addresses dynamically.

```
graph LR

    PC1((PC1)) --> switch1((switch1))
    PC2((PC2)) --> switch1((switch1))
    PC3((PC3)) --> switch1((switch1))
    PC4((PC4)) --> switch1((switch1))
    switch1((switch1)) --192.168.10.0/24--> router1((router1))
```

Steps

1. Drag a router and a switch onto the workspace.
2. Connect the switch to the router using a straight-through Ethernet cable.
3. Drag four PCs onto the workspace and connect them to the switch using straight-through Ethernet cables.
4. Configure the router interface with an IP address and **enable** it:

```
Router> enable
Router# configure terminal
Router(config)# interface GigabitEthernet0/0
Router(config-if)# ip address 192.168.10.1 255.255.255.0
Router(config-if)# no shutdown
```

Syntax:

5. Configure **DHCP** on the **router**:

```
Router> enable
Router# configure terminal
Router(config)# ip dhcp pool lab13
Router(dhcp-config)# network 192.168.10.0 255.255.255.0
Router(dhcp-config)# default-router 192.168.10.1
Router(dhcp-config)# dns-server 8.8.8.8
Router(dhcp-config)# exit
Router(config)# ip dhcp excluded-address 192.168.10.1
```

Syntax:

6. Verify the DHCP bindings by running the `show ip dhcp binding` command:

```
Router> enable
Router# show ip dhcp binding
```

Syntax:

7. **Test** the DHCP configuration by **changing** the IP addresses of the PCs to **dynamic**:
   - Click on a PC to select it.
   - Click on the "Config" tab in the right-hand pane.
   - Change the IP address to "DHCP".
   - Click "OK" to save the changes.
   - Repeat for each PC.

Lab14:

In Lab 14, you are asked to configure a network consisting of one PC, two switches, and two routers. The PC is connected to Switch 1, which in turn is connected to Router 1. Router 1 is configured with an IP address of 172.16.10/24 on its interface that is connected to Switch 1. Router 1 is also connected to Switch 2 via another interface with an IP address of 10.10.10.0/24. Switch 2 is then connected to Router 2 with the same IP address range of 10.10.10.0/24 on its interface that is connected to Router 2.

```
graph LR

    PC1((PC1)) --> switch1((switch1))
    switch1((switch1)) --172.16.10.0/24--> router1((router1))
    router1((router1)) --10.10.10.0/24--> switch2((switch2))
    switch2((switch2)) --10.10.10.0/24--> router2((router2))
```

Steps:

1. Drag two routers and two switches onto the workspace.
2. Connect the switches to the routers using straight-through Ethernet cables.
3. Connect PC1 to Switch1 using a straight-through Ethernet cable.
4. Configure **Router2** with an IP **address** and enable it:

```
Router> enable
Router# configure terminal
Router(config)# interface GigabitEthernet0/0
Router(config-if)# ip address 10.10.10.2 255.255.255.0
Router(config-if)# no shutdown
```

Syntax:

5. Configure **DHCP** on **Router2**:

```
Router> enable
Router# configure terminal
Router(config)# ip dhcp pool lab14
Router(dhcp-config)# network 172.16.10.0 255.255.255.0
Router(dhcp-config)# default-router 172.16.10.1
Router(dhcp-config)# dns-server 8.8.8.8
Router(dhcp-config)# exit
Router(config)# ip dhcp excluded-address 172.16.10.1
```

Syntax:

6. Configure **OSPF** on **Router2**:

```
Router> enable
Router# configure terminal
Router(config)# router ospf 1
Router(config-router)# network 10.10.10.0 0.0.0.255 area 0
```

Syntax:

7. Configure **Router1** with an IP **address** and enable it:

```
Router> enable
Router# configure terminal
Router(config)# interface GigabitEthernet0/0
Router(config-if)# ip address 172.16.10.1 255.255.255.0
Router(config-if)# no shutdown
```

Syntax:

8. Connect **Router1** to **Switch2** using a straight-through Ethernet cable and configure the interface:

```
Router> enable
Router# configure terminal
Router(config)# interface GigabitEthernet0/1
Router(config-if)# ip address 10.10.10.1 255.255.255.0
Router(config-if)# no shutdown
```

Syntax:

9. Configure **OSPF** on **Router1**:

```
Router> enable
Router# configure terminal
Router(config)# router ospf 1
Router(config-router)# network 10.10.10.0 0.0.0.255 area 0
Router(config-router)# network 172.16.10.0 0.0.0.255 area 0
```

Syntax:

10. Configure **Router1** to forward **DHCP** requests to **Router2**:

```
Router> enable
Router# configure terminal
Router(config)# interface GigabitEthernet0/1
Router(config-if)# ip helper-address 10.10.10.2
```

Syntax:

DNS (Domain Name System):

- Resolves URLs to IP addresses and vice versa.
    - **A URL (Uniform Resource Locator)** is a web address that identifies the location of a resource on the internet
    - **Structure:** scheme://host:port/path?query_parameters#fragment

Component	Description
Scheme	The protocol used to access the resource (e.g., HTTP, HTTPS)
Host	The domain name or IP address of the server hosting the resource
Port	The network port on the server that the client will use to access the resource (optional)
Path	The location of the resource on the server
Query parameters	Additional information passed to the server about the resource being requested (optional)
Fragment	A specific part of the resource being requested (optional)

- Works on UDP port 53.
- Reverse DNS also exists to resolve IP addresses to domain names.
- DNS is essential for internet communication and allows users to access websites by typing in a domain name instead of an IP address.
- DNS servers can be configured locally or can be provided by an ISP.
- **DNS Process:**
    - The user types a domain name, such as "www.example.com", into their web browser and clicks enter.
    - The web browser sends a DNS query to the local DNS resolver (usually provided by the ISP), asking for the IP address associated with the domain name.
    - If the local DNS resolver has the IP address in its cache, it responds to the web browser with the IP address.
    - If the local DNS resolver does not have the IP address in its cache, it sends a DNS query to the root DNS servers, asking for the IP address of the top-level domain (TLD) server associated with the domain name. For example, in the case of "www.example.com", the TLD is ".com".
    - The root DNS servers respond to the local DNS resolver with the IP address of the TLD server for the requested domain.
    - The local DNS resolver sends a DNS query to the TLD server, asking for the IP address of the authoritative DNS server for the domain name. The authoritative DNS server is the server that has the most up-to-date information about the domain name.
    - The TLD server responds to the local DNS resolver with the IP address of the authoritative DNS server for the requested domain.
    - The local DNS resolver sends a DNS query to the authoritative DNS server, asking for the IP address associated with the domain name.
    - The authoritative DNS server responds to the local DNS resolver with the IP address.
    - The local DNS resolver caches the IP address and responds to the web browser with the IP address.
    - The web browser establishes a connection to the server at the IP address and retrieves the requested web page.

    ```
    graph TD
        subgraph DNS
            A[DNS Resolver] -- Queries --> B[DNS Root Server]
            B -- Referral --> C[Top-Level Domain Server]
            C -- Referral --> D[Authoritative DNS Server]
            D -- Response --> A
        end
    ```

- **Methods that a DNS resolver:**
    - **iterative resolution:** the resolver takes an active role in resolving the domain name by sending queries to multiple DNS servers.
    - **recursive resolution:** the resolver relies on the DNS server to recursively resolve the domain name.

First Hop Redundancy Protocol (FHRP):

- **FHRP** is a network protocol that provides **redundancy** for the **default gateway** in case it fails.
- The default gateway is the router that **connects** the **local** network to **other** networks and the internet.
- If the default gateway **fails**, the **entire** network can become **unreachable**.
- FHRP allows a **redundant** gateway to take over if the primary gateway **fails**, ensuring that the **network remains** reachable.
- FHRP works by **assigning** a **virtual IP** address to a **group** of **routers** that **act** as a **single gateway**.
- When a client sends a packet to the **default** gateway, it sends it to the **virtual IP** address, which is then forwarded to the **active gateway**.
- If the active gateway **fails**, another gateway in the group **takes** over and becomes the active gateway.
- There are several types of **FHRP** protocols, **including**:
    - **Hot-Standby Redundancy Protocol (HSRP):**
        - HSRP is a **Cisco** proprietary protocol that provides redundancy for a group of routers.
        - It allows for the use of **two** gateways, **one** active and **one** standby.
        - The active gateway is responsible for **forwarding** packets, while the standby gateway **monitors** the active gateway and takes **over** if it **fails**.
        - HSRP does **not** support **load-balancing** between the gateways.
    - **Gateway Load-Balancing Protocol (GLBP):**
        - GLBP is also a **Cisco** proprietary protocol that provides redundancy for a group of routers.
        - It allows for the use of up to **four** gateways, all of which can share the **load** of **forwarding** packets.
        - GLBP can **balance** the **load** between the gateways by assigning different **virtual MAC** addresses to different clients.
        - If one of the gateways **fails**, the other gateways can take **over** its load.
    - **Virtual Router Redundancy Protocol (VRRP):**
        - VRRP is an **open standard** protocol that provides redundancy for a group of routers.
        - It allows for the use of **two** gateways, **one** active and **one** standby.
        - The active gateway is **responsible** for **forwarding** packets, while the standby gateway **monitors** the active gateway and takes **over** if it **fails**.
        - VRRP does **not** support **load-balancing** between the gateways.
        - VRRP is supported by **many vendors**, making it more **flexible** than **HSRP** and **GLBP**.

Network Address Translation (NAT):

- **Private IP Addresses:** Devices on a **local** network use **private** IP addresses to **communicate** with **each** other. These private IP addresses are not **routable** over the **internet**.
- **Public IP Addresses:** Public IP addresses are assigned by **Internet** Service Providers (**ISPs**) and can be used to **communicate** over the **internet**.
    - **End** devices like **PCs** cannot be assigned **public** IP addresses because they are not **directly routable** on **private** networks.
- **NAT:** Network Address Translation (**NAT**) is a technique used by **routers** to **translate private** IP addresses to **public** IP addresses, and **vice versa**.

- **NAT Types:** There are three types of NAT:
    - **Static NAT:** This involves a **one-to-one mapping** of private IP addresses to public IP addresses, typically used for **servers**.
    - **Dynamic NAT:** This involves **mapping a group** of private IP addresses to a **group of public** IP addresses, typically used for **clients** ,like **dhcp pool**.
    - **PAT:** Port Address Translation (**PAT**), also known as **NAT-Overload** or **NAPT**, is a variation of NAT that uses a **single public** IP address for **multiple private** IP addresses.
        - It translates the **private** IP address and **port number** to a unique **public** IP address and **port** number to allow **multiple** devices to **use** the **same public** IP address.
        - PAT **translates** the **source port** number of the outgoing packet to a **unique port number**, which is used to **identify** the **device** on the **local** network.
            - Example: 192.168.1.1 —**>** 5.10.10.10**:10 (port)**
- **NAT by Routers:** NAT is performed only by **routers**, **not** switches or MLS's.
- **Limitations of NAT:** NAT does **not solve** the problem of IP **exhaustion**, as there are a **limited** number of **public IP** addresses **available**.

Lab15:

In this lab, we have a network consisting of 3 PCs, 2 routers, 2 switches, and a server. The 3 PCs are connected to Switch 1, which is then connected to Router 1 with an IP address of 10.10.10.0/26. Router 1 is connected to the internet. Router 2 is acting as a router for Google with an IP address of 9.9.9.0/29 and is connected to Switch 2 with an IP address of 1.1.1.0/24. The server is connected to Switch 2. The objective of the lab is to enable communication between the PCs and the server using NAT.

```
graph LR

    PC1((PC1)) --> switch1((switch1))
    PC2((PC2)) --> switch1((switch1))
    PC3((PC3)) --> switch1((switch1))
    switch1((switch1)) --10.10.10.0/26--> router1((router1))
    router1((router1)) --9.9.9.0/29--> router2((router2))
    switch2((switch2)) --1.1.1.0/24--> router2((router2))
    Server1((Server1)) --> switch2((switch2))
```

Steps:

1. Drag and drop 3 PCs, 2 routers, 2 switches, and 1 server from the device list on the left.
2. Connect the devices as follows:
    - Connect each PC to Switch 1.
    - Connect Switch 1 to Router 1.
    - Connect Router 1 to Router 2.
    - Connect Switch 2 to Router 2.
    - Connect the server to Switch 2.
3. Assign IP addresses to the devices as follows:
    - **PC1:**
        - **IP address:** $10.10.10.10$
        - **Subnet mask:** $255.255.255.192$
        - **Default gateway:** $10.10.10.1$
    - **PC2:**
        - **IP address:** $10.10.10.11,$
        - **Subnet mask:** $255.255.255.192,$
        - **Default gateway:** $10.10.10.1$
    - **PC3:**
        - **IP address:** $10.10.10.12$
        - **Subnet mask:** $255.255.255.192$
        - **Default gateway:** $10.10.10.1$
    - **Server:**
        - **IP address:** $1.1.1.1$
        - **Subnet mask:** $255.255.255.0$
        - **Default gateway:** $1.1.1.2$
4. Configure Router1 with an IP address and enable it:

    ```
    Router> enable
    Router# configure terminal
    Router(config)# interface GigabitEthernet0/0
    Router(config-if)# ip address 10.10.10.1 255.255.255.192
    Router(config-if)# no shutdown
    ```

    Syntax:

5. Connect Router1 to the Internet using a straight-through Ethernet cable and configure the interface with an IP address:

```
Router> enable
Router# configure terminal
Router(config)# interface GigabitEthernet0/1
Router(config-if)# ip address 9.9.9.1 255.255.255.248
Router(config-if)# no shutdown
```

Syntax:

6. Configure OSPF on Router1:

```
Router> enable
Router# configure terminal
Router(config)# router ospf 10
Router(config-router)# network 10.10.10.0 0.0.0.63 area 0
Router(config-router)# network 9.9.9.0 0.0.0.7 area 0
```

Syntax:

7. Configure NAT on Router1 to translate the private IP addresses of the PCs to the public IP addresses of the server:

```
Router> enable
Router# configure terminal
Router(config)# ip nat inside source static 10.10.10.10 9.9.9.3
Router(config)# ip nat inside source static 10.10.10.11 9.9.9.4
Router(config)# ip nat inside source static 10.10.10.12 9.9.9.5
Router(config)# interface GigabitEthernet0/0
Router(config-if)# ip nat inside
Router(config-if)# end
Router(config)# interface GigabitEthernet0/1
Router(config-if)# ip nat outside
Router(config-if)# end
```

Syntax:

8. Verify the NAT configuration using the following commands:

```
Router> enable
Router# show ip nat translations
Router# show ip nat statistics
Router# show running-config
```

Syntax:

Lab16:

Configure dynamic NAT on the network topology used in Lab 15. This topology includes two routers, two switches, and a server. The goal is to allow the three PCs to communicate with the server using dynamic NAT.

Steps:

1. Configure Router1 with an access list to permit traffic from the PCs:

```
Router> enable
Router# configure terminal
Router(config)# access-list 1 permit 10.10.10.0 0.0.0.63
Router(config)# end
```

Synax:

2. Configure NAT on Router1 to dynamically translate the private IP addresses of the PCs to the public IP addresses of the server:

```
Router> enable
Router# configure terminal
Router(config)# ip nat pool lab16 9.9.9.3 9.9.9.6 netmask 255.255.255.248
Router(config)# ip nat inside source list 1 pool lab16
Router(config)# interface GigabitEthernet0/0
Router(config-if)# ip nat inside
Router(config-if)# end
Router(config)# interface GigabitEthernet0/1
Router(config-if)# ip nat outside
Router(config-if)# end
```

Syntax:

3. Verify the NAT configuration using the following commands:

   ```
   Router> enable
   Router# show ip nat translations
   Router# show ip nat statistics
   ```

   Syntax:

Network Time Protocol (NTP):

- **NTP** is a protocol used for **clock synchronization** in computer **networks**.
- It ensures that **all** devices in a network have the **same time** and **date** information, with **high accuracy**.
- NTP can **synchronize** an **internal** clock either **manually** or **automatically**.
  - **Accurate time** information is **important** for **security** and **logging** purposes, as well as for ensuring proper operation of time-sensitive applications, such as **financial transactions** or industrial **control** systems.
- NTP uses the User Datagram Protocol (**UDP**) on port **123**.
- In an NTP network, each device can be either a **server** or a **client**.
  - NTP servers **provide** time information to **clients**, which use this information to set their **internal** clocks.
- The accuracy of time information provided by an NTP server is determined by its **stratum**.
- Stratum represents the **level** of **hierarchy** in the NTP network, with 0 being the **most** accurate and 15 being the **least** accurate.
- **Stratum 0 servers** are **high**-**accuracy** time sources, such as **atomic clocks** or **GPS receivers**.
- S**tratum 1** servers are **directly** connected to Stratum **0** servers and provide time information to **Stratum 2** servers.
- **Stratum 2** servers receive time information from **Stratum 1** servers and provide time information to **Stratum 3** servers, and **so on.**
- The **lower** the stratum number, the more **reliable** and **accurate** the time information provided by the server.
- By **default**, a **Cisco router** running NTP is configured as a **Stratum 8** server, which means that it is **not synchronized** to any **external** time **source** and is not **recommended** for use as a **primary** time source.

Lab17:

In this lab, there are two routers connected to each other with an IP address of 192.168.1.0/24. The objective is to configure Network Time Protocol (NTP) between the two routers to ensure accurate time synchronization.

```
graph LR

    router1((router1)) --192.168.1.0/24---> router2((router2))
```

Steps:

1. Drag two routers onto the workspace and connect them using a serial DTE cable.
2. Configure Router1 with an IP address and enable it:

   ```
   Router> enable
   Router# configure terminal
   Router(config)# interface Serial0/0/0
   Router(config-if)# ip address 192.168.1.1 255.255.255.0
   Router(config-if)# no shutdown
   ```

   Syntax:

3. Set the clock on Router1:

   ```
   Router> enable
   Router# configure terminal
   Router(config)# clock timezone AST 3
   ```

   Syntax:

4. Configure Router1 as an NTP master:

   ```
   Router> enable
   Router# configure terminal
   Router(config)# ntp master
   ```

   Syntax:

5. Verify the clock on Router1:

```
Router> enable
Router# configure terminal
Router# show clock
```

Syntax:

6. Configure Router2 with an IP address and enable it:

```
Router> enable
Router# configure terminal
Router(config)# interface Serial0/0/0
Router(config-if)# ip address 192.168.1.2 255.255.255.0
Router(config-if)# no shutdown
```

Syntax:

7. Configure Router2 to use Router1 as its NTP server:

```
Router> enable
Router# configure terminal
Router(config)# ntp server 192.168.1.1
```

Syntax:

8. Verify the clock on Router2:

```
Router> enable
Router# configure terminal
Router# show clock
```

Syntax:

9. Verify the NTP associations on Router2:

```
Router> enable
Router# configure terminal
Router# show ntp associations
```

Syntax:

10. Verify the NTP status on Router2:

```
Router> enable
Router# configure terminal
Router# show ntp status
```

Syntax:

11. Test connectivity between the two routers using the ping command:

```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.1.1
```

Syntax:

Simple Network Management Protocol (SNMP):

- **SNMP** is a protocol used to **monitor** networks from a **single point of view**.
- **SNMP** uses a **server/client** relationship to **gather** information about **network devices**.
- SNMP uses User Datagram Protocol (**UDP**) port **161 / 162** for **trap**.
- The server is the **requester** and **recorder** of SNMP **data**.
  - The SNMP server **provides** a **GUI** that allows a network manager to **view** the **information collected** by the SNMP clients.
  - SNMP **sends "get"** messages to an SNMP client or clients to **retrieve** their **status** information.
- At the **agent side**, SNMP uses **two** main components: the Management Information Base (**MIB**) object and the **agent**.
  - The **MIB** object: is a **database** that contains **information** about the network **devices** and their **status**. It is like a factory that **produces** information.
  - The **agent** :is the **messenger** that **retrieves** information from the **MIB** object and **sends** it to the **SNMP** server.
- There are **three** versions of **SNMP**: **v1, v2c, and v3**.
  - **SNMP v1** is **obsolete** and has security **vulnerabilities**.
  - **SNMP v2c** is an **enhanced** version of SNMP v1 that **includes improvements** such as better error **handling** and support for **64-bit counters**.

    - **SNMP v3** is the **most secure** version of SNMP and supports **authentication** and **encryption** of SNMP messages.
- SNMP is used to **monitor** various network devices such as **routers**, **switches**, **servers**, **printers**, and more.
- SNMP can be used to monitor device **status**, network **traffic**, **bandwidth usage**, and other **metrics**.
- SNMP can be **configured** to send **alerts** and **notifications** when **certain conditions** are **met**, such as when a network **device** goes **offline**.
- SNMP is commonly **used** by network **administrators** to monitor and **troubleshoot** network **issues** and to ensure **optimal** network performance.
- To view the **information collected** by SNMP, you need **software** installed on a **PC** or **server**, such as **PRTG**.
    - **PRTG** is a network monitoring **tool** that can be used to **monitor** network devices and their **status**.
    - **PRTG** uses **SNMP** to **collect** information about network devices, such as **CPU** usage, **memory** usage, and **bandwidth** utilization.
    - **PRTG** provides a **GUI** that allows a network manager to view the **collected** information in **real-time**.

System Loggings (Syslog):

- **Syslog** helps you stay **aware** of **everything** happening on your network. By **collecting log** messages from **all** your network devices, you can **get** a **complete** picture of what's **happening behind** the scenes.
- **Syslog messages** are **prioritized** based on their **severity**. The severity levels range from **0** (**emergency**) to **7** (**debug**), with each level **indicating** a different level of **urgency**.
- A **server/clien**t relationship is **established** between the network devices that **generate log** messages (**the clients**) and the **central** server that **collects** and stores those messages (**the server**).
    - The **Syslog** server may have an installed **application** for **Syslog**, such as **Syslog-ng** or **rsyslog**, or it may use **specialized software** such as **Splunk** or **Kiwi** to manage and **analyze** the log data more **efficiently**.
    - The server can be a normal server that collects all the log messages, or it can use specialized software like Syslog or Splunk to manage the log data more efficiently.
- **Clients** are the networking devices that **generate** log messages. Examples of clients include **routers**, **switches**, **firewalls**, and servers.
- To remember the **severity levels**, some people use the **mnemonic** "**Every Awesome Cisco Engineer Will Need Ice-Cream Daily**," with each letter corresponding to a severity level **(0-7)**.
- Here's a breakdown of the **severity** levels and what they **represent**:
    - **0 (emergency):** The **highest** level of severity. Indicates a system is **unusable** or a **significant portion** of the network is **down**.
    - **1 (alert): Indicates** an **immediate** action is **needed**.
    - **2 (critical):** Indicates a **critical** condition that should be **addressed** as **soon** as **possible**.
    - **3 (error):** Indicates an **error** condition that **should** be **addressed**.
    - **4 (warning):** Indicates a potential **problem** that should be **investigated**.
    - **5 (notification):** Indicates a **normal** but **significant** condition, such as a **change** in **state**.
    - **6 (informational): Provides** information **about** the system, such as **startup** messages.
    - **7 (debug):** Provides **debugging information** for **troubleshooting purposes**. **(first letter)**
- **NTP** logs may be **collected** and **analyzed** using **Syslog** if **desired**.

Quality of Service (QoS):

- If there is more **traffic** than **available bandwidth**, **congestion** will occur. This can **lead** to **delays**, **dropped packets**, and **other** issues.
    - **Packet loss:**
        - **Packet loss** occurs **when** one or more packets of data **fail** to reach their **destination**.
        - Packet loss can be caused by a **variety** of factors, including network **congestion**, **faulty** hardware, or **improper** network configuration.
        - Packet loss can result in **degraded** network performance, **slower** data transfer rates, and **reduced** reliability.
    - **Congestion:**
        - Congestion can be **caused** by a **variety** of factors, including **inadequate** network capacity, **improperly** configured network devices, or **excessive** traffic from **individual** users or applications.
- **QoS** allows you to **prioritize** certain **types** of traffic over **others**, which can help **prevent** congestion and ensure that **critical** traffic gets **through**.
- In general, **UDP** (User Datagram Protocol) traffic is **preferred** over **TCP** (Transmission Control Protocol) traffic because **UDP** doesn't have the **overhead** of **retransmissions** that TCP does.
- QoS can be **implemented** using a variety of techniques, including **classification**, **marking**, **queuing**, **shaping**, and **policing**.
    - **Classification:** involves **identifying** traffic based on its **characteristics**, such as **source** and **destination** IP address, **protocol**, and **port** number. This allows you to **classify** traffic according to its **importance** or **priority**.
    - **Marking :** involves **adding** a special **tag** to packets to **indicate** their **priority** or importance. This tag can be used by **other** devices on the network to **prioritize** traffic **accordingly**.
        - Traffic can be **classified** and **marked** according to a **variety** of factors, such as the **level** of **importance (very high, high, medium, low)**, the type of traffic **(voice, video, data**), or the **source** or **destination** of the **traffic**.
    - **Queuing:** involves **managing** the order in which packets are transmitted based on their **priority**. **Higher**-priority packets are transmitted **first**, which can help ensure that **critical** traffic gets **through** even if there is **congestion**.
        - **For example**, **UDP** traffic may be given a **"very high**" priority with **40**% of the capacity, while lower-priority traffic may receive **less**. This helps ensure critical traffic gets **through during** congestion.
    - **Shaping:** involves **controlling** the **rate** at which traffic is **transmitted** to prevent **congestion**. This can be done by **slowing** down the **transmission** of **non-critical** traffic to make room for more **important** traffic.
    - **Policing:** involves **enforcing** traffic limits to **prevent** non-critical traffic from consuming **too much bandwidth**. This can be done by **dropping** or **delaying** packets that **exceed** certain limits.

Secure Shell (SSH):

- **SSH** (Secure Shell) is a **secure** and **trusted** method for **logging** in to a device **remotely**.
- It uses **TCP port 22** for communication.
- SSH **encrypts** the information that is transmitted between the **client** and **server**, providing a **high** level of security.
- SSH uses a **server/client** relationship, where the server is the device being **accessed** and the client is the device used to **access** the server.
- SSH is a **replacement** for **Telnet**, which is an **older** and less **secure** protocol for **remote** device access.
- SSH **requires** an application to be **installed** on the client device, especially for **Microsoft Windows users**.
- **Using console access:**
  - If **remote** access is not **possible** or **desirable**, console access can be used to configure a device from a **local** device, such as a **laptop** or **desktop** computer.
  - Console access **involves** physically **connecting** a cable from the local device to the **console port** of the **device** being configured.

Lab18:

In Lab18, there is a network consisting of one PC, three switches, and one router. The network is configured such that PC1 is connected to switch1, switch1 is connected to switch2, switch2 is connected to switch3, and switch3 is connected to router1. The IP address range for the network is 10.0.0.0/24.To enable SSH access for router1 from PC1

```
graph LR

    PC1((PC1)) ---> switch1((switch1))
    switch1((switch1)) ---> switch2((switch2))
    switch2((switch2)) ---> switch3((switch3))
    switch3((switch3)) --10.0.0.0/24--> router1((router1))
```

Steps:

1. Drag and drop one PC, three switches, and one router to the workspace.
2. Connect the devices as follows:
   - Connect PC1 to switch1
   - Connect switch1 to switch2
   - Connect switch2 to switch3
   - Connect switch3 to router1
3. Configure the IP settings on PC1 using the following commands:
   - **ip address:** $10.0.0.10$
   - **subnet mask:**$255.255.255.0$
   - **ip default-gateway:** $10.0.0.1$
4. Configure SSH access on router1 using the following commands:

   ```
   Router> enable
   Router# configure terminal
   Router(config)# hostname lab18
   lab18(config)# ip domain-name lab
   lab18(config)# crypto key generate rsa modulus 768
   lab18(config)# ip ssh version 2
   lab18(config)# line vty 0 15
   lab18(config-line)# transport input ssh
   lab18(config-line)# login local
   lab18(config)# username sshlab privilege 15 password 18
   lab18(config)# enable password en18
   ```

   Syntax:

5. Configure the IP settings and SSH access on switch1 using the following commands:

   ```
   Router> enable
   Router# configure terminal
   Router(config)# int vlan 1
   Router(config-if)# ip address 10.0.0.2 255.255.255.0
   Router(config-if)# no shutdown
   Router(config)# hostname lab18
   lab18(config)# ip domain-name lab
   lab18(config)# crypto key generate rsa modulus 768
   lab18(config)# ip ssh version 2
   lab18(config)# line vty 0 15
   lab18(config-line)# transport input ssh
   lab18(config-line)# login local
   ```

```
lab18(config)# username sshlab privilege 15 password 18
lab18(config)# enable password en18
```

Syntax:

6. Open the Command Prompt or Terminal on PC1.
7. Type the following command to initiate an SSH connection to router1:

```
Cisco Packet Tracer PC Command Line 1.0
C:\>ssh -l sshlab 10.0.0.1
sshlab@10.0.0.1's password: 18
Switch>en
Password: en18
Switch#
```

Syntax:

8. Once you have successfully authenticated, you should be able to access the router's command line interface (CLI) and issue commands.
9. You can test the connectivity between PC1 and router1 by pinging the router's IP address from PC1 using the following command:

```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 10.0.0.1
```

Syntax:

File Transfer Protocol (FTP):

- **FTP** is a protocol that enables devices to **transfer** files and **other** data between them.
- The **types** of data that can be transferred via FTP include files, **software images**, and **configurations** saved as **text files**.
- FTP uses TCP **ports 20** and **21** for communication.
  - The **reason** for **using two** TCP ports is that TCP **21** is used as the **control channel** to **establish** a connection between the **server** and the **client**, while TCP **20** is used as the **data** channel to **transfer** data between the **server** and **client**.
- FTP is a **reliable** protocol that provides **error** checking and **flow** control to ensure that data is **transferred** accurately and **efficiently**.
- FTP **software** can be **installed** on a server to **enable** file transfers between devices.
- To **connect** to the FTP server from a client device, the following **command** can be used: `ftp://<username>@<ip_address>`.
- When **prompted**, enter the **password** for the FTP server to **authenticate** and **establish** a connection.

Trivial FTP (TFTP):

- **TFTP** is a **relative** of FTP that is used for simple file transfers.
- TFTP uses **UDP** port **69** for communication.
- Unlike FTP, TFTP is an **unreliable** protocol that **does not** provide **error** checking or flow control mechanisms. This means that data **transferred** via TFTP may be **lost** or **corrupted**, but it is still **useful** for certain applications where **reliability** is **not** a major concern.
- TFTP is commonly used for transferring **firmware** updates to network devices, as well as for **booting diskless** workstations and **loading** operating system images during the **installation** process.

SECURITY FUNDAMENTALS:

Security Concepts & Programs:

- **Asset:**
  - An asset is anything that is **valuable** to an organization, such as **documents**, **information**, and other **resources**.
  - It is important to **identify** and **prioritize** assets in order to **protect** them from **threats**.
- **Threat:**
  - A threat is a **danger** or **risk** to an asset, such as a **hacker**, **software** bug, or **environmental disaster**.
    - **Software bug:**
      - A software bug is a **flaw** or **error** in a software program that causes it to **malfunction** or behave in **unexpected** ways.

- Bugs can be caused by **programming** errors, incorrect **assumptions**, or unexpected **interactions** between different parts of the **software**.
- Bugs can lead to **crashes**, data **corruption**, security **vulnerabilities**, and other issues that can impact the **performance** and **reliability** of the software.

- **Vulnerability:**
  - A vulnerability is a **weakness** in a system or process that can be **exploited** by a **threat**, such as an **old bug** or a missing **patch**.
    - **Patch:**
      - A patch is a **piece** of **software** that is designed to **fix** a **bug** or address a **security** vulnerability in an **existing** program.
      - Patches can be released by software **vendors**, **developers**, or security **researchers** in response to a known **bug** or vulnerability.
      - Patches can be applied to the **software** to **fix** the problem and **prevent** future **issues**.
      - Patches can also be used to add **new** features, **improve** performance, or **address** other **issues** with the software.
- **Mitigation:**
  - **Mitigation** refers to measures taken to **reduce** or **eliminate** the **risks** posed by **threats** and **vulnerabilities**.
  - There are **three** main types of **mitigation**: **technical/logica**l, **administrative**, and **physical**.
    - **Technical/logical mitigation:**
      - Technical/logical mitigation involves choosing the **correct firewall**, **IPS**, and **design** to protect assets.
        - Choosing the correct **firewall ,IPS** is important because different firewalls,IPS have different **features**, **capabilities**, and **performance** characteristics.
    - **Administrative mitigation:**
      - Administrative mitigation involves **policies** and **procedures** that are decided by the **network administrator**, such as **written documents**, background **checks** for **new** employees, and security **awareness training**.
      - **Alternatives:**
        - Alternatives to **password authentication** include **two-factor/multi-facto**r authentication, which involves using **biometrics** and **certificates** in **addition** to passwords.
    - **Physical mitigation:**
      - Physical mitigation involves **securing** devices inside **racks** and **data centers** using **locked** metal/glass doors, **keys**, **cards**, **fingerprints**, and other physical security **measures**.

Device Access Control:

Device Access Control:

- Device access control is the **process** of **controlling** and **managing** access to network devices, such as **routers**, **switches**, and **firewalls**.
- If a device is not **locked properly**, physical access to the device can be **gained**, and an attacker can **connect** to the **Console** and **Auxiliary** ports to gain **unauthorized** access.

Protecting Console/AUX Ports:

- Console and Auxiliary ports can be protected by **configuring** a **specified** password for **each** port, or by using **local credentials** and applying them **upon** the **ports**.
  - By using a **specified** password, an attacker must enter the **correct** password to gain **access** to the port.
  - By using **local** credentials, an attacker must **provide** the correct **username** and **password** to gain **access** to the port.
- Even if a user **logs** in to a **device**, access can be limited by **assigning** an `"enable secret/password"`.
  - This limits access to the **device's privileged exec** mode and prevents **unauthorized** configuration changes.

Lab19:

In Lab19, you have a network consisting of one PC and one router. Your task is to implement access control measures that will allow the router to access the PC through the console port, while preventing unauthorized access. Specifically, you need to set up passwords for the console and auxiliary ports of the router, create a local login for the console port, and configure an enable password for the router.

```
graph LR

    PC1((PC1)) ---> router1((router1))
```

Steps:

1. Drag and drop one PC and one router to the workspace.
2. Connect the devices using a console cable, with one end connected to the console port of the router and the other end connected to the PC.
3. Set the hostname for the router:

   ```
   Router> enable
   Router# configure terminal
   Router(config)# hostname lab19
   ```

   Syntax:

4. Set a password for the console port:

   ```
   Router> enable
   Router# configure terminal
   Router(config)# line console 0
   ```

```
Router(config-line)# password lab19
Router(config-line)# login
```

Syntax:

5. Set a password for the auxiliary port:

```
Router> enable
Router# configure terminal
Router(config)# line aux 0
Router(config-line)# password lab19
Router(config-line)# login
```

Syntax:

6. Set an enable password for the router:

```
Router> enable
Router# configure terminal
Router(config)# enable password enlab19
```

Syntax:

7. Create a username and password for the local login:

```
Router> enable
Router# configure terminal
Router(config)# username lab19user privilege 15 password lab19
```

Syntax:

8. Configure the console port to use local login:

```
Router> enable
Router# configure terminal
Router(config)# line console 0
Router(config-line)# login local
```

Syntax:

Virtual Private Networks (VPN):

- VPNs are a way to **securely** connect **remote** networks or **users** to a **private** network over the **public** internet.
- The **"virtual"** part of VPN refers to the fact that the **connection** is made over the **internet**, rather than a **physical private network**.This means that a VPN connection does **not** require a **dedicated** physical connection **between** the devices.
- The "**private**" part of VPN refers to the fact that the **connection** is **encrypted** and **secure**, providing full **separation** between the **private** network and the **public** internet.

Why VPN?

- VPNs are used to securely connect two or more networks or devices over the public internet.
- VPNs are often used when direct connections between networks or devices are **not possible** or practical due to **geographic** or **organizational constraints**.

Tunnels:

- VPNs use **tunnels** to establish a **secure** connection between **two** networks or **devices**.
- Tunnels create a **virtual connection** between the two networks, providing **end-to-end** encryption and **full separation**.

Site-to-Site VPN:

- Site-to-Site VPNs are used to connect **two** or **more** networks together over the **public** internet.
- In Site-to-Site VPN, the connection is established between **two routers**, allowing **all** devices on each network to communicate with each other **securely**.

> ⚠️ **Problem Senario :**
>
> - In a scenario where there are **two big departments** that need to be **linked** together, but they cannot be linked directly due to their **size** and **responsibilities**, **VPNs** can be used to create a secure **tunnel** between them.
> - The middle network between the two departments may have **unknown IP** addresses and unknown **structure**, making it **difficult** to establish a **direct** connection.

  - **Peer-to-Peer VPN:**
    - Peer-to-Peer VPNs are used to connect two devices together over the **public** internet.

        - In Peer-to-Peer VPN, the connection is established between **two devices**, allowing them to communicate with each other securely.
        - **solution Senario :**
            - In Peer-to-Peer VPN, the tunnel will be established from **one** department to the **middle** network, and then the **middle** network will **read** the data to **determine** the **routing algorithm**, and **continue** the tunnel to the **other** department.
            - This type of VPN allows for secure communication between two devices **without** the need for a **dedicated** infrastructure.
    - **Overlay VPN:**
        - An Overlay VPN is a type of VPN that allows users to obtain a circuit from an ISP and use their own IGP (Interior Gateway Protocol) all the way.
        - The ISP only provides the physical layer connectivity and does not participate in the routing and forwarding of the traffic.
        - **solution Senario :**
            - In Overlay VPN, a **dedicated** circuit is obtained from an **ISP** to create a **secure** tunnel between **two** networks.
            - This type of VPN is more **expensive**, but it provides a **higher** level of security and **control** over the **routing** and forwarding of traffic.
    - **Underlay:**
        - Underlay refers to the **physical** network **infrastructure** that provides connectivity between devices.
        - VPNs require an **IGP** for routing and forwarding over the **underlay** network.
        - The **IGP** will be exchanged at the **edges** with the **ISP** to ensure that the **traffic** is properly **routed**.
            - **An Interior Gateway Protocol (IGP)** is a type of routing protocol used by routers within a single autonomous system (**AS**) to **exchange** routing **information**. An AS is a **collection** of networks under a **single administrative** domain, such as a company or an internet service provider (**ISP**).
            - There are several types of IGPs, including:
                1. **Routing Information Protocol (RIP):**
                2. **Interior Gateway Routing Protocol (IGRP):**
                3. **Open Shortest Path First (OSPF):**
                4. **Intermediate Systemto-Intermediate System (IS-IS):**
    - **Client VPN:**
        - Client VPNs are used to connect an **end** user to a **private** network over the **public** internet.
        - Client VPNs require **software** to be installed on the **user's** device.
        - The connection is established **remotely** and requires **credentials** to access the **private** network.
        - The tunnel will be **established** between the **end** user's PC and the **router**.
        - **solution Senario :**
            - In Client VPN, **software** is installed on an **end** user's device to **establish** a secure connection to a **private** network over the **public** internet.
            - This allows the **end** user to **access** resources on the **private** network as if they were **physically** present on the network.

Access Control Lists (ACLs):

- ACLs are a set of **rules** that are used to define **specific permissions** for **users** or networks.
- ACLs **allow** or **deny** traffic based on specific **criteria**, such as **source** and **destination** IP **addresses**, **ports**, and **protocols**.
- ACLs are used in Layer 3 (**Network Layer**) of the OSI model to control access to a network or device.

ACL Rules:

- ACLs **consist** of **allow** or **deny** rules **only**.
- ACL rules are processed **serially**, which means they are read **line** by **line** in **order**.
    - If an Access Control List (**ACL**) rule **denies** access to a **particular** IP address, any **subsequent** ACL rules that allow access to the **same** IP address will be **ignored**.
- **Invisible** rules are **automatically** applied to an ACL if **no** rules are **defined**. The invisible rule **denies** access to **any** network or device.
- ACLs can be used to allow or deny access to specific **hosts** or **networks** from the internet.

ACLs in Checkpoint:

- In Checkpoint **firewall**, ACLs can be used to control traffic flowing **in** and **out** of the network.
- ACLs can be applied to only **one interfaces** (such as **inside** or **outside** interfaces) or to specific **devices** or **subnets**.
- ACLs are used to **permit** or **deny** traffic based on the **source** and **destination** IP **addresses**, **ports**, and **protocols**.

ACL Types:

- **Standard ACLs** use the **source host** or **network** to decide the **permissions**.
    - Standard ACLs have a range of **1-99** and do **not** provide **specific** permissions.
    - **Example:**
        - access-list 10 deny host 192.168.1.100
- **Extended ACLs** use the **source** and **destination** hosts, **networks**, **ports**, and **services** to decide the **permissions**.
    - Extended ACLs have a range of **100-199** and provide specific, **detailed** permissions.
    - **Example:**
        - access-list 101 permit tcp any host 172.16.1.100 eq 80
- **Named ACLs** are a **combination** of **standard** and **extended** ACLs and can be **organized** in a **hierarchy** mode. They are also **named** for **easier** management.

Lab20

In Lab 20, there are five PCs, two switches, and one multilayer switch. PC1 and PC2 are connected to Switch1, which is connected to MLS1 with an IP address of 10.1.1.0/24. MLS1 is also connected to Switch2 with an IP address of 10.1.3.0/28. PC3 is connected to MLS1 with an IP address of 10.1.2.0/27, while PC4 is connected to Switch2. Your task is to configure an Access Control List (ACL) on the switches and MLS1 to restrict access from the 10.1.1.0/24 network. Show your work.

```
graph LR

    PC1((PC1)) ---> Switch1((Switch1))
    PC2((PC2)) ---> Switch1((Switch1))
    Switch1((Switch1)) --10.1.1.0/24--> mls1((mls1))
    PC3((PC3)) --10.1.2.0/27--> mls1((mls1))
    Switch2((Switch2)) --10.1.3.0/28--> mls1((mls1))
    PC4((PC4)) ---> Switch2((Switch2))
    PC5((PC5)) ---> Switch2((Switch2))
```

Steps:

1. Drag and drop five PCs, one multilayer switch (MLS1), and two switches to the workspace.
2. Connect the devices as follows:
   - Connect PC1 and PC2 to switch1
   - Connect switch1 to MLS1
   - Connect MLS1 to switch2
   - Connect PC3 to MLS1
   - Connect PC4 and PC5 to switch2
3. Configure the IP settings on each device using the following commands:
   - **PC1:**
     - **ip address:** 10.1.1.10
     - **Subnet Mask:** 255.255.255.0
     - **ip default-gateway:** 10.1.1.1
   - **PC2:**
     - **ip address:** 10.1.1.11
     - **Subnet Mask:** 255.255.255.0
     - **ip default-gateway:** 10.1.1.1
   - PC3:
     - **ip address:** 10.1.2.10
     - **Subnet Mask:** 255.255.255.224
     - **ip default-gateway:** 10.1.2.1
   - PC4:
     - **ip address:** 10.1.3.10
     - **Subnet Mask:** 255.255.255.240
     - **ip default-gateway:** 10.1.3.1
   - PC5:
     - **ip address:** 10.1.3.11
     - **Subnet Mask:** 255.255.255.240
     - **ip default-gateway:** 10.1.3.1
4. Configure the Multilayer Switch (MLS1):
   - Enable IP routing with the command: `ip routing`.

   Syntax:

   - Configure the IP address for each interface with the commands:

     ```
     Switch> enable
     Switch# configure terminal
     Switch(config)# interface gigabitethernet0/0
     Switch(config-if)# no switchport
     Switch(config-if)# ip address 10.1.1.1 255.255.255.0
     Switch(config-if)# exit
     Switch(config)# interface gigabitethernet0/1
     Switch(config-if)# no switchport
     Switch(config-if)# ip address 10.1.2.1 255.255.255.224
     ```

     Syntax:

   - Configure OSPF routing with the command:

     ```
     Switch> enable
     Switch# configure terminal
     Switch(config)# router ospf 1
     Switch(config-router)# network 10.1.1.0 0.0.0.255 area 0
     Switch(config-router)# network 10.1.2.0 0.0.0.31 area 0
     Switch(config-router)# network 10.1.3.0 0.0.0.15 area 0
     ```

Syntax:

- Configure an ACL to deny traffic from the 10.1.1.0/24 network with the command:

  ```
  Switch> enable
  Switch# configure terminal
  Switch(config)# access-list 1 deny 10.1.1.0 0.0.0.255
  Switch(config)# access-list 1 permit any
  ```

  Syntax:

- Apply the ACL to the outbound traffic on the gigabitethernet0/0 interface with the command:

  ```
  Switch> enable
  Switch# configure terminal
  Switch(config)# interface gigabitethernet0/0
  Switch(config-if)# ip access-group 1 out
  ```

  Syntax:

5. Test the connectivity between the devices by pinging from each PC to the other PCs and the default gateway. You should also test the effectiveness of the access control list by attempting to ping from PC1 to MLS1's interface on the $10.1.1.0/24$ network. The ping should fail due to the access control list blocking it

Lab 21:

In this lab, there are five PCs, two switches, and one multilayer switch. PC1 and PC2 are connected to Switch1, which is connected to MLS1 with an IP address of 10.1.1.0/24. MLS1 is also connected to Switch2 with an IP address of 10.1.3.0/28. PC3 is connected to MLS1 with an IP address of 10.1.2.0/27, while PC4 is connected to Switch2. Additionally, there is a server connected to MLS1 with an IP address of 10.1.4.0/30. Your task is to configure an extended named ACL on the switches and MLS1 to restrict access from certain devices to the network and to allow ICMP traffic from any device to any device. Show your work.

```
graph LR

    PC1((PC1)) ---> Switch1((Switch1))
    PC2((PC2)) ---> Switch1((Switch1))
    Switch1((Switch1)) --10.1.1.0/24--> mls1((mls1))
    PC3((PC3)) --10.1.2.0/27--> mls1((mls1))
    Switch2((Switch2)) --10.1.3.0/28--> mls1((mls1))
    PC4((PC4)) ---> Switch2((Switch2))
    PC5((PC5)) ---> Switch2((Switch2))
    Server1((Server1)) --10.1.4.0/30--> mls1((mls1))
```

Steps:

1. Drag and drop five PCs, one multilayer switch (MLS1), two switches, and one server to the workspace.
2. Connect the devices as follows:
   - Connect PC1 and PC2 to switch1
   - Connect switch1 to MLS1
   - Connect MLS1 to switch2
   - Connect PC3 to MLS1
   - Connect PC4 and PC5 to switch2
   - Connect the server to MLS1
3. Configure the IP settings on each device using the following commands:
   - **Server:**
     - **ip address:** $10.1.4.2$
     - **Subnet Mask:** $255.255.255.252$
     - **ip default-gateway:** $10.1.4.1$
4. Configure the Multilayer Switch (MLS1):
   - Configure the IP address for each interface with the commands:

     ```
     Switch> enable
     Switch# configure terminal
     Switch(config)# interface G0/0/2
     Switch(config-if)# no switchport
     Switch(config-if)# ip address 10.1.4.1 255.255.255.252
     Switch(config-if)# exit
     ```

     Syntax:

   - Configure OSPF routing with the command:

     ```
     Switch> enable
     Switch# configure terminal
     ```

```
Switch(config)# router ospf 1
Switch(config-router)# network 10.1.4.0 0.0.0.3 area 0
```

Syntax:

- Test connectivity by pinging and sending HTTP requests to 10.1.4.2.
- configure the extended named ACL:
  - Deny 10.1.1.10 from reaching 10.1.3.0/28:

    ```
    Switch> enable
    Switch# configure terminal
    Switch(config)# ip access-list extended deny10
    Switch(config-ext-nacl)# deny ip host 10.1.1.10 10.1.3.0 0.0.0.15
    Switch(config-ext-nacl)# permit ip any any
    Switch(config-ext-nacl)# exit
    Switch(config)# interface G0/0/1
    Switch(config-if)# ip access-group deny10 in
    ```

    Syntax:

  - Deny 10.1.3.11 from accessing HTTP on 10.1.4.2:

    ```
    Switch> enable
    Switch# configure terminal
    Switch(config)# ip access-list extended HTTP
    Switch(config-ext-nacl)# deny tcp host 10.1.3.11 host 10.1.4.2 eq 80
    Switch(config-ext-nacl)# permit ip any any
    Switch(config-ext-nacl)# exit
    Switch(config)# interface G0/0/2
    Switch(config-if)# ip access-group HTTP out
    ```

    Syntax:

- Test ACLs by using the command `show access-lists`.

Syntax:

- Modify the extended named ACL:
  - Delete a rule with the number 10 in the HTTP ACL:

    ```
    Switch> enable
    Switch# configure terminal
    Switch(config)# ip access-list extended HTTP
    Switch(config-ext-nacl)# no 10
    ```

    Syntax:

  - Add a rule to allow ICMP traffic:

    ```
    Switch> enable
    Switch# configure terminal
    Switch(config)# ip access-list extended HTTP
    Switch(config-ext-nacl)# 15 permit icmp any any
    ```

    Syntax:

Port Security:

- Switch ports connect devices to a network **immediately**, which can lead to security **vulnerabilities**.
- To mitigate these vulnerabilities, a **limitation** is needed to the switch ports.
- Port security can also be used with **static** ports to **limit** the number of devices that can connect to the network.
- **examples of switch port vulnerabilities:**
  - **MAC address flooding:** an attacker can overload the switch's CAM table, causing it to enter a fail-open mode and potentially gain access to sensitive information.
  - **Rogue devices:** an attacker can connect an unauthorized device to an available switch port to gain access to the network.

    - **VLAN hopping:** an attacker can send spoofed packets to gain access to unauthorized VLANs.
    - **MAC address spoofing:** an attacker can impersonate an authorized device on the network by changing their MAC address.
    - **DHCP spoofing:** an attacker can provide fake DHCP responses to redirect network traffic to a device under their control and potentially gain access to sensitive information.

Limitations:

- Port Security limitations include the **number** of **learned MAC addresses** and the **allowance** of only **statically assigned MAC** addresses to connect.
- A combination of these **two limitations** can also be used to ensure **secure** port access.

Dynamic and Static Ports:

- All Cisco switch ports are **dynamic** by default, meaning they can **learn MAC** addresses over **time**.
- **Static** ports, on the other hand, do **not** have **timers** and require **manually assigned** MAC addresses, which are called "**sticky**."
    - **Dynamic switch** ports use **timers** to automatically **disable** a **port** if it has not received **any** traffic for a **certain period** of time.
    - **Static ports** do **not** have **timers**, which means they will **remain enabled** even if they are **not** in **use**.
    - This can be a security **vulnerability** because an attacker could **potentially** connect a device to an **unused** static **port** and **gain** access to the network.

Reaction to Unallowed MAC Addresses:

- When an **unallowed** MAC address **attempts** to connect to a secure port, the port security **behavior** can be set to either shutdown the port (**default**), **protect the port silently**, or **log the violation strictly**.

  Shutdown:

    - When the port security behavior is set to **shutdown**, the port will be **disabled**, preventing the **unallowed** device from accessing the network.

  Protect:

    - When the port security behavior is set to **protect**, the port will remain **active**, but the **unallowed** device will **not** be **able** to **transmit** packets. (**receiver will drop the frame**)

  Strict:

    - When the port security behavior is set to **strict**, the port will remain **active**, but the **unallowed** device will **not** be **able** to **transmit** packets, and the violation will be **logged**.

Lab22:

You have a network with 3 PCs, 2 switches, and 3 laptops. The PCs are connected to switch1, which is connected to switch2, which in turn is connected to the laptops. You need to implement port security to prevent unauthorized access to the network. Please provide the necessary commands to configure port security on the switches and demonstrate the steps you took to implement this security measure.

```
graph LR

    PC1((PC1)) ---> Switch1((Switch1))
    PC2((PC2)) ---> Switch1((Switch1))
    PC3((PC3)) ---> Switch1((Switch1))
    Switch2((Switch2)) ---> Switch1((Switch1))
    Laptop1((Laptop1)) ---> Switch2((Switch2))
    Laptop2((Laptop2)) ---> Switch2((Switch2))
    Laptop3((Laptop3)) ---> Switch2((Switch2))
```

Steps:

1. Add 3 PCs and 2 switches to the topology.
2. Connect the PCs to switch1, switch1 to switch2, and switch2 to the laptops.
3. Configure switch1 with the following commands to enable port security on its ports:

   ```
   Switch> enable
   Switch# configure terminal
   Switch(config)# interface range FastEthernet 0/1-3
   Switch(config-if-range)# switchport mode access
   Switch(config-if-range)# switchport port-security
   Switch(config-if-range)# switchport port-security maximum 1
   Switch(config-if-range)# switchport port-security mac-address sticky
   ```

   Syntax:

4. Configure switch2 with the following commands to enable port security on its ports:

   ```
   Switch> enable
   Switch# configure terminal
   Switch(config)# interface range FastEthernet 0/1-3
   Switch(config-if-range)# switchport mode access
   Switch(config-if-range)# switchport port-security
   Switch(config-if-range)# switchport port-security maximum 4
   Switch(config-if-range)# switchport port-security violation restrict
   ```

   Syntax:

5. Verify the port security configuration with the following command:

```
Switch> enable
Switch# show port-security interface FastEthernet 0/1
```

Syntax:

6. Test the port security by connecting more devices than the allowed maximum to the ports. This will trigger a violation and the port will be put in the error-disabled state.
7. To clear the violation and enable the port again, use the following commands:

```
Switch(config)# interface FastEthernet 0/1
Switch(config-if)# shutdown
Switch(config-if)# no shutdown
Switch(config-if)# end
```

Syntax:

DHCP Snooping:

- DHCP snooping is a **security** feature used to prevent **rogue DHCP servers** from **providing** IP addresses to devices on a network.
- Rogue DHCP servers can **respond** to DHCP **discovery** messages sent by devices on the network, which can cause devices to be assigned **incorrect** IP addresses or other network **settings**.
- DHCP snooping is implemented on switches and **intercepts** all DHCP messages between devices and DHCP servers.
- When DHCP snooping is enabled, the switch will **trust** only specific **interfaces** to receive **broadcast** messages from DHCP servers, making them the only interfaces **allowed** to receive DHCP offers.
- DHCP snooping is applied on a **per-VLAN** basis and is configured for each VLAN that requires protection against rogue DHCP servers.
- Rogue DHCP servers can act as a **"man in the middle**" and intercept network traffic, potentially **exposing** sensitive information or **causing** other security breaches.
- By using DHCP snooping, network administrators can protect their networks **against** rogue DHCP servers and ensure that devices receive **correct** network settings from **authorized** DHCP servers.

Lab23:

You have a network topology with 4 PCs, 2 switches, and 2 routers. The PCs are connected to switch1, which is connected to switch2, which in turn is connected to router1 with IP address 192.168.10.1/24. Additionally, switch1 is connected to router2 with IP address 192.168.11.1/24, which is acting as a man-in-the-middle. You need to configure DHCP snooping to prevent rogue DHCP servers from providing IP addresses to devices on the network. Please provide the necessary commands to configure DHCP snooping on the switches and routers and demonstrate the steps you took to implement this security measure.

```
graph LR

    PC1((PC1)) ---> Switch1((Switch1))
    PC2((PC2)) ---> Switch1((Switch1))
    PC3((PC3)) ---> Switch1((Switch1))
    PC4((PC4)) ---> Switch1((Switch1))
    Router2((Router2)) --192.168.11.0/24---> Switch1((Switch1))
    Switch1((Switch1)) ---> Switch2((Switch2))
    Switch2((Switch2))   --192.168.10.0/24--->Router1((Router1))
```

Steps:

1. Add 4 PCs, 2 switches, and 2 routers to the topology.
2. Connect the PCs to switch1, switch1 to switch2, switch2 to router1 with IP address 192.168.10.1/24, and switch1 to router2 with IP address 192.168.11.1/24.
3. Configure router1 with the following commands to enable DHCP server:

```
Router> enable
Router# configure terminal
Router(config)# hostname Mainrouter
Mainrouter(config)# interface FastEthernet0/0
Mainrouter(config-if)# ip address 192.168.10.1 255.255.255.0
Mainrouter(config-if)# no shutdown
Mainrouter(config-if)# exit
Mainrouter(config)# ip dhcp pool lab23
Mainrouter(dhcp-config)# network 192.168.10.0 255.255.255.0
Mainrouter(dhcp-config)# default-router 192.168.10.1
Mainrouter(dhcp-config)# dns-server 1.1.1.1
Mainrouter(dhcp-config)# exit
Mainrouter(config)# ip dhcp excluded-address 192.168.10.1
```

Syntax:

4. Configure router2 with the following commands to enable DHCP server:

   ```
   Router> enable
   Router# configure terminal
   Router(config)# hostname Man-in-middle
   Man-in-middle(config)# interface FastEthernet0/0
   Man-in-middle(config-if)# ip address 192.168.11.1 255.255.255.0
   Man-in-middle(config-if)# no shutdown
   Man-in-middle(config-if)# exit
   Man-in-middle(config)# ip dhcp pool lab23_Man
   Man-in-middle(dhcp-config)# network 192.168.11.0 255.255.255.0
   Man-in-middle(dhcp-config)# default-router 192.168.11.1
   Man-in-middle(dhcp-config)# dns-server 1.1.1.1
   Man-in-middle(dhcp-config)# exit
   Man-in-middle(config)# ip dhcp excluded-address 192.168.11.1
   ```

   Syntax:

5. Configure switch1 with the following commands to enable DHCP snooping:

   ```
   Switch> enable
   Switch# configure terminal
   Switch(config)# ip dhcp snooping
   Switch(config)# ip dhcp snooping vlan 1
   Switch(config)# interface range FastEthernet0/1-4
   Switch(config-if-range)# ip dhcp snooping trust
   ```

   Syntax:

6. Configure router1 with the following command to trust DHCP information relayed from other routers:

   ```
   Switch> enable
   Switch# configure terminal
   Switch(config)# ip dhcp relay information trust-all
   ```

   Syntax:

7. Verify the DHCP snooping configuration with the following command:

   ```
   Switch> enable
   Switch# show ip dhcp snooping
   ```

   Syntax:

8. Test the DHCP snooping by connecting a rogue DHCP server to the network and verifying that it is not able to provide IP addresses to the devices.

Dynamic ARP Inspection:

- **ARP** (Address Resolution Protocol) is a protocol used to **bind** an **IP** address to its corresponding **MAC** address.
- When a device needs to **communicate** with another device on the network, it sends an **ARP** request to determine the **MAC** address of the device with the corresponding **IP** address.
- ARP requests are **broadcast** messages, which means that they are sent to **all** devices on the network.
- This can create a security **vulnerability** in which an attacker can **intercept** and manipulate ARP messages to **redirect** traffic to a different device, also known as a **man-in-the-middle attack.**
  - The attacker sends a **fake ARP** reply to the sender, **claiming** to be the device with the **IP** address of the intended recipient. The fake ARP reply contains the MAC address of the attacker's device.
  - The sender **receives** the **fake ARP** reply and **updates** its ARP cache with the MAC address of the attacker's device, **thinking** it is the MAC address of the intended recipient.
  - The sender then **sends** the data packet to the attacker's device **instead** of the **intended** recipient.
  - The attacker can then **intercept** the data packet, **view** its contents, **modify** it, or **forward** it to the actual intended recipient **without** the sender's knowledge.
- **DAI** is a security feature used to **prevent** these types of attacks by allowing only **trusted interfaces** to **receive** and **forward ARP** messages on a network.
- DAI **works** by **inspecting** ARP messages and **verifying** their contents against a database of trusted **IP-to-MAC** address bindings, which is maintained by **DHCP snooping**.
- If the ARP message is from a **trusted** source and the IP-to-MAC address binding is **valid**, DAI will **forward** the ARP message to its intended recipient.
- If the ARP message is from an **untrusted** source or the IP-to-MAC address binding is **invalid**, DAI will **either drop** the ARP message or **log** the event, depending on the configuration.
- DAI also includes the ability to create ARP access control lists (**ACLs**) that can be used to **permit** or **deny** specific ARP messages based on **source** or **destination** IP addresses, source or destination MAC addresses, or other **criteria**.

- **Static** IP addresses that are not **assigned** by **DHCP** can still be protected by **DAI** by configuring the switch to **drop** ARP messages for those IP addresses or to **trust** the port to which the device is connected.

Authentication, Authorization, and Accounting:

AAA servers typically use the **RADIUS** or **TACACS+** protocol. Here's how AAA **works**,:

1. A user or device **attempts** to access a network resource.
2. The network device (e.g. a router, switch) **requests** authentication from the user/device.
3. The user/device **sends** their **credentials** (e.g. username and password) to the network device.
4. The network device forwards the **credentials** to the **AAA server** for verification.
5. The AAA server **checks** the **credentials** against a database or directory to **verify** their validity.
6. If the credentials are **valid**, the AAA server sends an **approval** message to the network device.
7. The network device **grants** the user/device access to the **requested** resource, based on their **authorization** level.
8. The AAA server **records** the user/device's activity on the network, **including** login/logout times and **resource** usage, for accounting purposes.

Authentication:

- **Verifies** the **identity** of a user or device **attempting** to access the network
- The user/device **provides credentials** (e.g. username and password, security certificate)
- The **credentials** are **checked** against a database or server to ensure they are **valid**
- If the credentials are **valid**, the user/device is **granted** access to the network

Authorization:

- Determines what **actions** and **resources** a user/device is **allowed** to access on the network
- The user/device's credentials are **checked** against a database or server to determine their **privileges**
- Based on their **privileges**, the user/device is **granted** access to **specific** resources and **actions** on the network

Accounting:

- Tracks and **records** the network **usage** of users/devices
- Records information such as **login**/**logout** times, **data** usage, and **actions** taken on the network
- This information can be used for **auditing**, **billing**, and **troubleshooting** purposes

WIRELESS NETWORKS FUNDAMENTALS:

WIRELESS NETWORKS FUNDAMENTALS:

- **Wireless** communication uses **electromagnetic** fields to **encode** data as **ones** and **zeros**.
  - **Electromagnetic** (EM) fields are a type of physical phenomenon that is produced by the **movement** of **electrically** charged **particles**.
    - **frequency-shift keying (FSK)**
    - **phase-shift keying (PSK)**
    - **amplitude-shift keying (ASK)**
  - In wireless communication, data can be encoded into an EM field in various ways.
- The **frequency** of a wave is **changed** to **represent** each **bit** of **data**.
  - Frequency is measured in **Hertz**, which represents the change in **frequency** per **second**.
- Modulation is used to **express** the **ones** and **zeros** in the encoded signal.
  - **Amplitude Modulation (AM)**
  - **Frequency Modulation (FM)**
  - **Phase Modulation (PM)**
  - **Quadrature Amplitude Modulation (QAM)**
- Wi-Fi has multiple generations, starting from **802.11a (2 Mbps)** and currently up to **802.11ax (14 Gbps Wi-Fi 6)**.
- several factors that can **impact** the actual data **transfer** speeds you can achieve over **Wi-Fi**, including:
  - **Distance**: The further away you are from the Wi-Fi access point, the weaker the signal strength will be, which can impact data transfer speeds.
  - **Interference**: Wi-Fi signals can be interfered with by other wireless signals, such as those from other Wi-Fi networks or nearby electronic devices, which can impact data transfer speeds.
  - **Network congestion**: If multiple devices are connected to the same Wi-Fi network and are transferring data simultaneously, this can cause network congestion and impact data transfer speeds.
  - **Device limitations:** The maximum data transfer speeds you can achieve over Wi-Fi will depend on the capabilities of your device's Wi-Fi hardware and software.
- The **encoder** responsible for **turning** ones and zeros into an **electromagnetic** field is called a **transceiver**.
  - A transceiver (**transmitter-receiver**) is a device that can both **transmit** and **receive** signals, and is responsible for **encoding** and **decoding** the data.
  - The number of **transceivers** available **affects** the **amount** of data that can be **encoded**.
- The **encoded** signal is pushed through an **antenna**, and the number of **antennas** available **affects** the **amount** of data that can be **transmitted**.
  - An **antenna** is a component that **converts** electrical signals into electromagnetic waves and **vice versa**, and is used to **transmit** and **receive** signals in wireless communication.
- Power is **required** to **generate** and **push** data through the air, which can come from a **battery** or an **AC adapter**.
- The **power** of a **frequency** is measured in **amplitude**.

    - **Amplitude** is the **strength** or magnitude of a waveform

Wireless Network Components:

Wi-Fi Client (End Point):

- Also called a **"Station"**
- **Generates**/**Consumes** Data
- Has **Transceivers** (to **encode** data)
- Has **Antennas** (to **push** the data)
- Needs Power to **operate**
- **Example:**
    - A laptop computer
    - A smartphone

Wi-Fi Access Point (AP):

- Acts as the **gateway** for the **stations**
- Stations **talk** to each other through the **AP**
- Has **Transceivers** (to **receive** data from stations and **send** to other stations)
- Has **Antennas** (to **push**/**receive** data)
- Needs Power to **operate**
- An Ethernet switch is typically used to connect **wired** devices to a network. In the case of a Wi-Fi network, the AP is connected to an **Ethernet** switch to **allow wired** devices to **communicate** with wireless devices connected to the **AP**.
- **Example:**
    - A wireless router used in a home or small office

Wi-Fi Controller:

- **Controls** Access Points (APs)
- Acts as the **central** point of **management** for the network
- **Controls Access** for clients (Authentication, Authorization, and Accounting)
- A **hardware**-based controller such as a Cisco Wireless LAN Controller (**WLC**)

Types of Wi-Fi Networks:

Ad-Hoc:

- Also known as a **peer-to-peer network**
- Wireless devices communicate **directly** with each other **without** the need for an access point (**AP**)
- Devices must be within **range** of each other to communicate
- Useful for temporary **networks** or **situations** where an infrastructure network is **not** available
- Requires **line of sight** between the two devices
- **Examples:**
    - **Two** laptops communicating with each other **without** the need for an **AP**

Infrastructure:

- The **most** common type of Wi-Fi network
- Wireless devices **communicate** with each other through **one** or **more** access points (**APs**)
- **Multiple** APs can be used to provide **coverage** over a **large** area
- **Examples:**
    - A home wireless network with a **wireless router** serving as the **AP**

Mesh:

- A network where **multiple APs** communicate with each other wirelessly to provide **coverage** over a **large** area
- No **single** AP acts as a **central** point of control - all **APs** communicate with **each** other **equally**
- Useful for providing coverage over **large** outdoor areas or in situations where **running** cables is not **practical**
- **Examples:**
    - A smart city deployment with **multiple** APs providing **Wi-Fi coverage** throughout a **large** outdoor area

Wi-Fi Terms:

Basic Service Set (BSS):

- A **single** access point (**AP**) and **its** coverage area
- Wireless devices can **communicate** with the **AP** and other devices **within** the BSS
- The **BSS** is identified by a unique **identifier** called the **BSSID**

Basic Service Set Identifier (BSSID):

- The **MAC address** of the access point (**AP**)
- Used to **uniquely identify** the BSS

Service Set Identifier (SSID):

- The **name** of the wireless local area network (**WLAN**)
- Used to **identify** the wireless network to wireless devices
- Devices must be **configured** with the **correct** SSID to connect to the network

Distribution System (DS):

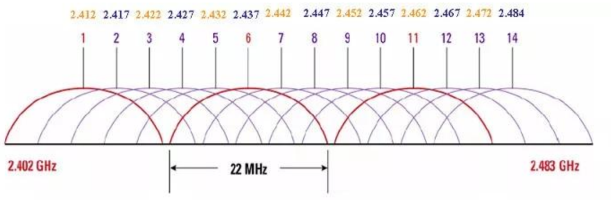
- The **wired** network that **connects** the AP to the local area network (**LAN**)
- **Allows** wireless devices connected to the **AP** to communicate with **devices** on the **LAN** and the internet

Extended Service Set (ESS):

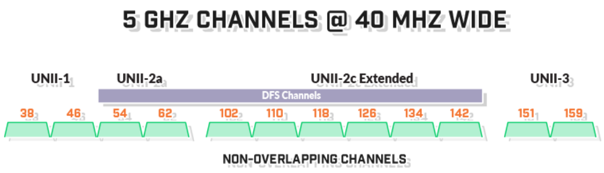
- A **collection** of **APs** connected to the **same** DS, offering the same **WLAN and SSID**
- Allows wireless devices to **move** between **coverage** areas of different APs **without** losing **network** connectivity
- Commonly used in **large-scale** Wi-Fi deployments such as **hotels** or **public** Wi-Fi hotspots

Wi-Fi Channels:

- Wi-Fi channels are a **range of radio frequencies** (**RF**) that are used by Wi-Fi **transceivers** to **encode** and **transmit** data.
- Each **frequency** can be **modulated** differently to **allow** for **more** encoding options and **better** data transfer rates.
- The **total RF bandwidth** of the channels is called **channel bandwidth**.
- Channels include **frequencies** either from the **2.4 GHz** range or from the **5 GHz** range.
  - The **2.4 GHz** range has been used for **Wi-Fi** for many years and is **commonly** used for **home** and **small** office networks.
  - The **5 GHz** range provides **more** channels and **less** interference, making it a **better** choice for **larger** and more **complex** networks.
- If two channels are too **close** together and are streaming some **common** frequencies, **overlapping** can happen, which can cause **interference** and **degrade** performance.
  - **Overlapping** can happen with **2.4 GHz** channels **only**, which come in a **20 MHz** width.This means that each channel includes a range of frequencies that is 20 MHz wide.



  - With **5 GHz** channels, a new channel starts with a frequency right after the last channel's frequency ended, so overlap **won't** happen.
  - **5 GHz** channels support from **20 MHz** width up to **160 MHz**, which means more frequencies can be included, allowing for **more data** to be **encoded**.



- The wider the **channel bandwidth**, the more **frequencies** are included, which can lead to **better** data **transfer** rates.

WLAN Architectures:

- **Autonomous Architecture:**
  - Uses autonomous **(independent) access** points.
  - Access points are managed **individually** through a **GUI**.
  - Supports one or more **SSIDs**, with each **SSID** corresponding to one **VLAN**.
    - For example, a network **administrator** might create **two** SSIDs, one for **employees** and one for **guests**.
  - When multiple SSIDs are used, the **back link** must be a **trunk**.
    - If the **back link** between the **access** point and the **switch** is **not** configured as a **trunk** link, traffic from both VLANs will be transmitted over the same VLAN, which can lead to **security issues** and **traffic** congestion.
  - Adding a **new** SSID requires **logging** in to **each** access point individually.
- **Split-MAC Architecture:**
  - Includes a Wireless LAN Controller (**WLC**).
  - Access points are called Lightweight APs (**LAPs**).
  - WLCs manage **RF**, **QoS**, **AAA**, and **policies**.
  - LAPs handle **transmitting** and **receiving** frames through the RF (Radio Frequency) , RF **collision detection**, **MAC** and data **management**.
  - There is a private tunnel between the WLC and LAPs called "Control and Provisioning of Wireless AP" (**CAPWAP**).
    - **CAPWAP** encapsulates and **transfers** all control and data information between the **WLC** and **LAPs.**
    - Two tunnels are used: **control tunnel** (UDP5246) and **data tunnel** (UDP5247).
    - Control tunnel is **encrypted** and authenticated, while data tunnel is **not encrypted** by default.
- **Cloud-Based Architecture:**
  - Includes a **WLC**, but it is **remotely** accessed through a **public** or **private** cloud.
  - Uses **LAPs**, which may be Cisco **Meraki** or Cisco Cat. 9800-CL.
  - Self-configures to the LAPs.

  - **CAPWAP** is used to transfer **control** and **data** information between the **WLC** and **LAPs**.

WLC Positioning:

- **Centralized WLAN Architecture:**
  - Uses a **single WLC** to control **all** the LAPs.
  - Can be placed in the **data center** or **near** the **edge** of the network.
  - **All** data must pass through the **CAPWAP** tunnel to reach the **WLC**, even if the **destination** is **closer** than the WLC.
  - Cisco **Flex Connect** can be used to **fix** this issue by enabling a mode on the **LAPs** that allows them to **pass** traffic **directly** to the **LAN**.
    - Flex Connect also allows **LAPs** to **authenticate** clients for access and work even if the CAPWAP tunnel goes **down**.
- **Converged WLAN Architecture:**
  - Connects a **WLC** and an **AP** to the **same** switch, typically an **access/distribution** layer switch.
  - **Multiple** WLCs may be needed in this scenario.
  - Results in a **shorter** distance CAPWAP and **faster** Wi-Fi with **fewer** delays.
  - Cisco **Catalyst 9300** series provides switches that can have a **WLC integrated** inside the switch itself (**embedded**).

AP Modes:

- **Local Mode:**
  - **Default** mode of a Lightweight AP (**LAP**).
  - Uses **CAPWAP** to communicate with the **WLC**.
  - **All** data passes through the **CAPWAP** tunnel, and if the tunnel **fails**, clients will be **disconnected**.
- **Bridged Mode:**
  - **Autonomous AP** works **independently** of a **WLC** and has its **own** management interface.
  - **Bridged** Mode allows an **Autonomous** AP to **connect** as a client to a **LAP**, managed by a **WLC**.
  - In Bridged Mode, the Autonomous AP can be **managed** and controlled by the **WLC**, just like **any** other LAP.
  - Bridged Mode provides greater **flexibility** between different **types** of APs.
  - Although Autonomous APs lack advanced features of LAPs managed by a WLC, they are **easier** to set up and manage.
  - Bridged Mode allows for centralized **management**, **security**, and advanced features **available** with LAPs managed by a WLC.
- **Flex Connect Mode:**
  - Allows **LAPs** to operate when the **CAPWAP** tunnel to the WLC is **down**.
  - Allows LAPs to **pass** traffic **directly** to the **LAN**.
  - LAPs can **authenticate** clients for access and work even if the **CAPWAP** tunnel goes **down**.
- **Monitor Mode:**
  - **Monitor** Mode is a feature on **LAPs**.
  - It generates **reports** and **statistics** on wireless network **activity**.
  - LAPs **send** the information to the **WLC** for analysis.
  - LAPs **don't** participate in data **transmission**.
  - It enables **monitoring** of wireless traffic without **active** participation.
- **Sniffer Mode:**
  - **Sniffer** Mode is available on some wireless access points, including **LAPs**.
  - It **scans** a **specific** channel or frequency **band**.
  - LAPs act as **passive** sniffers, **capturing** all wireless traffic.
  - The captured data is used for **analysis**.
  - **sends** scanning reports to the **WLC**.
- **Sensor Mode:**
  - **Sensor** Mode is available on some wireless access points.
  - It performs tests on **SSIDs,** can include **checking** for connectivity, **verifying** that the **correct** security settings.
  - Access points act as **sensors** and continuously **monitor** the wireless network.
  - Test reports are **sent** to the Cisco **DNA Center**.
  - The mode ensures that SSIDs are **functioning** correctly.
- **Mesh Mode:**
  - In **Mesh** Mode, access points form a **wireless mesh** network.
  - Mesh Mode **extends** wireless coverage in areas **without** wired connectivity.
  - Mesh nodes (**MAPs**) use (**AWPP)** to determine the **best** path to a root node or Access Point (**RAP**).
    - **MAPs** are wireless **access** points that **relay** wireless signals in a wireless mesh network.
    - **RAPs** are **access** points that are connected to the **wired** LAN and serve as **gateways** to the Internet or other network resources.
  - **AWPP** dynamically adjusts **routing** paths **based** on network topology or wireless interference.
  - AWPP ensures **timely** and **efficient** delivery of frames to the destination.

Other Information:

- Some APs have **PoE** and **AUX** ports in the back that can be **bundled** to form a **higher bandwidth** data interface.
- **WLCs** have a **Service**/**Management port** that can have an **IP** address assigned to it for **GUI** access.
- To **bundle**/**aggregate** ports:
  - **WLC**: use "`channel-group mode on`" on the **switch**, as it does not support **LACP**/**PAgP**.
  - **AP**: use either "**ON**" or "**LACP**," but only with **local** APs, **not** autonomous APs.
- **APs** and **WLCs** can be managed by **CLI** (console, telnet, ssh) and **GUI** (http and https).
- Authorization access can be done using **AAA**.

Cisco devices :

  - **Meraki MX65W:**
    - It is a security appliance that provides firewall, VPN, and content filtering capabilities.
    - It is designed for small to medium-sized businesses.
    - It can be managed through the cloud-based Meraki Dashboard.
    - Uses: It is used to secure the network and protect against various threats such as malware, viruses, and unauthorized access.
  - **LAP-PT:**
    - It is a Lightweight Access Point (LAP) that is used to provide wireless connectivity in a Cisco wireless network.
    - It can be managed through a Wireless LAN Controller (WLC).
    - Uses: It is used to provide wireless connectivity for laptops, mobile devices, and other wireless clients.
  - **WLC-PT:**
    - It is a Wireless LAN Controller (WLC) that is used to manage Cisco wireless access points in a centralized manner.
    - It can be simulated using the Packet Tracer software.
    - Uses: It is used to manage and configure Cisco wireless access points, provide security policies, and monitor network performance.
- **HomeRouter PT-AC:**
  - It is a home router that provides wireless connectivity and wired Ethernet ports for connecting devices to the Internet.
  - It can be simulated using the Packet Tracer software.
  - Uses: It is used to provide Internet connectivity for home or small office networks, and to share resources such as printers and storage devices.
- **WLC-2504:**
  - It is a Wireless LAN Controller (WLC) that is used to manage Cisco wireless access points in a centralized manner.
  - It can be accessed through a web-based management interface or through the command-line interface (CLI).
    1. Connect a computer to the WLC-2504 using a console cable or by connecting it to the same network as the WLC.
    2. Open a web browser on the computer and enter the IP address of the WLC-2504 in the address bar.
    3. Enter the username and password to log in to the WLC-2504. The default username is "admin" and the default password is "password".
    4. Once logged in, you can access the web-based management interface of the WLC-2504. From here, you can manage and configure the various settings of the WLC-2504 and its associated wireless access points.
  - Uses: It is used to manage and configure Cisco wireless access points, provide security policies, and monitor network performance.

Wi-Fi Security:

Unsecured WLANs:

- WLANs with **no** password, **free**, and **public**.

Secured WLANs:

- WLANs that may have a **hidden** SSID, **authentication**, and **encrypted** data.
- Authentication can be done by authenticating the user's **credentials**, authenticating the device's **MAC** address, or using a **captive** portal.
  - **Authenticating the user's credentials:**
    - This method requires the user to enter a **username** and **password** to access the network.
  - **Authenticating the device's MAC address:**
    - This method involves creating a **whitelist** of **MAC** addresses that are allowed to access the network.
  - **Using a captive portal:**
  - A captive portal is a **web page** that is displayed to users when they **attempt** to access the network.
  - The user is required to enter some form of **authentication**, such as a **username** and password or a **voucher** code, to access the network.

Encryption:

- **Data** frames are **encrypted**, but **management** frames are **not**.
- Encryption happens between the **client** and the **AP** only.
- To have end-to-end encryption, use **HTTPS**, which sends a **digital** certificate between the **source** and **destination**.

Wi-Fi Protected Access (WPA):

- **Two** types: **personal** and **enterprise**.
- **Personal** uses a **passphrase** and a **256-bit pre-shared key** ex **RC4 , TKIP,** and **MIC** for encryption.
  - A **passphrase** is a combination of **characters** used in Wi-Fi network security.
  - TKIP encrypt **every** packet with a **unique** key
- **Enterprise** uses **802.1X** and **EAP** packets for **authentication** and **encryption**.
  - **Authentication:**

        - 802.1X involves **three** components: the supplicant (**client**), the authenticator (**access point**), and the **authentication server**.
        - The **supplicant** sends packets carried by the Extensible Authentication Protocol (**EAP**) to the **authenticator** for **authentication**.
        - 802.1X authentication occurs only between the **supplicant** and the **authenticator**.
        - The **authenticator** then forwards the **EAP** packets to the **authentication** server, which **verifies** the supplicant's identity.
        - The authentication server is typically a Remote Authentication Dial-In User Service (**RADIUS**) server.
            - **RADIUS** is a network protocol used for **AAA** in computer networks.
    - **Encryption:**
        - After authentication is **complete**, encryption is **applied**.
        - Encryption is done by the **authentication** server, which generates a **unique** key for each **client** device.
        - The key is used to encrypt **data** between the **client** device and the **access** point.
        - This provides a **higher** level of security than the **pre-shared** key used in WPA-**Personal** mode.

WPA2:

- Also has **personal** and **enterprise** modes.
    - **Personal mode:**
        - In Personal mode, a **passphrase** is used to generate a pre-shared key.
        - The pre-shared key is used to **encrypt** data between the **client** and the **access** point.
        - Encryption can use either **AES-CCMP** or **RC4+TKIP**.
        - Personal mode provides **encryption** from the **client** to the **access** point **only**.
    - **Enterprise mode:**
        - In Enterprise mode, **802.1X** is used for **authentication**, which involves a **RADIUS** server.
        - 802.1X can support **re-authentication**, which allows for **faster** authentication and **better** network performance.
        - Encryption is also used here, and it can use **AES-CCMP** or **RC4+TKIP**.

WPA3:

- Has **personal** and **enterprise** modes.
- Supports "**Enhanced Open**" Wi-Fi and "**Wi-Fi Easy Connect**" for IoT devices.
    - **Enhanced Open:**
        - Enhanced Open is a new security standard introduced in WPA3.
        - It provides improved **security** for **public** Wi-Fi networks, such as those found in **coffee** shops, **airports**, and **hotels**.
        - Enhanced Open uses Opportunistic Wireless Encryption (**OWE**) to **encrypt** traffic between the **client** device and the **access** point.
        - OWE allows for **encryption without** requiring any **authentication** or pre-shared key.
        - Enhanced Open provides **protection** against **passive** eavesdropping attacks, but it does **not** provide protection against **active** attacks.
- Personal uses SAE instead of a pre-shared key and protects against offline dictionary attacks.
    - **SAE:**
        - SAE provides **stronger** security against offline dictionary attacks by using a more **complex** and **random** key generation process.
        - SAE also provides **forward** secrecy, which means that **even** if a **key** is **compromised**, previous sessions are not **compromised**.
        - SAE reduces the risk of password-based attacks by using a **password**-based key **derivation** function.
    - **Offline dictionary attacks:**
        - An offline dictionary attack is a type of attack where an attacker attempts to **guess** a password by using a **list** of **known** passwords.
        - In Wi-Fi network security, an offline dictionary attack involves an attacker **gaining** access to the **encrypted** password or **pre-shared key** used for authentication and attempting to guess the **password** by using a **list** of **known** passwords.
        - This type of attack can be carried out **offline**, meaning that the attacker does not need to be **connected** to the **Wi-Fi** network to carry out the attack.
- Enterprise uses 192-bit minimum cryptographic security suite.
    - A 192-bit cryptographic security suite provides a higher level of security than a 128-bit security suite, making it more difficult for attackers to carry out these types of attacks.

AUTOMATION & PROGRAMMABILITY:

Automation:

Traditional Network Management:

- **Installation** and **initial** configuration of network devices.
    - **Examples**
        - **Connect the device**
        - **Assign IP Address**
        - **Access the Configuration Interface**
        - **Configure the Network Settings**
        - **Set Administrative Password**
        - **Configure Protocol Settings**
        - **Configure Security Settings**
- **Modifying** and updating existing configurations.

- **Upgrading** software.
- Achieved through **console**, **Telnet**, **SSH,** applying scripts, or **copying** configurations.
- Monitoring achieved through **SNMP** and **Netflow**.
- Managed "**box-by-box**".
    - refers to the management of network devices **individually**, **rather** than using a **centralized** management system.

Automation:

- New devices can **automatically** find an **initial** configuration.
    - **DHCP**
    - **Zero-Touch Provisioning**: Zero-Touch Provisioning (ZTP) allows new devices to automatically **download** and **install** an initial configuration from a **centralized** management system.
    - **Plug-and-Play**: Some devices support plug-and-play, which allows them to **automatically** detect **network settings** and configure themselves **accordingly**.
    - A**uto-Discovery:** Devices can use auto-discovery protocols, such as Cisco Discovery Protocol (**CDP**), to **discover** neighboring devices and obtain **information** about their configuration.
- **Automated QoS** and **AAA** profiles/configurations.
    - **QoS Profiles:** Automated QoS profiles can be pre-configured for different types of traffic, such as voice or video, to ensure that they receive the appropriate level of priority and bandwidth.
    - **AAA Configurations**: Automated AAA (Authentication, Authorization, and Accounting) configurations can be used to define access policies, such as who is allowed to access the network and what level of access they have.
    - **Templates:** QoS and AAA configurations can be pre-configured as templates, which can be applied to new devices automatically when they are added to the network.
    - **Centralized Management**: Automated QoS and AAA profiles/configurations can be centrally managed and deployed across the network, ensuring consistency and reducing the likelihood of configuration errors.
    - **Dynamic Updates**
- Utilizes **scripts** and **tools** to **standardize** procedures.
    - **Software** image **per** device model and **upgrade** procedure.
- **Scheduled** operations.
- Automated **troubleshooting** .
    - Managed through **CLI**, **SSH**, **SNMP**, **NETCONF**, and **RESTCONF**.
- Topology visualization and monitoring achieved through **SNMP** Manager and **Netflow** Collector.
- Reduces or eliminates "**box-by-box**" management.
- Requires **smaller** staff, saves **time**, and ensures configuration **consistency**.

Software-Defined Networking (SDN):

- SDN is a network architecture that utilizes **software** to **control** and **manage** the network.
- Automation is achieved by **SDN**, allowing for more **efficient** configuration and **management** of network devices.
- A software-based approach is used to run and **administrate** the network, using a **controller** to manage and automate network functions.
    - The SDN controller is a **key** component of the SDN architecture, responsible for **controlling** and **implementing** automation and **administration** across the network.
    - The SDN controller can be either a **software** application installed on a **server**, an appliance with a **built-in controller** (such as Cisco's **APIC** or **DNA Center**), or a **remote** controller through the **cloud**.
    - SDN controllers use **scripting** languages like **Cisco TCL** or **Python** to enable automation and **customization** of network functions.
- Some tools and applications, such as **Puppet**, **Chef**, and **Ansible**, can be used to assist with SDN automation and management tasks.
- SDN offers several **benefits**, including increased network **flexibility**, improved **scalability**, and **simplified** management and administration.

SDN Implementation:

Imperative Approach:

- In the Imperative approach, the **control plane** logic **resides** completely in the **controller**.
- The SDN controller has **complete** control over **programming** the forwarding **decisions** of the networking devices.
- Devices will **ask** the controller **before** any **forwarding** or **routing** action is taken.
- The controller **makes all** the **decisions** and provides **specific** instructions to the devices on how to **forward** packets.
- This approach requires that the **controller** be **highly available** and reliable, as it is responsible for **all** forwarding **decisions** in the network.

Declarative Approach:

- In the Declarative approach, the control plane **resides within** the network device, as it does in **traditional** networking.
- The SDN controller **declares** the **requirements** of **all** the **forwarding**/**routing** decisions to the **networking** devices.
- The network **devices** then **decide** how to translate the controller instructions into actions.
- This approach is more **distributed** than the Imperative approach and can be more **scalable** in larger networks.
- The network **devices** have more **autonomy** in making forwarding **decisions**, but they must still **comply** with the controller's **requirements**.

SDN Architecture:

Underlay Network:

- The Underlay Network is **responsible** for providing **basic connectivity** between network devices.
- It uses **protocols** and features to **establish reachability** between devices.

- All links in the Underlay Network must be **Layer 3** and **point-to-point**.
    - **Layer 2** connectivity can be used in the Underlay Network, but it is generally **not recommended** in an SDN **architecture** due to several **limitations**.
        - broadcast and flooding
        - Spanning Tree Protocol (STP),
        - scalability limitations,
        - security vulnerabilities such as MAC address and ARP spoofing.
        - As a result, **Layer 3** connectivity is typically **preferred** for the Underlay Network in an SDN architecture as it provides more **efficient routing**, **scalability**, and **security**.
- Open standard protocols like **OSPF** and **IS-IS** are commonly used in the Underlay Network.

Overlay Network:

- The Overlay Network is a **virtual** network created on **top** of the Underlay Network.
- The **Underlay** Network becomes the **physical** connectivity for the **Overlay** Network.
- **Protocols** like **VRF**, **MPLS-VPN**, and **VXLAN** are commonly used to create the Overlay Network.
- The Overlay Network is used to provide network **services** and **enable** network **virtualization**.

SDN Fabric:

- The SDN Fabric consists of the **physical** devices used to **build** the Underlay Network.
- These devices can be **controlled** by an SDN **Controller**.
- The SDN Controller is responsible for **managing** and **automating** the devices in the SDN Fabric.
- The SDN Fabric can be composed of **various types** of network devices, including **routers**, **switches**, and **firewalls**.

SDN Effect upon Planes:

Three Planes of Network Devices:

- **Control Plane:** Responsible for **learning information** from **protocols** and **downloading** that information to the **Data Plane** as **tables**.
- **Data Plane:** Also known as the **Forwarding** Plane, responsible for **controlling** the **forwarding** of frames and packets using **tables**.
- **Management Plane:** Responsible for **managing** device **configuration**, remote **authentication**, and **console** access.

Effect of SDN on Planes:

- The **effect** of SDN on the **Control** and **Data** Planes depends on the **implementation** approach.
    - In an **Imperative** approach, also known as **Stateful** SDN, the **controller** is responsible for learning information and downloading it to the Data Plane. If devices **lose** connectivity to the controller, they may become **powerless**.
    - In a **Declarative** approach, also known as **Stateless** SDN, the controller only **declares** how it wishes the network to **operate**, and the **devices** make their **own** forwarding decisions **based** on that information.

Cisco DNA Center:

- Cisco DNA Center is an **appliance** designed to provide **centralized management**, **automation**, and **analysis** for networks.
- It enables **Intent-Based** Networking, which **allows** networks to be **controlled** through **software**.(**SDN**).
- It allows the **creation** and management of **topology** maps, WLAN **SSIDs**, and **more** through a **GUI**.
- It includes a built-in **APIC** (Application Policy Infrastructure Controller) for network **control** and a built-in **NDP** (Network Data Platform) for problem **analysis** and **solution** suggestions.
- The Cisco DNA Center Sandbox is a cloud-hosted environment that provides a virtual instance of Cisco DNA Center for testing and learning purposes.

Cisco DNA Center Platform

Automate and operate your network to configure and observe at scale, using the Cisco DNA Center REST API.

https://developer.cisco.com/docs/dna-center/#!cisco-dna-center-sandbox/cisco-dna-center-sandbox-overview

Application Programming Interface:

APIs:

- APIs are **code** that **encodes data** and transforms it between **different applications** and **devices**.
- **Northbound** APIs transform data between **applications** and **controllers**, while **Southbound** APIs transform data between **controllers** and **network devices**.
    - **Southbound** APIs include **Openflow**, **Cisco OpFlex**, Command Line Interface (**CLI**), Simple Network Management Protocol (**SNMP**), and Network Configuration Protocol (**NETCONF**).
    - **Cisco Intent** is a **Northbound** REST-Based API used in Cisco DNA Center for intent-based networking.
- APIs use a **Server/Client** relationship, with the **Server providing** the data and the **Client** making **requests** for that data.

API Types:

- **Internal APIs** are used between **applications** to **exchange** data, such as **transferring** data from **HTML** to **PDF**.
- **Web-Service APIs** are used to **exchange** data between **remote devices** and use **IP** addresses to **communicate**, such as **REST**-**Based APIs**.

REST-Based APIs:

- REST-Based APIs are the **most** common type of **web-service** API and are mostly found in **Northbound** APIs.
- They use **HTTP** verbs such as **GET**, **PUT**, **POST**, and **DELETE**.
    - **GET** is used to **retrieve** data from the server.
    - **PUT** is used to **update** or **replace** data on the server.
    - **POST** is used to **create** new data on the server.

  - **DELETE** is used to **delete** data from the server.
- During development, a **developer** would use **CRUD** (**Create**, **Read**, **Update**, and **Delete**) to develop the API's **HTTP** verbs.
  - CRUD is a framework used by developers to create REST-Based APIs.
  - For example, to create data, the developer would use the HTTP POST verb. To read data, the developer would use the HTTP GET verb. To update data, the developer would use the HTTP PUT verb. To delete data, the developer would use the HTTP DELETE verb.
- The most common **languages** used to **encode** data in a REST-Based API are **XML** and **JSON**.
- **Encoding** means **standardizing** a data structure between the **application**, **controller**, and **nodes**.

Configuration Management Mechanisms:

Configuration Management Mechanisms:

- Configuration management mechanisms are applications used to **automate** the **configuration** of network devices.
- These mechanisms typically require **CLI/scripting knowledge**, but also often include a **GUI** for ease of use.
- They allow for **scheduling** of **tasks** and the manual instantiation of **events**.

Puppet and Chef:

- Puppet and Chef are configuration management tools that use a **Master/Agent** model.
- There are **two codes** involved in the process: **one** in the **server** and the **other** in the **node**.
- These tools use a **Pull** Model, where an agent **periodically asks** the **master** for events and actions, and **pulls** the script from it.
- Puppet and Chef use the **RUBY** language.

Ansible:

- Ansible is a configuration management tool that is **agentless**, meaning it does **not** require an **agent** to be **installed** on the **node**.
- Ansible uses a **Push** Model, where the master **pushes** a **config** to the agent for it to **apply** to the **node**.
- Ansible uses the **YAML** language.

Java-Script Object Notation (JSON):

- JSON is a programming language used to create **APIs**, commonly used by **REST**-**Based APIs**.
- It is human-readable and lightweight.
- The "**Object**" in JSON is a container that encloses one or more $\{\text{name} : \text{value}\}$ pairs, also called key-value pairs.
- JSON **values** are always surrounded by curly brackets $\{\}$.
- A **string** value must be enclosed with double quotes **""**.

Name:Value Pairs in JSON:

- The value **types** in a name:value pair can be **categorized** into several **types**.
  - **String pairs** consist of both the name and value being strings, enclosed in **double quotes**.
  - **Number pairs** consist of a string name and a numeric value, **without** double quotes.
  - **Array pairs** consist of a string name and an array of values, enclosed in square brackets **[]**.
  - **Boolean pairs** consist of a string name and a boolean value (**True/False**), **without** double quotes.
  - **Null** pairs consist of a string name and a **null** value.

Extra

Extra-Labs

Lab1

Scenario:

A building has 4 floors - Account, Sales, IT, and HR. The IP address 192.168.1.0/24 needs to be subnetted for the 4 floors to avoid wasting IP addresses. Account requires 10 IPs, Sales requires 25 IPs, IT requires 50 IPs, and HR requires 34 IPs. Each floor has a switch connected to a main switch, which is connected to a router. The router is connected to a server through another switch, and the server's IP is 172.16.1.0/30.

```
graph TD
    subgraph Account
        PC1((PC1))
        PC2((PC2))
    end
    subgraph Sales
        PC3((PC3))
        PC4((PC4))
        PC5((PC5))
        PC6((PC6))
    end
    subgraph IT
        PC7((PC7))
        PC8((PC8))
        PC9((PC9))
        PC10((PC10))
        PC11((PC11))
        PC12((PC12))
        PC13((PC13))
        PC14((PC14))
    end
    subgraph HR
        PC15((PC15))
        PC16((PC16))
        PC17((PC17))
        PC18((PC18))
        PC19((PC19))
        PC20((PC20))
    end

    subgraph Switches
        Switch1((Switch1))
        Switch2((Switch2))
        Switch3((Switch3))
        Switch4((Switch4))
        Switch5((Switch5))
        Switch6((Switch6))
        Router1((Router1))
        Server1((Server1))
    end

    PC1 --192.168.1.160/28--- Switch1
    PC2 --192.168.1.160/28--- Switch1
    PC3 --192.168.1.128/27--- Switch2
    PC4 --192.168.1.128/27--- Switch2
    PC5 --192.168.1.128/27--- Switch2
    PC6 --192.168.1.128/27--- Switch2
    PC7 --192.168.1.0/26--- Switch3
    PC8 --192.168.1.0/26--- Switch3
    PC9 --192.168.1.0/26--- Switch3
    PC10 --192.168.1.0/26--- Switch3
    PC11 --192.168.1.0/26--- Switch3
    PC12 --192.168.1.0/26--- Switch3
    PC13 --192.168.1.0/26--- Switch3
    PC14 --192.168.1.0/26--- Switch3
    PC15 --192.168.1.64/26--- Switch4
    PC16 --192.168.1.64/26--- Switch4
    PC17 --192.168.1.64/26--- Switch4
    PC18 --192.168.1.64/26--- Switch4
    PC19 --192.168.1.64/26--- Switch4
    PC20 --192.168.1.64/26--- Switch4

    Switch1 --- Switch5
    Switch2 --- Switch5
    Switch3 --- Switch5
    Switch4 --- Switch5
    Switch5 --- Router1
    Router1 --172.16.1.0/30--- Switch6
    Switch6 --- Server1
```

Network Topology:

- Connect PC0 and PC1 to Switch 1.
- Connect PC3, PC4, PC5, and PC6 to Switch 2.
- Connect PC7, PC8, PC9, PC10, PC11, PC12, PC13, and PC14 to Switch 3.
- Connect PC15, PC16, PC17, PC18, and PC19 to Switch 4.
- Connect Switch 1, Switch 2, Switch 3, and Switch 4 to Switch 5 using trunk links.
- Connect Switch 5 to Router 1.
- Connect Router 1 to Switch 6.
- Connect the server to Switch 6.

Problem:

The problem is that each floor is separated by a switch, so VLANs will be used. However, there is a small problem with the default gateway. ROAS (Router on a Stick) can be used to solve this issue.

Solution:

1. **Subnetting:**

   Sorting the subnets from the largest to the smallest when subnetting is a best practice that helps you allocate IP addresses efficiently and minimize wasted IP addresses.

   - **Maximizing address usage:** Allocating the largest subnet first ensures that you use the available address space effectively. If you allocate smaller subnets first, you may find that there isn't enough contiguous address space left for the largest subnet, which can lead to wasted IP addresses.
     - **Easier management:** Allocating subnets in a consistent order makes it easier to manage your network. When you have a clear structure in place, it's simpler to track IP allocations, troubleshoot issues, and plan for future expansion.
     - **Simpler calculations:** When you allocate subnets from largest to smallest, you can use the remaining address space more effectively. This method allows you to allocate smaller subnets within the leftover address space without needing to perform complex calculations.

   Step 1: Determine the smallest subnet size for each department

   1. **Account:** 10 IP addresses **require** $10 + 2$ (2 address for network id and broadcast id)$- > 2^4 = 16$(smallest subnet size is $/28$) since (32-4)
   2. **Sales:** 25 IP addresses **require** $25 + 2$ (2 address for network id and broadcast id) $- > 2^5 = 32$ (smallest subnet size is $/27$) since (32-5)
   3. **IT:** 50 IP addresses **require** $50 + 2$ (2 address for network id and broadcast id) $- > 2^6 = 64$(smallest subnet size is $/26$) since (32-6)
   4. **HR:** 34 IP addresses **require** $34 + 2$ (2 address for network id and broadcast id) $- > 2^6 = 64$ (smallest subnet size is $/26$) since (32-6)

   Step 2: Allocate subnets

   1. **IT:**
      - **Subnet:** $192.168.1.0/26$
      - **Usable IP range:** $192.168.1.1 - 192.168.1.62$
      - **Broadcast address:** $192.168.1.63$
      - **Subnet mask:** $255.255.255.192$
   2. **HR:**
      - **Subnet:** $192.168.1.64/26$
      - **Usable IP range:** $192.168.1.65 - 192.168.1.126$
      - **Broadcast address:** $192.168.1.127$
      - **Subnet mask:** $255.255.255.192$
   3. **Sales:**

    - **Subnet:** $192.168.1.128/27$
    - **Usable IP range:** $192.168.1.129 - 192.168.1.158$
    - **Broadcast address:** $192.168.1.159$
    - **Subnet mask:** $255.255.255.224$

4. **Account:**
    - **Subnet:** $192.168.1.160/28$
    - **Usable IP range:** $192.168.1.161 - 192.168.1.174$
    - **Broadcast address:** $192.168.1.175$
    - **Subnet mask:** $255.255.255.240$

   since for example **IT:** • Total IP addresses: $2^6 = 64$ • Required IP addresses (including network and broadcast): $50 + 2 = 52$ • Wasted IP addresses: $64 - 52 = 12$

2. **Configure VLANs on switches:**
    - **For each switch (switch1, switch2, switch3, switch4, switch5), perform the following configurations:**

      ```
      Switch> enable
      Switch# configure terminal
      Switch(config)# vlan 10
      Switch(config-vlan)# name Account
      Switch(config-vlan)# exit
      Switch(config)# vlan 20
      Switch(config-vlan)# name Sales
      Switch(config-vlan)# exit
      Switch(config)# vlan 30
      Switch(config-vlan)# name IT
      Switch(config-vlan)# exit
      Switch(config)# vlan 40
      Switch(config-vlan)# name HR
      Switch(config-vlan)# exit
      ```

      Syntax:

    - **verify**

      ```
      Switch> enable
      Switch# show vlan brief
      ```

      Syntax:

3. **Assign access ports for each floor's switch:**
    - **switch1 (Account):**

   ```
   Switch> enable
   Switch# configure terminal
   Switch(config)# interface range fa0/2-3
   Switch(config-if-range)# switchport mode access
   Switch(config-if-range)# switchport access vlan 10
   ```

   Syntax:

    - **switch2 (Sales):**

   ```
   Switch> enable
   Switch# configure terminal
   Switch(config)# interface range fa0/2-5
   Switch(config-if-range)# switchport mode access
   Switch(config-if-range)# switchport access vlan 20
   ```

   Syntax:

    - **switch3 (IT):**

   ```
   Switch> enable
   Switch# configure terminal
   Switch(config)# interface range fa0/2-9
   Switch(config-if-range)# switchport mode access
   Switch(config-if-range)# switchport access vlan 30
   ```

   Syntax:

    - **switch4 (HR):**

```
Switch> enable
Switch# configure terminal
Switch(config)# interface range fa0/2-7
Switch(config-if-range)# switchport mode access
Switch(config-if-range)# switchport access vlan 40
```

Syntax:

4. **Configure trunk ports :**

- **for each floor's switch (except switch5):**

```
Switch> enable
Switch# configure terminal
Switch(config)# interface fa0/1
Switch(config-if)# switchport mode trunk
```

Syntax:

- **or For switch5, since it is connected to the router and server, you can configure its trunk ports using the following commands:**

```
Switch> enable
Switch# configure terminal
Switch(config)# interface range fa0/1-5
Switch(config-if-range)# switchport mode trunk
```

Syntax:

- **Verify**

```
Switch> enable
Switch# show interface trunk
```

Syntax:

5. **Configure Router-on-a-Stick (ROAS) on the router:**

- **for activates the interface**

```
Switch(config)# interface fa0/0
Switch(config-if)# no shutdown
Switch(config-if)# exit
```

Syntax:

- **Create subinterface, assign VLAN 10, set IP and subnet mask.**

```
Router> enable
Router# configure terminal
Router(config)# interface fa0/0.10
Router(config-subif)# encapsulation dot1Q 10
Router(config-subif)# ip address 192.168.1.161 255.255.255.240
```

Syntax:

- **Create subinterface, assign VLAN 20, set IP and subnet mask.**

```
Router> enable
Router# configure terminal
Router(config)# interface fa0/0.20
Router(config-subif)# encapsulation dot1Q 20
Router(config-subif)# ip address 192.168.1.129 255.255.255.224
```

Syntax:

- **Create subinterface, assign VLAN 30, set IP and subnet mask.**

```
Router> enable
Router# configure terminal
```

```
Router(config)# interface fa0/0.30
Router(config-subif)# encapsulation dot1Q 30
Router(config-subif)# ip address 192.168.1.1 255.255.255.192
```

Syntax:

- **Create subinterface, assign VLAN 40, set IP and subnet mask.**

```
Router> enable
Router# configure terminal
Router(config)# interface fa0/0.40
Router(config-subif)# encapsulation dot1Q 40
Router(config-subif)# ip address 192.168.1.65 255.255.255.192
```

Syntax:

Verify

```
Router> enable
Router# show interface brief
```

Syntax:

6. **Configure OSPF on the router:**

```
Router> enable
Router# configure terminal
Router(config)# router ospf 1
Router(config-router)# network 192.168.1.160 0.0.0.15 area 0
Router(config-router)# network 192.168.1.128 0.0.0.31 area 0
Router(config-router)# network 192.168.1.0 0.0.0.63 area 0
Router(config-router)# network 192.168.1.64 0.0.0.63 area 0
```

Syntax:

7. **Configure the interface between the router and server:**

```
Router> enable
Router# configure terminal
Router(config)# interface fa0/1
Router(config-if)# ip address 172.16.1.1 255.255.255.252
Router(config-if)# no shutdown
```

Syntax:

Add network to OSPF.

```
Router> enable
Router# configure terminal
Router(config)# router ospf 1
Router(config-router)# network 172.16.1.0 0.0.0.3 area 0
```

Syntax:

8. **Assign IP addresses, subnet masks, and default gateways for PCs and the server.**

- **PC1 (Account VLAN 10):**

  IP address: 192.168.1.162

  Subnet mask: 255.255.255.240

  Default gateway: 192.168.1.161

- **PC2 (Sales VLAN 20):**

  IP address: 192.168.1.130

  Subnet mask:255.255.255.224

  Default gateway: 192.168.1.129

- **PC3 (IT VLAN 30):**

  IP address: 192.168.1.2

  Subnet mask: 255.255.255.192

   Default gateway: 192.168.1.1

- **PC4 (HR VLAN 40):**

  IP address: 192.168.1.66

  Subnet mask: 255.255.255.192

  Default gateway: 192.168.1.65

- **Configure the server:**

  IP address: 172.16.1.2

  Subnet mask: 255.255.255.252

  Default gateway: 172.16.1.1

9. **Verify connectivity using pings between devices.**

- **From PC1, ping PC2:**

```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.1.130
```

- **From PC1, ping PC3:**

```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.1.2
```

Lab2

Scenario:

There are two buildings, each having a ring topology consisting of 6 switches. Each switch is connected to 2 PCs and the main switch is connected to an MLS for each ring. The two rings are connected to each other with 3 routers. The task is to create 1 VLAN per two switches, configure DHCP, and ensure successful communication between PCs. The IP addresses for the VLANs and connections between rings are as follows:

- **VLAN 11:** 10.10.11.0/24
- **VLAN 12:** 10.10.12.0/25
- **VLAN 13:** 10.10.13.0/26
- **VLAN 21:** 10.10.21.0/27
- **VLAN 22:** 10.10.22.0/28
- **VLAN 23:** 10.10.23.0/29

Connections between rings:

- 172.16.1.0/30
- 172.16.2.0/30
- 192.168.2.0/30
- 192.168.1.0/30

```
graph RL
  subgraph Ring1
    SW1[Switch1] --- PC1_1[PC1_1]
    SW1 --- PC1_2[PC1_2]
    SW1 --- SW2[Switch2]

    SW2 --- PC2_1[PC2_1]
    SW2 --- PC2_2[PC2_2]
    SW2 --- SW3[Switch3]

    SW3 --- PC3_1[PC3_1]
    SW3 --- PC3_2[PC3_2]
    SW3 --- SW4[Switch4]

    SW4 --- PC4_1[PC4_1]
    SW4 --- PC4_2[PC4_2]
    SW4 --- SW5[Switch5]

    SW5 --- PC5_1[PC5_1]
    SW5 --- PC5_2[PC5_2]
    SW5 --- SW6[Switch6]

    SW6 --- PC6_1[PC6_1]
    SW6 --- PC6_2[PC6_2]
    SW6 --- SW1
  end

  subgraph Ring2
    SW7[Switch7] --- PC7_1[PC7_1]
    SW7 --- PC7_2[PC7_2]
    SW7 --- SW8[Switch8]

    SW8 --- PC8_1[PC8_1]
    SW8 --- PC8_2[PC8_2]
    SW8 --- SW9[Switch9]

    SW9 --- PC9_1[PC9_1]
    SW9 --- PC9_2[PC9_2]
    SW9 --- SW10[Switch10]

    SW10 --- PC10_1[PC10_1]
    SW10 --- PC10_2[PC10_2]
    SW10 --- SW11[Switch11]

    SW11 --- PC11_1[PC11_1]
    SW11 --- PC11_2[PC11_2]
    SW11 --- SW12[Switch12]

    SW12 --- PC12_1[PC12_1]
    SW12 --- PC12_2[PC12_2]
    SW12 --- SW7
  end
subgraph VLAN_Info
    vlan11["vlan11 10.10.11.0/24"]
    vlan12["vlan12 10.10.12.0/25"]
    vlan13["vlan13 10.10.13.0/26"]
    vlan21["vlan21 10.10.21.0/27"]
    vlan22["vlan22 10.10.22.0/28"]
    vlan23["vlan23 10.10.23.0/29"]
  end
  Mls1[Multilayer Switch1] --- SW1
  Mls1 --- Router1
  Router2 --- Router1
  Router2 --- Router3
  Router3 --- Mls2[Multilayer Switch2]
  Mls2 --- SW7
```

Switches	VLAN
1, 2	VLAN 11
3, 4	VLAN 12
5, 6	VLAN 13
7, 8	VLAN 21
9, 10	VLAN 22
11, 12	VLAN 23

Solution:

Step 1: Configure VLANs on MLS1

Create and name VLANs 11, 12, and 13:

```
vlan 11
name customer11
vlan 12
name customer12
vlan 13
name customer13
```

Step 2: Verify VLANs on MLS1

Verify the VLAN configuration by running `show vlan brief`.

Step 3: Configure LACP on MLS1

Set up LACP between the switch1 and MLS1:

```
int range fa0/1-2
channel-group 1 mode active
```

Step 4: Verify LACP on MLS1

Verify the LACP configuration by running `show etherchannel summary`.

Step 5: Configure inter-VLAN routing on MLS1

```
ip routing
int vlan 11
description GW-VL-11
ip address 10.10.11.1 255.255.255.0
no shutdown
int vlan 12
description GW-VL-12
ip address 10.10.12.1 255.255.255.128
no shutdown
int vlan 13
description GW-VL-13
ip address 10.10.13.1 255.255.255.192
no shutdown
```

Step 6: Verify inter-VLAN routing on MLS1

Verify the inter-VLAN routing configuration by running `show vlan brief`.

Step 7: Configure DHCP pools and excluded addresses on MLS1

```
ip dhcp pool Ring1-VL-11
network 10.10.11.0 255.255.255.0
default-router 10.10.11.1
ip dhcp pool Ring1-VL-12
network 10.10.12.0 255.255.255.128
default-router 10.10.12.1
ip dhcp pool Ring1-VL-13
network 10.10.13.0 255.255.255.192
default-router 10.10.11.1
ip dhcp pool Ring1-VL-21
network 10.10.21.0 255.255.255.224
default-router 10.10.21.1
ip dhcp pool Ring1-VL-22
network 10.10.22.0 255.255.255.240
default-router 10.10.22.1
ip dhcp pool Ring1-VL-23
network 10.10.21.0 255.255.255.248
default-router 10.10.21.1

ip dhcp excluded-address 10.10.11.1
ip dhcp excluded-address 10.10.12.1
ip dhcp excluded-address 10.10.13.1
ip dhcp excluded-address 10.10.21.1
ip dhcp excluded-address 10.10.22.1
ip dhcp excluded-address 10.10.23.1
```

Step 8: Configure DTP to use dot1q encapsulation for the port channel on MLS1

```
interface portchannel 1
switchport trunk encapsulation dot1q
switchport mode trunk
```

Step 9: Set up OSPF routing on MLS1

```
interface gig0/1
no switchport
ip address 172.16.1.1 255.255.255.252
router ospf 1
network 10.10.11.0 0.0.0.255 area 1
network 10.10.12.0 0.0.0.127 area 1
network 10.10.13.0 0.0.0.63 area 1
network 172.16.1.0 0.0.0.3 area 1
```

Step 10: Verify OSPF routing on MLS1

Verify OSPF routing by running `show ip ospf neighbor` and `show ip ospf database`.

Step 11: Configure VLANs on switches 1 to 6

Create and name VLANs 11, 12, and 13 on each switch (switch 1 to 6):

```
vlan 11
name customer11
vlan 12
name customer12
vlan 13
name customer13
```

Step 12: Verify VLANs on switches 1 to 6

Verify the VLAN configuration on each switch by running `show vlan brief`.

Step 13: Configure switch 1 as the root bridge for VLANs 11, 12, and 13 (only on switch 1)

```
spanning-tree vlan 11 root primary
spanning-tree vlan 12 root primary
spanning-tree vlan 13 root primary
```

Step 14: Verify spanning-tree configuration on switch 1

Verify the spanning-tree configuration by running `show spanning-tree`.

Step 15: Configure LACP between switch 1 and MLS1 (only on switch 1)

```
int range fa03-4
channel-group 1 mode active
```

Step 16: Verify LACP on switch 1

Verify the LACP configuration by running `show etherchannel summary`.

Step 17: Configure access ports on switches 1 to 6

- Switch 1 and switch 2: access ports for VLAN 11

  ```
  interface range fa0/3-4
  switchport mode access
  switchport access vlan 11
  ```

- Switch 3 and switch 4: access ports for VLAN 12

  ```
  interface range fa0/3-4
  switchport mode access
  switchport access vlan 12
  ```

- Switch 5 and switch 6: access ports for VLAN 13

  ```
  interface range fa0/3-4
  switchport mode access
  switchport access vlan 13
  ```

Step 18: Verify access port configuration on switches 1 to 6

Verify the access port configuration on each switch by running `show vlan brief`.

Step 19: Configure trunk ports on switches 1 to 6

- Switch 1:

  ```
  interface range fa0/1-2
  switchport mode trunk
  interface portchannel 1
  switchport mode trunk
  ```

- Switch 2 to switch 6:

  ```
  interface range fa0/1-2
  switchport mode trunk
  ```

Step 20: Verify trunk port configuration on switches 1 to 6

Verify the trunk port configuration on each switch by running `show interfaces trunk`.

Step 21: Configure VLANs on MLS2

```
vlan 21
name customer21
vlan 22
name customer22
vlan 23
name customer23
```

This step creates VLANs 21, 22, and 23 on the Multi-Layer Switch MLS2 and gives them names.

Step 22: Verify VLANs on MLS2

```
show vlan brief
```

This command verifies that the VLANs have been created on MLS2.

Step 23: Configure LACP on MLS2

Configure Link Aggregation Control Protocol (LACP) between MLS2 and the connected switch to ensure that service does not get interrupted.

```
int fa0/2
channel-group 1 mode active
```

This command creates a new LACP channel group (group 1) and sets the mode to active on interface fa0/2.

Step 24: Verify EtherChannel on MLS2

```
show etherchannel summary
```

This command shows the summary of the EtherChannel configuration on MLS2.

Step 25: Add the second link to LACP on MLS2

```
int fa0/1
channel-group 1 mode active
```

This command adds interface fa0/1 to the LACP channel group 1 on MLS2.

Step 26: Verify EtherChannel on MLS2

```
show etherchannel summary
```

This command shows the summary of the EtherChannel configuration on MLS2.

Step 27: Configure InterVLAN Routing on MLS2

Enable IP routing and configure the VLAN interfaces with IP addresses and descriptions.

```
ip routing
interface vlan21
description GW-VL-21
ip address 10.10.21.1 255.255.255.224
no shutdown
interface vlan22
description GW-VL-22
ip address 10.10.22.1 255.255.255.240
no shutdown
interface vlan23
description GW-VL-23
ip address 10.10.23.1 255.255.255.248
no shutdown
```

Step 28: Verify IP Interface on MLS2

```
show ip interface brief
```

This command shows a summary of the IP interface configuration on MLS2.

Step 29: Configure DTP on MLS2

Change the Dynamic Trunking Protocol (DTP) on MLS2 from auto to dot1q.

```
interface portchannel 1
switchport trunk encapsulation dot1q
switchport mode trunk
```

Step 30: Configure OSPF on MLS2

```
interface gig0/1
no switchport
ip address 172.16.2.1 255.255.255.252
router ospf 1
network 172.16.2.0 0.0.0.3 area 2
network 10.10.21.0 0.0.0.31 area 2
network 10.10.22.0 0.0.0.15 area 2
network 10.10.23.0 0.0.0.7 area 2
```

This step configures OSPF routing on MLS2. It assigns an IP address to the Gigabit Ethernet interface, creates an OSPF routing process, and adds the networks to the OSPF area.

Step 31: Verify OSPF on MLS2

```
show ip ospf database
show ip ospf neighbor
```

These commands verify the OSPF configuration on MLS2.

Step 32: Configure DHCP Relay on MLS2

```
interface vlan 21
ip helper-address 172.16.1.1
interface vlan 22
ip helper-address 172.16.1.1
interface vlan 23
ip helper-address 172.16.1.1
```

This step configures the DHCP relay on MLS2 to forward DHCP requests to the DHCP server at 172.16.1.1.

Step 33: Configure VLANs on switches 7 to 12

Create and name VLANs 21, 22, and 23 on each switch (switch 7 to 12):

```
vlan 21
name customer21
vlan 22
name customer22
vlan 23
name customer23
```

Step 34: Verify VLANs on switches 7 to 12

Verify the VLAN configuration on each switch by running `show vlan brief`.

Step 35: Configure switch 7 as the root bridge for VLANs 21, 22, and 23 (only on switch 7)

```
spanning-tree vlan 21 root primary
spanning-tree vlan 22 root primary
spanning-tree vlan 23 root primary
```

Step 36: Verify spanning-tree configuration on switch 7

Verify the spanning-tree configuration by running `show spanning-tree`.

Step 37: Configure LACP between switch 7 and MLS1 (only on switch 7)

```
int fa0/4
channel-group 1 mode active
```

Step 38: Verify LACP on switch 7

Verify the LACP configuration by running `show etherchannel summary`.

Step 39: Add the second link to LACP between switch 7 and MLS1 (only on switch 7)

```
int fa0/3
channel-group 1 mode active
```

Step 40: Verify LACP on switch 7 again

Verify the LACP configuration by running `show etherchannel summary`.

Step 41: Configure access ports on switches 7 to 12

- Switch 7 and switch 8: access ports for VLAN 21

  ```
  interface range fa0/5-6
  switchport mode access
  switchport access vlan 21
  ```

- Switch 9 and switch 10: access ports for VLAN 22

  ```
  interface range fa0/3-4
  switchport mode access
  switchport access vlan 22
  ```

- Switch 11 and switch 12: access ports for VLAN 23

  ```
  interface range fa0/3-4
  switchport mode access
  switchport access vlan 23
  ```

Step 42: Verify access port configuration on switches 7 to 12

Verify the access port configuration on each switch by running `show vlan brief`.

Step 43: Configure trunk ports on switches 7 to 12

- Switch 7:

  ```
  interface range fa0/1-2
  switchport mode trunk
  interface portchannel 1
  switchport mode trunk
  ```

- Switch 8 to switch 12:

  ```
  interface range fa0/1-2
  switchport mode trunk
  ```

Step 44: Verify trunk port configuration on switches 7 to 12

Verify the trunk port configuration on each switch by running `show interfaces trunk`.

Step 45: Configure OSPF on router1

Configure OSPF on router1:

```
interface gig0/1
 ip address 172.168.1.2 255.255.255.252
 no shutdown

interface gig0/0
 ip address 192.168.1.2 255.255.255.252
 no shutdown

router ospf 1
 network 172.168.1.0 0.0.0.3 area 1
 network 192.168.1.0 0.0.0.3 area 0
```

Step 46: Verify OSPF on router1

Verify OSPF configuration on router1:

```
show ip ospf neighbor
show ip ospf database
```

Step 47: Configure OSPF on router2

Configure OSPF on router2:

```
interface gig0/2
 ip address 192.168.2.2 255.255.255.252
 no shutdown

interface gig0/1
 ip address 172.16.2.2 255.255.255.252
 no shutdown

router ospf 1
 network 172.16.2.0 0.0.0.3 area 2
 network 192.168.2.0 0.0.0.3 area 0
```

Step 48: Verify OSPF on router2

Verify OSPF configuration on router2:

```
show ip ospf neighbor
show ip ospf database
```

Step 49: Configure OSPF on router3

Configure OSPF on router3:

```
interface gig0/0
 ip address 172.16.2.2 255.255.255.252
 no shutdown

interface gig0/2
 ip address 192.168.2.2 255.255.255.252
 no shutdown

router ospf 1
 network 172.16.2.0 0.0.0.3 area 2
 network 192.168.2.0 0.0.0.3 area 2
```

Step 50: Verify OSPF on router3

Verify OSPF configuration on router3:

```
show ip ospf neighbor
show ip ospf database
```

Step 51: Configure PCs to get IP addresses automatically

Set the PCs to obtain IP addresses automatically using DHCP.

Step 52: Test connectivity between PCs

Ping between PCs to verify connectivity.

Lab3

Scenario:

A company requires a highly redundant network connected to a public network using NAT. There are five switches connected to each other to provide redundancy, and each switch is connected to a PC. An MLS is connected to these switches. There are three routers: one for the DHCP server, two others for routing (one primary and one standby), connected to the public network with two routers (one primary and one standby). These routers are connected to the MLS and two switches connected to two servers. The IP addresses for the private company are in the 10.10.10.0/24 range, the public network is in the 50.10.10.0/27 range, and the private branch 2 network is in the 172.16.10.0/26 range.

```
graph TB
subgraph Branch1_Private_Company
        pc1[PC1] --- switch1[Switch1]
        pc2[PC2] --- switch2[Switch2]
        pc3[PC3] --- switch3[Switch3]
        pc4[PC4] --- switch4[Switch4]
        pc5[PC5] --- switch5[Switch5]

        switch1 --- switch2
        switch1 --- mls1[MLS1]
        switch1 --- switch4

        switch2 --- switch3
        switch2 --- switch4
        switch2 --- switch5

        switch3 --- switch5

        switch4 --- switch5
        switch4 --- mls1

    end
    subgraph Public_Network
        mls1 --- router1[Router1]
        mls1 --- router2[Router2]
        mls1 --- router3[Router3]
        router1 --- router4[Router4]
        router2 --- router5[Router5]
    end
    subgraph Branch2_Private_Company
        mls2[MLS2] --- router4
        mls2 --- router5
        mls2 --- switch6[Switch6]
        mls2 --- switch7[Switch7]
        switch6 --- server1[Server1]
        switch7 --- server2[Server2]
    end
```

Step 1: Configure Router3 as a DHCP server for the private company network.

1. Assign IP address and enable the interface connected to the private company network.

~~~
Router3(config)# interface fa0/0
Router3(config-if)# ip address 10.10.10.4 255.255.255.0
Router3(config-if)# no shutdown
```

Explanation: This configures the IP address of the interface connecting Router3 to the private company network and enables the interface.
~~~

2. Create a DHCP pool and set the network and default gateway for clients.

~~~
Router3(config)# ip dhcp pool Elab3
Router3(config-dhcp)# network 10.10.10.0 255.255.255.0
Router3(config-dhcp)# default-router 10.10.10.1
```

Explanation: This creates a DHCP pool named Elab3, sets the network range for the pool, and specifies the default gateway (the virtual IP address of the HSRP routers) for clients.
~~~

3. Exclude some IP addresses from the DHCP pool to prevent conflicts with static IPs.

~~~
Router3(config)# ip dhcp excluded-address 10.10.10.1
Router3(config)# ip dhcp excluded-address 10.10.10.2
Router3(config)# ip dhcp excluded-address 10.10.10.3
Router3(config)# ip dhcp excluded-address 10.10.10.4
```

Explanation: This excludes the specified IP addresses from the DHCP pool, preventing conflicts with static IP addresses assigned to routers and other devices.
~~~

Step 2: Configure Router1 as the primary HSRP router for the private company network.

1. Assign IP address and enable the interface connected to the private company network.

~~~
Router1(config)# interface fa0/0
Router1(config-if)# ip address 10.10.10.2 255.255.255.0
Router1(config-if)# no shutdown
```

Explanation: This configures the IP address of the interface connecting Router1 to the private company network and enables the interface.
~~~

2. Configure the HSRP settings for the interface.

~~~
Router1(config-if)# standby 1 ip 10.10.10.1
Router1(config-if)# standby 1 priority 120
Router1(config-if)# standby 1 preempt
```

Explanation: This sets the standby group number, virtual IP address, priority (higher than Router2), and enables preemption, which allows Router1 to become the active router when it has a higher priority.
~~~

Step 3: Configure Router2 as the standby HSRP router for the private company network.

1. Assign IP address and enable the interface connected to the private company network.

~~~
Router2(config)# interface fa0/0
Router2(config-if)# ip address 10.10.10.3 255.255.255.0
Router2(config-if)# no shutdown
```
~~~

````
Explanation: This configures the IP address of the interface connecting Router2 to the private company network and enables the interface.
````

2. Configure the HSRP settings for the interface.

~~~
Router2(config-if)# standby 1 ip 10.10.10.1
Router2(config-if)# standby 1 priority 80
Router2(config-if)# standby 1 preempt
```

Explanation: This sets the standby group number, virtual IP address, priority (lower than Router1), and enables preemption, allowing Router1 to become the active router
when it has a higher priority.
~~~

Step 4: Configure NAT settings for Router1 and Router2.

1. Configure NAT for Router1:

~~~
Router1(config)# access-list 1 permit 10.10.10.0 0.0.0.255
Router1(config)# ip nat inside source list 1 interface fa0/1 overload
Router1(config)# interface fa0/0
Router1(config-if)# ip nat inside
Router1(config)# interface fa0/1
Router1(config-if)# ip address 50.10.10.2 255.255.255.224
Router1(config-if)# ip nat outside
Router1(config-if)# no shutdown
```

Explanation: This creates an access list to permit the private network range, sets up NAT with the access list and the public-facing interface, and enables NAT inside a
nd outside on the respective interfaces.
~~~

2. Configure NAT for Router2:

~~~
Router2(config)# access-list 1 permit 10.10.10.0 0.0.0.255
Router2(config)# ip nat inside source list 1 interface fa0/1 overload
Router2(config)# interface fa0/0
Router2(config-if)# ip nat inside
Router2(config)# interface fa0/1
Router2(config-if)# ip address 50.10.10.3 255.255.255.224
Router2(config-if)# ip nat outside
Router2(config-if)# no shutdown
```

Explanation: This is similar to the NAT configuration for Router1, but uses the IP address assigned to Router2's public-facing interface.
~~~

Step 7: Configure OSPF on Router1, Router2, and Router3.

OSPF (Open Shortest Path First) is a link-state routing protocol that enables routers to exchange routing information and dynamically update their routing tables. In this scenario, you can use OSPF to ensure that routers have accurate routing information for the private company network, public network, and private branch network.

1. Configure OSPF on Router1:

~~~
Router1(config)# router ospf 1
Router1(config-router)# network 10.10.10.0 0.0.0.255 area 0
Router1(config-router)# network 50.10.10.0 0.0.0.31 area 0
Router1(config-router)# network 172.16.10.0 0.0.0.63 area 0
```

Explanation: This enables OSPF process 1 on Router1 and associates it with the networks for the private company, public, and private branch networks. All networks a
re placed in Area 0 (the OSPF backbone area).
~~~

2. Configure OSPF on Router2:

~~~
Router2(config)# router ospf 1
Router2(config-router)# network 10.10.10.0 0.0.0.255 area 0
Router2(config-router)# network 50.10.10.0 0.0.0.31 area 0
Router2(config-router)# network 172.16.10.0 0.0.0.63 area 0
```

Explanation: This is similar to the OSPF configuration on Router1, enabling OSPF process 1 on Router2 and associating it with the networks for the private company,
 public, and private branch networks. All networks are placed in Area 0.
~~~

3. Configure OSPF on Router3:

~~~
Router3(config)# router ospf 1
Router3(config-router)# network 10.10.10.0 0.0.0.255 area 0
Router3(config-router)# network 172.16.10.0 0.0.0.63 area 0
```

Explanation: This enables OSPF process 1 on Router3 and associates it with the networks for the private company and private branch networks. Both networks are place
d in Area 0. Since Router3 does not have a direct connection to the public network, there is no need to include the public network in the OSPF configuration for Rou
ter3.
~~~

With OSPF configured, the routers will now exchange routing information and dynamically update their routing tables, allowing for efficient routing between the private company network, public network, and private branch network.

Lab4:

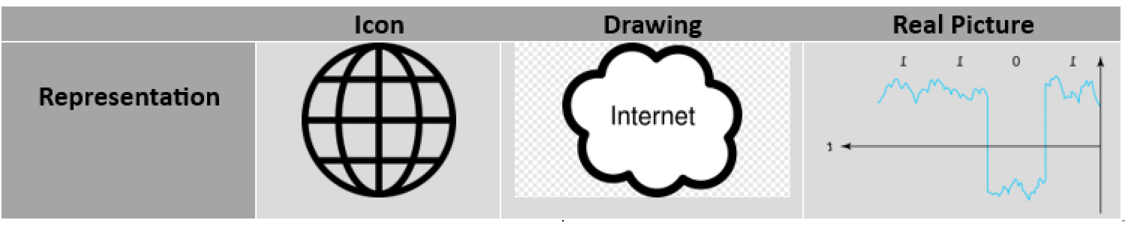
````
graph TB

        Router1[Router1] --10.10.10.1/24--- Router2[Router2]
        Router1[Router1] --10.10.10.2/24--- Router2[Router2]
        Router3[Router3] --10.10.10.3/24--- Router4[Router4]
        Router3[Router3] --10.10.10.4/24--- Router4[Router4]
````

Extra-Information

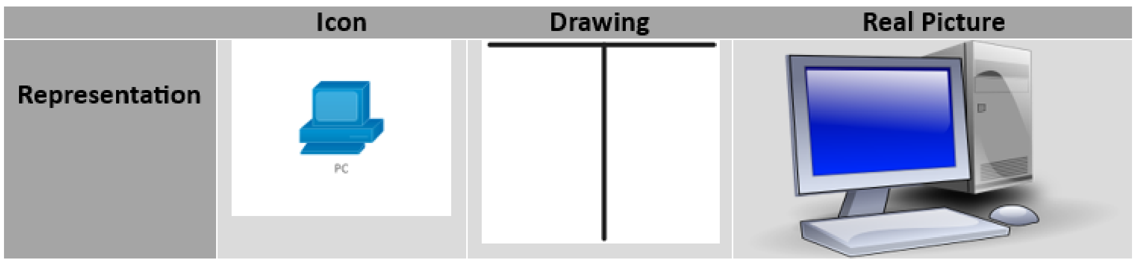
Drawing

- When referring to the internet, a cloud drawing is often used to represent the vast network of interconnected devices and servers that make up the internet.



	Icon	Drawing	Real Picture
Representation		Internet	

- In network diagrams, end devices are typically represented as rectangles or squares with the network connections on one side, creating a "T" shape.



	Icon	Drawing	Real Picture
Representation	PC		

Creating a VM using VMware:

- Install VMware on the physical server or computer.
- Create a new virtual machine and select the operating system to be installed on the virtual machine.
- Allocate resources such as CPU, memory, and storage to the virtual machine.
- Install the operating system on the virtual machine using a CD/DVD or ISO image.
- Configure the virtual machine settings such as network adapters, virtual disks, and virtual devices.

CCNP and Packet Tracer/EVE-NG:

- CCNP is a Cisco certification program that covers advanced concepts in network engineering and design.
- Packet Tracer is a network simulation tool developed by Cisco that allows users to create network topologies and simulate network behaviors.
- EVE-NG (Emulated Virtual Environment - Next Generation) is a network emulation tool that provides a more realistic simulation of network behaviors and protocols compared to Packet Tracer.
- CCNP uses EVE-NG instead of Packet Tracer because EVE-NG provides a more realistic simulation of network protocols and behaviors, which is important for advanced network engineering and troubleshooting.

Cisco Catalyst 3560-24PS and HP ProCurve 2650-24PS:

The Cisco Catalyst 3560-24PS and HP ProCurve 2650-24PS are both multilayer switches, but they have some differences in terms of their capabilities and features.

- The Cisco Catalyst 3560-24PS is a Layer 3 switch, while the HP ProCurve 265024PS is primarily a Layer 2 switch but supports some Layer 3 features.
- The Cisco Catalyst 3560-24PS has a higher switching capacity (32 Gbps) compared to the HP ProCurve 2650-24PS (12.8 Gbps).
- Both switches support Power over Ethernet (PoE), but the Cisco Catalyst 356024PS has a higher PoE budget.
- The Cisco Catalyst 3560-24PS has 24 Ethernet ports, while the HP ProCurve 2650-24PS has 24 Ethernet ports as well as four dual-personality ports that can be used as either Ethernet or SFP ports.
- The Cisco Catalyst 3560-24PS can be managed through a variety of interfaces including a console port, Telnet, SSH, and SNMP, while the HP ProCurve 2650-24PS can be managed through a console port, Telnet, or SNMP.

Types of message transmission:

there are **three main types** of message transmission:

- **Unicast:** A message that is sent from one sender to one receiver. This is a point-to-point transmission, where the sender sends a message to a specific receiver.
- **Broadcast:** A message that is sent from one sender to all devices on the network. This is a one-to-many transmission, where the sender sends a message to all devices on the network, and each device checks to see if the message is intended for them.
- **Multicast:** A message that is sent from one sender to a specific group of receivers. This is a many-to-many transmission, where the sender sends a message to a group of devices that have subscribed to receive messages.

Factors that affect the age of network devices:

1. Technology Obsolescence: Network devices can become outdated due to advances in technology. As newer devices with more advanced features become available, older devices can become obsolete and need to be replaced.
2. Wear and Tear: Network devices can also wear out over time due to constant use and exposure to environmental factors such as heat, dust, and humidity.
3. Security Issues: Security threats can also affect the age of network devices. As new security threats emerge, older devices may not be able to handle them and may need to be upgraded or replaced.
4. End-of-Support: Manufacturers typically provide support for their devices for a certain period of time. Once the support ends, the devices may no longer receive software updates, making them vulnerable to security threats.

Extra Commands:

- `"ping -t"` is a command used in the command prompt or terminal of a computer to continuously send ICMP

Dijkstra's algorithm:

- Dijkstra's algorithm is a graph traversal algorithm used to find the shortest path between two nodes in a weighted graph.
- The algorithm works by starting at the source node and exploring all its neighbors, updating their distances from the source node.
- It then selects the node with the smallest distance and repeats the process until it reaches the destination node or all nodes have been explored.
- The algorithm maintains a priority queue of nodes to explore, sorted by their current distance from the source node.
- The distance between nodes is determined by the sum of the weights of the edges between them.
- Initially, the distance to the source node is set to 0, and the distance to all other nodes is set to infinity.
- When exploring a node, the algorithm updates the distances of its neighbors based on the distance to the current node and the weight of the edge between them.
- If the new distance is smaller than the current distance, it replaces the current distance and updates the parent node of the neighbor.
- The algorithm terminates when the destination node is reached or all nodes have been explored.
- After the algorithm terminates, the shortest path can be reconstructed by following the parent nodes from the destination node back to the source node.
- Dijkstra's algorithm is guaranteed to find the shortest path in a graph with non-negative edge weights.
- **Example:**
  - Consider the following undirected weighted graph:

    ```
        7        2
     A ----- B ----- C
     | \\     |      / |
     |   \\   |     /  |
    3|     \\ |   /    |4
     |      \\| /      |
     D ----- E ----- F
        5        6
    ```

  - We want to find the shortest path from node A to node F using Dijkstra's algorithm.
  - Solution:
    1. Initialize the distance to all nodes to infinity, except for the source node A, which has a distance of 0. Also, keep track of the parent node of each node.

       ```
       A: distance = 0, parent = null
       B: distance = infinity, parent = null
       C: distance = infinity, parent = null
       D: distance = infinity, parent = null
       E: distance = infinity, parent = null
       F: distance = infinity, parent = null
       ```

    2. Add the source node A to a priority queue sorted by the current distance.

       ```
       Queue: A(0)
       ```

    3. While the queue is not empty, select the node with the smallest distance and explore its neighbors.

       ```
       Queue: B(7), D(3)
       ```

    4. Explore node D and update the distances of its neighbors E and A.

       ```
       Queue: B(7), E(8), A(3)
       A: distance = 3, parent = D
       ```

    5. Explore node A and update the distance of its neighbor B.

       ```
       Queue: B(7), E(8), C(infinity), F(infinity)
       B: distance = 7, parent = A
       ```

    6. Explore node B and update the distances of its neighbors C and E.

       ```
       Queue: E(8), C(9), F(13)
       C: distance = 9, parent = B
       ```

    7. Explore node E and update the distances of its neighbors F and C.

       ```
       Queue: F(14), C(9)
       F: distance = 14, parent = E
       ```

    8. Explore node C and update the distance of its neighbor F.

       ```
       Queue: C(15)
       F: distance = 15, parent = C
       ```

    9. The destination node F has been reached, and the shortest path is A -> D -> E -> F with a total distance of 14.

Atomic clocks:

- Atomic clocks are devices that measure time with extraordinary accuracy, based on the properties of atoms.
- The most commonly used atomic clocks are based on the properties of the cesium atom or the rubidium atom.
- In a cesium atomic clock, a beam of cesium atoms is heated, causing them to emit radiation at a specific frequency.
- The frequency of the radiation is precisely defined and can be used as a standard for measuring time.
- A detector is used to measure the radiation and compare it with the standard frequency. Any deviation indicates a change in time.

- The output of the detector is used to control the frequency of an oscillator, which generates a signal that can be used as a time reference.
- The accuracy of atomic clocks is typically measured in terms of their stability or uncertainty, usually expressed as the number of seconds that the clock will deviate from the correct time over a period of one day or year.
- Atomic clocks are used as primary standards for defining the International System of Units (SI) second, which is based on the properties of the cesium atom.
- Atomic clocks are used for a wide range of applications that require accurate time measurement, such as navigation systems, telecommunications networks, and scientific experiments.
- The Global Positioning System (GPS) uses atomic clocks to provide precise time information that is essential for accurately determining location.
- The most accurate atomic clocks can measure time with an uncertainty of around one second over the age of the universe, or about 13.8 billion years.
- The development of atomic clocks has revolutionized the field of timekeeping, enabling unprecedented levels of accuracy and precision in time measurement.

Automatic Private IP Addressing (APIPA):

- **Automatic Private IP Addressing** (APIPA) is a feature of some operating systems that allows devices to **automatically** assign **themselves** an IP address when a DHCP server is **not available**.
- **APIPA** is used in **small** networks where there is no **dedicated** network **administrator** and **no** DHCP server is **present**.
- When a device is configured to use **APIPA**, it will automatically assign **itself** an IP address in the range of $169.254.0.1$ to $169.254.255.254$, with a subnet mask of $255.255.0.0$.
- The **device** will attempt to **verify** that the IP address is **not already** in use by sending an **Address Resolution Protocol** (ARP) request to the network.
- If the address is **already** in **use**, the device will assign itself a **different** IP address and **repeat** the **verification** process until a **unique** IP address is **found**.
- APIPA only works within a **single** subnet, so devices using **APIPA** cannot **communicate** with devices on other subnets **without** the use of a router.
- APIPA is a **simple** and **convenient** way to configure IP addresses on a **small** network **without** the need for a **DHCP** server.
- However, APIPA addresses are not **routable** on the Internet, so they **cannot** be used for devices that need to **communicate** with other networks.
- APIPA should be used only as a **temporary** solution until a properly configured **DHCP** server becomes **available**.
- APIPA can be **disabled** by configuring a **static** IP address or by enabling **DHCP** on the network.

Dual Gateway or Dual WAN:

- **Dual Gateway** or **Dual WAN** (Wide Area Network) is a network **configuration** that allows a device to connect to **two** separate **internet** connections **simultaneously**.
- The purpose of Dual Gateway is to provide **redundant** connectivity and **load balancing**, which can **improve** network **performance** and **reduce downtime**.
- In a Dual Gateway setup, a device such as a **router** or firewall is configured to use two **separate** internet **connections**, each with its **own unique IP** address and **gateway**.
- The device is then configured to use **both gateways** simultaneously, either by alternating between them or by **load balancing traffic** between them.
- Redundant connectivity is achieved by ensuring that if **one internet connection fails**, the device can **automatically** switch to the other connection, ensuring that the network **remains** operational.
- Load balancing is achieved by **spreading** network **traffic** between the two connections, which can **reduce** congestion and improve performance.
- Dual Gateway requires a device that is capable of supporting **multiple** internet connections, such as a **router** or firewall with **multiple** WAN **interfaces**.
- Some devices offer **built-in** support for Dual **Gateway**, while others may require **additional** configuration or the use of **specialized software** or hardware.
- Dual Gateway is commonly used in businesses or organizations that require high **availability** and **fast** internet connectivity, such as **data centers**, **e-commerce** sites, or **online gaming** platforms.
- Dual Gateway can also be used in residential settings where there are **multiple** internet connections available, such as a cable modem and a **DSL** modem, to provide a more **reliable** and **faster internet** connection.

Crossover cables, straight-through fiber optic cable, and serial cables:

- **Crossover cable:** This is a **type** of Ethernet cable that is used to connect **two similar devices**, such as **two** computers or **two** switches, without the need for a **hub** or a **switch**. The wiring arrangement of the cable is different on both ends, meaning that the transmit and receive signals are **swapped**. This allows the two devices to communicate with each other **directly**. In contrast, a straight-through cable is used to connect **dissimilar** devices, such as a computer to a switch.
- **straight-through fiber optic cable:** This **type** of cable is used to connect **dissimilar** devices, such as a **computer** to a **switch**, using fiber optic technology. **Unlike** copper Ethernet cables, fiber optic cables use **light** to transmit data over longer distances and at higher speeds. Straight-through fiber optic cables have the same wiring arrangement on both ends of the cable and are commonly used in data centers and other high-speed networking environments.
- **Serial cable:** This is a type of cable that is used to connect devices with serial ports, such as a **computer** to a **modem** or a router to a console port. Serial cables have a **connector** at each end that **plugs** into the serial port on each device. They are commonly used for configuring and managing networking equipment. Serial cables can have different **pinouts** depending on the devices being connected, and they typically have a **maximum** cable length of **50** feet.

Startup configuration in nonvolatile memory:

- The **startup configuration** is a **file** that contains the **configuration** settings for a **Cisco** device.
- The startup configuration is stored in nonvolatile memory (**NVRAM**), which is a **type** of memory that **retains** its **contents** even when the device is powered **off**.
- The startup configuration is **loaded** into the device's **running configuration** when the device **boots** up.
- The startup **configuration** can be **modified** using the `config-register` command, which specifies the **boot options** for the device.

Installing FTP software:

- FTP **software** can be **installed** on a server to **enable** file transfers between devices.
- To **install** FTP **software** on a server, the following steps can be taken:
  - **Download** the FTP software from a **trusted** source.
  - **Install** the software on the server by running the **installation** wizard and following the prompts.
  - Once the software is installed, **configure** the FTP server settings, such as the **port number** and user **authentication** settings.

Main types of memory used in Cisco devices:

- **RAM (Random Access Memory):** This is volatile memory that is used for **temporary** storage of data and **instructions** while the device is powered **on**. **RAM** is used for the device's **running configuration**, **routing** tables, **ARP** cache, and **other** data structures that are required for the device's **operation**.
- **NVRAM (Non-Volatile RAM):** This is a type of memory that retains its contents even when the device is powered **off**. **NVRAM** is used to store the device's **startup** configuration, which is **loaded** into **RAM** when the device is powered **on**.
- **Flash Memory:** This is a type of **non-volatile** memory that is used for **long-term storage** of data and **software images**. Flash memory is used to store the device's **operating system**, **configuration** files, **IOS** images, and **other** software components.
- **ROM (Read-Only Memory):** This is **non-volatile memory** that is used to store the device's **bootstrap** program, which is used to **initialize** the device's **hardware** and **load** the **operating** system from **flash memory**.
- **CAM (Content-Addressable Memory):** This is a type of memory that is used in **switches** to store the device's **MAC address table**. CAM memory is used to perform **fast lookups** of MAC addresses to enable **switching** of data between **ports**.
- **TCAM (Ternary Content-Addressable Memory):** This is a type of memory that is used in **switches** to support advanced features such as access control lists (**ACLs**) and Quality of Service (**QoS**). TCAM memory is used to **accelerate** the **lookup** and processing of packets based on their **attributes**.

Syntax:

Interface configuration:

Interface:

```
interface [interface_type] [interface_number]/[subinterface_number]
```

- `interface`**:** Main command used to configure a specific network interface.
- `interface_type`**:** This argument specifies the type of interface you want to configure. Examples include Ethernet, FastEthernet, GigabitEthernet, Port-channel, and Vlan.
- `interface_number`**/**`subinterface_number`**:** These arguments specify the interface number and subinterface number (if applicable) of the selected interface type.

Example usage:

```
interface FastEthernet 0/1
```

interface range:

```
interface range [interface_type] [first_interface_number]/[first_subinterface_number] - [last_interface_number]/[last_subinterface_number]
```

- `interface range`**:** Main command used to configure multiple network interfaces simultaneously.
- `interface_type`**:** This argument specifies the type of interface you want to configure. Examples include Ethernet, FastEthernet, GigabitEthernet, Port-channel, and Vlan.
- `first_interface_number`**/**`first_subinterface_number` **-** `last_interface_number`**/**`last_subinterface_number`**:** These arguments specify the range of interface numbers and subinterface numbers (if applicable) of the selected interface type.

For example, if you want to configure GigabitEthernet interfaces 0/1 through 0/5, you would use the following command:

```
interface range GigabitEthernet 0/1 - 0/5
```

This will allow you to apply the same configuration to all interfaces in the specified range.

show interface brief:

`show interface brief` is not an actual command in Cisco IOS. Instead, the command providing a similar brief summary of interface information is `show interfaces status`. Here's a breakdown of this command's syntax:

- `show`**:** This keyword is used to display information about various aspects of the device's configuration and status.
- `interfaces`**:** This keyword specifies that you want to display information about interfaces on the device.
- `status`**:** This keyword is used to display a brief summary of the interfaces' status, including details such as interface name, status (up or down), VLAN, duplex, and speed.

An example of using this command is:

```
Switch# show interfaces status
```

This command provides a brief summary of the status of all interfaces on the device, which is useful for quickly identifying interface-related issues or verifying the configuration.

Please note that the `ping` command syntax explanation does not apply directly to `show interfaces status`. The example provided for the `ping` command is simply an illustration of how to explain command syntax in general.

switchport:

The `switchport` command is the main command used to configure and manage Layer 2 (L2) switch port settings on a network switch.

```
switchport [sub-command] [options]
```

- `switchport access`: Set access mode characteristics of the interface.

    - `vlan`**:** Set VLAN when the interface is in access mode.
        - `<1-4094>`**:** VLAN ID of the VLAN when this port is in access mode.
- `switchport mode`**:** Set trunking mode of the interface.
    - `access`**:** Set trunking mode to ACCESS unconditionally.
    - `dynamic`**:** Set trunking mode to dynamically negotiate access or trunk mode.
    - `trunk`**:** Set trunking mode to TRUNK unconditionally.
- `switchport nonegotiate`**:** Device will not engage in negotiation protocol on this interface.
- `switchport port-security`**:** Security related command.
- `switchport priority`**:** Set appliance 802.1p priority.
- `switchport protected`**:** Configure an interface to be a protected port.
- `switchport trunk`**:** Set trunking characteristics of the interface.
- `switchport voice`**:** Voice appliance attributes.

For example, if you want to set an interface to access mode and assign it to VLAN 10, you would use the following commands:

```
switchport mode access
switchport access vlan 10
```

exit:

command is a simple command used to exit the current mode or context in a command-line interface (CLI) or shell environment. The syntax for the `exit` command is:

```
exit
```

show interface trunk:

The `show interface trunk` command is used to display information about the trunk interfaces on a network switch. The syntax for the `show interface trunk` command is:

```
show interface trunk
```

There are no options or arguments for the `show interface trunk` command. Its primary purpose is to display the current trunking status and configuration of the switch interfaces, including information on trunking mode, allowed VLANs, and native VLAN.

channel-group:

The `channel-group` command is used to configure an EtherChannel on a Cisco network device. The syntax for the `channel-group` command is:

```
channel-group <group-number> mode {active | auto | desirable | on | passive}
```

- `<group-number>`**:** This argument specifies the channel group number, which is typically a value within a certain range depending on the platform (e.g., 1-6, 1-48, or 1-64).
- `mode`**:** This argument specifies the EtherChannel mode of the interface.
    - `active`**:** This option enables LACP unconditionally, meaning that the switch will actively initiate negotiations with the other device to form the EtherChannel.
    - `auto`**:** This option enables the Port Aggregation Protocol (PAgP) only if a PAgP device is detected.
    - `desirable`**:** This option enables PAgP unconditionally, meaning that the switch will actively initiate negotiations with the other device to form the EtherChannel.
    - `on`**:** This option enables Etherchannel only, meaning that the EtherChannel is formed without any negotiation.
    - `passive`**:** This option enables LACP only if a LACP device is detected, meaning that the switch will wait for the other device to initiate negotiations to form the EtherChannel.

show interfaces port-channel:

The `show interfaces port-channel` command is used to display information about the status and configuration of port-channel (EtherChannel) interfaces on a Cisco network device. The syntax for the `show interfaces port-channel` command is:

```
show interfaces port-channel [options]
```

- `options`**:** These arguments are optional and allow you to further filter or customize the output of the `show interfaces port-channel` command. Some common options include specifying a particular port-channel number to display information only for that specific EtherChannel. The availability of options depends on the specific implementation of the command on the network device.

The `show interfaces port-channel` command provides information about the port-channel interfaces, such as their current status (up or down), protocol status, bandwidth, delay, reliability, and other statistics.

show etherchannel summary:

The `show etherchannel summary` command is used to display a summary of EtherChannel configurations and their status on a Cisco network device. The syntax for the `show etherchannel summary` command is:

```
show etherchannel summary [options]
```

- `options`**:** These arguments are optional and allow you to further filter or customize the output of the `show etherchannel summary` command. Some common options include specifying a particular group number to display information only for that specific EtherChannel group. The availability of options depends on the specific implementation of the command on the network device.

The `show etherchannel summary` command provides a summarized view of EtherChannel configurations, including the port-channel number, port-channel type (Layer 2 or Layer 3), protocol used (PAgP or LACP), port list, and the status of each port in the EtherChannel.

ip address:

```
ip address <assign an IP address and subnet mask>
```

- `ip address` **:** This command is used to assign an IP address and subnet mask to a network interface on a device.
- `<assign an IP address and subnet mask>` **:** This is a placeholder for entering the IP address and subnet mask. You have two options:
  - `A.B.C.D` **:** Enter the IP address in dotted-decimal notation, where each octet is a decimal number between 0 and 255.
  - `dhcp` **:** Use this keyword to enable the device to obtain an IP address automatically via the Dynamic Host Configuration Protocol (DHCP).
- `IP subnet mask` **:** Enter the subnet mask to be assigned to the interface, which is used to identify the network portion and host portion of an IP address.

Example:

```
ip address 192.168.1.1 255.255.255.0
```

no shutdown:

```
no shutdown
```

- `no shutdown` **:** This command is used on network devices, such as routers and switches, to enable a disabled interface and bring it back up. When an interface is administratively shut down, it doesn't forward traffic. The "no shutdown" command is used to reverse this state and make the interface operational.

VLAN:

vlan:

```
vlan [VLAN_ID]
```

- `vlan` **:** Main command used to create or modify a VLAN (Virtual Local Area Network).
- `VLAN_ID` **:** This argument specifies the VLAN ID number you want to create or modify. The ID should be in the range of 1-4094, with ISL VLAN IDs 1-1005 referring to the Inter-Switch Link (ISL) encapsulation protocol used to transmit VLAN information between Cisco switches.

Example usage:

```
vlan 10
```

vlan name:

```
name [VLAN_NAME]
```

- `name` **:** Main command used to assign a name to a VLAN.
- `VLAN_NAME` **:** This argument specifies the name you want to assign to the VLAN. The name can be any string of up to 32 characters.

Example usage:

```
name Sales_VLAN
```

This command assigns the name "Sales_VLAN" to the VLAN. Note that the `name` command should be used after specifying the VLAN ID with the `vlan` command.

show vlan brief:

The `show vlan brief` command is used to display a summary of the VLAN configurations on a network device, such as a switch. The syntax for the `show vlan brief` command is:

```
show vlan brief
```

There are no options or arguments for the `show vlan brief` command. Its primary purpose is to provide a concise overview of the VLANs configured on the device, including VLAN ID, VLAN name, and associated interfaces.

Spanning tree:

show spanning-tree:

The `show spanning-tree` command is used to display information about the Spanning Tree Protocol (STP) status and configuration on a network switch. The syntax for the `show spanning-tree` command is:

```
show spanning-tree [options]
```

- `options` **:** These arguments are optional and allow you to further filter or customize the output of the `show spanning-tree` command. Some common options include specifying a specific VLAN ID or instance number to display STP information for that particular VLAN or instance. The availability of options depends on the specific implementation of STP on the network device.

The `show spanning-tree` command provides detailed information about the STP status, including root bridge, root path cost, bridge priority, designated ports, and forwarding/blocking states of the ports.

CDP and LLDP:

show cdp neighbors:

The `show cdp neighbors` command is used to display information about neighboring devices discovered using the Cisco Discovery Protocol (CDP) on a Cisco network device. The syntax for the `show cdp neighbors` command is:

```
show cdp neighbors [options]
```

- `options`**:** These arguments are optional and allow you to further filter or customize the output of the `show cdp neighbors` command. Some common options include specifying a particular interface to display only the neighbors connected to that interface. The availability of options depends on the specific implementation of CDP on the network device.

The `show cdp neighbors` command provides information about neighboring Cisco devices, including device ID, local interface, hold time, capability, platform, and remote interface.

show lldp neighbors:

The `show lldp neighbors` command is used to display a list of all the neighboring devices discovered using Link Layer Discovery Protocol (LLDP) on a network device, along with basic information about each device. The syntax for the `show lldp neighbors` command is:

```
show lldp neighbors [options]
```

- `options`: These arguments are optional and allow you to further filter or customize the output of the `show lldp neighbors` command. Some common options include specifying a particular interface to display only the neighbors connected to that interface. The availability of options depends on the specific implementation of LLDP on the network device.

The `show lldp neighbors` command provides a summarized view of neighboring devices, including device ID, local interface, hold time, capability, platform, and remote interface.

lldp run:

The `lldp run` command is used in Cisco networking devices to enable the Link Layer Discovery Protocol (LLDP). The syntax for the `lldp run` command is:

```
lldp run
```

There are no options or arguments for the `lldp run` command. Its primary purpose is to enable the LLDP functionality on the Cisco networking device, allowing it to discover and share information with its directly connected neighbors.

Routing configuration:

Router OSPF:

```
router ospf <process ID>
```

- `router ospf`**:** This command is used to enable the OSPF routing process on a network device, such as a router.
- `<process ID>`**:** This argument specifies the OSPF process ID, which is a number between 1 and 65535 that identifies the OSPF routing process on the router. Each OSPF process must have a unique process ID.

Example:

```
Router(config)#router ospf 1
```

OSPF network:

```
network <A.B.C.D> <wildcard mask> [area <area ID>]
```

- `network`**:** This command is used under OSPF router configuration mode to specify the networks that will participate in OSPF.
- `<A.B.C.D>`**:** This argument specifies the IP address of the network that will participate in OSPF. The IP address must be in dotted decimal notation.
- `<wildcard mask>`**:** This argument specifies the wildcard mask that defines which bits of the IP address should be considered. The wildcard mask is also in dotted decimal notation and is used to match the interfaces that should participate in OSPF. A 0 bit in the wildcard mask means that the corresponding bit in the IP address must match exactly, while a 1 bit means that the corresponding bit can be any value.
- `[area <area ID>]`**:** This optional argument specifies the OSPF area ID for the network. If this argument is not specified, the network will be assigned to the default OSPF area, which is area 0.

Example:

```
Router(config-router)#network 192.168.1.0 0.0.0.255 area 0
```

show IP OSPF neighbor:

```
show ip ospf neighbor
```

- `show ip ospf neighbor`: This command is used to display information about OSPF neighbors on a Cisco router. When OSPF is enabled on a router, it forms neighbor relationships with other OSPF-enabled routers on directly connected networks. The command output includes information such as the neighbor router's IP address, the interface that the neighbor is connected to, the state of the neighbor relationship, and other information.

show IP OSPF database:

```
show ip ospf database
```

- `show ip ospf database`: This command is used to display information about the OSPF link-state database (LSDB) on a Cisco router. The OSPF LSDB contains information about all the links in the network topology that are participating in OSPF. This information is used by OSPF routers to calculate the shortest path to a destination network.

show IP interface brief:

```
show ip interface brief
```

- `show ip interface brief`: This command is used to display a summary of the IP addresses assigned to all interfaces on a router or switch. The summary typically includes information such as interface name, IP address, status (up or down), and protocol status (up or down).

show IP route:

```
show ip route
```

- `show ip route`: This command is used to display a table that summarizes the routing information for a network device, such as a router or a switch. The routing table includes information about directly connected networks, static routes, and routes learned from dynamic routing protocols (e.g., OSPF, EIGRP, RIP, and BGP).

no switchport:

```
no switchport
```

- `no switchport`: This command is used on Cisco IOS switches to configure an interface as a Layer 3 (routed) interface rather than a Layer 2 (switched) interface. By default, physical interfaces on Cisco switches operate as Layer 2 switchports. The "no switchport" command is typically entered in the interface configuration mode.

IP routing:

```
ip routing
```

- `ip routing`: This command is used to enable IP routing functionality on a network device, such as a Layer 3 switch or a router. By enabling IP routing, the device can make routing decisions based on IP addresses, allowing traffic to be forwarded between different IP networks.

IP route:

```
ip route A.B.C.D Destination prefix mask [Forwarding router's address | Exit Interface] [distance]
```

- `ip route`: This command is used to configure a static route on a network device, such as a router.
- `A.B.C.D`: The destination prefix to which the static route applies, represented in dotted-decimal notation.
- `Destination prefix mask`: The subnet mask of the destination prefix, used to identify the network and host portions of the destination address.
- `Forwarding router's address`: The IP address of the next hop router that will forward the packet towards the destination prefix.
- Alternatively, you can specify the exit interface for the static route, such as a Dialer, Ethernet, FastEthernet, GigabitEthernet, Loopback, Null, Serial, or VLAN interface.
- `<1-255>` (optional): Distance metric for this route. Lower values indicate more preferred routes. The default administrative distance for a static route is 1.

Example:

```
Router(config)#ip route 1.1.1.1 255.255.255.0 1.1.1.2 10
```

IPv6:

IPv6 unicast-routing:

```
ipv6 unicast-routing
```

- `ipv6 unicast-routing`: This command is used to enable IPv6 routing functionality on a network device, such as a Layer 3 switch or a router. By enabling IPv6 routing, the device can make routing decisions based on IPv6 addresses, allowing IPv6 traffic to be forwarded between different IPv6 networks.

IPv6 address:

```
ipv6 address <WORD | X:X:X:X::X | X:X:X:X::X/<0-128> | autoconfig | dhcp>
```

- `ipv6 address`: This command is used to configure an IPv6 address on a network interface.

- `<WORD>` **:** This is a general prefix name that can be used to specify the IPv6 address. This can be any alphanumeric string of up to 19 characters.
- `<X:X:X:X::X>` **:** This is an IPv6 linklocal address in the standard IPv6 notation. The "X" characters represent the hexadecimal value of each segment of the address.
- `<X:X:X:X::X/<0-128>>` **:** This is an IPv6 prefix in the standard IPv6 notation, followed by a prefix length from 0 to 128 bits.
- `autoconfig` **:** This option enables the interface to obtain an IPv6 address using stateless autoconfiguration.
- `dhcp` **:** This option enables the interface to obtain an IPv6 address using DHCPv6.

```
Router(config)#interface GigabitEthernet0/1
Router(config-if)#ipv6 address 2001:db8:1::1/64
```

IPv6 route:

```
ipv6 route <X:X:X:X::X/<0-128>> [interface | next-hop] [<administrative-distance>]
```

- `ipv6 route` **:** This command is used to configure static IPv6 routes in a network device such as a router.
- `<X:X:X:X::X/<0-128>>` **:** This is the IPv6 prefix and prefix length for the destination network.
- `interface` **:** This is the optional interface through which the traffic should be forwarded (e.g., `Ethernet`, `FastEthernet`, `GigabitEthernet`, `Loopback`, `Serial`, `Vlan`, or `Dialer`).
- `next-hop` **:** This is the optional IPv6 address of the next-hop router to which the traffic should be forwarded.
- `<administrative-distance>` **:** This is the optional administrative distance for the static route, ranging from 1 to 254. Lower values indicate higher preference.

```
Router(config)#ipv6 route fec2::/64 GigabitEthernet0/1 2001::2 10
```

show IPv6 interface brief:

```
show ipv6 interface brief
```

- `show ipv6 interface brief` **:** This command is used to display a table with a summary of IPv6-related information for all interfaces on a network device, such as a router or a switch. The table includes the interface name, the IPv6 address assigned to the interface, the status of the interface (up or down), and other relevant information such as the link-local address and the MAC address of the interface.

show IPv6 route:

```
show ipv6 route
```

- `show ipv6 route` **:** This command is used to display the IPv6 routing table on a network device, such as a router or a Layer 3 switch. The routing table lists the available routes to different IPv6 destinations. It includes information such as the destination IPv6 address, the prefix length, the next-hop address or interface, and the routing protocol used to learn the route.

NAT:

ip nat:

```
ip nat [subcommand] [arguments]
```

Here's a breakdown of the subcommands and their arguments:

1. `ip nat inside source`
   - `inside` **:** Inside address translation
   - `source` **:** Source address translation
2. `ip nat outside source`
   - `outside` **:** Outside address translation
   - `source` **:** Source address translation
3. `ip nat pool WORD A.B.C.D E.F.G.H Netmask I.J.K.L`
   - `pool` **:** Define a pool of addresses
   - `WORD` **:** Pool name
   - `A.B.C.D` **:** Start IP address
   - `E.F.G.H` **:** End IP address
   - `I.J.K.L` **:** Netmask

Here's a breakdown of the `ip nat inside source` command and its arguments:

1. `ip nat inside source list [access-list] pool WORD`
   - `list` **:** Specify access list describing local addresses
   - `[access-list]` **:** Access list number or name for local addresses
   - `pool` **:** Connects the access list with a pool of addresses
   - `WORD` **:** Pool name
2. `ip nat inside source static A.B.C.D E.F.G.H [protocol] [local-port] [global-port]`
   - `static` **:** Specify static local-to-global mapping

- `A.B.C.D`: Inside local IP address
- `E.F.G.H`: Inside global IP address
- `[protocol]`: (Optional) TCP or UDP
- `[local-port]`: (Optional) Inside local port number
- `[global-port]`: (Optional) Inside global port number

Here's a breakdown of the `ip nat outside source` command and its arguments:

1. `ip nat outside source list [access-list] pool WORD`
   - `list`: Specify access list describing local addresses
   - `[access-list]`: Access list number or name for local addresses
   - `pool`: Connects the access list with a pool of addresses
   - `WORD`: Pool name
2. `ip nat outside source static A.B.C.D E.F.G.H [protocol] [global-port] [local-port]`
   - `static`: Specify static global-to-local mapping
   - `A.B.C.D`: Outside global IP address
   - `E.F.G.H`: Outside local IP address
   - `[protocol]`: (Optional) TCP or UDP
   - `[global-port]`: (Optional) Outside global port number
   - `[local-port]`: (Optional) Outside local port number

The `ip nat` command is used for configuring Network Address Translation (NAT) on a router. It has various subcommands and arguments that allow you to configure different types of NAT, such as static NAT, dynamic NAT, and Port Address Translation (PAT).

show ip nat translations:

```
show ip nat translations
```

This command does not have any additional arguments or options. It displays the current NAT translations on the router, including inside local IP address and port, outside global IP address and port, the protocol (TCP or UDP), and the age of the translation.

show ip nat statistics:

```
show ip nat statistics
```

This command does not have any additional arguments or options. It displays statistics about the NAT process on the router, including the number of packets translated by NAT, the number of active translations, and other related statistics.

DHCP:

ip dhcp:

```
ip dhcp [subcommand] [arguments]
```

Here's a breakdown of the subcommands and their arguments:

1. `ip dhcp excluded-address A.B.C.D E.F.G.H`
   - `excluded-address`: The subcommand to prevent DHCP from assigning certain addresses.
   - `A.B.C.D`: The low IP address of the range to be excluded.
   - `E.F.G.H`: (Optional) The high IP address of the range to be excluded. If not provided, only the single IP address specified by A.B.C.D will be excluded.

   Example:

   ```
   ip dhcp excluded-address 192.168.1.1 192.168.1.10
   ```
2. `ip dhcp pool WORD`
   - `pool`: The subcommand to configure DHCP address pools.
   - `WORD`: The name of the pool to be configured.

   Example:

   ```
   ip dhcp pool mypool
   ```
3. `ip dhcp relay [arguments]`
   - `relay`: The subcommand for DHCP relay agent parameters.
   - `[arguments]`: Additional arguments to configure the DHCP relay agent.

DHCP pool:

```
Router(dhcp-config)# [command] [arguments]
```

Here's a breakdown of the commands and their arguments:

1. `Router(dhcp-config)# default-router A.B.C.D`
    - `default-router`: The command to set the default router (gateway) for the DHCP pool.
    - `A.B.C.D`: The IP address of the default router.

    Example:

    ```
    Router(dhcp-config)# default-router 192.168.1.1
    ```

2. `Router(dhcp-config)# dns-server A.B.C.D`
    - `dns-server`: The command to set the DNS server for the DHCP pool.
    - `A.B.C.D`: The IP address of the DNS server.

    Example:

    ```
    Router(dhcp-config)# dns-server 8.8.8.8
    ```

3. `Router(dhcp-config)# domain-name WORD`
    - `domain-name`: The command to set the domain name for the DHCP pool.
    - `WORD`: The domain name to be used.

    Example:

    ```
    Router(dhcp-config)# domain-name example.com
    ```

4. `Router(dhcp-config)# network A.B.C.D E.F.G.H`
    - `network`: The command to set the network number and mask for the DHCP pool.
    - `A.B.C.D`: The network number in dotted-decimal notation.
    - `E.F.G.H`: The network mask in dotted-decimal notation.

    Example:

    ```
    Router(dhcp-config)# network 192.168.1.0 255.255.255.0
    ```

5. `Router(dhcp-config)# option [arguments]`
    - `option`: The command to configure raw DHCP options.
    - `[arguments]`: Additional arguments to configure the raw DHCP options.

ip dhcp binding:

```
show ip dhcp binding
```

This command does not have any additional arguments or options. It is used to display the DHCP bindings on a router or switch. DHCP is a protocol that automatically configures IP addresses and related network settings for hosts on a network.

When you execute the `show ip dhcp binding` command, the router or switch will display a table containing the IP addresses, MAC addresses, lease expiration times, and associated VLANs (if applicable) for each DHCP binding.

ip helper-address:

```
ip helper-address A.B.C.D
```

Here's a breakdown of the command and its argument:

- `ip helper-address`: This command is used to configure a DHCP relay agent on a router or switch. It forwards DHCP requests from clients on one subnet to a DHCP server on another subnet.
- `A.B.C.D`: This argument specifies the IP destination address, which is the IP address of the remote DHCP server that the router or switch should forward DHCP requests to.

Example:

```
ip helper-address 192.168.2.1
```

In this example, the command tells the router or switch to forward DHCP requests to the DHCP server located at IP address 192.168.2.1.

Access Control Lists:

ip access-group:

```
ip access-group [access-list-identifier] [direction]
```

- `ip access-group`: The command used to apply an access control list (ACL) to an interface to filter traffic.
- `access-list-identifier`: This argument specifies the identifier of the access control list you want to apply. It can be one of the following:
    - `<1-199>`: IP access list number (standard or extended). In this range, numbers 1-99 are reserved for standard ACLs and 100-199 are reserved for extended ACLs.

    - `WORD`: Access-list name. This is an alphanumeric name to identify the access control list, providing a more descriptive way to reference the ACL.
- `direction`: This argument indicates the direction of traffic that the access control list will filter. There are two options:
    - `in`: Apply the access control list to inbound packets entering the interface.
    - `out`: Apply the access control list to outbound packets leaving the interface.

The `ip access-group` command is used to apply an access control list (ACL) to an interface in order to filter traffic based on the rules defined within the ACL. By specifying the access control list identifier (number or name) and the direction of traffic to filter (inbound or outbound), you can control the flow of traffic through the interface. This command is commonly used in network devices like routers and switches to enforce security policies and limit access to specific network resources or services.

show access-lists:

```
show access-lists [access-list-name]
```

- `access-list-name` (Optional): This argument allows you to display a specific access list by its name. If you don't provide an access-list name, the command will display all access lists configured on the device.

The `show access-lists` command displays the access control lists (ACLs) configured on a networking device, such as a router or switch. It shows the details of each access list, including its name, sequence number, actions (permit or deny), and the matching criteria (such as source and destination IP addresses, protocol, and port numbers). This command helps you verify the current ACL configuration and monitor its usage.

For example, if you want to display the details of the access list named "HTTP", you would use the following syntax:

```
show access-lists HTTP
```

This command will display the access list "HTTP" and its rules, including the sequence numbers, actions (permit or deny), and the matching criteria.

Router(config-ext-nacl)# (under extended ACL):

```
Router(config-ext-nacl)# [sequence_number] [action] [protocol] [source] [source_wildcard] [destination] [destination_wildcard]
```

- `sequence_number` (Optional): An integer between 1 and 2147483647, specifying the order in which the access control entries (ACEs) are evaluated. Lower numbers have higher priority.
- `action`: Specifies whether to permit or deny the packet. Options include:
    - `permit`: Specify packets to forward.
    - `deny`: Specify packets to reject.
- `protocol`: Specifies the protocol of the packet. Some common options include `ip` for any Internet Protocol, `icmp` for Internet Control Message Protocol, `tcp` for Transmission Control Protocol, `udp` for User Datagram Protocol, and others like `ahp`, `eigrp`, `esp`, `gre`, and `ospf`.
- `source`: Specifies the source IP address or host. It can be in the format `A.B.C.D` for a specific IP address, `any` for any source host, or `host` followed by an IP address for a single source host.
- `source_wildcard`: Specifies the source wildcard bits (inverse netmask) for the source IP address.
- `destination`: Specifies the destination IP address or host. It can be in the format `A.B.C.D` for a specific IP address, `any` for any destination host, or `host` followed by an IP address for a single destination host.
- `destination_wildcard`: Specifies the destination wildcard bits (inverse netmask) for the destination IP address.

The `Router(config-ext-nacl)#` command is used to configure the rules for an extended access control list (ACL) on a router. Extended ACLs allow you to filter traffic based on both the source and destination IP addresses, as well as the protocol and port numbers. By specifying the action (permit or deny), protocol, source and source wildcard, destination and destination wildcard, you can create precise rules to control the flow of traffic in your network.

For example, to create an ACL rule that denies all ICMP traffic from any source to the destination IP address 192.168.1.0/24, you would use the following syntax:

```
Router(config-ext-nacl)# deny icmp any 192.168.1.0 0.0.0.255
```

This command denies ICMP traffic from any source IP address to destination IP addresses in the range of 192.168.1.0 to 192.168.1.255.

access-list:

1. `access-list`

    ```
    access-list [list-number] [action] [criteria]
    ```

    - `list-number`: A number indicating the access list type. Ranges are as follows:
        - `<1-99>`: IP standard access list
        - `<100-199>`: IP extended access list
    - `action`: The action to take when the criteria in the access list are met. Options include:
        - `deny`: Specify packets to reject
        - `permit`: Specify packets to forward
        - `remark`: Access list entry comment
    - `criteria`: The criteria to match against packets. This may differ depending on the access list type and action.
2. `access-list 1 permit`

    ```
    access-list 1 permit [source-address] [wildcard-bits]
    ```

    - `source-address`: The source IP address to match. Options include:

    - `A.B.C.D`: Address to match
    - `any`: Any source host
    - `host`: A single host address
  - `wildcard-bits`: (Only for `A.B.C.D` source address) Wildcard bits to apply to the source IP address

3. `access-list 1 permit host`

```
access-list 1 permit host [host-address]
```

- `host-address`: The single host address to match (A.B.C.D)

The `access-list` command is used to create and configure access control lists (ACLs) on a router. ACLs are used to filter traffic based on specified criteria, such as source or destination IP addresses, and to control which traffic is permitted or denied through the router. They are commonly used for security and traffic management purposes.

Switch security:

switchport port-security:

```
switchport port-security [options]
```

- `options`:
  - `aging`: This argument enables port-security aging commands.
    - `time`: This option allows you to set port-security aging time.
      - `<1-1440>`: Enter a value between 1 and 1440 to specify aging time in minutes.
  - `mac-address`: This argument sets a secure MAC address.
    - `H.H.H`: Enter a 48-bit MAC address.
    - `sticky`: This option configures dynamically learned secure addresses as sticky (retained across device reloads).
  - `maximum`: This argument sets the maximum number of secure addresses allowed on the interface.
    - `<1-132>`: Enter a value to specify the maximum number of addresses.
  - `violation`: This argument determines the action to take when a security violation occurs.
    - `protect`: Security violation protect mode – drop packets from violating MAC addresses.
    - `restrict`: Security violation restrict mode – drop packets from violating MAC addresses and send an SNMP trap.
    - `shutdown`: Security violation shutdown mode – disable the interface upon violation.

The `switchport port-security [options]` command allows you to enable and configure port security on a specific interface with various options for aging, MAC address handling, maximum secure addresses, and violation actions.

show port-security interface:

```
show port-security interface [interface]
```

- `interface`: This argument specifies the target interface for which you want to display port security information.

In this case, the command is:

`show port-security interface FastEthernet 0/1`

- `FastEthernet 0/1`: This argument specifies the FastEthernet 0/1 interface as the target interface for which you want to display port security information.

The `show port-security interface [interface]` command allows you to view the port security configuration and status of the specified interface.

show ip dhcp snooping:

```
show ip dhcp snooping
```

This command displays the current status and configuration of DHCP Snooping on the switch. It does not require any additional options or arguments.

When you enter the `show ip dhcp snooping` command, the switch displays information about the DHCP Snooping status, such as whether it is enabled globally, the list of VLANs for which it is enabled, and the configuration of individual interfaces with respect to DHCP Snooping.

ip dhcp relay:

```
ip dhcp relay [options]
```

- `options`:
  - `information`: Configure the relay agent information option.

`ip dhcp relay information [options-information]`

- `options-information`:
  - `trust-all`: Allow received DHCP packets to contain relay info option with zero giaddr (gateway IP address).

The `ip dhcp relay [options]` command allows you to configure the DHCP relay agent settings on the switch or router. The `ip dhcp relay information [options-information]` command is used to configure the relay agent information option, such as trusting all received DHCP packets that contain relay info option with a zero Gateway IP Address.

ip dhcp snooping:

```
ip dhcp snooping [options]
```

- `options`:
  - `database`: Configure the DHCP Snooping database agent.
  - `information`: Configure DHCP Snooping information options.
  - `verify`: Configure DHCP Snooping verification options.
  - `vlan`: Configure DHCP Snooping for specific VLANs.
  - `<cr>`: Press Enter to enable DHCP Snooping globally without additional options.

`ip dhcp snooping vlan [vlan_range]`

- `vlan_range`: Specify the first VLAN number or a range of VLANs (e.g., 1, 3-5, 7, 9-11) to enable DHCP Snooping for those VLANs.

Under the interface configuration mode **(Switch(config-if)#):**

- `ip dhcp snooping [options-interface]`
  - `options-interface`:
    - `limit`: Configure the DHCP Snooping rate limit.
    - `trust`: Configure the DHCP Snooping trust setting for the interface.

`ip dhcp snooping limit rate [rate_value]`

- `rate_value`: Specify a value between 1 and 2048 to set the DHCP Snooping rate limit.

The `ip dhcp snooping [options]` command enables and configures DHCP Snooping on the switch globally and for specific VLANs. In interface configuration mode, you can configure rate limits and trust settings for individual interfaces.

Administration:

hostname:

```
hostname <WORD>
```

- `<WORD>`: This argument is the name you want to assign to the device's hostname.

The `hostname` command is used to set or change the hostname of a network device, such as a router or switch. The hostname serves as an identifier for the device and is particularly useful in larger networks or when managing multiple devices. By assigning a unique and descriptive hostname, you can more easily identify and manage the device.

ip domain-name:

```
ip domain-name <WORD>
```

- `<WORD>`: This argument is the actual domain name you want to set as the default for the network device.

The `ip domain-name` command is used to set or configure the default domain name on a network device, such as a router or switch. This default domain name is appended to any unqualified hostnames (hostnames without a domain) during DNS resolution. Setting a default domain name can help simplify network configurations and make it easier to manage devices within a specific domain.

description:

The `description` command is used to add a descriptive text to a network device interface, such as a switch or router interface. The syntax for the `description` command is:

```
description <description_text>
```

- `description_text`: This argument is a text string that provides information about the purpose or function of the interface. It can be up to 240 characters long and helps to identify the interface's role in the network.

The `description` command is typically used in the interface configuration mode of a network device. For example:

```
interface FastEthernet0/1
description Up to 240 characters describing this interface
```

crypto:

```
crypto key generate rsa general-keys modulus <360-4096>
```

- `crypto`: This command is used to configure cryptographic features.
- `key`: This specifies that you are performing long term key operations.
- `generate`: This indicates that you want to generate new keys.
- `rsa`: This specifies that you want to generate RSA keys.
- `general-keys`: Generate a general-purpose RSA key pair for signing and encryption.
- `modulus`: The next argument will provide the number of modulus bits for the RSA key.
- `<360-4096>`: This is a placeholder for the actual key size, in bits. Replace this with a value between 360 and 4096 to specify the desired key size.

The `crypto` command is used to configure cryptographic features on a network device, such as a router or switch. In this specific use case, the command is focused on generating an RSA key pair for secure communications. The key pair consists of a public and private key, which are used for encryption and decryption, as well as signing and verifying messages.

ip ssh:

```
ip ssh [option]
```

- `option`**:** This argument allows you to customize the behavior of the SSH settings. Some common options include:
  - `authentication-retries`**:** Specify the number of authentication retries.
  - `time-out`**:** Specify the SSH time-out interval.
  - `version`**:** Specify the protocol version to be supported.
    - `<1-2>`**:** The desired SSH protocol version, expressed as an integer (either 1 or 2).

The `ip ssh` command is used to configure Secure Shell (SSH) settings on a network device, such as a router or switch. SSH is a secure protocol for remote device management, providing encryption and authentication to ensure that data transmitted between devices remains confidential and protected from unauthorized access.

line:

```
line [option]
```

- `line`**:** Main command to configure the different types of terminal lines for device access.
- `option`**:** This argument allows you to choose the type of terminal line you want to configure. Options include:
  - `<2-499>`**:** First line number for a range of line numbers to configure.
  - `aux`**:** Auxiliary line.
  - `console`**:** Primary terminal line.
  - `tty`**:** Terminal controller.
  - `vty`**:** Virtual terminal.
    - `<0-15>`**:** This argument specifies the first line number for the range of VTY lines to be configured. The range of line numbers is from 0 to 15, indicating a total of 16 available VTY lines.
  - `x/y/z`**:** Slot/Subslot/Port for modems.

The `line` command is used to configure the different types of terminal lines for device access on a network device, such as a router or switch. Terminal lines are used for remote or local device management, and configuring them correctly is essential for secure and efficient device access.

transport:

```
transport [direction] [option]
```

- `transport`**:** Main command to configure the transport protocols for terminal lines.
- `direction`**:** This argument allows you to choose the direction of the transport protocol configuration. Options include:
  - `input`**:** Define which protocols to use when connecting to the terminal server (incoming connections).
  - `output`**:** Define which protocols to use for outgoing connections.
- `option`**:** This argument allows you to choose the protocol(s) to be used for the selected direction. Options include:
  - `all`**:** All available protocols.
  - `none`**:** No protocols, effectively disabling connections in the selected direction.
  - `ssh`**:** TCP/IP SSH protocol for secure remote connections.
  - `telnet`**:** TCP/IP Telnet protocol for remote connections.

The `transport` command is used to configure the transport protocols for terminal lines on a network device, such as a router or switch. Terminal lines are used for remote or local device management, and configuring the transport protocols is essential to establish secure and efficient connections.

login:

```
login [option]
```

- `login`**:** Main command to configure the authentication method for terminal lines.
- `option`**:** This argument allows you to choose the authentication method to be used for terminal lines. Options include:
  - `authentication`**:** Authenticate using an AAA (Authentication, Authorization, and Accounting) method list. This option typically requires you to specify the name of the AAA method list configured on the device.
  - `local`**:** Use local password checking for authentication. This option relies on the locally configured username and password on the device.

The `login` command is used to configure the authentication method for terminal lines on a network device, such as a router or switch. Terminal lines are used for remote or local device management, and configuring the authentication method is essential for secure and efficient access to the device.

username:

```
username [WORD] [option] [sub-option]
```

- `username`**:** Main command to configure user accounts on the device.
- `WORD`**:** User name for the account being configured.
- `option`**:** This argument allows you to choose an option for configuring the user account. Options include:
  - `password`**:** Specify the password for the user.

  - `privilege`**:** Set the user privilege level.
- `sub-option`**:** This argument provides additional options for the chosen option, such as specifying the privilege level or password encryption type.
  - `level`**:** `<0-15>`**:** User privilege level, where 0 is the lowest and 15 is the highest level. Higher levels grant more access rights to the user.
    - `password`: Specify the password for the user.
      - `encryption`: This argument allows you to choose the type of password encryption. Options include:
        - `0`**:** Specifies an unencrypted (cleartext) password will follow.
        - `7`**:** Specifies a hidden (encrypted) password will follow.
          - `password`: The actual password for the user, either in unencrypted or encrypted form based on the chosen encryption type.

The `username` command is used to configure user accounts on a network device, such as a router or switch. Configuring user accounts is crucial for managing access control and ensuring proper security for the device.

enable:

```
enable <option> [encryption] [password]
```

- `enable`**:** Main command to assign the privileged level password or secret.
- `<option>`**:** This argument allows you to choose between assigning a password or a secret for the privileged level. Options include:
  - `password`**:** Assign the privileged level password.
    - `encryption`**:** This argument allows you to choose the type of password encryption. Options include:
      - `7`**:** Specifies a hidden (encrypted) password will follow.
      - `LINE`**:** The unencrypted (cleartext) 'enable' password.
        - `password`: The actual privileged level password, either in unencrypted or encrypted form based on the chosen encryption type.
  - `secret`**:** Assign the privileged level secret.

The `enable` command is used to configure the privileged level password or secret on a network device, such as a router or switch. The privileged level (also called "enable mode") grants users additional access rights, allowing them to execute more powerful commands for device management and configuration.

ssh:

```
ssh [options] [destination]
```

- `ssh`**:** Main command to initiate an SSH (Secure Shell) session with a remote device or server.
- `options`**:** This argument allows you to customize the behavior of the SSH command. Some common options include:
  - `l`**:** Specify the username for the SSH session. This option is followed by the username.
- `destination`**:** This argument specifies the target device or server you want to establish an SSH session with. It is usually an IP address or a hostname.

The `ssh` command is used to initiate a secure and encrypted remote terminal session with a network device or server, such as a router, switch, or server. SSH is commonly used for remote device management, configuration, and troubleshooting, providing a secure way to access and control remote systems over an unsecured network.

password:

```
password [option]
```

- `password`**:** The command used to set a password for various contexts, such as a console or vty line.
- `option`**:** This argument allows you to customize the behavior of the `password` command. Some options include:
  - `7`**:** Specifies that a hidden (encrypted) password will follow. The password will be encrypted using a weak Type 7 encryption, which can be easily decrypted but provides a basic level of obfuscation.
  - `LINE`**:** Specifies that an unencrypted (cleartext) password will follow. The password will be stored in the configuration file as plain text, which is less secure but easier to read and understand when viewing the configuration.

The `password` command is used to set a password for various contexts in network devices, such as console lines or vty lines for remote access. By specifying the desired option, you can choose to store the password as encrypted (hidden) or unencrypted (cleartext). Note that the Type 7 encryption used for hidden passwords is considered weak and can be easily decrypted, so it should not be relied upon for strong security. It is generally recommended to use stronger methods for securing access, such as using SSH with public key authentication.

Time Protocol:

show clock:

```
show clock
```

- No arguments: This command has no arguments or options. When executed in a command-line interface, the device will display the current time and date according to the device's internal clock.

The `show clock` command is used to display the current date and time on a network device, such as a router or switch. This is useful for verifying the device's internal clock settings or for checking the time when troubleshooting time-sensitive issues.

show ntp associations:

```
show ntp associations
```

- No arguments: This command has no arguments or options. When executed in a command-line interface, the device will display a list of the NTP peers with which it is currently synchronized.

The `show ntp associations` command is used to display the current Network Time Protocol (NTP) associations on a network device, such as a router or switch. This is useful for verifying the NTP configuration and ensuring that the device is correctly synchronizing its time with the appropriate NTP servers or peers.

show ntp status:

```
show ntp status
```

- No arguments: This command has no arguments or options. When executed in a command-line interface, the device will display information about the device's NTP configuration and synchronization status.

The `show ntp status` command is used to display the current Network Time Protocol (NTP) status on a network device, such as a router or switch. This is useful for verifying the NTP configuration, understanding the device's synchronization status, and ensuring that the device is maintaining accurate time.

clock timezone:

```
clock timezone <timezone-name> <hours-offset-from-utc> [<minutes-offset-from-utc>]
```

- `<timezone-name>`**:** This argument is used to specify the name of the time zone you want to configure on the device.
- `<hours-offset-from-utc>`**:** This argument is used to specify the hours offset from UTC (Coordinated Universal Time) for the time zone. The range is between -23 and +23 hours.
- `[<minutes-offset-from-utc>]`**:** (Optional) This argument is used to specify the minutes offset from UTC for the time zone. The range is between 0 and 59 minutes.

The `clock timezone` command is used to configure the time zone on a network device, such as a router or switch. This is important for accurate timekeeping and log entries on the device. The command requires you to specify the time zone name, as well as the hours and optionally minutes offset from Coordinated Universal Time (UTC).

ntp:

```
ntp <setting> [arguments]
```

- `authenticate`**:** Enable authentication for time sources.
- `authentication-key`**:** Configure an authentication key for trusted time sources. The authentication key is used to verify that the time source is trusted.
  - `<key-number> md5 <key>`**:** Specify the key number, the authentication algorithm (md5), and the key.
- `master`**:** Configure the device as an NTP master clock. An NTP master clock is a device that provides accurate time to other devices on the network.
  - `[stratum]`: (Optional) Specify the stratum level of the NTP master (1-15).
- `server`**:** Configure an NTP server. An NTP server is a device that provides time to other devices on the network.
  - `<ip-address>`**:** Specify the IP address of the NTP server.
  - `[key <key-number>]`**:** (Optional) Specify the authentication key number for the NTP server.
- `trusted-key`**:** Configure key numbers for trusted time sources. The key numbers are used to identify the trusted time sources.
  - `<key-number>`**:** Specify the key number.
- `update-calendar`**:** Configure the device to update the calendar using NTP time.

The `ntp` command is used to configure Network Time Protocol (NTP) settings on a network device, such as a router or switch. NTP ensures that devices on the network have accurate and synchronized time, which is important for log entries, time-based access control, and other time-sensitive operations. The command has various settings and arguments, allowing you to configure NTP authentication, master clock, server, trusted keys, and calendar updates.

ROAS:

encapsulation:

`encapsulation dot1Q` is a command used to configure a subinterface on a router so that it can understand and process VLAN-tagged frames. Here's a breakdown of the command's syntax:

- `encapsulation`**:** This keyword indicates that you want to specify the encapsulation method for the subinterface.
- `dot1Q`**:** This keyword represents IEEE 802.1Q, which is a VLAN standard for carrying VLAN-tagged frames over Ethernet networks.
- `<1-4094>`**:** This is a placeholder for the VLAN ID, which is a unique identifier for the VLAN. The valid range for VLAN IDs is 1 to 4094.
- `native`**:** This keyword, when used after the VLAN ID, designates the specified VLAN as the native VLAN for the subinterface. Frames belonging to the native VLAN are not tagged when sent over the trunk link.

An example of using this command is:

```
Router(config-subif)# encapsulation dot1Q 10
```

This command configures the subinterface to process frames with the VLAN ID 10 using the IEEE 802.1Q encapsulation method.

To configure the native VLAN, you would use the command like this:

```
Router(config-subif)# encapsulation dot1Q 1 native
```

This command designates VLAN 1 as the native VLAN for the subinterface.

Basic config:

ping:

```
ping [options] [destination]
```

- **`ping`:** Main command to send ICMP echo request packets to a target device or server and receive ICMP echo replies, allowing you to test the network connectivity between the source and target.
- **`options`:** This argument allows you to customize the behavior of the `ping` command. Some common options include:
  - **`c`:** Specify the number of ping packets to send. This option is followed by the desired number of packets.
  - **`t`:** Send ping packets continuously until you interrupt the command.
  - **`s`:** Specify the size of the ping packet. This option is followed by the desired packet size in bytes.
  - `i`: Specify the interval between ping packets. This option is followed by the desired interval in seconds.
- **`destination`:** This argument specifies the target device or server you want to ping. It can be an IP address or a hostname.

Example usage with options:

```
ping -c 4 -s 100 -i 2 192.168.1.1
```

This command sends 4 ICMP echo request packets to the device or server at IP address 192.168.1.1 with a packet size of 100 bytes and an interval of 2 seconds between packets. It then receives ICMP echo replies, testing the network connectivity between the source and the target.

```
ping -c 3 -s 64 -i 1 192.168.1.2
```

This command sends 3 ICMP echo request packets to the device or server at IP address 192.168.1.2 with a packet size of 64 bytes and an interval of 1 second between packets. It then receives ICMP echo replies, testing the network connectivity between the source and the target.

show:

```
show [sub-command] [options]
```

- **`sub-command`:** This argument is used to specify the type of information you want to display. There are many sub-commands available, depending on the network device and its capabilities. Some common sub-commands include `interfaces`, `running-config`, `vlan`, and `ip route`.
- **`options`:** These arguments are optional and allow you to further filter or customize the output of the `show` command. The availability of options depends on the specific sub-command being used.

show run:

```
show run
```

- **`show run`:** This command, short for "show running-config," is used to display the configuration stored in a network device's RAM memory. This active configuration is what the device is currently using to operate. The command can be executed in privileged EXEC mode for routers and switches, such as those running Cisco IOS.

show running-config:

```
show running-config
```

This command does not have any additional arguments or options. It displays the current configuration of the router, including all entered commands related to NAT, VLANs, interfaces, routing protocols, and other features.

no:

```
no [command]
```

- **`no`:** This keyword is used in Cisco IOS configuration mode to negate or remove a previously configured command or setting. It is typically placed before the command you want to reverse.
- **`[command]`:** This is a placeholder for the command or configuration setting you want to negate or remove.

end:

```
end
```

- **`end`:** This command is used in the CLI (Command-Line Interface) of a Cisco device to exit the current configuration mode and return to privileged EXEC mode. Privileged EXEC mode allows you to access more advanced commands and perform administrative tasks on the device.