ENCOR Study Guide

Alexander

May 3, 2025

OCG

Forwarding and Layer 2

VLANs are defined in the IEEE 802.1Q standard, which states that 32 bits are added to the packet header in between the Source MAC and the EtherType/Size fileds. 16 bits for the Tag protocol identifier (TPID) and 16 for the Tag control information (TCI). The TPID is a 16-bit field set to a value of 0x8100 in order to identify the frame as an IEEE 802.1Q-tagged frame. This field is located at the same position as the EtherType field in untagged frames, and is thus used to distinguish the frame from untagged frames. The TCI can be broken into the following subfields: Priority code point (PCP) which is 3 bits and refers to 802.1p CoS and maps to the frame priority level, Drop eligible indicator (DEI) which is a 1 bit and may be used in conjuction or separately from PCP to indicate if the frame is eligible to be dropped in the presence of congestion, VLAN identifier (VID) is a 12 bit field specifying the VLAN to which the frame belongs.

The VLAN identifier has only 12 bits, which provide 4094 unique VLANs. Catalyst switches use the following logic for VLAN identifiers:

- VLAN 0 is reserved for 802.1p traffic and cannot be modified or deleted.
- VLAN 1 is the default VLAN and cannot be modified or deleted.
- VLANs 2 to 1001 are in the normal VLAN range and can be added, deleted, or modified as necessary.
- VLANs 1002 to 1005 are reserved and cannot be deleted.
- VLANs 1006 to 4094 are in the extended VLAN range and can be added, deleted, or modified as necessary.

A routed subinterface is required when there are multiple VLANs on a switch that require routing, and it is not desirable to use a dedicated physical routed interface per VLAN, or there are not enough physical router interfaces to accommodate all the VLANs. To overcome this issue, it is possible to create a trunk port on the switch and create a logical subinterface on the router. A subinterface is created by appending a period and a numeric value after the period. Then the VLAN needs to be associated with the subinterface with the command encapsulation dot1q vlan-id.

With Catalyst switches, it is possible to assign an IP address to a switched virtual interface (SVI), also known as a VLAN interface. An SVI is configured by defining the VLAN on the switch and then defining the VLAN interface with the command interface vlan vlan-id.The switch must have an interface associated to that VLAN in an up state for the SVI to be in an up state. If the switch is a multilayer switch, the SVIs can be used for routing packets between VLANs without the need of an external router.

Some network designs include a point-to-point link between switches for routing. For example, when a switch needs to connect to a router, some network engineers would build out a transitVLAN (for example, VLAN 2001), associate the port connecting to the router to VLAN 2001 could exist elsewhere in the Layer 2 realm, or that a spanning tree could impact the topology. Instead, the multilater switch port can be converted from a Layer 2 switch port to a routed switch port with the interface configuration command no switchport. Then the IP address can be assigned to it.

Access Port: An access port is a type of switch port that is configured to carry traffic for only one VLAN (Virtual Local Area Network). It does not tag the Ethernet frames with VLAN identifiers and is used to connect end devices such as computers and printers to the network.

ARP (Address Resolution Protocol): ARP is a network protocol used to map an IP address (Layer 3) to a MAC address (Layer 2) within a local network. When a device needs to communicate with another device on the same local network and knows its IP address, ARP is used to discover the corresponding MAC address to facilitate frame delivery.

Broadcast Domain: A broadcast domain is a logical segment of a network in which any broadcast packet sent by a device is received by all other devices within the same domain. Broadcast domains are typically bounded by routers or layer 3 devices, as these devices prevent broadcast packets from crossing into other segments of the network.

CEF (Cisco Express Forwarding): CEF is a high-performance, Layer 3 packet switching technique used in Cisco routers. It improves packet forwarding efficiency by maintaining a Forwarding Information Base (FIB) and an adjacency table to quickly route packets through the network without requiring extensive routing table lookups.

Collision Domain: A collision domain is a network segment where data packets can collide with one another when being sent over the same network medium. In Ethernet networks, collision domains are typically bounded by switches or bridges that separate segments to reduce packet collisions.

CAM (Content Addressable Memory): CAM is a type of memory used in network switches to store MAC addresses and their associated ports. It allows for rapid lookup of MAC addresses when forwarding frames by comparing the incoming MAC address against stored entries to determine the appropriate outgoing port.

Layer 2 Forwarding: Layer 2 forwarding refers to the process of switching Ethernet frames based on MAC addresses at the Data Link layer (Layer 2) of the OSI model. The switch uses the MAC address table to forward frames to the appropriate port.

Layer 3 Forwarding: Layer 3 forwarding involves routing packets based on IP addresses at the Network layer (Layer 3) of the OSI model. Routers use routing tables and algorithms to determine the best path for forwarding packets across different network segments.

FIB (Forwarding Information Base): The FIB is a data structure used by routers and switches to store routing or forwarding information for quick lookup and packet forwarding decisions. It is used by CEF in Cisco devices to determine the outgoing interface for packets based on their destination IP addresses.

MAC Address Table: The MAC address table, also known as the forwarding table or content addressable memory (CAM) table, is used by network switches to map MAC addresses to specific switch ports. This table enables efficient frame forwarding by directing traffic only to the port associated with the destination MAC address.

Native VLAN: The native VLAN is a VLAN that is associated with an 802.1Q trunk port where untagged frames are sent. The native VLAN's purpose is to ensure that untagged traffic is correctly associated with a VLAN, preventing VLAN tag miscommunication on trunk links.

Process Switching: Process switching is a method of packet forwarding in which the CPU of a router processes each packet individually. This method involves a high degree of CPU intervention, which can result in slower forwarding performance compared to hardware-based methods.

RIB (Routing Information Base): The RIB is a data structure used by routers to store routing information learned from various routing protocols. It contains details about network routes and their metrics, which are used by the router to make forwarding decisions.

Trunk Port: A trunk port is a type of switch port that can carry traffic for multiple VLANs by tagging frames with VLAN identifiers. It is used to connect switches or other networking devices to maintain VLAN information across

network segments.

TCAM (Ternary Content Addressable Memory): TCAM is a specialized type of memory used in network devices to perform high-speed lookups of IP addresses and other data. Unlike traditional CAM, TCAM supports three states (0, 1, and "don't care"), enabling more complex matching operations used in features like access control lists (ACLs) and routing tables.

VLAN (Virtual Local Area Network): A VLAN is a logical network segment that groups together devices on different physical LANs into a single broadcast domain. VLANs are used to segment network traffic for security, performance, and organizational purposes, isolating traffic between different groups of devices even if they share the same physical network infrastructure.

Forwarding architectures refer to the methods and systems that network devices (like routers and switches) use to process and forward packets through a network. The main types of forwarding architectures: store-and-forward, cut-through, fragment-free, Virtual Output Queuing (VOQ), Application-Specific Integrated Circuits (ASICs), network processors, Software-Defined Networking (SDN). Store-and-forward is a packet-switching method in which a network device receives the entire packet before forwarding it to the next hop. This approach allows the device to perform error checking on the received packet, typically using Cyclic Redundancy Check (CRC) mechanisms to ensure data integrity. By buffering the entire packet, the device can also handle packets of varying sizes and manage them more effectively. While this method enhances reliability and can reduce the chances of forwarding corrupted packets, it introduces latency, as the device must wait until the complete packet is received before any forwarding action is taken. This delay can impact overall network performance, especially in high-throughput environments where speed is critical. Cut-through switching is a method that allows a network device to begin forwarding a packet as soon as it reads the destination address, without waiting for the entire packet to be received. This technique significantly reduces latency, making it faster than the store-and-forward method. As soon as the switch identifies the destination port, it starts transmitting the packet towards its destination, which is particularly beneficial in scenarios where low latency is paramount, such as in real-time applications. However, cut-through switching lacks error checking until the entire packet has been transmitted, which can lead to forwarding of corrupted packets if issues arise during transmission. Consequently, while cut-through offers speed advantages, it does so at the potential cost of data integrity. Fragment-free switching represents a compromise between the store-and-forward and cut-through methods. In this approach, the switch reads the first 64 bytes of a packet, which typically contain the header information and any collision data. If the initial segment appears valid, the switch begins forwarding the packet, thus avoiding the transmission of fragmented packets caused by collisions. This method reduces the likelihood of forwarding corrupted packets while still allowing for lower latency than pure store-and-forward systems. While fragment-free switching enhances performance in environments with high collision rates, it does not offer the same level of reliability as full store-and-forward, since it only checks a portion of the packet before forwarding. Virtual Output Queuing (VOQ) is a sophisticated switching architecture that addresses the issue of head-of-line blocking in traditional queuing systems. In VOQ, each input port maintains a separate queue for each output port, allowing packets destined for different outputs to be processed concurrently. This architecture enables more efficient use of bandwidth and minimizes latency by allowing multiple flows to be forwarded without being impeded by packets queued at the front of the line. VOQ is particularly advantageous in high-capacity switches and routers, as it optimizes resource utilization and enhances throughput. However, implementing VOQ can be complex, requiring additional memory and management resources, making it more challenging to deploy in simpler network environments. Application-Specific Integrated Circuits (ASICs) are specialized hardware components designed for high-speed packet processing in networking devices. ASICs are optimized for specific tasks, such as packet forwarding, routing, or encryption, enabling them to operate with high efficiency and low latency. By utilizing parallel processing and dedicated pathways for data, ASICs can handle vast amounts of traffic simultaneously, making them ideal for high-performance switches and routers. Their customization allows for tailored functionality to meet the specific needs of network applications. However, the design and manufacturing of ASICs can be costly and time-consuming, resulting in reduced flexibility compared to general-purpose processors, as any changes to functionality typically require redesigning the hardware. Network processors are programmable chips designed to handle complex networking tasks, such as packet inspection, classification, and traffic management. Unlike ASICs, which are hardwired for specific functions, network processors provide flexibility, allowing for the implementation of various protocols and features through software. This programmability enables network operators to adapt to changing requirements and to perform deep packet inspection for enhanced security measures. While network processors can

provide significant versatility and support advanced functionalities, they generally operate at slower speeds compared to ASICs due to their more complex architectures. As such, they are often used in environments where flexibility and advanced processing capabilities are prioritized over raw performance. Software-Defined Networking (SDN) is an innovative network architecture that decouples the control plane from the data plane, allowing centralized control over network resources. In SDN, a central controller manages the network, directing traffic flows and implementing policies across multiple devices, which can be configured and monitored programmatically. This separation enables greater flexibility, as network administrators can quickly adapt to changing demands, implement dynamic traffic management, and automate network operations through software applications. SDN also facilitates more efficient resource utilization and simplifies network management, as changes can be made from a single point of control. However, this centralized approach can introduce challenges such as increased latency for control messages and potential risks associated with a single point of failure, necessitating robust security measures and high availability strategies.

In a centralized forwarding architecture within a single network device, the control plane and data plane functions are tightly integrated, with the control plane responsible for making all forwarding decisions. The device contains a central routing processor that manages the routing table, handles protocol updates, and calculates the best paths for incoming packets. When a packet arrives, the ingress line card sends it to the routing processor, which analyzes the packet's header and consults the routing table to determine the appropriate output port. This decision is then communicated back to the line card for forwarding. While this architecture simplifies processing and allows for a cohesive decision-making process, it may introduce latency, as packets must traverse the routing processor before forwarding. Additionally, if the routing processor becomes overwhelmed or fails, packet forwarding can be severely disrupted, highlighting the importance of robust design and redundancy within the device.

In a distributed forwarding architecture within a single network device, the control and data plane functions are more decentralized, allowing individual components to operate independently. Each ingress line card is equipped with its own local forwarding engine and routing information, enabling it to make forwarding decisions without having to rely on a central processor. When a packet arrives, the line card can immediately consult its local forwarding table to determine the appropriate output port, allowing for faster processing and reduced latency. This independence enhances performance, especially under high traffic loads, as each line card can handle packets concurrently without bottlenecking at a central control point. However, this decentralized approach can lead to challenges in maintaining consistent routing information across multiple line cards, as updates must be propagated and synchronized among them. Overall, distributed forwarding within a single device promotes efficiency and speed, making it well-suited for handling high volumes of traffic.

Ingress line cards are responsible for receiving incoming packets from the network. They perform initial processing, such as buffering and determining the destination of the packets. In a centralized forwarding architecture the ingress line cards may send packets to a centralized control plane for processing. After receiving forwarding decisions from the central controller, they then pass the packets to the switch fabric. For distributed forwarding architectures the ingress line cards can make forwarding decisions locally based on routing tables stored on the line cards themselves, reducing the need to communicate with a central controller.

Egress line cards handle outgoing packets, sending them to the appropriate destination on the network. For centralized forwarding, egress cards like ingress cards depend on the central controller's instructions for the proper routing to packets. In a distributed forwarding architecture would have the egress line cards making independent forwarding decisions based on local tables, helping to expedite the forwarding process.

The role of the switches fabric is to facilitate communication between ingress and egress line cards. It transfers packets from the input ports to the output ports. In a centralized model, the switch fabric is often controlled by the central processor, which directs traffic based on the global forwarding decisions. For distributed forwarding the fabric operates in a way that allows for fast, direct communication between line cards without needing to reference a centralized control point for each packet.

Route Processor Engines (RPEs) are responsible for maintaining routing tables and making routing decisions. It runs the control plane functions. In centralized architectures, the RPE is often part of a central controller that oversees the entire network. It processes incoming route information and disseminates updates to line cards. In a distributed

scenario, each device has its own RPE that independently processes routing protocols and updates its local routing table, enabling localized decision-making.

Forwarding engines roll is to use the routing information from the route processor to make realtime forwarding decisions for packets. With a centralized model the forwarding engine will rely on direction form the centralized route processor, making forwarding decisions based on the information received from it. Imagine a distributed model where the forwarding engine works independently, utilizing local routing tables to make fast, efficient forwarding decisions without needing to consult a central point.

In centralized forwarding architectures, the route processor and forwarding decisions are made at a central point, while ingress and egress line cards primarily handle the input and output of packets with the switch fabric facilitating the movement of data. In contrast, distributed forwarding allows for localized decision-making, where each component (line cards, RPE, forwarding engine) operates independently, enhancing speed and scalability. Both architectures utilize the same components but in different configurations and operational dynamics, with centralized architectures relying on a singular control point and distributed architectures spreading the decision-making processes across multiple devices.

Software Cisco Express Forwarding (CEF) is a packet-switching method that leverages the device's CPU to manage routing and forwarding tasks. In this architecture, when a packet arrives at a network device, the software CEF examines the packet headers and makes forwarding decisions based on the routing table stored in the device's memory. This process involves several steps: the incoming packet is received, the routing processor performs a lookup in the forwarding information base (FIB), and then the packet is sent out through the appropriate interface. While software CEF provides flexibility and supports advanced features like policy-based routing and deep packet inspection, it can introduce latency due to reliance on the CPU for forwarding decisions. This architecture is typically utilized in smaller devices or scenarios where complex processing is required, but it may not scale well under high traffic conditions due to potential CPU overload.

Hardware CEF (hCEF) is a more efficient forwarding mechanism that offloads the packet processing tasks from the CPU to specialized hardware components, such as application-specific integrated circuits (ASICs) or dedicated forwarding engines. In hardware CEF, the control plane still manages routing protocols and updates the FIB, but the actual packet forwarding is performed by dedicated hardware, enabling high-speed processing and reduced latency. When a packet arrives, the hardware component quickly performs lookups in the FIB and forwards packets with minimal CPU intervention. This architecture is highly scalable and can handle larger volumes of traffic with greater efficiency, making it suitable for high-performance network devices. The use of hardware CEF allows for features such as load balancing and fast switching, while maintaining lower latency and higher throughput compared to software CEF.

Switch Database Management (SDM) templates are configurations applied to Cisco switches that define how system resources are allocated for various features, such as routing, VLANs, and access control lists (ACLs). SDM templates help optimize the switch's performance based on the specific needs of the network. By configuring SDM templates, administrators can allocate resources to match the expected traffic patterns, whether the focus is on IP routing, VLAN management, or other functionalities. For instance, a template might prioritize IP unicast routing if the switch is primarily used for Layer 3 routing, or it might optimize for bridging if the primary function is Layer 2 switching. Administrators can select from various predefined templates or create custom configurations to tailor the device's performance. Properly managing SDM templates is crucial, as improper configurations can lead to resource shortages or degraded performance, particularly in high-traffic scenarios.

COMMANDS

Define a VLAN

- vlan *vlan-id*
- name *vlan-name*

Configure an interface as a trunk port

- switchport mode trunk

Configure an interface as an access port assigned to a specific VLAN

- switchport mode access
- switchport access vlan {*vlan-id* | name *name*}

Configure a static MAC address entry

- mac address-table static mac-address vlan *vlan-id* interface *interface-id*

Clear MAC address from the MAC address table

- clear mac address-table dynamic [{address *mac-address* | interface *interface-id* | vlan *vlan-id*}]

Assign an IPv4 address to an interface

- ip address *ip-address subnet-mask*

Assign a secondary IPv4 address to an interface

- ip address *ip-address subnet-mask* secondary

Assign an IPv6 address to an interface

- ipv6 address *ipv6-address/prefix-length*

Modify the SDM database

- sdm prefer {vlan | advanced}

Display the interface that are configured as a trunk port an all the VLANs that they permit

- show interfaces trunk

Display the list of VLANs and their associated ports

- show vlan [{brief | id *vlan-id* | name *vlan-name* | summary}]

Display the MAC address table for a switch

- show mac address-table [address *mac-address* | dynamic | vlan *vlan-id*]

Display the current interface state, including duplex, speed, and link state

- show interfaces status

Display the Layer 2 configuration information for a specific switch port

- show interfaces *interface-id* switchport

Display the ARP table

- show ip arp [*mac-address* | *ip-address* | vlan *vlan-id* | *interface-id*]

Display the IP interface table

- show ip interface [brief | *interface-id* | vlan *vlan-id*]

Display the IPv6 interface table

- show ipv6 interface [brief | *interface-id* | vlan *vlan-id*]

BPDU (Bridge Protocol Data Unit): A type of network message exchanged between switches (bridges) in a network to maintain the Spanning Tree Protocol (STP). BPDUs help switches to discover and maintain the network topology, determine the root bridge, and calculate the shortest paths.

Configuration BPDU: A type of BPDU used by switches to exchange information about the network topology, including the root bridge identifier, path cost to the root bridge, and port roles. Configuration BPDUs are used during STP calculations to build and maintain the spanning tree.

Designated Port: In STP, a port on a switch that is selected to be the forwarding port for a particular network segment. It is the port that has the best path to the root bridge for that segment. Each segment has one designated port.

Forward Delay: A timer used in STP that defines how long a port stays in the Listening and Learning states before transitioning to the Forwarding state. This delay helps to ensure that all network topology changes are properly processed before the port starts forwarding traffic.

Hello Time: The interval at which BPDUs are sent out by the root bridge to maintain the STP topology. This timer helps in detecting changes in the network and ensures that all switches are synchronized with the current topology.

Local Bridge Identifier: A unique identifier assigned to each switch (bridge) in an STP network, usually consisting of the switch's Bridge Priority and MAC address. It helps to uniquely identify each switch in the network for STP operations.

Max Age: A timer that defines how long a switch should keep a BPDU before discarding it if no newer BPDU is received. This timer helps to ensure that outdated information about the network topology is eventually removed.

Root Bridge: The central switch in an STP network that serves as the reference point for all path calculations. The root bridge is selected based on the lowest Bridge ID, which is a combination of the bridge's priority and MAC address.

Root Bridge Identifier: A unique identifier for the root bridge in an STP network, consisting of the bridge's priority and MAC address. This identifier is used to distinguish the root bridge from other switches in the network.

Root Path Cost: The cumulative cost of the path from a switch to the root bridge. It helps in determining the best path to the root bridge. Each switch calculates this cost based on the cost of each link in the path.

Root Port: The port on a switch that has the lowest path cost to the root bridge. It is the port used to forward traffic towards the root bridge. Each non-root switch has one root port.

System Priority: A value used in combination with the MAC address to form the Bridge ID. The system priority helps in determining the root bridge and other switch roles in the STP process.

System ID Extension: A part of the Bridge ID that extends the MAC address space to uniquely identify switches. This extension allows for a larger number of unique bridge identifiers, especially in networks with many switches.

Topology Change Notification (TCN): A message sent by a switch to notify all other switches in the STP network of a topology change. TCNs trigger the recalculation of the spanning tree to adapt to the changes in the network topology.

COMMANDS

Set the STP max age

- spanning-tree vlan *vlan-id* max-age

Set the STP hello interval

- spanning-tree vlan *vlan-id* hello-time *hello-time*

Set the STP forwarding delay

- spanning-tree vlan *vlan-id* forward-time *forward-time*

Display the STP root bridge cost

- show spanning-tree root

Display the STP information (root bridge, local bridge, and interfaces) for one or more VLANs

- show spanning-tree [vlan *vlan-id*]

Identify when the last TCN occurred and which port was the reason for it

- show spanning-tree [vlan *vlan-id*] detail

802.1D Port States

- Disabled: The port is in an administratively off position (that is, shut down).
- Blocking: The switch port is enabled, but the port is not forwarding any traffic to ensure that a loop is not created. The switch does not modify the MAC address table. It can only receive BPDUs from other switches.
- Listening: The switch port has tgransitioned from a blocking state and can now send or receive BPDUs. It cannot forward any other network traffic. The duration of the state correlates to the STP forwarding time. The next port state is learning.
- Learning: The switch port can now modify the MAC address table with any network traffic that it receives. The switch still does not forward any other network traffic besides BPDUs.; The duration of the state correlates to the STP forwarding time. The next port state is forwarding.
- Forwarding: The switch port can forward all network traffic an can update the MAC address table as expected. This is the final state for a switch port to forward network traffic.
- Broken: The switch has detected a configuration or an operational problem on a port that can have major effects. The port discards packets as long as the problem continues to exist.

802.1D Port Roles

- Root port: A network port that connects to the root bridge or an upstream switch in the spanning-tree topology. There should be only one root port per VLAN on a switch.
- Designated port: A network port that receives and forwards BPDU frames to other swithces. Designated ports provide connectivity to downstream devices and switches. There should be only on active designated port on a link.
- Blocking port: A network port that is not forwarding traffic becuase of STP calulations.

These port types are expected on Catalyst switches:

- P2P: This port type connects with another network device (PC or RSTP switch).
- P2P edge: This port type specifies that portfast is enabled on this port.

The interface STP cost is an essential compnent for root path calulation because the root path is found based on the cumulative interface STP cost to reach the root bridge. The interface STP cost was originally stored as a 16-bit value with a reference value of 20 Gbps. As switches have developed with higher-speed interfaces, 10 Gbps might not be enough. Another method, called long mode, uses a 32-but value and uses a reference speed of 20 Tbps. The original method, known as short mode, has been the default for most switches, but has been transitioning to long mode based on specific platform and OS versions. Devices can be configured with the long-mode interface cost with thecommand spanning-tree pathcost method long. The entire Layer 2 topology should use the same setting for every device in the environment to ensure a consistent topology. Before you enable this setting in an environment, it is important to conduct an audit to ensure that the setting will work.

The root bridge election process is a critical component of the Spanning Tree Protocol (STP), which is used to prevent loops in network topologies with multiple switches. Here's a detailed explanation of how the root bridge election process works:

1. Bridge IDs: Each switch in an STP topology has a unique identifier called the Bridge ID, which consists of two parts: the Bridge Priority (a configurable value) and the MAC address of the switch. The default Bridge Priority is typically set to 32768.

2. Initiation of Election: When STP begins, all switches consider themselves potential root bridges. Each switch starts by advertising its Bridge ID to its neighbors through Bridge Protocol Data Units (BPDUs).

3. BPDU Tansmission: Each switch sends out BPDUs containing its own Bridge ID to all of its neighboring switches. This process occurs periodically, allowing switches to communicate and evaluate their IDs.

4. Comparing Bridge IDs: Upon receiving BPDUs from other switches, each switch compares the Bridge IDs. The switch with the lowest Bridge ID is elected as the root bridge. The comparison process involves:

    - First comparing the Bridge Priority values. The switch with the lowest priority value is favored.
    - If the priorities are equal, the switch with the lowest MAC address wins.

5. Consensus on Root Bridge: As BPDUs are exchanged, all switches will eventually agree on a single root bridge, as they all will learn and propagate the same BPDU with the lowest Bridge ID.

6. Convergence: Once the root bridge is elected, the other switches determine their roles (designated ports and blocked ports) in the network topology based on their distance from the root bridge, which is measured in terms of path cost. The switches will then update their forwarding tables to reflect the new topology.

7. Stable State: The network converges to a stable state with one root bridge and a loop-free topology. This state is maintained until there is a change in the network, such as a switch failure or configuration change, prompting a reelection process.

Locating Root Ports

After the switches have identified the root bridge, they must determine their root port (RP). The root bridge contunues to advertise configuration BPDUs out all of its ports. The switch compares the BPDU information received on its ports to identify the RP. The RP is selected using the following logic (where the next criterion is used in the event of a tie):

1. The interface associated to lowest path cost is more preferred.

2. The interface associated to the lowest system priority of the advertising switch is preferred next.

3. The interface associated to the lowest system MAC address of the advertising switch is preferred next.

4. When multiple links are associated to the same switch, the lowest port priority from the advertising switch is preferred.

5. When multiple links are associated to the same switch, the lower port number from the advertising switch is preferred.

Locating Blocked and Designated Switch Ports

Now that the root bridge and RPs have been identified, all other ports are considered designated ports. However, if two non-root switches are connected to each other on their designated ports, one of those switch ports must be set to a block state to prevent a forwarding loop.

1. The interface is a designated port and must not be considered an RP.

2. The switch with the lower path cost to the root bridge forwards packets, and the one with the higher path cost blocks. If they tie, they move on to the next step.

3. The system priority of the local switch is compared to the system priority of the remote switch. The local port is moved to a blocking state if the remote system priority is lower than that of the local swtich. If they tie, they move on to the next step.

4. The system MAC address of the local switch is compared to the system MAC address of the remote switch. The local designated port is moved to a blocking state if the remote system MAC address is slower than that of the local switch.

Port Fast is a feature in networking, specifically in the context of Spanning Tree Protocol (STP) on switches. When enabled on a switch port, Port Fast allows the port to skip the usual STP states (Listening and Learning) and immediately enter the Forwarding state. This is particularly useful for ports connected to end devices like computers or printers, where you want quick connectivity without the delays caused by STP convergence. By using Port Fast, you can significantly reduce the time it takes for a port to become operational after being enabled, improving network performance and responsiveness. However, it's important to use this feature carefully, as it can lead to network loops if misconfigured, especially when connecting switches or hubs.

In a stable Layer 2 topology, configuration BPDUs always flow from the root bridge toward the edge switches. However, changes in the topology have an impact on all the switch in the Layer 2 topology. The switch that detects a link status change sends a TCN BPDU towards the root bridge, out its RP. If an upstream switch receives the TCN, it sends out an acknowledgment and forwards the TCN out its RP to the root bridge. Upon receipt of the TCN, the root bridge creates a new configuration BPDU with the Topology Change flat set, and it is then flooded to all the switches. When a switch receives a configuration BPDU with the Topology Change flag set, all switches change their MAC address timer to the forwarding delay timer (witha default of 15s). This flushes out MAC address for devices that have not communicated in that 15-second window but maintains MAC addresses for devices that are actively communicating. Flushing the MAC address table prevents a switch from sending traffic to a host that is no longer reachable by that port. However, a side effect of flushing the MAC address table is that it temporarily increases the unknown unicast flooding while it is rebuilt. Remember thath this can impact hosts because of their CSMA/CD behavior. The MAC address timer is then reset to normal (300s by default) after the second configuration BPDU is received.

TCNs are generated on a VLAN basis, so the impact of TCNs directly correlates to the number of hosts in a VLAN. As the number of hosts increases, the more likely TCN generation is to occur and the more hosts that are impacted by the broadcasts. Topology changes should be checked as part of the troubleshooting process. Ports configured with Port Fast do not generate TCN BPDUs when they transition from blocking to forwarding. This means that if a device connected to a Port Fast-enabled port goes online or offline, the switch does not notify the rest of the network about this change through TCN BPDUs. Since Port Fast is typically used on ports connected to end devices (not switches), the likelihood of network loops is minimized. Therefore, not generating TCNs on these ports is generally safe. For ports that are not configured with Port Fast (typically inter-switch links), any changes in their state (like a link going down) will generate TCN BPDUs to inform the rest of the network about the topology change. In essence, enabling Port Fast prevents TCN BPDUs from being generated for state changes on those specific ports. This design choice helps maintain a faster connection for end devices while reducing unnecessary notifications about changes that are unlikely to affect the overall topology of the network. Topology changes are seen with the command show spanning-tree [vlan vlan-id] detail on a switch bridge. The output of this command shows the topology change count and time since the last change has occurred. A sudden or continuous increase in TCNs indicates a potential problem and should be investigated further for flapping ports or events on a connected switch. Flapping ports refer to network switch ports that frequently alternate between the up (active) and down (inactive) states. This behavior can cause disruptions in network performance and stability. Flapping can occur for several reasons, including: cable issues, hardware problems, misconfigurations, environment factors, overloaded network. The effects of flapping ports are: network instability, increased CPU utilization, STP convergence issues. It is important to check cables and connections, inspect configurations including duplex and speed, monitor logs for patterns, and stabilize the environment.

When a switch loses power or reboots, or when a cable is removed from a port, the Layer 1 signaling places the port into a down state, which can notify other processes, such as STP. STP considers such an event a direct link failure and can react in one of three ways, depending on the topology.

Configuration BPDU Format:

- Protocol ID (2 bytes): This is set to 0x0000 for STP.
- Version (1 byte): Indicates the version of the Spanning Tree Protocol (0 for STP).

- BPDU Type (1 byte): Set to 0 for Configuration BPDUs.
- Flags (1 byte): Contains various flags, such as the Root Port flag, which indicates whether the bridge is a root bridge or not.
- Root Bridge ID (8 bytes): The Bridge ID of the root bridge (MAC address + priority).
- Root Path Cost (4 bytes): The cost to reach the root bridge from this bridge.
- Bridge ID (8 bytes): The Bridge ID of the bridge sending the BPDU.
- Port ID (2 bytes): The ID of the port that sent the BPDU.
- Message Age (1 byte): The age of the BPDU in seconds.
- Maximum Age (1 byte): The maximum age for which the BPDU is considered valid.
- Hello Time (1 byte): The interval at which BPDUs are sent.
- Forward Delay (1 byte): The time the bridge should wait before changing states.

Topology Change Notification (TCN) BPDU Format:

- Protocol ID (2 bytes): Set to 0x0000 for STP.
- Version (1 byte): Indicates the version of the Spanning Tree Protocol (0 for STP).
- BPDU Type (1 byte): Set to 1 for TCN BPDUs.
- Flags (1 byte): Typically not used for TCN BPDUs.
- Root Bridge ID (8 bytes): The Bridge ID of the root bridge.
- Root Path Cost (4 bytes): The cost to reach the root bridge from this bridge.
- Bridge ID (8 bytes): The Bridge ID of the bridge sending the BPDU.
- Port ID (2 bytes): The ID of the port that sent the BPDU.

An example:

- s1 (root switch)
  - g1/0/2 to s2
  - g1/0/3 to s3
- s2
  - g1/0/1 to s1 (root port)
  - g1/0/3 to s3 (designated port)
- s3
  - g1/0/1 to s1 (root port)
  - g1/0/2 to s2 (blocked)

Three scenarios:

1. the link between s2 and s3 fails...how does stp react? since the link is already in a blocking state, there's no need for a TCN, and STP doesn't initiate any topology changes. Both s2 and s3 will advertise a TCN towards the root, which results in the Layer 2 topology flushing its MAC address table. Flushing the MAC table is a proactive measure to ensure that it does not use potentially outdated or invalid MAC address entries, especially since the topology may have changed. Without flushing, S1 might still have entries for MAC addresses learned from devices connected to S2 and S3. Since the topology has changed, these entries could lead to misrouting or dropped frames if traffic continues to flow based on outdated information. After the flush, as devices start communicating again, S1 can relearn MAC addresses based on the current active paths and connections. This is particularly useful in dynamic environments where devices may come and go, or ports may change states. The blocked port remains blocked, and there's no transition time needed because no ports change state. The network remains stable, and both s1 and s3 continue through their respective connections with no interruption.

2. the link between s1 and s3 fails...how does stp react? First s1 detects a link failure on its g1/0/3 interface. s3 detects a link failure on its g1/0/1 interface. Then normally, s1 would generate a TCN flag out its root port, but it is the root bridge, so it does not. s1 would advertise a TCN if it were not the root bridge. s3 removes it best BPDU received from s1 on its g1/0/1 interface because it is now in a down state. At this point, s3 would attempt to send a TCN toward the root switch to notify it of a topology change; however, its root port is down. Then s1 advertises a configuration BPDU with the Topology Change flag out of all its ports. This BPDU is received and relayed to all switches in the environment. NOTE If other switches were connected to s1, they would receive a configuration BPDU with the Topolgy Change flag set also. These packets have an impact for all switches in the same Layer 2 domain. Next s2 and s3 receive the configuration BPDU with the Topology Change flag. These switches then reduce the MAC entries. In this phase, s2 does not know what changed in the topology. Finally, there is no need to wait for the Max Age timer (default value of 20s) to age out with a link failure. s3 restarts the STP listening and learning states to learn about the root bridge on the g1/0/2 interface (which was in the blocking state previously). The total convergence time for s3 is 30s: 15s for the listening state and 15s for the learning state before s3s g1/0/2 can be made the RP.

3. the link between s1 and s2 fails...how does stp react? First s1 detects a link failure on its g1/0/2 interface. s2 detects a link failure on its g1/0/1 interface. Normally s1 would generate a TCN flag out its root port, but it is the root bridge, so it does not. s1 would advertise a TCN if it were not the root bridge. s2 removes it best BPDU received from s1 on its g1/0/1 interface because it is now in a down state. At this point, s2 would attempt to send a TCN toward the root switch to notify it of a topology change; however, its root port is down. Next s1 advertises a configuration BPDU with the Topology Change flag out of all its ports. This BPDU is then received and relayed to s3. s3 cannot relay this to s2 because it g1/0/2 port is still in a blocking state. s2 assumes that it is now the root bridge. s3 receives the configuration BPDU with the Topology Change flag from s1. s3 reduces the MAC address age timer to the forwarding delay timer to flush out older MAC entries. s3 receives s2s inferior BPDUs and discards them as it is still receiving superior BPDUs from s1. Next the Max Age timer on s3 expires, and now s3s g1/02 port transitions from blocking to listening state. s3 can now forward the next configuration BPDU it receives from s1 to s2. Finally, s2 receives s1s configuration BPDU via s3 and recognizes it as superior. It marks its g1/0/3 interface as the root port and transitions it to the listening state.The total convergence time for s2 is 50s: 20s for the Max Age timer on s3, 15s for the listening state on s2, and 15s for the learning state.

802.1W Port States:

- Discarding: The switch port is enabled, but the port is not forwarding any traffic to ensure that a loop is not created. This state combines the traditional STP states disabled, blocking, and listening
- Learning: The switch port modifies the MAC address table with any network traffic it receives. The switch still does not forward any other network traffic besides BPDUs.
- Forwarding: The switch port forwards all network traffic and updates the MAC address table as expected. This is the final state for a switch port to forward network traffic.

802.1W Port Roles:

- RP: A network port that connects to the root switch or an upstream switch in the spanning-tree topology. There should be only one root port per VLAN on a switch.

- DP: A network port that receives and forwards frames to other switches. DPs provide connectivity to downstream devices and switches. There should be only one active DP port on a link.
- Alternate port: A network port that provides alternate connectivity toward the root switch through a different switch. This is a port that provides an alternate path to the root bridge. It is a backup for a root port, which is the port that has the lowest cost to the root bridge. If the root port goes down, the alternate port can take over, helping to maintain connectivity.
- Backup port: A network port that provides link redundancy toward the shared segment within the same collision domain, which is typically a network hub.This is a port that provides a backup connection to a designated port. The designated port is the port that has the lowest cost to the segment and is responsible for forwarding traffic to and from that segment. If the designated port fails, the backup port can take over for that segment.

802.1W Port Types:

- Edge port: A port at the edge of the network where hosts connect to the Layer 2 topology with one interface and cannot form a loop. These ports directly correlate to ports that have the STP portfast feature enabled.
- Non-Edge port: A port that has received a BPDU.
- Point-to-point port: Any port that connects to another RSTP switch with full duplex. Full-duplex links do not permit more than two devices on a network segment, so determining whether a link is full duplex is the fastest way to check the feasibility of bein connected to a switch.

NOTE: Multi-access Layer 2 devices such as hubs can connect only at half duplex. If a port can connect only via half duplex, it must operate under traditional 802.1D forwarding states.

With RSTP, swtiches exchange handshakes with other RSTP switches to transition through the following STP states faster. When two switches first connect, they establish a bidriectional handshake across the shared link to identify the root bridge. This is straightforward for an environment with only two switches; however, large environments require greater care to avoid creating a forwarding loop. RSTP uses a synchronization process to add a switch to the RSTP topology without introducing a forwarding loop. The synchronization process starts when two switches are first connected. The process proceeds as follows:

1 As the first two switches connect to each other, they verify that they are connected with a point-to-point link by checking the full-duplex status.

2 They establish a handshake with each other to advertise a proposal (in configuration BPDUs) that their interface should be the DP for the segment.

3 There can be only one DP per segment, so each switch identifies whether it is the superior or inferior switch, using the same logic as in 802.1D for the system identifier (that is, the lowest priority and then the lowest MAC address). Using the MAC address from sw (0062.ec9d.c500) is the superior switch to s2 (0081.c4ff.8b00).

4 The inferior switch recognized that it is inferior and marks its local port as the RP. At that same time, it moves all non-edge ports to a discarding state. At this point in time, the switch has stopped all local switching for non-edge ports.

5 The inferior switch sends an agreement (configuration BPDU) to the root bridge, which signifies to the root bridge that synchronization is occurring on that switch.

6 The inferior switch moves its RP to a forwarding stat. The superior switch moves its DP to a forwarding state too.

7 The inferior switch repeats the process for any downstream switches connected to it.

The RSTP convergence process can occur quickly. RSTP ages out the port inforamtion after it has not received hellows in three consecutive cycles. Using default timers, the Max Age would take 20s, but RSTP requires only 6s. And thanks to the new synchronization, ports can transition from discarding to forwarding in an extremely low amount

of time.

If a downstream switch fails to acknowledge the proposal, the RSTP switch must default to 802.1D behaviors to prevent a forwarding loops.

- BPDU filter: a feature that prevents a switch port from sending or receiving Bridge Protocol Data Units (BPDUs). When enabled, the port operates in a mode where it ignores all BPDUs, effectively preventing the port from participating in STP operations. Typically used on edge ports where devices connected do not participate in STP (like end-user devices) to avoid unnecessary processing and to keep the port in a forwarding state.
- BPDU guard: a protective feature that places a port in an error-disabled state if it receives a BPDU on a port that is configured to be an edge port (a port where no BPDUs should be received). This is useful to prevent potential loops caused by mistakenly connecting switches to edge ports. If a BPDU is received, the port is disabled to protect the network.
- root guard: a feature that prevents a designated port from becoming a root port for a bridge that is not supposed to be the root bridge. If a BPDU from a superior bridge (indicating a new root) is received, the port is placed in a blocked state. It ensures that a specific switch maintains its status as the root bridge, helping to preserve the intended topology and prevent unexpected topology changes.
- STP portfast: a feature that allows a switch port to transition directly to the forwarding state from the blocking state when a device is connected. This bypasses the usual STP listening and learning states. Typically used on edge ports where end devices connect, as it speeds up the connection process and reduces the delay in getting devices online.
- STP loop guard: a feature that prevents a switch port from entering a forwarding state if it stops receiving BPDUs. It helps to protect against loops that can occur if a segment goes down and the port mistakenly assumes it can forward traffic. It is used in scenarios where unidirectional links might cause a switch to believe it is in a valid topology, thus preventing loops from forming.
- Unidirectional Link Detection (UDLD): a protocol used to detect unidirectional links between switches. It actively monitors the link status by sending messages between the two ends of a link to ensure that traffic can flow in both directions. If a unidirectional link is detected, UDLD can automatically place the affected port into an error-disabled state, preventing loops and ensuring network stability.

COMMANDS

Configure the STP priority for a switch so that it is a root bridge or a backup root bridge

- spanning-tree vlan *vlan-id* root {primary — secondary} [diameter *diameter*]
- spanning-tree [vlan *vlan-id*] priority *priority*

Configure the STP port cost

- spanning tree [vlan *vlan-id*] cost *cost*

Configure the STP port priority on the downstream port

- spanning tree [vlan *vlan-id*] port-priority *priority*

Enable root guard on an interface

- spanning-tree guard root

Enable STP portfast globally, for a specific port, or for a trunk port

- spanning-tree portfast default
- spanning-tree portfast

- spanning-tree portfast trunk

Enable BPDU guard globally or for a specific switch port

- spanning-tree portfast bpduguard default
- spanning-tree bpduguard {enable — disable}

Enable BPDU filter globally or for a specific interface

- spanning-tree portfast bpdufilter default
- spanning-tree bpdufilter enable

Enable STP loop guard globally or for a specific interface

- spanning-tree loopguard default
- spanning-tree guard loop

Enable automatic error recovery time

- errdisable recovery cause bpduguard

Change the automatic error recovery time

- errdisable recovery interval *time-seconds*

Enable UDLD globally or for a specific port

- udld enable [aggressive]
- udld port [aggressive]

Display the list of STP ports in an inconsistent state

- show spanning-tree inconsistentports

Display the list of neighbor devices running UDLD

- show udld neighbors

The placement of the root bridge is an important decision and often should be chsen to minimize the number of hops to th efurthest switch in the topology. The design should consider where redundant connections exist, connections that will be blocked, and the ability (performance) for the root switch to handle cross-switch traffic. Generally, root switches are at L2/L3 boundaries. To prevent preemptive attacks you should set the root bridge to priority of 0 and the backup at 4096...in addition to using root guard.

Using the diameter keyword when setting a new root bridge in the Spanning Tree Protocol (STP) can serve specific purposes, particularly in larger or more complex network topologies. Here's why you might consider using it:

1 control over path selection:

   - By specifying the network diameter, you can influence how STP calculates paths. If your network has specific topological requirements (e.g., limited hop counts), setting a diameter can help ensure that STP considers only appropriate paths.

2 avoiding loops:

   - By specifying the network diameter, you can influence how STP calculates paths. If your network has specific topological requirements (e.g., limited hop counts), setting a diameter can help ensure that STP considers only appropriate paths.

3 performance optimization:

- Setting a diameter can help optimize performance by limiting the complexity of the STP calculations. This can lead to faster convergence times, as STP will not need to analyze paths that exceed the specified hop count.

4 supporting larger topologies:

- In extensive networks, defining the diameter ensures that STP can handle the structure without running into issues caused by excessive hops. This can be particularly useful in environments with multiple VLANs and interconnected switches.

5 facilitating redundancy:

- By controlling the diameter, you can create redundancy in your network while ensuring that traffic paths remain efficient and manageable. This helps balance load and maintain reliability.

By changing the STP port costs, you can modify the STP forwarding path. You can lower a path that is currently an alternate port while making it designated, or you can raise the cost on a port that is designated to turn it into a blocking port. Note that the spanning-tree command modifies the cost for all VLANs unless the optional vlan keyword is used to specify a VLAN.

Port priority is a feature used in the Spanning Tree Protocol (STP) to influence the selection of which ports on a switch will be used to forward traffic. It helps determine the roles of ports in the network topology, especially in redundant network designs. Port priority helps determine which ports should be preferred for forwarding traffic versus those that should be in a blocking state. This is crucial in preventing loops in a redundant network. Each port on a switch can have a default priority value (typically 128). The priority can be adjusted to give preference to certain ports when establishing the spanning tree topology.When determining the best path to the root bridge, STP uses a combination of port priority and port costs (which are based on the speed of the link) to decide which ports to put in forwarding state and which to block. Port priority values typically range from 0 to 255. A lower value indicates a higher priority. For example, a port with a priority of 100 would be preferred over a port with a priority of 200. For switches that are not the root bridge, STP selects one port as the root port (the port with the best path to the root bridge) based on the lowest cost. If multiple ports have the same cost, port priority is used to determine which port to select. The designated port is the port on a network segment that has the lowest cost to the root bridge. If there are ties in cost, port priority is compared. Switch A has two ports connected to a network. If Port 1 has a priority of 128 and Port 2 has a priority of 64, STP will prefer Port 2 for forwarding traffic because it has a lower priority value. Port priority is a crucial component of STP that helps ensure efficient and loop-free network operation. By allowing administrators to adjust port priorities, it provides a way to influence the traffic paths in redundant network configurations.

The following scenarios are common for Layer 2 forwarding loops:

- STP disabled on a switch
- A misconfigured load balancer that transmit traffic out multiple ports with the same MAC address
- A misconfigured virtual switch that bridges two physical ports (Virtual switches typically do not participate in STP.)
- End users using a dumb network switch or hub

Preemption can be potentially dangerous in networks with spurious root switches due to several reasons:

- Flapping Root Bridge Role: If a switch constantly comes online and offline, it may cause the root bridge role to flip between different switches frequently. This ”flapping” can lead to network instability as the topology is constantly changing.
- Increased Convergence Time: When preemption occurs frequently, the network has to go through the convergence process repeatedly. Each convergence event involves re-evaluating paths and states (blocking, listening, learning, and forwarding) for all switches, which can cause temporary outages and increased latency.

- Broadcast Storms: Frequent changes in the root bridge can lead to broadcast storms, especially if multiple switches are trying to advertise their presence as the root. This can overwhelm the network with traffic, leading to degraded performance.
- Configuration Complexity: In a complex network, having multiple switches configured for preemption can make it difficult to predict which switch will be the root bridge at any given time. This unpredictability can complicate troubleshooting and network management.
- Potential Loops: If a switch that is not intended to be a stable root bridge preempts, it could cause loops in the topology if other switches don't quickly recognize the change and adjust their forwarding states accordingly.
- Inconsistent Policy Enforcement: Network policies that rely on a stable root bridge (such as Quality of Service configurations) can be disrupted, leading to inconsistent behavior across the network.

To mitigate these risks, network administrators should carefully plan and monitor their network topology, consider using techniques like Rapid Spanning Tree Protocol (RSTP) for faster convergence, and ensure that only stable and well-placed switches are configured for preemption.

Fiber-optic cables consist of strands of glass/plastic that transmit light. A cable typically consists of one strand for sending data and another strand for receiving data on one side; the order is directly opposite at the remote site. Network devices that use fiber for connectivity can encounter unidirectional traffic flows if one strand is broken. In such scenarios, the interface still shows a line-protocol up state; however, BPDUs are not able to be transmitted, and the downstream switch eventually times out the existing root pot and identifies a different port as the root port. Traffic is then received on the new root port and forwardrd out the strand that is still working, thereby creating a forwarding loop.

Unidirectional Link Detection (UDLD) allows for the bidirectional monitoring of fiber-optic cables. UDLD operates by transmitting UDLD packets to a neighbor device that includes the system ID and port ID of the interface transmitting the UDLD packet. The receiving device then repeats that information, including its system ID and port ID, back to the originating device. The process continues indefinitely. UDLD operates in two different modes:

- Normal: In normal mode, if a frame is not acknowledged, the link is considered undetermined and the port remains active.
- Aggressive: In aggressive mode, when a frame is not acknowledged, the switch sends another eight packets in 1-second intervals. If those packets are not acknowledged, the port is placed into an error state.
- BPDU filter: a feature that prevents a switch port from sending or receiving Bridge Protocol Data Units (BPDUs). When enabled, the port operates in a mode where it ignores all BPDUs, effectively preventing the port from participating in STP operations. Typically used on edge ports where devices connected do not participate in STP (like end-user devices) to avoid unnecessary processing and to keep the port in a forwarding state.
- BPDU guard: a protective feature that places a port in an error-disabled state if it receives a BPDU on a port that is configured to be an edge port (a port where no BPDUs should be received). This is useful to prevent potential loops caused by mistakenly connecting switches to edge ports. If a BPDU is received, the port is disabled to protect the network.
- root guard: a feature that prevents a designated port from becoming a root port for a bridge that is not supposed to be the root bridge. If a BPDU from a superior bridge (indicating a new root) is received, the port is placed in a blocked state. It ensures that a specific switch maintains its status as the root bridge, helping to preserve the intended topology and prevent unexpected topology changes.
- STP portfast: a feature that allows a switch port to transition directly to the forwarding state from the blocking state when a device is connected. This bypasses the usual STP listening and learning states. Typically used on edge ports where end devices connect, as it speeds up the connection process and reduces the delay in getting devices online.

- STP loop guard: a feature that prevents a switch port from entering a forwarding state if it stops receiving BPDUs. It helps to protect against loops that can occur if a segment goes down and the port mistakenly assumes it can forward traffic. It is used in scenarios where unidirectional links might cause a switch to believe it is in a valid topology, thus preventing loops from forming.
- Unidirectional Link Detection (UDLD): a protocol used to detect unidirectional links between switches. It actively monitors the link status by sending messages between the two ends of a link to ensure that traffic can flow in both directions. If a unidirectional link is detected, UDLD can automatically place the affected port into an error-disabled state, preventing loops and ensuring network stability.

COMMANDS

Configure the switch for a basic MST refion that includes all VLANS and the version number 1

- spanning-tree mode mst
- spanning-tree mst configuration
- instance 0 vlan 1-4094
- revision 1

Modify a switch's MSTI priority or make it the root bridge for the MSTI

- spanning-tree mst *instance-number* priority *priority*
- spanning-tree mst *instance-number* root {primary | secondary}[diameter *diameter*]

Specify additional VLANs to an MSTI

- spanning-tree mst configuration
- instance *instance-number* vlan *vlan-id*

Change the MST version number

- spanning-tree mst configuration
- revision *version*

Change the port cost for a spcific MSTI

- spanning-tree mst *instance-number* cost *cost*

Change the port priority for a spcific MSTI

- spanning-tree mst *instance-number* port-priority *priority*

Display the MST configuration

- show spanning-tree mst configuration

Verify the MST switch status

- show spanning-tree mst [*instance-number*]

View the STP topology for the MST

- show spanning-tree mst interface *interface-id*

Common Spanning Tree (CST): a single spanning tree that is used to represent the entire network topology in a multi-instance environment, such as in Multiple Spanning Tree Protocol (MSTP). The CST is used to prevent loops and ensure that all VLANs can communicate efficiently over a single tree structure.

internal spanning tree (IST): a spanning tree that encompasses all VLANs in a single instance. It is used in MSTP to define the path for traffic across the network. The IST is responsible for maintaining the loop-free topology within the entire MST region.

MST instance (MSTI): a specific spanning tree configuration for a particular set of VLANs within an MST region. Each MSTI can have its own unique topology and can be used to optimize traffic paths based on the VLAN traffic patterns. MSTP allows for multiple MSTIs to be defined within the same MST region.

MST region: a collection of switches that share the same MST configuration and participate in the same MSTP. All switches within an MST region must have the same MST configuration parameters (such as region name, revision number, and mappings of VLANs to MSTIs). This ensures consistent spanning tree behavior across the switches in the region.

MST region boundary: the point at which one MST region ends and another begins. It is defined by switches that do not share the same MST configuration. These boundary switches may use different spanning tree protocols (e.g., RSTP, CST) and must interact carefully to avoid loops and ensure proper communication between regions.

PVST simulation check: a process used to verify that a network's configuration adheres to the principles of Per-VLAN Spanning Tree (PVST) operation. It ensures that the network topology can accommodate the requirements of PVST, including the management of multiple spanning trees for different VLANs. The simulation check often involves analyzing the switch configurations, port roles, and state to ensure that they are correctly set up to prevent loops and optimize traffic flow.

COMMANDS

Configure the VTP version

- vtp version {1 | 2 | 3}

Configure the VTP domain name

- vtp domain *domain-name*

Configure the VTP mode for a switch

- vtp mode {server | client | transparent | off}(required for the VTP v3 server)
- vtp primary

Configure a switch port to actively attempt to establish a trunk link

- switchport mode dynamic desirable

Configure a switch port to respond to remote attempt to establish a trunk link

- switchport mode dynamic auto

Configure the member ports for a static EtherChannel

- channel-group *etherchannel-id* mode on

Configure the member ports for an LACP EtherChannel

- channel-group *etherchannel-id* mode {active | passive}

Configure the memeber ports for a PAgP

- channel-group *etherchannel-id* mode {auto | desirable} [non-silent]

Configure the LACP packet rate

- lacp rate {fast | slow}

Configure the minimum number of member links for the LACP EtherChannel to become active

- port-channel min-links *min-links*

Configure the maximum number of member links in an LACP EtherChannel

- lacp max-bundle *max-links*

Configure a switch's LACP system priority

- lacp system-priority *priority*

Configure a switch's LACP port priority

- lacp port-priority *priority*

Configure the EtherChannel load-balancing algorithm

- port-channel load-balance *hash*

Display the VTP system settings

- show vtp status

Display the switch port DTP settings, native VLANs, and allowed VLANs

- show interface [*interface-id*] trunk

Display a brief summary update on EtherChannel interfaces

- show etherchannel summary

Display detailed information for the local EtherChannel interfaces

- show interface port-channel

Display information about LACP neighbors

- show lacp neighbor [detail]

Display the local LACP system identifier and priority

- show lacp *system-id*

Display the LACP counters for congfigure interfaces

- show lacp counters

Display information about PAgP neighbors

- show pagp neighbor

Display the PAgP counters for configured interfaces

- show pagp counters

Display the algorithm for load balancing network traffic based on the traffic type

- show etherchannel load-balance

Dynamic Trunking Protocol (DTP): a Cisco proprietary protocol used to negotiate the establishment of trunk links between switches. DTP allows switches to automatically determine whether a link should be a trunk or an access link based on the configuration of both ends. It simplifies the configuration process by enabling or disabling trunking dynamically based on the devices connected to the port.

EtherChannel bundle: a logical link that aggregates multiple physical Ethernet links into a single logical interface. This configuration provides increased bandwidth and redundancy. EtherChannel allows for load balancing across the physical links and can help improve network performance by combining the capacity of the links.

LACP interface priority: the priority value assigned to individual interfaces in a Link Aggregation Control Protocol (LACP) configuration. This priority helps determine which links are included in the EtherChannel bundle if the total number of links exceeds the configured limit. The link with the highest priority (lowest numerical value) is preferred for inclusion in the bundle.

LACP system priority: a value assigned to the entire system (switch) that participates in LACP. It helps in resolving conflicts when multiple switches attempt to form an EtherChannel. The switch with the highest system priority (lowest numerical value) will be preferred as the aggregator for the EtherChannel, which is particularly important in cases of redundancy or multiple connections.

load-balancing hash: an algorithm used to distribute traffic across the member links of an EtherChannel or other aggregated links. The hash function takes certain criteria (such as source and destination IP addresses, MAC addresses, or Layer 4 port numbers) to create a hash value that determines which link will carry the traffic. This approach optimizes bandwidth usage and reduces congestion on individual links.

member links: the individual physical Ethernet links that are aggregated together to form an EtherChannel. Each member link contributes to the overall bandwidth and redundancy of the EtherChannel. Member links must be configured identically in terms of speed, duplex mode, and encapsulation type to ensure proper functioning of the EtherChannel.

VLAN Trunking Protocol (VTP): a Cisco proprietary protocol designed to manage VLANs (Virtual Local Area Networks) across a network of switches, enabling centralized VLAN management for easier creation, modification, and deletion of VLANs throughout the network. VTP allows switches to share VLAN configuration information, ensuring that changes made on one switch are propagated to all others within the same VTP domain. There are three modes of VTP operation: Server Mode, where switches can create and modify VLANs and send updates; Client Mode, where switches can only receive updates; and Transparent Mode, where switches forward VTP messages without applying them. All participating switches must be configured with the same VTP domain name to exchange VLAN information. Additionally, VTP supports pruning, which optimizes bandwidth by preventing unnecessary VLAN traffic on trunk links to switches without assigned ports in those VLANs. Different versions of VTP exist (VTPv1, VTPv2, VTPv3), with VTPv3 offering enhancements such as improved security and support for private VLANs. While VTP simplifies VLAN management, it requires careful configuration to avoid issues like accidental deletion of VLANs.

Imagine this example again:

- s1 (root switch)
  - g1/0/2 to s2
  - g1/0/3 to s3
- s2
  - g1/0/1 to s1 (root port)
  - g1/0/3 to s3 (designated port)
- s3
  - g1/0/1 to s1 (root port)

- g1/0/2 to s2 (blocked)

If VLAN 4 contained devices only on s2 and s3, the topology could not be tuned with traffic going directly between the two switches. With PVST, the root bridge can be placed on a different switch or can cost ports differently, on a VLAN-by-VLAN basis. This allows for a link to be blocked for one VLAN and forwarding for another.

Routing

"A system of interconnected routers and related components managed under a common network administration is known as an AS, or a routing domain. The internet is composed of thousands of these AS's spanning the globe."

The common dynamic routing protocols found on most routing platforms today are as follows: Routing Information Protocol Version 2 (RIPv2), Enhanced Interior Gateway Routing Protocol (EIGRP), Open Shortest Path First (OSPF), Intermediate System-To-Intermediate System (IS-IS), Border Gateway Protocol (BGP)

- Distance Vector algorithms

  distance: the distance is the route metric to reach the network
  vector: the vector is the interface or direction to reach the network

  When routing information is received from another router, it stores that information in a local routing DB. A distance vector algorithm (Bellman-Ford/Ford-Fulkerson) is used to determine which paths are the best loop-free paths to each reachable destination. The best paths are determined, they are installed into the routing table and are advertised (from its perspective) to each neighbor router.

  Routers running distance vector protocols advertise the routing information to their neighbors from their own perspective, modified from the original route received. Therefore, a distance vector protocol does not have a complete map of the whole network; instead, its DB reflects that a neighbor router knows how to reach the destination network and how far the neighbor router is from the destination network. The advantage of distance vector protocols is that they require less CPU and memory and can run on low-end routers.

  "An analogy commonly used to describe distance vector protocols is a road sign at an intersection indicating that the destination is 2 miles to the west; drivers trust and blindly follow this information, without really knowing whether there is a shorter or better way to the destination or whether the sign is even correct."

  A distance vector protocol selecs a path purely based on distance. It does not account for link speeds or other factors.

- Enhanced Distance Vector algorithms

  The diffusing update algorithm (DUAL) is an enhanced distance vector algorithm that EIGRP uses to calculate the shortest path to a destination network within a routing domain. EIGRP is called a hybrid because of it's link-state characteristics as well as distance vector characteristics. It uses more advanced metrics than hop count (for example, bandwidth) for its best-path calculations. Information is advertised out in all directions, like other distance vector routing protocols. By default, EIGRP advertises the total path delay and minimum bandwidth for a route. By default, EIGRP advertises the total path delay and minimum bandwidth for a route. Once advertised, each router can calculate the best path based on the information provided to it's neighbors.

  - It offers rapid convergence time for changes in the network topology
  - It sends updates only when there is a topological change. It does not send full routing table updates in a periodic fasion, as distance vector protocols typically do.
  - It uses hellos and forms neighbor relationships just as link-state protocols do.
  - It can use bandwidth, delay, reliability, load, and MTU size instead of hop count for path calculations
  - It has the option to load balance traffic across equal-or unequal-cost paths.

- Link-state algorithms

  A link-state dynamic IP routing protocol advertises the link state and link metric for each of its conntected links and directly connected routers to every router in the network. OSPF advertisements are called Link-State Advertisements (LSAs), and IS-IS uses Link-State packets (LSPs) for its advertisements. As a router receives

an advertisement from a neighbor, it stores the information in a local database called the link-state database (LSDB) and advertises the link-state information on to each of its neighbor routers exactly as it was received. The link-state information is essentially flooded throughout the network, unchanged, from router to router, just as the originating router advertised it. This allows all the routers in the network to have synchronized and identical map of the network.

Using the complete map of the network, every router in the network then runs the Dijkstra shortest path first (SPF) algorithm to calculate the best shortest loop-free paths. The link-state algorithm then populates the routing table with this information.

Becausee they have the complete map of the network, link-state protocols usually require more CPU and memory than distance vector protocols, but they are less prone to routing loops and make better path decisions. In addition, link-state protocols are equipped with extended capabilities such as opaque LSAs for OSPF and TLVs (type/length/value) for IS-IS that allow them to support features commonly used by service providers, such as MPLS traffic engineering.

An analogy for link-state protocols is a GPS navigation system. The GPS navigation system has a complete map and can make the best decision about which way is the shortest and best path to reach a destination.

- Path Vector algorithm

  A path vector protocol such as BGP is similar to a distance vector protocol; the difference is that instead of looking at the distance to determine the best loop-free path, it looks at various BGP path attributes. BGP path attributes include autonomous system path (AS_Path), multi-exit discriminator (MED), origin, next hop, local preference, atomic aggregate, and aggregator.

  A path vector protocol guarantees loop-free paths by keeping a record of each autonomous system that the routing advertisement traverses. Any time a router receives an advertisement in which it is already part of the AS_Path would effectively result in a routing loop.

- path selection

  A router identifies the path that a packet should take by evaluating the prefix length that is programmed in the Forwarding Information Base (FIB). The FIB is programmed through the routing table, which is also known as the Routing Information Base (RIB). The RIB is composed of routes presented from the routing protocol processes. Path selection has three main components:

  - prefix length: the prefix length represents the number of leading binary bits in the subnet mask that are in the on position
  - administrative distance: a rating of the trustworhiness of a routing information source. If a router learns about a route to a destination from more than one routing protocol, and all the routes have the same prefix length, then the AD is compared.
  - metrics: A metric is a unit of measure used by a routing protocol in the best-path calculation. The metrics vary from one routing protocol to another.

  note: the logic for selecting the best path for a routing protocol can vary. Most IGPs prefer internally learned routes over external routes and further prioritize the path with the lowest metric.

- Equal-Cost Multipathing

If a routing protocol identifies multiple paths as a best path and supports multiple path entries, the router installs the maximum number of paths allowed per destination. This is known as equal-cost multipathing (ECMP) and provides load sharing across all links. RIPv2, EIGRP, OSPF, and IS-IS all support ECMP. ECMP provides a mechanism to increase bandwidth across multiple paths by splitting traffic equally across the links.

By default, routing protocols install only routes with the lowest path metric. However, EIGRP can be configured (not enabled by default) to install multiple routes with different path metrics. This configuration allows for unequal-cost load balancing across multiple paths. Traffic is transmitted out the router's interfaces in a ratio to the path metric associated with that interface.

- static routing

  Static routes provide precise control over routing but may create an administrative burden as the number of routers and network segments grows. Using static routing rquires zero network bandwidth because implementing manual route entries does not require communication with other routers.

  Unfortunately, because the routers are not communicating, there is no network intelligence. If a link goes down, other routers will not be aware that the network path is no longer valid. Static routes are useful when:

  - Dynamic routing protocols cannot be used on a router because of limited router CPU or memory
  - Routes learned from dynamic routing protocols need to be superseded

  static route types:

  - directly attchted static routes

    P2P serial interfaces do not have to worry about maintaining an adjacency table and do not use ARP, so static routes can directly reference the outbound interface of a router. A static route that uses only the outbound next-hop interface is known as a directly attached static route, and it requires that the outbound interface be in a up state for the route to be installed into the RIB.

    note: configuring a directly attached static route to an interface that uses ARP (that is, Ethernet) causes problems and is not recommended. The router must repeat the ARP process for every destination that matches that static route, which consumes CPU and memory. Depending on the size of the prefix of the static route and the number of lookups, the configuration can cause system instability.

  - recursive static routes

    The forwarding engine on Cisco devices need to know which interface an outbound packet should use. A recursive static route specifies the IP address of the next-hop address. The next recursive lookup occurs when the router queries the RIB to locate the route towards the next hop IP address (connected, static, or dynamic) and then cross-references the adjacency table.

    Recursive static routes are configured with the command ip route network subnet-mask next-hop-ip. Recursive static routes require the route's next-hop address to exist in the routing table to install the static route into the RIB. A recursive static route may not resolve the next-hop forwarding address using the default route (0.0.0.0/0) entry. The static route will fail next-hop reachability requirements and will not be inserted into the RIB.

  - fully specified static routes

    Static route recursion can simplify topologies if a link fails because it may allow the static route to stay installed while it changes to a different outbound interface in the same direction as the destination. However, problems arise if the recursive lookup resolves to a different interface pointed in the opposite direction.

    To correct this issue, the static route configuration should use the outbound interface and the next-hop IP address. A static route with both an interface and a next-hop IP address is known as a **fully specified static route**. If the interface listed is not in an up state, the router removes the static route from the RIB. Specifying the next-hop address along with the physical interface removes the recursive lookup and does not involve the ARP processing problems that occur when using only the outbound interface.

Fully specified static routes around configured with the command ip route network subnet-mask interface-id next-hop-ip

- floating static routing

  The default AD on a static route is 1, but a static route can be configured with an AD value of 1 to 255 for a specific route. The AD is set on a static route by appending the AD as part of the command structure.

  Using a **floating static route** is a common technique for providing backup connectivity for prefixes learned via dynamic routing protocols. A floating static route is configured with an AD higher than that of the primary route. Because the AD is higher than that of the primary route, it is installed in the RIB only when the primary route is withdrawn.

- static route to null interfaces

  The null interface is a virtual interface that is always in an up state. Null interfaces do not forward or receive network traffic; instead, they drop all traffic destined toward them without adding overhead to a router's CPU.

  Configuring a static route to a null interface provides a method of dropping network traffic without requiring the configuration of an access list. Creating a static route to the Null0 interface is a common technique to prevent routing loops. The static route to the Null0 interface uses a summarized network range, and routes that are more specific point toward the actual destination.

- IPv6 static routes

  The static routing principles for IPv4 routes are exactly the same for IPv6. It is important to ensure that IPv6 routing is enabled by using the configuration command **ipv6 unicast routing**. IPv6 routing is enabled by using the configuration command **ipv6 route** network/prefix-length next-hop-interface-id | [next-hop-interface-id] next-ip-address.

  NOTE: If the next-hop address is an IPv6 link-local address, the static route must be a fully specified static route.

- Policy Based Routing

  A router makes forwarding decisions based on the destination address in IP packets. Some scenarios accommodate other factors, such as packet length or source address, when deciding where the router should forward a packet.

  Policy-based routing (PBR) allows for conditional forwarding of packets based on packet characteristics besides the destination IP address.

  PBR provides the following capabilities:

  - Routing by protocol type (ICMP, TCP, UDP, and so on)
  - Routing by source IP address, destination IP address, or both
  - Manual assignment of different network paths to the same destination, based on tolerance for latency, link speed, or utilzation for specific tansient traffic

  Some of the drawbacks of conditional routing include the following:

  - Administrative burden in scalability
  - Lack of network Intelligence
  - Troubleshooting complexity
  - Not all configuration options are supported by all hardware platforms

  Packets are examined for PBR processing as they are received on the router interface. Local PBR policies can also identify traffic originating from the router.

  PBR verifies the existence of the next-hop IP address and then forwards packets using the specified next-hop address. Additional next-hop addresses can be configured so that if the first next-hop address is not in the RIB,

the secondary next-hop addresses can be used. If none of the specified next-hop addresses exist in the routing table, the packets are not conditionally forwarded.

NOTE: PBR policies do not modify the RIB because the policies are not universal for all packets. Because PBR does not change the RIB, troubleshooting can become complicated as the routing table displays the next-hop address learned from the routing protocol but does not display a different next-hop address learned from the routing protocol but does not display a different next-hop address for the conditional traffic.

- Virtual Routing and Forwarding

  Virtual routing and forwarding (VRF) is a technology that creates separate virtual routers on physical router. Router interfaces, routing tables, and forwarding tables are completely isolated between VRFs, preventing traffic from one VRF from forwarding into another VRF. All router interfaces belong to the global VRF until they are specifically assigned to a user-defined VRF. The global VRF is synonymous to regular routing table of non-VRF routers.

  Every router's VRF maintains a separate routing table, so it is possible to allow for overlap ping IP address ranges. VRF creates segmentation between network interface, network supinterfaces, IP addresses, and routing tables. Configuring VRF on a router ensures that the paths of each VRF are isolated, network security is increased through segmentation, and encrypting traffic on the network is not needed to maintain privacy between VRF instances.

  The creation of multiprotocol VRF instances requires the global configuration command **vrf definition** vrf-name. Under the VRF definition submode, the command **address-family ipv4 — ipv6** is required to specify the appropriate address family. The VRF instance is then associated to the interface with the command **vrf forwarding** vrf-name under the interface configuration submode.

  The following steps are required to create a VRF and assign it to an interface:

  1.0 Create a multiprotocol VRF routing table by using the command **vrf definition** vrf-name.

  2.0 Initialize the appropriate address family can be IPv4, IPv6, or both.

  3.0 Enter interface configuration submode and specify the interface to be associated with the VRF instance by using the command **interface** interface-id

  4.0 Associate the VRF instance to the interface or subinterface by entering the command **vrf forwarding** vrf-name under interface configuration submode.

  5.0 Configure an IP address (IPv4, IPv6, or both) on the interface or subinterface by entering either or both of the following commands: 1. **ip address** ip-address subnet-mask [**secondary**] 2. **ipv6 address** ipv6-address/prefix-length

  VRF instances on a router can be compared to that of virtual local area networks (VLANs) on a switch. However, instead of relying on Layer 2 technologies such as 802.1Q for separation, VRF insttances allow for interaction and segmentation with Layer 3 dynamic routing protocols.

- EIGRP Fundamentals

  EIGRP overcomes the deficiencies of other distance vector routing protocols like RIPv2 with features such as unequal-cost load balancing, support for networks 255 hops away, and rapic convergence features. EIGRP uses a diffusing update algorithm (DUAL) to indentify networks paths and enable fast convergence using precalculated loop-free backup paths. Most distance vector routing protocols use hop count as the metric for routing decisions. However, using hop count for path selection does not take into account link speed and total delay. EIGRP adds to the route selection algorithm logic that uses factors other than hop count alone.

A router can run multiple EIGRP processes. Each process operates under the context of an /textbfautonomous system, which represents a common routing domain. Routers within the same domain use the same metric calculation formula and exchange routes only with members of the same autonomous system. An EIGRP autonomous system should not be confused with a BGP autonomous system.

Successor route: the route with the lowest path metric to reach a destination

Successor: the first next-hop router for the successor route

Feasible distance (FD): the metric value for the lowest-metric path to reach a destination. the feasible distance is calculated locally using the formula shown in the "Path Metric Calculattion"

Reported distance (RD): the distance reported by a router to reach a destination. the reported distance value is the feasible distance for the advertising router

Feasibility condition: a condition under which, for a route to be considered a backup route, the reported distance received for that route must be less than the feasible distance calculated locally. this logic guarantees a loop-free path

Feasible successor: a route that satisfies the feasibility condition and is maintained as a backup route. the feasibility condition ensures that the backup route is loop free

EIGRP contains a **topology table** that makes it different from a "true" distance vector routing protocol. EIGRP's topology table is a vital component to DUAL and contains information to identify loop-free backup routes. The topology table contains all the network prefixes advertised within an EIGRP autonomous system. Each entry in the table contains the following:

- Network prefix
- EIGRP neighbors that have advertised that prefix
- Metrics from each neighbor (for example, reported distance, hop count)
- Values used for calculating the metric (for example, load, reliability, total delay, minimum bandwidth)

EIGRP neighbors exchange the entire routing table when forming an adjacency, and they advertise only incremental updates as topology changes occur within a network. The neighbor adjacency table is vital for tracking neighbor status and the updates sent to each neighbor.

EIGRP uses five different packet types to communicate with other routers. It uses IP protocol number 88 in the IP header, and it uses multicast packets where possible and unicast packets when necessary. Communication between routers is done with multicast, using the group address 224.0.0.10 when possible.

Opcode Value : Packet Type : Function

- 1 : Update : used to transmit routing and reachability information with other EIGRP neighbors
- 2 : Request : used to get specific information from one or more neighbor
- 3 : Query : Sent out to search for another path during convergence
- 4 : Reply : Sent in response to a query packet
- 5 : Hello : Used for discovery of EIGRP neighbors and for detecting when a neighbor is no longer available

- Path Metric Calculation

Metric calculation is a critical component for any routing protocol. EIGRP uses multiple factors to calculate the metric for a path. Metric calculation uses bandwidth and delay by default, but it can include interface load and reliability too.

$$Metric = 256 * [(K_1 * BW + \frac{K_2 * BW}{256 - Load} + K_3 * Delay) * \frac{K_5}{K_4 + Reliability}]$$

EIGRP uses K values to define which factors the formula uses and the associated impact of a factor when calculating the metric. A common misconception is that K values directly both reference bandwdith (BW).

BW represents the slowest link in the path scaled to a 10 Gbps link (10^7). Link speed correlates to the configured interface bandwidth on an interface and is measured in kilobits per second. Delay is the total measure of delay in the path, measured in tens of microseconds (μ seconds).

$$Metric = 256 * [(K_1 * \frac{10^7}{Min.Bandwidth} + \frac{K_2 * \frac{10^7}{Min.Bandwidth}}{256 - Load} + \frac{K_3 * Delay}{10}) * \frac{K_5}{K_4 + Reliability}]$$

NOTE: RFC 7868 states that if the $K_5 = 0$ then the reliability quotient is defined to be 1.

By default, K_1 and K_3 have the value 1, and K_2, K_4 and K_5 are set to 0.

The EIGRP update packet includes path attributes associated with each prefix. The EIGRP path attributes can include hop count, cumulative delay, minimum bandwidth link speed, and RD. The attributes are updated each hop along the way, allowing each router to independently identify the shortest path.

- Wide Metrics The original EIGRP specifications measured delay in 10 $\mu seconds$ units and bandwidth in kilobits per second, which did not scale well with higher-speed interfaces. In fact, EIGRP classic metrics do not differentiate between an 11 Gbps link and a 50 Gbps link.

  EIGRP includes support for a second set of metrics, known as **wide metrics**, that addresses the issue of scalability with higher-capacity interfaces. Just as EIGRP scaled by 256 to accommodate IGRP, EIGRP wide metrics scale by 65,536 to accommodate higher-speed links. This provides support for interface speeds up to 655 Tbps without any scalability issues. The following shows the explicit EIGRP wide metrics formula. Notice that an additional K value (K_6) is included; it adds an extended attribute to measure jitter, energy, or other future attributes.

  $$WideMetrics = 65,536 * [(K_1 * BW + \frac{K_2 * BW}{256 - Load} + K_3 * Latency + K_6 * Extended) * \frac{K_5}{K_4 + Reliability}]$$

  Latency is the total interface delay measured in picoseconds (10^{-12}) instead of measuring in microseconds (10^{-6})...the following is an updated EIGRP wide metric formula that takes into account the conversions in latency and scalability

  $$WideMetric =$$
  $$65,536 * [(\frac{K_1 * 10^7}{Min.Bandwidth} + \frac{\frac{K_2 * 10^7}{Min.Bandwidth}}{256 - Load} + \frac{K_3 * Latency}{10^-6} + K_6 * Extended) * \frac{K_5}{K_4 + Reliability}]$$

- Metric Backward Compatibility
  EIGRP wide metrics were designed with backward compatibility in mind. With EIGRP wide metrics, K_1 and K_3 are set to a value of 1, and K_2, K_4, K_5, and K_6 are set to 0, which allows backward compatibility because the K value metrics match with classic metrics. As long as K_1 through K_5 are the same and K_6 is not set, the two metrics styles allow adjacency between routers. EIGRP is able to detect when peering with a router using classic metrics, and it unscales the metric.

- Load Balancing
  EIGRP allows multiple successor routes (using the same metric) to be installed into the RIB. Installing multiple paths into the RIB for the same prefix is called equal-cost multipathing (ECMP). EIGRP supports unequal-cost load balancing, which allows installation of both successor routes and feasible successors into the EIGRP RIB. EIGRP supports unequal-cost load balancing by changing EIGRP's variance multiplier. The EIGRP supports unequal-cost load balancing by changing EIGRP's variance multiplier. The EIGRP

variance value is the feasible distance (FD) for a route multiplied by the EIGRP variance multiplier. Any feasible successor's FD with a metric below the EIGRP variance value is installed into the RIB. EIGRP installs multiple routes where the FD for the routes is less than the EIGRP variance value up to the maximum number of ECMP routes, as discussed earlier

Dividing the feasible successor metric by the successor route metric provides the variance multiplier. The variance multipler is a whole number, so any remainders should always round up.

- Failure Detection and Timers
  A secondary function for the EIGRP **hello packets** is to ensure that EIGRP neighbors are still healthy and available. EIGRP hello packets are sent out in intervals determined by the **hello timer**. The default EIGRP hello timer is 5 seconds, but it is 60 seconds on slow-speed interfaces (T1 or lower).

  EIGRP uses a second timer for the hold time, which is the amount of time EIGRP deems the router reachable and functioning. The hold time value defaults to 3 times the hello interval. The default value is 15 seconds, and it is 180 seconds for slow-speed interface (T1 or lower). The hold time decrements, and upon receipt of a hello packet, the hold time resets and restarts the count down. If the hold time reaches 0, EIGRP declares the neighbor unreachable and notifies DUAL of a topology change.

  - Convergence
    When a link fails, and the interface protocol moves to a down state, any neighbor attached to that interface moves to a down state too. When an EIGRP neighbor moves to a down state, path recomputation must occur for any prefix where thath EIGRP neighbor was a successor (upstream router).

    When EIGRP detects that it has lost its successor for a path, the feasible successor, if one exists, instantly becomes the successor route, providing a backup route. the router sends out an update packet for that path because of the new EIGRP path metrics. Downstream routers run their own DUAL for any impacted prefixes to account for the new EIGRP metrics. It is possible that a change of the successor route or feasible successor may occur upon receipt of new EIGRP metrics from a successor router for a prefix.

    If a feasible successor is not available for a prefix. DUAL must perform a new route calculation. The route state changes from passive (P) to active (A) in the EIGRP topology table. the router detecting the topology change sends out query packets to EIGRP neighbors for the route. The query packet includes the prefix with the delay set to infinity so that other routers are aware that it has gone active. When the router sends the EIGRP query packets, it sets the reply status flag set for each neighbor on a per prefix basis. The router tracks the reply status for each of the EIGRP query packets on a prefix basis.

    Upon receipt of a query packet, an EIGRP router does one of the following:

    * It might reply to the query that the router does not have a route to the prefix.
    * If the query did not come from the successor for that route, it detects the delay set for infinity, but ignores it because it did not come from the successor. The receiving router replies with the EIGRP attributes for that route
    * If the query came from the successor for the route, the receiving router detects the delay set for infinity, sets the prefix as active in the EIGRP topology, and sends out a qquery packet to all downstream EIGRP neighbors for that route.

    The query process countinues from router to router until a query reaches a query boundary. A query boundary is established when a router does not mark the prefix as active, meaning that it responds to a query as follows:

    * It says it does not have a route to the prefix
    * It replies with EIGRP attributes because the query did not come from the successor.

    When a router receives a reply for every downstream query that was sent out, it completes the DUAL, changes the route to passive, and sends a reply packet to any upstream routers that sent a query packet to it. Upon receiving the reply packet for a prefix, the reply packet is notated for that neighbor and prefix. The reply process continues upstream for the queries until the first router's queries are received.

- Route Summarization
  EIGRP works well with minimal optimizations. Scalability of an EIGRP autonomous system depends on **summarization**. As the size of an EIGRP autonomous system increases, convergence may take longer. Scaling an EIGRP topology requires summarizing routes in a hierarchical fashion.

EIGRP summarizes routes on a per-interface basis and not under the routing process. Summarization is enabled by configuring a summary address range are referred to as componnet routes. With summarization enabled, the compnent routes are suppressed (not advertised), and only the summary route is advertise. The summary route is not advertised until one of the component route matches it. Interface-specific summarization can be performed in any portion of the network topology. In additon to shrinking the routing tables of all the routers, summarization creates a query boundary and shrinks the query domain when a route goes active during convergece.

COMMANDS

Configure a directly attached IPv4 static route

- **ip route network** *subnet-mask next-hop-interface-id*

Configure a recursive IPv4 static route

- **ip route** *network subnet-mask next-hop-ip*

Configure a fully specified IPv4 static route

- **ip route network** *subnet-mask interface-id next-hop-ip*

OSPF

- OSPF Fundamentals OSPF sends to neighboring routers link-state advertisements (LSAs) that contain the link state and link metric. The received LSAs are stored in a local database called the link-state database (LSDB), and they are flooded throughout the OSPF routing domain, just as the advertising router advertised them. All OSPF routers maintain a synchronized identical copy of the LSDB for the same area.

  The LSDB provides the topology of the network, in essence providing for the router a complete map of the network. All OSPF routers run the Dijkstra shortest path first (SPF) algorithm to construct a loop-free topology of shortest paths. OSPF dynamically detects topology changes within the network and calculates loop-free paths in a short amount of time with minimal routing protocol traffic.

  Each router sees itself as the root or top of the SPF tree, and the **shortest path tree (SPT)** contains all destination networks within the OSPF domain. The SPT differs for each OSPF router, but the LSDB used to calculate the SPT is identical for all OSPF routers.

  The SPTs give the illusion that no redundancy exists to the networks, but remeber that the SPT shows the shortest path to reach a network and is built from the LSDB, which contains all the links for an area. During a topology change, the SPT is rebuilt and may change.

  OSPF provides scalability for the routing table by using multiple OSPF areas within the routing domain. Each OSPF area provides a collection of connected networks and hosts that are grouped together. OSPF uses a two-tier hierarchical architecture, where Area 0 is a special area known as the backbone, to which all other areas must connect. In other word, Area 0 provides transit connectivity between non-backbone then advertises routes into other non-backbone areas.

  The exact topology of the area is invisible from outside the area while still providing connectivity to routers outside the area. This means that routers outside the are do not have a complete topological map for that area, which reduces OSPF traffic in that area.

  When you segment an OSPF routing domain into multiple areas, it is no longer true that all OSPF routers will have identical LSDBs; however, all routers within the same area will have identical area LSDBs.
  The reduction in routing traffic uses less router memory and resources and therefore provides scalability.

A router can run multiple OSPF processes. Each process maintains its own unique database and routes learned in one OSPF process are not availible to a different OSPF process without redistribution of routes between processes. The OSPF process numbers are locally significant and do not have to match amoung routers. Running OSPF process number 1 on one router and running OSPF process number 1234 will still allow the two routers to become neighbors.

- Inter-Router Communication OSPF runs directly over IPv4, using protocol number 89 in the IP header, which is reserved for OSPF by the Internet Assigned Numbers Authority (IANA). OSPF uses multicast where possible to reduce unnecessary traffic. the two OSPF multicast addresses are as follows:
  - **ALLSPFRouters**: IPv4 address 244.0.0.5 or MAC address 01:00:5E:00:00:05. all routers running OSPF should be able to receive these packets.
  - **ALLDRouters**: IPv4 address 244.0.0.6 or MAC address 01:00:5E:00:00:06. communication with designated routers (DRs) uses this address.

  Within the OSPF protocol, five types of packets are communicated.
  - Hello These packets are for discovering and maintaining neighbors. Packets are sent out periodically on all OSPF interfaces to discover new neighbors while ensuring that other adjacent neighbors are still online.
  - Database description (DBD) or (DDP) These packets are for summarizing database contents. Packets are exchanged when an OSPF adjacency is first being formed. These packets are used to describe the contents of the LSDB.
  - Link-state request These packets are for database downloads. When a router thinks that part of its LSDB is stale, it may request a portion of a neighbor's database by using this packet type.
  - Link-state update (LSU) These packets are for database updates. This is an explicit LSA for a specific network link and normally is sent in direct response to and LSR.
  - Link-state ack These packets are for flooding acknowledgments. this packets are sent in response to the flooding of LSAs, thus making flooding a reliable transport feature.
- OSPF Hello Packets OSPF **hello packets** are responsible for discovering and maintaining neighbors. In most instances, a router sends hello packets to the AllSPFRouters address (244.0.0.5).
  - **Router ID (RID)**: A unique 32-bit ID within an OSPF domain used to build a topology
  - Authentication options: A field that allows secure communication between OSPF routers to prevent malicious activity. Options are none, clear text, or Message Digest 5 (MD5) authentication
  - Area ID: The OSPF area that the OSPF interface belongs to. It is a 32-bit number that can be written in dotted-decimal dormat (0.0.1.0) or decimal (256)
  - Interface address mask: The network mask for the primary IP address for the interface out which the hello is sent
  - **Interface priority**: The router interface priority for DR elections
  - **Hello interval**: The time span, in seconds, that a router sends out hello packets on the interface
  - **Dead interval**: The time span, in seconds, that a router waits to hear a hello from a neighbor router before it declares that router down
  - Designated router and backup designated router: The IP address of the DR and backup DR (BDR) for the network link
  - Active neighbor: A list of OSPF neighbors seen on the network segment. A router must have received a hello from the neighbor within the dead interval
- Router ID
  The OSPF router ID (RID) is a 32-bit number that uniquely identifies an OSPF router, and is an essential component for building an OSPF topology. In some OSPF output commands. neighbor ID refers to the RID; the terms are synonymous. The RID must be unique for each OSPF process in an OSPF domain and must be unique between OSPF processes on a router.

- Neighbors
  An OSPF neighbor is a router that shares a common OSPF-enabled network link. OSPF routers discorver other neighbors via the OSPF hello packets. An adjacent OSPF neighbor is an OSPF neighbor that shares a synchronized OSPF database between the two neighbors.

  Each OSPF process maintains a table for adjacent OSPF neighbors and the state of each router.

  * Down: This is the initial state of a neighbor relationship. It indicates that the router has not received any OSPF hello packets.
  * Attempt: This state is relevant to NBMA networks that do not support broadcast and require explicit neighbor configuration. This state indicates that no information has been received recently, but the router is still attempting
  * Init: This state indicates that a hello packet has been received from another router, but bidirectional communication has not been established
  * 2-Way: Bidirectional communication has been established. If a DR or BDR is needed, the election occurs during this state
  * ExStart: This is the first state in forming an adjacency. Routers identify which router will be the primary or secondary for the LSDB synchronization
  * Exchange: During this state, routers are exchanging link states by using DBD packets
  * loading: LSR packets are sent to the neighbor, asking for the more recent LSAs that have been discovered (but not received) in the Exchange state
  * Full: Neighboring routers are fully adjacent
- Designated router and Backup Designated Router
  Multi-access networks such as Ethernet (LANs) and Frame Relay allow more than two routers to exist on a network segment. Such a setup could cause scalability problems with OSPF as the number of routers on a segment increases. Additional routers flood more LSAs on the segment, and OSPF traffic becomes excessive as OSPF neighbor adjacencies increases. If occurrences of database flooding on a network.

  The number of edges formula, $n(n-1)/2$ where n represents the number of routers, is used to identify the number of sessions in a full mesh topology. If 5 routers were present on a segment, $5(5-1)/2 = 10$ then 10 OSPF adjacencies would exist for that segment. Continuing the logic, adding 1 additional router would make 15 OSPF adjacencies on a network segtment. Having so many adjacencies per segment consumes more bandwidth, more CPU processing, and more memory to maintain each of the neighbor states.

  OSPF over comes this inefficiency by created a pseudonode (virtual router) to manage the adjacency state with all the other routers on that broadcast network segment. A router on the broadcast segment, known as the **designated router (DR)**, assumes the role of the pseudonode. The DR reduces the number of OSPF adjacencies on a multi-access netowrk segment because routers form a full OFPF adjacency only with the DR and not with each other. The DR is responsible for flooding updates to all OSPF routers on that segment as the updates occur.

  If the DR were to fail, OSPF would need to form new adjacencies, invoking all new LSAs, and could potentially cause a temporary loss of routes. In the even of DR failure, a **backup designated router (BDR)** becomes the new DR; then an election occurs to replace the BDR. To minimuzed transition time, the BDR also forms full OSPF adjacencies with all OSPF routers on that segment.

  The DR/BDR process distributes LSAs in the following manner, assuming that all OSPF routers (DR, BR, and DROTHER) on a segment form a full OSPF adjacency with the DR and BDR.

  1.0 As an OSPF router learns of a new route, it sends the updated LSA to the AllDRouters (224.0.0.6) address, which only the DR and BDR accept and process
  2.0 The DR sends a unicast acknowledgment to the router that sent the initial LSA update
  3.0 The DR floods the LSA to all the routers on the segment via the AllSPFrouters (224.0.0.5) address

- OSPF Configuration
  The configuration process for OSPF resides mostly under the OSPF process, but some OSPF options go directly on the interface ocnfiguration submode. The command **router ospf** process-id defines and initializes the OSPF process. The OSPF process ID is locally significant but is generally kept the same for operational consistency. OSPF is enabled on an interface using two methods: 1. An OSPF network statement 2. Interface-specific configuraiton

  - OSPF Network Statement
    The OSPF network statement identifies the interfaces that the OSPF process will use and the area that those interfaces participate in. The network statements match against the primary IPv4 address and netmask associated with an interface.

    A common misconception is that the network statement advertises the networks into OSPF; in reality, though, the network statement is selecting and enabling OSPF on the interface. The interface is then advertised in OSPF through the LSA. The network statement uses a wildcard mask, which allows the configuration to be as specific or vague as necessary. The selection of interfaces within the OSPF process is accomplished by using the command **network** ip-address wildcard-mask **area** area-id.

    The concept is similar to the configuration of Enhanced Interior Gateway Routing Protocol (EIGRP), except that the OSPF area is specified. If the IP address for an interface matches two network statements with different areas, the most explicit network statement (that is, the longest match) preempts the other network statements for area allocation.

    The connected network for the OSPF-enabled interface is added to the OSPF LSDB under the corresponding OSPF area in which the interface participates. Secondary connected networks are added to the LSDB only if the secondary IP address matches a network statement associated with the same area.

  - Interface-Specific Configuration
    The second method for enabling OSPF on an interface for IOS XE is to configure it specifically on an interface with the command **ip ospf** process-id **area** area-id [**secondaries none**]. This method also adds secondary connected networks to the LSDB unless the secondaries **none** option is used.

    This method provides explicit control for enabling OSPF; however, the configuration is not centralized and increases in complexity as the number of interfaces on the routers increases. If a hybrid configuration exists on a router, interface-specific settings take precedence over the network statement with the assignment of the areas.

  - Statically Setting the Router ID
    By default, the RID is dynamically allocated using the highest IP address of any up loopback interfaces. If there are no up loopback interfaces, the highest IP address of any active up physical interfaces becomes the RID when the OSPF process initializes.

    The OSPF process selects the RID when the OSPF process initializes, and it does not change until the process restarts. Interface changes (such as the addition or removal of IP addresses) on a router are detected when the OSPF process restarts, and the RID changes accordingly. The OSPF topology is built on the RID. Setting a static RID helps with troubleshooting and reduces LSAs when an RID changes in an OSPF environment. The RID is four octets in length and generally represents an IPv4 address that resides on the router for operational simplicity; however, this is not a requirement. The command **router-id** router-id statically assigns the OSPF RID under the OSPF process.

    The EXEC command **clear ip ospf process** restarts the OSPF process on a router so the OSPF can use the new RID.

- Passive Interfaces
  Enable an interface with OSPF is the quickest way to advertise a network segement to other OSPF routers. However, it might be easy for someone to plug in an unauthorized OSPF router on an OSPF-enabled network segment and introduce false routes, thus causing havoc in the network. Making the network interface passive still adds the network segment into the LSDB but prohibits the interface from forming OSPF adjacencies. A **passive interface** does not send out OSPF hellos and does not process any received OSPF packets.

  The command **passive-interface default** makes all interfaces passive. To allow for an interface to process OSPF packets, the command **no passive-interface** interface-id is used.
- Requirements for Neighbor Adjacency

  * RIDs must be unique between the two devices. They should be unique across all areas of the entire OSPF routing domain to prevent errors
  * The interfaces must share a common subnet. OSPF uses the interface's primary IP address when sending out OSPF hellos. The network mask (netmask) in the hello packet is used to extract the network ID of the hello packet. Network masks on the interfaces must match, except for OSPF point-to-point network type interfaces or virtual links
  * The maximum transsion units (MTUs) on the interfaces must match. The OSPF protocol does not support fragmentation, so the MTUs on the interfaces should match
  * The area ID must match for the segment
  * The DR enablement must match for the segment
  * OSPF hello and dead timers must match for the segment
  * Authentication type and credentials (if any) must match for the segment
  * Area type flags must match for the segment (for example, Stuf, NSSA), (These are not discussed in this book)
- Confirmation of Interfaces
  It is a good practice to verify that the correct interfaces are running OSSPF after making changes to the OSPF configuration. The command **show ip ospf interface [brief | interface-id]** displays the OSPF-enabled interfaces. OSPF Interface Columns (Field: Description)

  * Interface: Interfaces with OSPF enabled
  * PID: The OSPF process ID associated with this interface
  * Area: The area that this interface is associate with
  * IP Address/Mask: The IP address and subnet mask (network mask) for the interface
  * Cost: The cost metric assigned to an interface that is used to calculate a path metric
  * State: The current interface state, which could be DR, BDR, DROTHER, LOOP, or Down
  * Nbrs F: The number of neighbor OSPF routers for a segmenet that are fully adjacent
  * Nbrs C: The number of neighbor OSPF routers for a segment that have been detected and are in a 2-Way state

  NOTE: The DROTHER is a router on the DR-enabled segmenet that is not the DR or the BDR; it is simply the other router. DROTHERs do not establish full adjacency with other DROTHERs
- Verification of OSPF Neighbor Adjacencies
  COMMAND: **show ip ospf neighbor [detail]** ... OSPF Neighbor State Fields (Field : Description)

  * Neighbor ID: The router ID (RID) of the neighboring router
  * PRI: The priority for the neighbor's interface, which is used for DR/BDR elections
  * State: The first field is the neighbor state ... The second field is the DR, BDR, or DROTHER role if the interface requires a DR. For non-DR network links, the second field shows just a hyphen (-)
  * Dead Time: The time left until the router is declared unreachable
  * Address: The primary IP address for the OSPF neighbor

    * Interface: The local interface to which the OSPF neighbor is attached
  - Verification of OSPF Routes
    The next step is to verify the OSPF routes installed in the IP routing table. OSPF routes that install into the Routing Information Base (RIB) are shown with the command **show ip route ospf**.

- Default Route Advertisement
  OSPF suuports advertising the default route into the OSPF domain. The default route is advertised by using the command **default-information [always] [metric *metric-value*]** [**metric-type** *type-value*] underneath the OSPF process

  If a default route does not exist in a routing table, the **always** optional keyword advertises a default route regardless of whether a default route exists in the RIB or not. In addition, the route metric can be changed with the **metric** metric-value option, and the metric type can be changed with the **metric-type** type-value option

- Common OSPF Optimizations
  Almost every network requires tuning based on the equipment, technical requirements, or a variety of other factors. The following section explain common concepts involved with the tuning of an OSPF network
  - Link Costs Interface cost is an essential component of Dijkstra's SPF calculation because the shortest path metric is based on the cumulative interface cost (that is, metric) from the router to the destination. OSPF assigns the OSPF link cost (that is, metric) for an interface by using the formula:

    $$Cost = \frac{ReferenceBandwidth}{InterfaceBandwidth}$$ (the default metric bandwidth is 100 Mbps)

    OSPF Interface Costs Using Default Settings (Interface Type: OSPF Cost)
    * T1: 64
    * Ethernet: 10
    * FastEthernet: 1
    * GigabitEthernet: 1
    * 10 GigabitEthernet: 1

    Notice that there is no differentiation in the link cost associated with a FastEthernet interface and a 10 GigabitEthernet interface. Changing the reference bandwidth to a higher value allows for a differentiation of cost between higher-speed interface. Making the value too high could cause issues because low-bandwidth interfaces would not be distinguishable. The OSPF LSA metric field is 16 bits, and the interface cost cannot exceed 65,535.

    Under the OSPF process, the command **auto-cost reference-bandwidth** bandwidth-in-mbps changes the reference bandwidth for all OSPF interfaces associated with that process. If the reference bandwidth is changed on one router, the reference bandwidth should be changed on all OSPF routers to ensure that SPF uses the same logic to prevent routing loops. It is a best practice to set the same reference bandwidth for all OSPF routers.

    NOTE: NX-OS uses a default reference cost of 40,000 Mbps. To align with other routers and incorporate higher-speed interfaces, setting the reference bandwidth to 40,000 Mbps could standardize the reference bandwidth across multiple platforms.

    The OSPF cost can be set manually with the command **ip ospf cost** 1-65535 under the interface configuration submode. Wile the interface cost is limited to 65,535 because of LSA field limitations, the path metric can exceed a 16-bit value (65,535) because all the link metrics are calculated locally.
  - Failure Detection
    A secondary function of the OSPF hello packets is to ensure that adjacent OSPF neighbors are still healthy and available. OSPF sends hello packets at set intervals, based on the hello time. OSPF uses a second

timer called the OSPF dead interval time, which defaults to four times the hello timer. Upon receipt of a hello packet from a neighboring router, the OPSF dead timer resets to the initial value and then starts to decrement again.

If a router does not receive a hello before the OSPF dead interval timer reaches 0, the neighbor state is changed to down. The OSPF router immediately sends out the appropriate LSA, reflecting the topology change, and the SPF algorithm processes on all routers within the area.

* Hello Timer
  The default OSPF hello timer interval varies based on the OSPF network type. OSPF allows modification to the hello timer interval with values between 1 and 65,535 seconds. Changing the hello timer interval modifies the default dead interval too. The OSPF hello timer is modified with the interface configuration submode command **ip ospf hello-interval** 1-65535 under the interface configuration submode.
* Dead Interval Timer
  The dead interval timer can be changed to a value between 1 and 65,535 seconds. The OSPF dead interval timer can be changed with the command **ip ospf dead-interval** 1-65535 under the interface configuration submode.
  NOTE: Always make sure that the dead interval timer setting is greater than the hello timer setting to ensure that the dead interval timer does not reach 0 in between hello packets
* OSPF Timers
  The timers for an OSPF interface are shown with the command **show ip ospf interface**

– DR Placement The DR and BDR roles for a broadcast network consume CPU and memory on the host routers in order to maintain states with all the other routers for that segement. Placing the DR and BDR roles on routers with adequate resources is recommended.

The following sections explain the DR election process and how the DR role can be assigned to specific hardware.

* Designated Router Elections
  The DR/BDR election occurs during OSPF adjacency-specifically during the last phase of 2-Way neighbor state and just before the ExStart state. When a router enters the 2-Way state, it has already received a hello from the neighbor. If the hello packet includes an RID other than 0.0.0.0 for the DR and BDR, the new router assumes that the current routers are the actual DR and BDR.
  Any router with an OSPF priority of 1 to 255 on its OSPF interface attempts to become the DR. By default, all OSPF interfaces use a priority of 1. The routers place their RID and OSPF priorities in their OSPF hellos for that segment.

  Routers then receive and examine OSPF hellos from neighboring routers. If a router identifies itself as being a more favorable routers than the OSPF hellos it receives, it continues to send out hellos with its RID and priority listed. If the hello received is more favorable, the router updates its OSPF hello packet to use the more preferable RID in the DR field. OSPF deems a router more preferable if the priority for the interface is the highest for that segment. If the OSPF priority is the same, the higher RID is more favorable.

  After all the routers have agreed on the same DR, all routers for that segment become adjacent with the DR. Then the election for the BDR takes place. The election follows the same logic for the DR election, except that the DR does not add its RID to the BDR field of the hello packet.

  The OSPF DR and BDR roles cannot be preempted after the DR/BDR election. Only upon the failure (or process restart of the DR or BDR) does the election start to replace the role that is missing.
  NOTE: To ensure that all routers on a segment have fully initialized, OSPF initiates a wait timer when OSPF hello packets do not contain a DR/BDR router for a segment. The default value for the wait timer is the dead interval timer. After the wait timer has expired, a router participates in the DR electio
  . The wait timer startts when OSPF first sttarts on an interface; so that a router can still elect itsself as

the DR for a segment without other OSPF routers, it waits until the wait timer expires.

The easiest way to determine the interface role is by viewing the OSPF interface with the command **show ip ospf interface brief**

The neighbo's full adjacency field reflects the number of routers that have become adjacent on that network segment; the neighbors count field is the number of other OSPF routers on that segment. You might assume that all routers will become adjacent with each other, but that would default the purpose of using a DR. Only the DR and BDR become adjacent with routers on a network segment.

* DR and BDR Placement
  Modifying a router's RID for DR placement is a bad design strategy. A better technique involves modifying the interface priority to a higher value than the existing DR has. Remeber that OSPF does not preempt the DR or BDR roles, and the OSPF proces might need to be restarted on the current DR/BDR for the changes to take effect.

  The priority can be set manually under the interface configuration with the command **ip ospf priority** 0-255. Setting an interface priority to 0 removes that interface from the DR/BDR election immediately. Raising the priority above the default value (1) makes that interface more favorable compared to interface with the default value.

– OSPF Network Types Different media can provide different characteristics or might limit the number of nodes allowed on a segment. Frame Relay and Ethernet are common multiple-access media, and because they support more than two nodes on a network segment, the need for a DR exists. Other network circuits, such as serial links (with HDLC or PPP encapsulation), do not require a DR, and having one would just waste router CPU cycles.

The default OSPF network type is set based on the media used for the connection and can be changed independently of the actual media type used. Cisco's implementation of OSPF considers the various media and provides five OSPF network types

OSPF Network Types (Type : Description: DR/BDR Field in OSPF Hellos : Timers)

* Broadcast : Default setting on OSPF-enabled Ethernet links. : Yes : Hello 40, Wait 40, Dead 40
* Non-broadcast : Default setting on OSPF-enabled Frame Relay main interface or Frame Relay multipoint subinterfaces : No : Hello 30, Wait 120, Dead 120
* Point-to-point : Default setting on OSPF-enabled Frame Relay point-to-point subinterfaces : No : Hello 10, Wait 40, Dead 40
* Point-to-multipoint : Not enabled by default on any interface type. Interface is advertised as a host route (`\32`) and sets the next-hop address to the outbound interface. Primarily used for hub-and-spoke topologies : No : Hello 30, Wait 120, Dead 120
* Loopback : Default setting on OSPF-enabled loopback interfaces. Interface is advertised as a host route (`\32`) : N/A : N/A

The non-broadcast or point-to-multipoint network types are beyond the scope of the Enterprise Core exam, but the other OSPF network types are explained in the following sections.

* Broadcast
  Broadcast media such as Ethernet are better defined as broadcast multi-access to distinguish them from non-broadcast multi-access (NBMA) networks. Broadcast networks are multiaccess in that they are capable of connecting more than two devices, and broaadcast sent out one interface are capable of reaching all interfaces attached to that segment.

  The OSPF network type is set to broadcast by default for Ethernet interfaces. A DR is required for this OSPF network type because of the possibility that multiple nodes can exist on a segment, and LSA flooding needs to be controlled. The hello timer defaults to 10 seconds, as defined in RFC 2328.

The interface parameter command **ip ospf network broadcast** overrides the automatically configured setting and statically sets an interface as an OSPF broadcast network type.

* Point-to-Point Networks
  A network circuit that allows only two devices to communicate is considered a point-to-point (P2P) network. Because of the nature of the medium, point-to-point networks do not use Address Resolution Protocol (ARP), and broadcast traffic does not become the limiting factor.

  The OSPF network type is set to point-to-oiunt by default for serial interfaces (HDLC or PPP encapsulation), generic routing encapsulation (GRE) tunnels, and point-to-point Frame Relay subinterfaces. Only two nodes can exist on this type of network medium, so OSPF does not waste CPU cycles on DR functionality. The hello timer is set to 10 seconds on OSPF point-to-point network types.
* Loopback Networks
  The OSPF network type loopback is enabled by default for loopback interface and can be used only on loopback interfaces. The OSPF loopback network type states that the IP address is always advertised with a `\32` prefix length, even if the IP address configured on the loopback interface does not have a `\32` prefix length.

Advanced OSPF

- Areas
  An OSPF area is a logical rouping of routers or, more specifically, a logical grouping of router interfaces. Area membership is set at the interface level, and the area ID is included in the OSPF hello packet. An interface can belong to only one area. All routers within the same OSPF area maintiain an identical copy of the link-state database (LSDB).

  An OSPF area grows in size as network links and the number of routers increase in the area. While using a single area simplifies the topology, there are trade offs:

  – A full shortest path first (SPF) tree calculation runs when a link flaps within the area
  – The LSDB increases in size and becomes unmanageable if the network grows because the LSDB will consume more memory, and take longer during the SPF calculation process
  – No summarization of route information occurs

  Proper design addresses each of this issues by segmenting the OSPF routing domain into multiple OSPF areas, thereby keeping the LSDB to a manageable size. Sizing and design of OSPF networks should account for the hardware constraints of the smallest router in that area.

  If a router has interfaces in multiple LSDBs (one for each area). The internal topology of one area is invisible from outside that area. If a topology change occurs (such as a link flap or an additional network being added) within an area, all routers in the same OSPF area calculate the SPF tree again. Routers outside that area do not calculate the full SPF tree again but perform a partial SPF calculation if the metrics have changed or a prefix is removed.

  In essence, an OSPF area hides the topology from another area but enables the networks to be visible in otther areas within the OSPF domain. Segmenting the OSPF domain into multiple areas reduces the size of the LSDB for each area, making SPF tree calculations faster, and decreasing LSDB flooding between routers when a link flaps.

  Just because a router connects to multiple OSPF areas does not mean the routes from one area will be injected into another area.

  Area 0 is a special area called th ebackbone. By design, all areas must connect to Area 0 because OSPF expects all areas to inject routing information into the backbone, and Area 0 advertises the routes into other areas. The

backbone design is crucial to preventing routing loops.

Area border routers (ABRs) are OSPF routers connected to Area 0 and another OSPF area, per Cisco definition and according to RFC 3509. ABRs are responsible for advertising routes from one area and injecting them into a different OSPF area. Every ABR needs to participate in Area 0; otherwise, routes will not advertise into another area. ABRs compute an SPF tree for every area that they participate in.

- Area ID
  The area ID is a 32-bit field and can be formatted in simple decimal (0 through 4,284,967,295) or dotted decimal (0.0.0.0 through 255.255.255.255). During router configuration, the area can use decimal format on one router and dotted-decimal format on a different router, and the routers can still form an adjacency. OSPF advertises the area ID in dotted-decimal format in the OSPF hello packet.
- OSPF Route Types
  OSPF routes to destination networks within the same area are known as **intra-area routes.**

  OSPF routtes to destination networks from outside the OSPF domain that are injected into the OSPF domain through redistribution are known as external routes. External OSPF routes can come from a different OSPF domain or from a different routing protocol. External OSPF routes are beyond the scope of the CCNP and CCIE Enterprise Core ENCOR 350-401 exam and are not covered.

- Link-State Advertisements
  When OSPF neighbors become adjacent, the LSDBs synchronize between the OSPF routers. As an OSPF router adds or removes a directly connected network link to or from its database, the router floods the link-state advertisement (LSA) out all active OSPF interfaces. The OSPF LSA contains a complete list of networks advertised from that router.

  OSPF uses six LSA types for IPv4 routing:

  - **Type 1, router LSA**: Advertises the LSAs that originiate within an area
  - **Type 2, network LSA**: Advertises a multi-access network segment attached to a DR
  - **Type 3, summary LSA**: Advertises network prefixes that originated from a different area
  - **Type 4, ASBR summary LSA**: Advertises a summary LSA for a specific ASBR
  - **Type 5, AS external LSA**: Advertises LSAs for routes thatt have been redistributed
  - **Type 7, NSSA external LSA**: Advertises redistributed routes in NSSAs

  LSA types 1, 2, and 3, which are used for building the SPF tree for intra-area and inter-area routes, are explained here.

  - LSA Sequences
  - LSA Age and Flooding
  - LSA types
    - LSA Type 1: Router Link
    - LSA Type 2: Network Link
    - LSA Type 3: Summary Link

- Discontiguous Networks

- OSPF Path Selection

- Summarization of Routes
- Route Filtering

OSPFv3

-
-
-

COMMANDS:

-
-
-
-
-
-
-
-
-

BGP

-
-
-
-

Advanced BGP

-
-
-
-
-
-

COMMANDS:

-
-

-
-
-
-
-
-
-

Services

Overlay

Wireless

Architecture

Security

SDN

1.0 Architecture 15%

1.1 Explain the different design principles used in an enterprise network

- 1.1.a High-level enterprise network design such as 2-tier, 3-tier, fabric, and cloud
- 1.1.b High availability techniques such as redundancy, FHRP, and SSO

1.2 Describe wireless network design principles

- 1.2.a Wireless deployment models (centralized, distributed, controller-less, controller-based, cloud, remote branch)
- 1.2.b Location services in a WLAN design
- 1.2.c Client density

1.3 Explain the working principles of the Cisco SD-WAN solution

- 1.3.a SD-WAN control and data planes elements
- 1.3.b Benefits and limitations of SD-WAN solutions

1.4 Explain the working principles of the Cisco SD-Access solution

- 1.4.a SD-Access control and data planes elements
- 1.4.b Traditional campus interoperating with SD-Access

1.5 Interpret wired and wireless QoS configurations

- 1.5.a QoS components
- 1.5.b QoS policy

1.6 Describe hardware and software switching mechanisms such as CEF, CAM, TCAM, FIB, RIB, and adjacency tables

2.0 Virtualization 10%

2.1 Describe device virtualization technologies

- 2.1.a Hypervisor type 1 and 2
- 2.1.b Virtual machine
- 2.1.c Virtual switching

2.2 Configure and verify data path virtualization technologies

- 2.2.a VRF
- 2.2.b GRE and IPsec tunneling

2.3 Describe network virtualization concepts

- 2.3.a LISP
- 2.3.b VXLAN

3.0 Infrastructure 30%

3.1 Layer 2

- 3.1.a Troubleshoot static and dynamic 802.1q trunking protocols
- 3.1.b Troubleshoot static and dynamic EtherChannels
- 3.1.c Configure and verify common Spanning Tree Protocols (RSTP, MST) and Spanning Tree enhancements such as root guard and BPDU guard

3.2 Layer 3

- 3.2.a Compare routing concepts of EIGRP and OSPF (advanced distance vector vs. link state, load balancing, path selection, path operations, s, and area types)
- 3.2.b Configure simple OSPFv2/v3 environments, including multiple normal areas, summarization, and filtering (neighbor adjacency, point-to-point, and broadcast network types, and passive-interface)
- 3.2.c Configure and verify eBGP between directly connected neighbors (best path selection algorithm and neighbor relationships)
- 3.2.d Describe policy-based routing

3.3 Wireless

- 3.3.a Describe Layer 1 concepts, such as RF power, RSSI, SNR, interference, noise, bands, channels, and wireless client devices capabilities
- 3.3.b Describe AP modes and antenna types
- 3.3.c Describe access point discovery and join process (discovery algorithms, WLC selection process)
- 3.3.d Describe the main principles and use cases for Layer 2 and Layer 3 roaming
- 3.3.e Troubleshoot WLAN configuration and wireless client connectivity issues using GUI only
- 3.3.f Describe wireless segmentation with groups, profiles, and tags

3.4 IP Services

- 3.4.a Interpret network time protocol configurations such as NTP and PTP
- 3.4.b Configure NAT/PAT
- 3.4.c Configure first hop redundancy protocols, such as HSRP, VRRP
- 3.4.d Describe multicast protocols, such as RPF check, PIM and IGMP v2/v3

4.0 Network Assurance 10%

4.1 Diagnose network problems using tools such as debugs, conditional debugs, traceroute, ping, SNMP, and syslog
4.2 Configure and verify Flexible NetFlow
4.3 Configure SPAN/RSPAN/ERSPAN
4.4 Configure and verify IPSLA
4.5 Describe Cisco DNA Center workflows to apply network configuration, monitoring, and management
4.6 Configure and verify NETCONF and RESTCONF

5.0 Security 20%

5.1 Configure and verify device access control

- 5.1.a Lines and local user authentication
- 5.1.b Authentication and authorization using AAA

5.2 Configure and verify infrastructure security features

- 5.2.a ACLs
- 5.2.b CoPP

5.3 Describe REST API security
5.4 Configure and verify wireless security features

- 5.4.a 802.1X
- 5.4.b WebAuth
- 5.4.c PSK
- 5.4.d EAPOL (4-way handshake)

5.5 Describe the components of network security design

- 5.5.a Threat defense
- 5.5.b Endpoint security
- 5.5.c Next-generation firewall
- 5.5.d TrustSec and MACsec
- 5.5.e Network access control with 802.1X, MAB, and WebAuth

6.0 Automation 15%

6.1 Interpret basic Python components and scripts
6.2 Construct valid JSON-encoded files
6.3 Describe the high-level principles and benefits of a data modeling language, such as YANG
6.4 Describe APIs for Cisco DNA Center and vManage
6.5 Interpret REST API response codes and results in payload using Cisco DNA Center and RESTCONF
6.6 Construct an EEM applet to automate configuration, troubleshooting, or data collection
6.7 Compare agent vs. agentless orchestration tools, such as Chef, Puppet, Ansible, and SaltStack