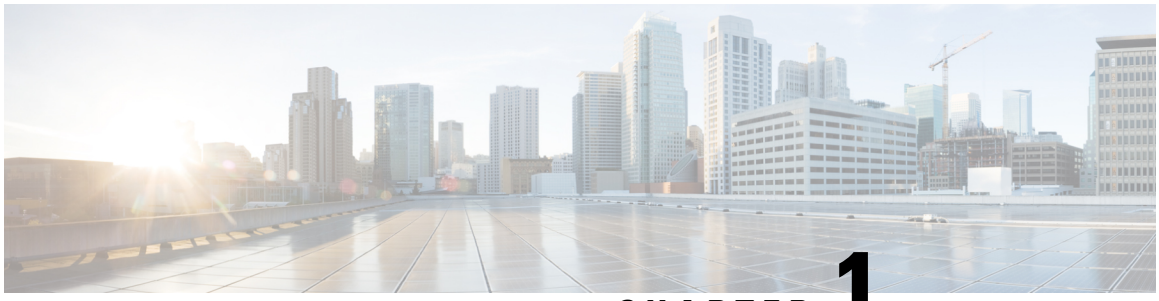
Layer 2 Configuration Guide, Cisco IOS XE Gibraltar 16.10.x (Catalyst 9400 Switches)

First Published: 2018-12-10

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

CONTENTS



CHAPTER 1

Configuring Spanning Tree Protocol

This chapter describes how to configure the Spanning Tree Protocol (STP) on port-based VLANs on the Catalyst devices. The device can use either the per-VLAN spanning-tree plus (PVST+) protocol based on the IEEE 802.1D standard and Cisco proprietary extensions, or the rapid per-VLAN spanning-tree plus (rapid-PVST+) protocol based on the IEEE 802.1w standard. A switch stack appears as a single spanning-tree node to the rest of the network, and all stack members use the same bridge ID.

Restrictions for STP

- An attempt to configure a device as the root device fails if the value necessary to be the root device is less than 1.
- If your network consists of devices that support and do not support the extended system ID, it is unlikely that the device with the extended system ID support will become the root device. The extended system ID increases the device priority value every time the VLAN number is greater than the priority of the connected devices running older software.
- The root device for each spanning-tree instance should be a backbone or distribution device. Do not configure an access device as the spanning-tree primary root.

Information About Spanning Tree Protocol

Spanning Tree Protocol

Spanning Tree Protocol (STP) is a Layer 2 link management protocol that provides path redundancy while preventing loops in the network. For a Layer 2 Ethernet network to function properly, only one active path can exist between any two stations. Multiple active paths among end stations cause loops in the network. If a loop exists in the network, end stations might receive duplicate messages. Devices might also learn end-station

MAC addresses on multiple Layer 2 interfaces. These conditions result in an unstable network. Spanning-tree operation is transparent to end stations, which cannot detect whether they are connected to a single LAN segment or a switched LAN of multiple segments.

The STP uses a spanning-tree algorithm to select one device of a redundantly connected network as the root of the spanning tree. The algorithm calculates the best loop-free path through a switched Layer 2 network by assigning a role to each port based on the role of the port in the active topology:

- Root—A forwarding port elected for the spanning-tree topology
- Designated—A forwarding port elected for every switched LAN segment
- Alternate—A blocked port providing an alternate path to the root bridge in the spanning tree
- Backup—A blocked port in a loopback configuration

The device that has *all* of its ports as the designated role or as the backup role is the root device. The device that has at least *one* of its ports in the designated role is called the designated device.

Spanning tree forces redundant data paths into a standby (blocked) state. If a network segment in the spanning tree fails and a redundant path exists, the spanning-tree algorithm recalculates the spanning-tree topology and activates the standby path. Devices send and receive spanning-tree frames, called bridge protocol data units (BPDUs), at regular intervals. The devices do not forward these frames but use them to construct a loop-free path. BPDUs contain information about the sending device and its ports, including device and MAC addresses, device priority, port priority, and path cost. Spanning tree uses this information to elect the root device and root port for the switched network and the root port and designated port for each switched segment.

When two ports on a device are part of a loop, the spanning-tree and path cost settings control which port is put in the forwarding state and which is put in the blocking state. The spanning-tree port priority value represents the location of a port in the network topology and how well it is located to pass traffic. The path cost value represents the media speed.

Note The short path cost method is the default STP path cost method.

Note By default, the device sends keepalive messages (to ensure the connection is up) only on interfaces that do not have small form-factor pluggable (SFP) modules. You can change the default for an interface by entering the [**no**] **keepalive** interface configuration command with no keywords.

Note In addition to STP, the device uses keepalive messages to detect loops. By default, keepalive is enabled on Layer 2 ports. To disable keepalive, use the **no keepalive** command in interface configuration mode.

Spanning-Tree Topology and BPDUs

The stable, active spanning-tree topology of a switched network is controlled by these elements:

- The unique bridge ID (device priority and MAC address) associated with each VLAN on each device. In a device stack, all devices use the same bridge ID for a given spanning-tree instance.

- The spanning-tree path cost to the root device.
- The port identifier (port priority and MAC address) associated with each Layer 2 interface.

When the devices in a network are powered up, each functions as the root device. Each device sends a configuration BPDU through all of its ports. The BPDUs communicate and compute the spanning-tree topology. Each configuration BPDU contains this information:

- The unique bridge ID of the device that the sending device identifies as the root device
- The spanning-tree path cost to the root
- The bridge ID of the sending device
- Message age
- The identifier of the sending interface
- Values for the hello, forward delay, and max-age protocol timers

When a device receives a configuration BPDU that contains *superior* information (lower bridge ID, lower path cost, and so forth), it stores the information for that port. If this BPDU is received on the root port of the device, the device also forwards it with an updated message to all attached LANs for which it is the designated device.

If a device receives a configuration BPDU that contains *inferior* information to that currently stored for that port, it discards the BPDU. If the device is a designated device for the LAN from which the inferior BPDU was received, it sends that LAN a BPDU containing the up-to-date information stored for that port. In this way, inferior information is discarded, and superior information is propagated on the network.

A BPDU exchange results in these actions:

- One device in the network is elected as the root device (the logical center of the spanning-tree topology in a switched network). See the figure following the bullets.

  For each VLAN, the device with the highest device priority (the lowest numerical priority value) is elected as the root device. If all devices are configured with the default priority (32768), the device with the lowest MAC address in the VLAN becomes the root device. The device priority value occupies the most significant bits of the bridge ID, as shown in the following figure.

- A root port is selected for each device (except the root device). This port provides the best path (lowest cost) when the device forwards packets to the root device.

  When selecting the root port on a device stack, spanning tree follows this sequence:

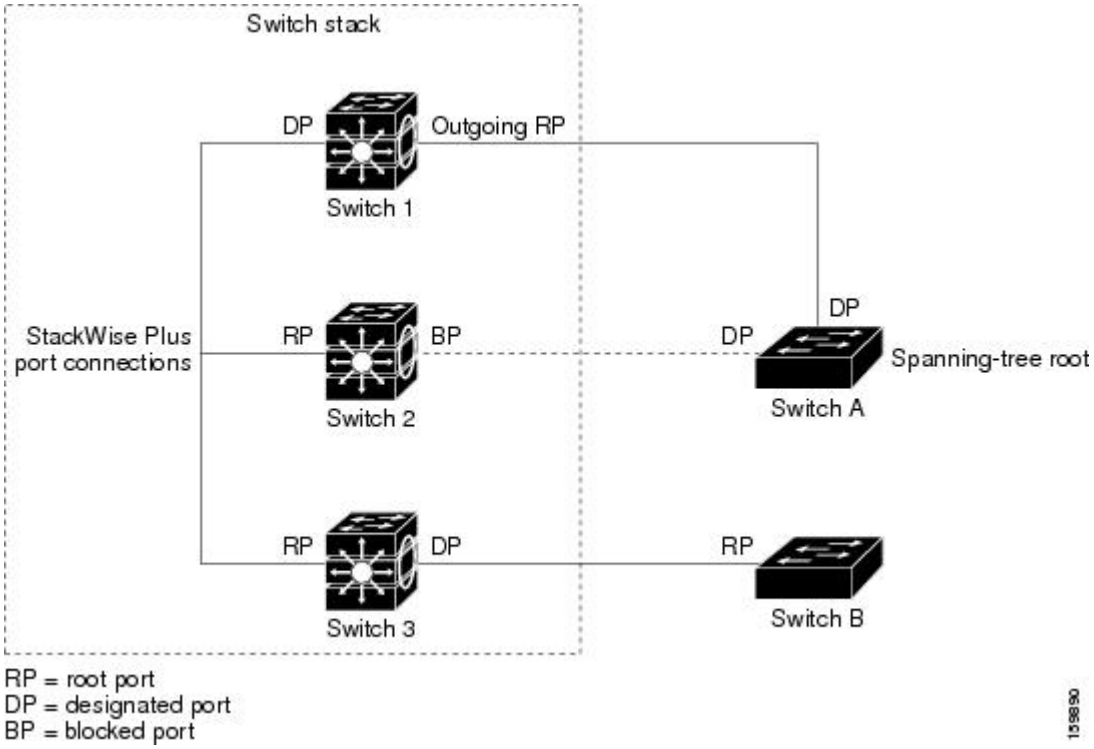
  - Selects the lowest root bridge ID
  - Selects the lowest path cost to the root device
  - Selects the lowest designated bridge ID
  - Selects the lowest designated path cost
  - Selects the lowest port ID

- Only one outgoing port on the stack root device is selected as the root port. The remaining devices in the stack become its designated devices (Device 2 and Device 3) as shown in the following figure.
- The shortest distance to the root device is calculated for each device based on the path cost.

- A designated device for each LAN segment is selected. The designated device incurs the lowest path cost when forwarding packets from that LAN to the root device. The port through which the designated device is attached to the LAN is called the designated port.

Figure 1: Spanning-Tree Port States in a Device Stack

One stack member is elected as the stack root device. The stack root device contains the outgoing root port (Device 1).



All paths that are not needed to reach the root device from anywhere in the switched network are placed in the spanning-tree blocking mode.

Bridge ID, Device Priority, and Extended System ID

The IEEE 802.1D standard requires that each device has an unique bridge identifier (bridge ID), which controls the selection of the root device. Because each VLAN is considered as a different *logical bridge* with PVST+ and Rapid PVST+, the same device must have a different bridge ID for each configured VLAN. Each VLAN on the device has a unique 8-byte bridge ID. The 2 most-significant bytes are used for the device priority, and the remaining 6 bytes are derived from the device MAC address.

The 2 bytes previously used for the device priority are reallocated into a 4-bit priority value and a 12-bit extended system ID value equal to the VLAN ID.

Table 1: Device Priority Value and Extended System ID

Priority Value				**Extended System ID (Set Equal to the VLAN ID)**										
Bit 16	**Bit 15**	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit**
32768	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2

Spanning tree uses the extended system ID, the device priority, and the allocated spanning-tree MAC address to make the bridge ID unique for each VLAN.

Support for the extended system ID affects how you manually configure the root device, the secondary root device, and the device priority of a VLAN. For example, when you change the device priority value, you change the probability that the device will be elected as the root device. Configuring a higher value decreases the probability; a lower value increases the probability.

If any root device for the specified VLAN has a device priority lower than 24576, the device sets its own priority for the specified VLAN to 4096 less than the lowest device priority. 4096 is the value of the least-significant bit of a 4-bit device priority value as shown in the table.

Port Priority Versus Path Cost

If a loop occurs, spanning tree uses port priority when selecting an interface to put into the forwarding state. You can assign higher priority values (lower numerical values) to interfaces that you want selected first and lower priority values (higher numerical values) that you want selected last. If all interfaces have the same priority value, spanning tree puts the interface with the lowest interface number in the forwarding state and blocks the other interfaces.

The spanning-tree path cost default value is derived from the media speed of an interface. If a loop occurs, spanning tree uses cost when selecting an interface to put in the forwarding state. You can assign lower cost values to interfaces that you want selected first and higher cost values that you want selected last. If all interfaces have the same cost value, spanning tree puts the interface with the lowest interface number in the forwarding state and blocks the other interfaces.

If your device is a member of a device stack, you must assign lower cost values to interfaces that you want selected first and higher cost values that you want selected last instead of adjusting its port priority. For details, see Related Topics.

Spanning-Tree Interface States

Propagation delays can occur when protocol information passes through a switched LAN. As a result, topology changes can take place at different times and at different places in a switched network. When an interface transitions directly from nonparticipation in the spanning-tree topology to the forwarding state, it can create temporary data loops. Interfaces must wait for new topology information to propagate through the switched LAN before starting to forward frames. They must allow the frame lifetime to expire for forwarded frames that have used the old topology.

Each Layer 2 interface on a device using spanning tree exists in one of these states:

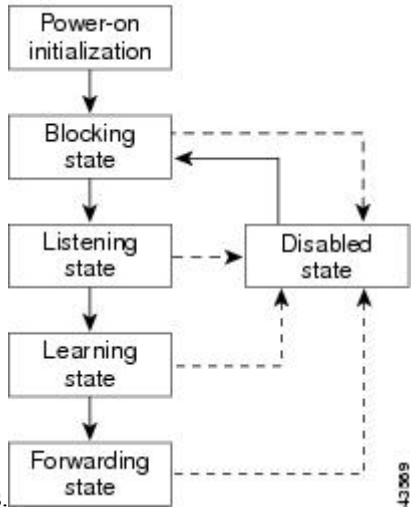
- Blocking—The interface does not participate in frame forwarding.
- Listening—The first transitional state after the blocking state when the spanning tree decides that the interface should participate in frame forwarding.
- Learning—The interface prepares to participate in frame forwarding.
- Forwarding—The interface forwards frames.
- Disabled—The interface is not participating in spanning tree because of a shutdown port, no link on the port, or no spanning-tree instance running on the port.

An interface moves through these states:

- From initialization to blocking

- From blocking to listening or to disabled
- From listening to learning or to disabled
- From learning to forwarding or to disabled
- From forwarding to disabled

Figure 2: Spanning-Tree Interface States



An interface moves through the states.

When you power up the device, spanning tree is enabled by default, and every interface in the device, VLAN, or network goes through the blocking state and the transitory states of listening and learning. Spanning tree stabilizes each interface at the forwarding or blocking state.

When the spanning-tree algorithm places a Layer 2 interface in the forwarding state, this process occurs:

1. The interface is in the listening state while spanning tree waits for protocol information to move the interface to the blocking state.
2. While spanning tree waits for the forward-delay timer to expire, it moves the interface to the learning state and resets the forward-delay timer.
3. In the learning state, the interface continues to block frame forwarding as the device learns end-station location information for the forwarding database.
4. When the forward-delay timer expires, spanning tree moves the interface to the forwarding state, where both learning and frame forwarding are enabled.

Blocking State

A Layer 2 interface in the blocking state does not participate in frame forwarding. After initialization, a BPDU is sent to each device interface. A device initially functions as the root until it exchanges BPDUs with other devices. This exchange establishes which device in the network is the root or root device. If there is only one device in the network, no exchange occurs, the forward-delay timer expires, and the interface moves to the listening state. An interface always enters the blocking state after device initialization.

An interface in the blocking state performs these functions:

- Discards frames received on the interface

- Discards frames switched from another interface for forwarding
- Does not learn addresses
- Receives BPDUs

Listening State

The listening state is the first state a Layer 2 interface enters after the blocking state. The interface enters this state when the spanning tree decides that the interface should participate in frame forwarding.

An interface in the listening state performs these functions:

- Discards frames received on the interface
- Discards frames switched from another interface for forwarding
- Does not learn addresses
- Receives BPDUs

Learning State

A Layer 2 interface in the learning state prepares to participate in frame forwarding. The interface enters the learning state from the listening state.

An interface in the learning state performs these functions:

- Discards frames received on the interface
- Discards frames switched from another interface for forwarding
- Learns addresses
- Receives BPDUs

Forwarding State

A Layer 2 interface in the forwarding state forwards frames. The interface enters the forwarding state from the learning state.

An interface in the forwarding state performs these functions:

- Receives and forwards frames received on the interface
- Forwards frames switched from another interface
- Learns addresses
- Receives BPDUs

Disabled State

A Layer 2 interface in the disabled state does not participate in frame forwarding or in the spanning tree. An interface in the disabled state is nonoperational.

A disabled interface performs these functions:

- Discards frames received on the interface

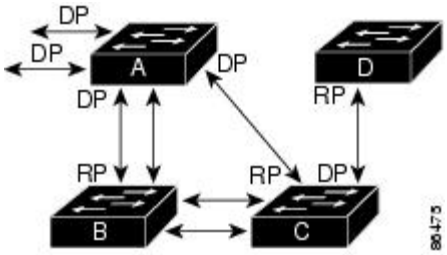
- Discards frames switched from another interface for forwarding
- Does not learn addresses
- Does not receive BPDUs

How a Device or Port Becomes the Root Device or Root Port

If all devices in a network are enabled with default spanning-tree settings, the device with the lowest MAC address becomes the root device.

Figure 3: Spanning-Tree Topology

Device A is elected as the root device because the device priority of all the devices is set to the default (32768) and Device A has the lowest MAC address. However, because of traffic patterns, number of forwarding interfaces, or link types, Device A might not be the ideal root device. By increasing the priority (lowering the numerical value) of the ideal device so that it becomes the root device, you force a spanning-tree recalculation



to form a new topology with the ideal device as the root.

When the spanning-tree topology is calculated based on default parameters, the path between source and destination end stations in a switched network might not be ideal. For instance, connecting higher-speed links to an interface that has a higher number than the root port can cause a root-port change. The goal is to make the fastest link the root port.

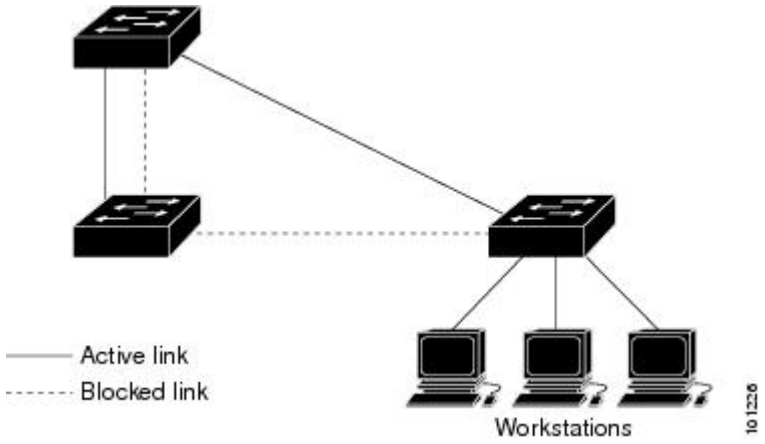
For example, assume that one port on Device B is a Gigabit Ethernet link and that another port on Device B (a 10/100 link) is the root port. Network traffic might be more efficient over the Gigabit Ethernet link. By changing the spanning-tree port priority on the Gigabit Ethernet port to a higher priority (lower numerical value) than the root port, the Gigabit Ethernet port becomes the new root port.

Spanning Tree and Redundant Connectivity

Figure 4: Spanning Tree and Redundant Connectivity

You can create a redundant backbone with spanning tree by connecting two device interfaces to another device or to two different devices. Spanning tree automatically disables one interface but enables it if the other one fails. If one link is high-speed and the other is low-speed, the low-speed link is always disabled. If the speeds

are the same, the port priority and port ID are added together, and spanning tree disables the link with the highest value.



You can also create redundant links between devices by using EtherChannel groups.

Spanning-Tree Address Management

IEEE 802.1D specifies 17 multicast addresses, ranging from 0x00180C2000000 to 0x0180C2000010, to be used by different bridge protocols. These addresses are static addresses that cannot be removed.

Regardless of the spanning-tree state, each device in the stack receives but does not forward packets destined for addresses between 0x0180C2000000 and 0x0180C200000F.

If spanning tree is enabled, the CPU on the device or on each device in the stack receives packets destined for 0x0180C2000000 and 0x0180C2000010. If spanning tree is disabled, the device or each device in the stack forwards those packets as unknown multicast addresses.

Accelerated Aging to Retain Connectivity

The default for aging dynamic addresses is 5 minutes, the default setting of the **mac address-table aging-time** global configuration command. However, a spanning-tree reconfiguration can cause many station locations to change. Because these stations could be unreachable for 5 minutes or more during a reconfiguration, the address-aging time is accelerated so that station addresses can be dropped from the address table and then relearned. The accelerated aging is the same as the forward-delay parameter value (**spanning-tree vlan** *vlan-id* **forward-time** *seconds* global configuration command) when the spanning tree reconfigures.

Because each VLAN is a separate spanning-tree instance, the device accelerates aging on a per-VLAN basis. A spanning-tree reconfiguration on one VLAN can cause the dynamic addresses learned on that VLAN to be subject to accelerated aging. Dynamic addresses on other VLANs can be unaffected and remain subject to the aging interval entered for the device.

Spanning-Tree Modes and Protocols

The device supports these spanning-tree modes and protocols:

- PVST+—This spanning-tree mode is based on the IEEE 802.1D standard and Cisco proprietary extensions. The PVST+ runs on each VLAN on the device up to the maximum supported, ensuring that each has a loop-free path through the network.

  The PVST+ provides Layer 2 load-balancing for the VLAN on which it runs. You can create different logical topologies by using the VLANs on your network to ensure that all of your links are used but that no one link is oversubscribed. Each instance of PVST+ on a VLAN has a single root device. This root

device propagates the spanning-tree information associated with that VLAN to all other devices in the network. Because each device has the same information about the network, this process ensures that the network topology is maintained.

- Rapid PVST+—Rapid PVST+ is the default STP mode on your device.This spanning-tree mode is the same as PVST+ except that is uses a rapid convergence based on the IEEE 802.1w standard. . To provide rapid convergence, the Rapid PVST+ immediately deletes dynamically learned MAC address entries on a per-port basis upon receiving a topology change. By contrast, PVST+ uses a short aging time for dynamically learned MAC address entries.

  Rapid PVST+ uses the same configuration as PVST+ (except where noted), and the device needs only minimal extra configuration. The benefit of Rapid PVST+ is that you can migrate a large PVST+ install base to Rapid PVST+ without having to learn the complexities of the Multiple Spanning Tree Protocol (MSTP) configuration and without having to reprovision your network. In Rapid PVST+ mode, each VLAN runs its own spanning-tree instance up to the maximum supported.

- MSTP—This spanning-tree mode is based on the IEEE 802.1s standard. You can map multiple VLANs to the same spanning-tree instance, which reduces the number of spanning-tree instances required to support a large number of VLANs. The MSTP runs on top of the RSTP (based on IEEE 802.1w), which provides for rapid convergence of the spanning tree by eliminating the forward delay and by quickly transitioning root ports and designated ports to the forwarding state. In a device stack, the cross-stack rapid transition (CSRT) feature performs the same function as RSTP. You cannot run MSTP without RSTP or CSRT.

Supported Spanning-Tree Instances

In MSTP mode, the device or device stack supports up to 65 MST instances. The number of VLANs that can be mapped to a particular MST instance is unlimited.

Spanning-Tree Interoperability and Backward Compatibility

In a mixed MSTP and PVST+ network, the common spanning-tree (CST) root must be inside the MST backbone, and a PVST+ device cannot connect to multiple MST regions.

When a network contains devices running Rapid PVST+ and devices running PVST+, we recommend that the Rapid PVST+ devices and PVST+ devices be configured for different spanning-tree instances. In the Rapid PVST+ spanning-tree instances, the root device must be a Rapid PVST+ device. In the PVST+ instances, the root device must be a PVST+ device. The PVST+ devices should be at the edge of the network.

All stack members run the same version of spanning tree (all PVST+, all Rapid PVST+, or all MSTP).

Table 2: PVST+, MSTP, and Rapid-PVST+ Interoperability and Compatibility

	PVST+	**MSTP**	**Rapid PVST+**
PVST+	Yes	Yes (with restrictions)	Yes (reverts to PVST+)
MSTP	Yes (with restrictions)	Yes	Yes (reverts to PVST+)
Rapid PVST+	Yes (reverts to PVST+)	Yes (reverts to PVST+)	Yes

STP and IEEE 802.1Q Trunks

The IEEE 802.1Q standard for VLAN trunks imposes some limitations on the spanning-tree strategy for a network. The standard requires only one spanning-tree instance for *all* VLANs allowed on the trunks. However, in a network of Cisco devices connected through IEEE 802.1Q trunks, the devices maintain one spanning-tree instance for *each* VLAN allowed on the trunks.

When you connect a Cisco device to a non-Cisco device through an IEEE 802.1Q trunk, the Cisco device uses PVST+ to provide spanning-tree interoperability. If Rapid PVST+ is enabled, the device uses it instead of PVST+. The device combines the spanning-tree instance of the IEEE 802.1Q VLAN of the trunk with the spanning-tree instance of the non-Cisco IEEE 802.1Q device.

However, all PVST+ or Rapid PVST+ information is maintained by Cisco devices separated by a cloud of non-Cisco IEEE 802.1Q devices. The non-Cisco IEEE 802.1Q cloud separating the Cisco devices is treated as a single trunk link between the devices.

Rapid PVST+ is automatically enabled on IEEE 802.1Q trunks, and no user configuration is required. The external spanning-tree behavior on access ports and Inter-Switch Link (ISL) trunk ports is not affected by PVST+.

Spanning Tree and Device Stacks

When the device stack is operating in PVST+ or Rapid PVST+ mode:

- A device stack appears as a single spanning-tree node to the rest of the network, and all stack members use the same bridge ID for a given spanning tree. The bridge ID is derived from the MAC address of the active stack.
- When a new device joins the stack, it sets its bridge ID to the active stack bridge ID. If the newly added device has the lowest ID and if the root path cost is the same among all stack members, the newly added device becomes the stack root.
- When a stack member leaves the stack, spanning-tree reconvergence occurs within the stack (and possibly outside the stack). The remaining stack member with the lowest stack port ID becomes the stack root.
- If a neighboring device external to the device stack fails or is powered down, normal spanning-tree processing occurs. Spanning-tree reconvergence might occur as a result of losing a device in the active topology.
- If a new device external to the device stack is added to the network, normal spanning-tree processing occurs. Spanning-tree reconvergence might occur as a result of adding a device in the network.

Default Spanning-Tree Configuration

Table 3: Default Spanning-Tree Configuration

Feature	Default Setting
Enable state	Enabled on VLAN 1.
Spanning-tree mode	Rapid PVST+ (PVST+ and MS disabled.)
Device priority	32768

Feature	Default Setting
Spanning-tree port priority (configurable on a per-interface basis)	128
Spanning-tree port cost (configurable on a per-interface basis)	1000 Mb/s: 4 100 Mb/s: 19 10 Mb/s: 100
Spanning-tree VLAN port priority (configurable on a per-VLAN basis)	128
Spanning-tree VLAN port cost (configurable on a per-VLAN basis)	1000 Mb/s: 4 100 Mb/s: 19 10 Mb/s: 100
Spanning-tree timers	Hello time: 2 seconds Forward-delay time: 15 seconds Maximum-aging time: 20 seconds Transmit hold count: 6 BPDUs

Note Beginning in Cisco IOS Release 15.2(4)E, the default STP mode is Rapid PVST+.

How to Configure Spanning-Tree Features

Changing the Spanning-Tree Mode

The switch supports three spanning-tree modes: per-VLAN spanning tree plus (PVST+), Rapid PVST+, or multiple spanning tree protocol (MSTP). By default, the device runs the Rapid PVST+ protocol.

If you want to enable a mode that is different from the default mode, this procedure is required.

SUMMARY STEPS

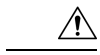
1. **enable**
2. **configure terminal**
3. **spanning-tree mode** {**pvst** | **mst** | **rapid-pvst**}
4. **interface** *interface-id*
5. **spanning-tree link-type point-to-point**
6. **end**
7. **clear spanning-tree detected-protocols**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree mode** {**pvst** | **mst** | **rapid-pvst**} **Example:** `Device(config)# spanning-tree mode pvst`	Configures a spanning-tree mode. All stack members run the same version of spanning tree. • Select **pvst** to enable PVST+. • Select **mst** to enable MSTP. • Select **rapid-pvst** to enable rapid PVST+.
Step 4	**interface** *interface-id* **Example:** `Device(config)# interface GigabitEthernet1/0/1`	Specifies an interface to configure, and enters interface configuration mode. Valid interfaces include physical ports, VLANs, and port channels. The VLAN ID range is 1 to 4094. The port-channel range is 1 to 48.
Step 5	**spanning-tree link-type point-to-point** **Example:** `Device(config-if)# spanning-tree link-type point-to-point`	Specifies that the link type for this port is point-to-point. If you connect this port (local port) to a remote port through a point-to-point link and the local port becomes a designated port, the device negotiates with the remote port and rapidly changes the local port to the forwarding state.
Step 6	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 7	**clear spanning-tree detected-protocols** **Example:** `Device# clear spanning-tree detected-protocols`	If any port on the device is connected to a port on a legacy IEEE 802.1D device, this command restarts the protocol migration process on the entire device. This step is optional if the designated device detects that this device is running rapid PVST+.

Disabling Spanning Tree

Spanning tree is enabled by default on VLAN 1 and on all newly created VLANs up to the spanning-tree limit. Disable spanning tree only if you are sure there are no loops in the network topology.



Caution When spanning tree is disabled and loops are present in the topology, excessive traffic and indefinite packet duplication can drastically reduce network performance.

This procedure is optional.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no spanning-tree vlan** *vlan-id*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no spanning-tree vlan** *vlan-id* **Example:** `Device(config)# no spanning-tree vlan 300`	For *vlan-id*, the range is 1 to 4094.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring the Root Device

To configure a device as the root for the specified VLAN, use the **spanning-tree vlan** *vlan-id* **root** global configuration command to modify the device priority from the default value (32768) to a significantly lower

value. When you enter this command, the software checks the device priority of the root devices for each VLAN. Because of the extended system ID support, the device sets its own priority for the specified VLAN to 24576 if this value will cause this device to become the root for the specified VLAN.

Use the **diameter** keyword to specify the Layer 2 network diameter (that is, the maximum number of device hops between any two end stations in the Layer 2 network). When you specify the network diameter, the device automatically sets an optimal hello time, forward-delay time, and maximum-age time for a network of that diameter, which can significantly reduce the convergence time. You can use the **hello** keyword to override the automatically calculated hello time.

This procedure is optional.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spanning-tree vlan** *vlan-id* **root primary** [**diameter** *net-diameter*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree vlan** *vlan-id* **root primary** [**diameter** *net-diameter* **Example:** `Device(config)# spanning-tree vlan 20-24 root primary diameter 4`	Configures a device to become the root for the specified VLAN. • For *vlan-id*, you can specify a single VLAN identified by VLAN ID number, a range of VLANs separated by a hyphen, or a series of VLANs separated by a comma. The range is 1 to 4094. • (Optional) For **diameter** *net-diameter*, specify the maximum number of devices between any two end stations. The range is 2 to 7.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

What to do next

After configuring the device as the root device, we recommend that you avoid manually configuring the hello time, forward-delay time, and maximum-age time through the **spanning-tree vlan** *vlan-id* **hello-time**, **spanning-tree vlan** *vlan-id* **forward-time**, and the **spanning-tree vlan** *vlan-id* **max-age** global configuration commands.

Configuring a Secondary Root Device

When you configure a device as the secondary root, the device priority is modified from the default value (32768) to 28672. With this priority, the device is likely to become the root device for the specified VLAN if the primary root device fails. This is assuming that the other network devices use the default device priority of 32768, and therefore, are unlikely to become the root device.

You can execute this command on more than one device to configure multiple backup root devices. Use the same network diameter and hello-time values that you used when you configured the primary root device with the **spanning-tree vlan** *vlan-id* **root primary** global configuration command.

This procedure is optional.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spanning-tree vlan** *vlan-id* **root secondary** [**diameter** *net-diameter*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree vlan** *vlan-id* **root secondary** [**diameter** *net-diameter* **Example:** `Device(config)# spanning-tree vlan 20-24 root secondary diameter 4`	Configures a device to become the secondary root for the specified VLAN. • For *vlan-id*, you can specify a single VLAN identified by VLAN ID number, a range of VLANs separated by a hyphen, or a series of VLANs separated by a comma. The range is 1 to 4094.

	Command or Action	Purpose
		• (Optional) For **diameter** *net-diameter*, specify the maximum number of devices between any two end stations. The range is 2 to 7. Use the same network diameter value that you used when configuring the primary root device.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring Port Priority

Note If your device is a member of a device stack, you must use the **spanning-tree** [**vlan** *vlan-id*] **cost** *cost* interface configuration command instead of the **spanning-tree** [**vlan** *vlan-id*] **port-priority** *priority* interface configuration command to select an interface to put in the forwarding state. Assign lower cost values to interfaces that you want selected first and higher cost values that you want selected last.

This procedure is optional.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *interface-id*
4. **spanning-tree port-priority** *priority*
5. **spanning-tree vlan** *vlan-id* **port-priority** *priority*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/0/2`	Specifies an interface to configure, and enters interface configuration mode. Valid interfaces include physical ports and port-channel logical interfaces (**port-channel** *port-channel-number*).
Step 4	**spanning-tree port-priority** *priority* **Example:** `Device(config-if)# spanning-tree port-priority 0`	Configures the port priority for an interface. For *priority*, the range is 0 to 240, in increments of 16; the default is 128. Valid values are 0, 16, 32, 48, 64, 80, 96, 112, 128, 144, 160, 176, 192, 208, 224, and 240. All other values are rejected. The lower the number, the higher the priority.
Step 5	**spanning-tree vlan** *vlan-id* **port-priority** *priority* **Example:** `Device(config-if)# spanning-tree vlan 20-25 port-priority 0`	Configures the port priority for a VLAN. • For *vlan-id*, you can specify a single VLAN identified by VLAN ID number, a range of VLANs separated by a hyphen, or a series of VLANs separated by a comma. The range is 1 to 4094. • For *priority*, the range is 0 to 240, in increments of 16; the default is 128. Valid values are 0, 16, 32, 48, 64, 80, 96, 112, 128, 144, 160, 176, 192, 208, 224, and 240. All other values are rejected. The lower the number, the higher the priority.
Step 6	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Configuring Path Cost

This procedure is optional.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *interface-id*
4. **spanning-tree cost** *cost*
5. **spanning-tree vlan** *vlan-id* **cost** *cost*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/0/1`	Specifies an interface to configure, and enters interface configuration mode. Valid interfaces include physical ports and port-channel logical interfaces (**port-channel** *port-channel-number*).
Step 4	**spanning-tree cost** *cost* **Example:** `Device(config-if)# spanning-tree cost 250`	Configures the cost for an interface. If a loop occurs, spanning tree uses the path cost when selecting an interface to place into the forwarding state. A lower path cost represents higher-speed transmission. For *cost*, the range is 1 to 200000000; the default value is derived from the media speed of the interface.
Step 5	**spanning-tree vlan** *vlan-id* **cost** *cost* **Example:** `Device(config-if)# spanning-tree vlan 10,12-15,20 cost 300`	Configures the cost for a VLAN. If a loop occurs, spanning tree uses the path cost when selecting an interface to place into the forwarding state. A lower path cost represents higher-speed transmission. • For *vlan-id*, you can specify a single VLAN identified by VLAN ID number, a range of VLANs separated by a hyphen, or a series of VLANs separated by a comma. The range is 1 to 4094. • For *cost*, the range is 1 to 200000000; the default value is derived from the media speed of the interface.
Step 6	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

The **show spanning-tree interface** *interface-id* privileged EXEC command displays information only for ports that are in a link-up operative state. Otherwise, you can use the **show running-config** privileged EXEC command to confirm the configuration.

Configuring the Device Priority of a VLAN

You can configure the device priority and make it more likely that a standalone device or a device in the stack will be chosen as the root device.

Note Exercise care when using this command. For most situations, we recommend that you use the **spanning-tree vlan** *vlan-id* **root primary** and the **spanning-tree vlan** *vlan-id* **root secondary** global configuration commands to modify the device priority.

This procedure is optional.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spanning-tree vlan** *vlan-id* **priority** *priority*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree vlan** *vlan-id* **priority** *priority* **Example:** `Device(config)# spanning-tree vlan 20 priority 8192`	Configures the device priority of a VLAN. • For *vlan-id*, you can specify a single VLAN identified by VLAN ID number, a range of VLANs separated by a hyphen, or a series of VLANs separated by a comma. The range is 1 to 4094. • For *priority*, the range is 0 to 61440 in increments of 4096; the default is 32768. The lower the number, the more likely the device will be chosen as the root device. Valid priority values are 4096, 8192, 12288, 16384, 20480, 24576, 28672, 32768, 36864, 40960, 45056, 49152, 53248, 57344, and 61440. All other values are rejected.

	Command or Action	Purpose
Step 4	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Configuring the Hello Time

The hello time is the time interval between configuration messages generated and sent by the root device.

This procedure is optional.

SUMMARY STEPS

1. **enable**
2. **spanning-tree vlan** *vlan-id* **hello-time** *seconds*
3. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**spanning-tree vlan** *vlan-id* **hello-time** *seconds* **Example:** `Device(config)# spanning-tree vlan 20-24 hello-time 3`	Configures the hello time of a VLAN. The hello time is the time interval between configuration messages generated and sent by the root device. These messages mean that the device is alive. • For *vlan-id*, you can specify a single VLAN identified by VLAN ID number, a range of VLANs separated by a hyphen, or a series of VLANs separated by a comma. The range is 1 to 4094. • For *seconds*, the range is 1 to 10; the default is 2.
Step 3	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Configuring the Forwarding-Delay Time for a VLAN

This procedure is optional.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spanning-tree vlan** *vlan-id* **forward-time** *seconds*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree vlan** *vlan-id* **forward-time** *seconds* **Example:** `Device(config)# spanning-tree vlan 20,25 forward-time 18`	Configures the forward time of a VLAN. The forwarding delay is the number of seconds an interface waits before changing from its spanning-tree learning and listening states to the forwarding state. • For *vlan-id*, you can specify a single VLAN identified by VLAN ID number, a range of VLANs separated by a hyphen, or a series of VLANs separated by a comma. The range is 1 to 4094. • For *seconds*, the range is 4 to 30; the default is 15.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring the Maximum-Aging Time for a VLAN

This procedure is optional.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spanning-tree vlan** *vlan-id* **max-age** *seconds*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree vlan** *vlan-id* **max-age** *seconds* **Example:** `Device(config)# spanning-tree vlan 20 max-age 30`	Configures the maximum-aging time of a VLAN. The maximum-aging time is the number of seconds a device waits without receiving spanning-tree configuration messages before attempting a reconfiguration. • For *vlan-id*, you can specify a single VLAN identified by VLAN ID number, a range of VLANs separated by a hyphen, or a series of VLANs separated by a comma. The range is 1 to 4094. • For *seconds*, the range is 6 to 40; the default is 20.
Step 4	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Configuring the Transmit Hold-Count

You can configure the BPDU burst size by changing the transmit hold count value.

Note Changing this parameter to a higher value can have a significant impact on CPU utilization, especially in Rapid PVST+ mode. Lowering this value can slow down convergence in certain scenarios. We recommend that you maintain the default setting.

This procedure is optional.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spanning-tree transmit hold-count** *value*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree transmit hold-count** *value* **Example:** `Device(config)# spanning-tree transmit hold-count 6`	Configures the number of BPDUs that can be sent before pausing for 1 second. For *value*, the range is 1 to 20; the default is 6.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Monitoring Spanning-Tree Status

Table 4: Commands for Displaying Spanning-Tree Status

show spanning-tree active	Displays spanning-tree information on active interfaces only.
show spanning-tree detail	Displays a detailed summary of interface information.
show spanning-tree vlan *vlan-id*	Displays spanning-tree information for the specified VLAN.
show spanning-tree interface *interface-id*	Displays spanning-tree information for the specified interface.
show spanning-tree interface *interface-id* **portfast**	Displays spanning-tree portfast information for the specified interfac
show spanning-tree summary [**totals**]	Displays a summary of interface states or displays the total lines of t state section.

To clear spanning-tree counters, use the **clear spanning-tree** [**interface** i*nterface-id*] privileged EXEC command.

Additional References for Spanning-Tree Protocol

Related Documents

Related Topic	Document Title
For complete syntax and usage information for the commands used in this chapter.	See the Layer 2/3 Commands section of the*Command Reference (Catalyst 9400 Series Switches)*

Standards and RFCs

Standard/RFC	Title
None	—

MIBs

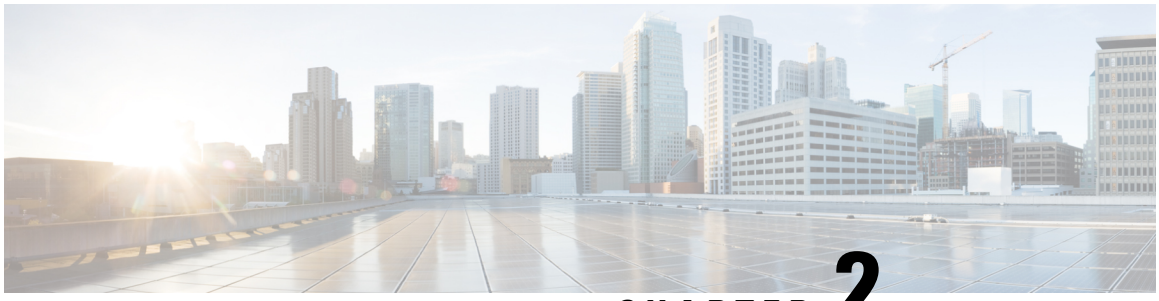
MIB	MIBs Link
All the supported MIBs for this release.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for STP

Release	Modification
Cisco IOS XE Everest 16.6.1	This feature was introduced.



CHAPTER 2

Configuring Multiple Spanning-Tree Protocol

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/. An account on Cisco.com is not required.

Prerequisites for MSTP

- For two or more devices to be in the same multiple spanning tree (MST) region, they must have the same VLAN-to-instance map, the same configuration revision number, and the same name.
- For two or more stacked switches to be in the same MST region, they must have the same VLAN-to-instance map, the same configuration revision number, and the same name.
- For load-balancing across redundant paths in the network to work, all VLAN-to-instance mapping assignments must match; otherwise, all traffic flows on a single link. You can achieve load-balancing across a device stack by manually configuring the path cost.
- For load-balancing between a per-VLAN spanning tree plus (PVST+) and an MST cloud or between a rapid-PVST+ and an MST cloud to work, all MST boundary ports must be forwarding. MST boundary ports are forwarding when the root of the internal spanning tree (IST) of the MST cloud is the root of the common spanning tree (CST). If the MST cloud consists of multiple MST regions, one of the MST regions must contain the CST root, and all of the other MST regions must have a better path to the root

contained within the MST cloud than a path through the PVST+ or rapid-PVST+ cloud. You might have to manually configure the devices in the clouds.

Restrictions for MSTP

- The switch stack supports up to 65 MST instances. The number of VLANs that can be mapped to a particular MST instance is the maximum active VLAN supported by a given switch.
- PVST+, Rapid PVST+, and MSTP are supported, but only one version can be active at any time. (For example, all VLANs run PVST+, all VLANs run Rapid PVST+, or all VLANs run MSTP.)
- All stack members must run the same version of spanning tree (all PVST+, Rapid PVST+, or MSTP).
- VLAN Trunking Protocol (VTP) propagation of the MST configuration is not supported. However, you can manually configure the MST configuration (region name, revision number, and VLAN-to-instance mapping) on each device within the MST region by using the command-line interface (CLI) or through the Simple Network Management Protocol (SNMP) support.
- Partitioning the network into a large number of regions is not recommended. However, if this situation is unavoidable, we recommend that you partition the switched LAN into smaller LANs interconnected by routers or non-Layer 2 devices.
- A region can have one member or multiple members with the same MST configuration; each member must be capable of processing rapid spanning tree protocol (RSTP) Bridge Protocol Data Units (BPDUs). There is no limit to the number of MST regions in a network, but each region can only support up to 65 spanning-tree instances. You can assign a VLAN to only one spanning-tree instance at a time.
- After configuring a device as the root device, we recommend that you avoid manually configuring the hello time, forward-delay time, and maximum-age time through the **spanning-tree mst hello-time**, **spanning-tree mst forward-time**, and the **spanning-tree mst max-age** global configuration commands.

Table 5: PVST+, MSTP, and Rapid PVST+ Interoperability and Compatibility

	PVST+	**MSTP**	**Rapid PVST+**
PVST+	Yes	Yes (with restrictions)	Yes (reverts to PVST+)
MSTP	Yes (with restrictions)	Yes	Yes (reverts to PVST+)
Rapid PVST+	Yes (reverts to PVST+)	Yes (reverts to PVST+)	Yes

Information About MSTP

MSTP Configuration

MSTP, which uses RSTP for rapid convergence, enables multiple VLANs to be grouped into and mapped to the same spanning-tree instance, reducing the number of spanning-tree instances needed to support a large number of VLANs. The MSTP provides for multiple forwarding paths for data traffic, enables load balancing, and reduces the number of spanning-tree instances required to support a large number of VLANs. It improves

the fault tolerance of the network because a failure in one instance (forwarding path) does not affect other instances (forwarding paths).

Note The multiple spanning-tree (MST) implementation is based on the IEEE 802.1s standard.

The most common initial deployment of MSTP is in the backbone and distribution layers of a Layer 2 switched network. This deployment provides the highly available network required in a service-provider environment.

When the device is in the MST mode, the RSTP, which is based on IEEE 802.1w, is automatically enabled. The RSTP provides rapid convergence of the spanning tree through explicit handshaking that eliminates the IEEE 802.1D forwarding delay and quickly transitions root ports and designated ports to the forwarding state.

Both MSTP and RSTP improve the spanning-tree operation and maintain backward compatibility with equipment that is based on the (original) IEEE 802.1D spanning tree, with existing Cisco-proprietary Multiple Instance STP (MISTP), and with existing Cisco PVST+ and rapid per-VLAN spanning-tree plus (Rapid PVST+).

A device stack appears as a single spanning-tree node to the rest of the network, and all stack members use the same device ID.

MSTP Configuration Guidelines

- When you enable MST by using the **spanning-tree mode mst** global configuration command, RSTP is automatically enabled.
- For configuration guidelines about UplinkFast, BackboneFast, and cross-stack UplinkFast, see the relevant sections in the Related Topics section.
- When the device is in MST mode, it uses the long path-cost calculation method (32 bits) to compute the path cost values. With the long path-cost calculation method, the following path cost values are supported:

Speed	Path Cost Value
10 Mb/s	2,000,000
100 Mb/s	200,000
1 Gb/s	20,000
10 Gb/s	2,000
100 Gb/s	200

Root Switch

The device maintains a spanning-tree instance for the group of VLANs mapped to it. A device ID, consisting of the device priority and the device MAC address, is associated with each instance. For a group of VLANs, the device with the lowest device ID becomes the root device.

When you configure a device as the root, you modify the device priority from the default value (32768) to a significantly lower value so that the device becomes the root device for the specified spanning-tree instance.

When you enter this command, the device checks the device priorities of the root devices. Because of the extended system ID support, the device sets its own priority for the specified instance to 24576 if this value will cause this devices to become the root for the specified spanning-tree instance.

If any root device for the specified instance has a device priority lower than 24576, the device sets its own priority to 4096 less than the lowest device priority. (4096 is the value of the least-significant bit of a 4-bit device priority value. For more information, select "Bridge ID, Device Priority, and Extended System ID" link in Related Topics.

If your network consists of devices that support and do not support the extended system ID, it is unlikely that the device with the extended system ID support will become the root device. The extended system ID increases the device priority value every time the VLAN number is greater than the priority of the connected switches running older software.

The root device for each spanning-tree instance should be a backbone or distribution device. Do not configure an access device as the spanning-tree primary root.

Use the **diameter** keyword, which is available only for MST instance 0, to specify the Layer 2 network diameter (that is, the maximum number of device hops between any two end stations in the Layer 2 network). When you specify the network diameter, the device automatically sets an optimal hello time, forward-delay time, and maximum-age time for a network of that diameter, which can significantly reduce the convergence time. You can use the **hello** keyword to override the automatically calculated hello time.

Multiple Spanning-Tree Regions

For switches to participate in multiple spanning-tree (MST) instances, you must consistently configure the switches with the same MST configuration information. A collection of interconnected switches that have the same MST configuration comprises an MST region.

The MST configuration controls to which MST region each device belongs. The configuration includes the name of the region, the revision number, and the MST VLAN-to-instance assignment map. You configure the device for a region by specifying the MST region configuration on it. You can map VLANs to an MST instance, specify the region name, and set the revision number. For instructions and an example, select the "Specifying the MST Region Configuration and Enabling MSTP" link in Related Topics.

A region can have one or multiple members with the same MST configuration. Each member must be capable of processing RSTP bridge protocol data units (BPDUs). There is no limit to the number of MST regions in a network, but each region can support up to 65 spanning-tree instances. Instances can be identified by any number in the range from 0 to 4094. You can assign a VLAN to only one spanning-tree instance at a time.

IST, CIST, and CST

Unlike PVST+ and Rapid PVST+ in which all the spanning-tree instances are independent, the MSTP establishes and maintains two types of spanning trees:

- An internal spanning tree (IST), which is the spanning tree that runs in an MST region.

  Within each MST region, the MSTP maintains multiple spanning-tree instances. Instance 0 is a special instance for a region, known as the internal spanning tree (IST). All other MST instances are numbered from 1 to 4094.

  The IST is the only spanning-tree instance that sends and receives BPDUs. All of the other spanning-tree instance information is contained in M-records, which are encapsulated within MSTP BPDUs. Because the MSTP BPDU carries information for all instances, the number of BPDUs that need to be processed to support multiple spanning-tree instances is significantly reduced.

All MST instances within the same region share the same protocol timers, but each MST instance has its own topology parameters, such as root device ID, root path cost, and so forth. By default, all VLANs are assigned to the IST.

An MST instance is local to the region; for example, MST instance 1 in region A is independent of MST instance 1 in region B, even if regions A and B are interconnected.

- A common and internal spanning tree (CIST), which is a collection of the ISTs in each MST region, and the common spanning tree (CST) that interconnects the MST regions and single spanning trees.

  The spanning tree computed in a region appears as a subtree in the CST that encompasses the entire switched domain. The CIST is formed by the spanning-tree algorithm running among switches that support the IEEE 802.1w, IEEE 802.1s, and IEEE 802.1D standards. The CIST inside an MST region is the same as the CST outside a region.

Operations Within an MST Region

The IST connects all the MSTP switches in a region. When the IST converges, the root of the IST becomes the CIST regional root. It is the device within the region with the lowest device ID and path cost to the CIST root. The CIST regional root is also the CIST root if there is only one region in the network. If the CIST root is outside the region, one of the MSTP switches at the boundary of the region is selected as the CIST regional root.

When an MSTP device initializes, it sends BPDUs claiming itself as the root of the CIST and the CIST regional root, with both of the path costs to the CIST root and to the CIST regional root set to zero. The device also initializes all of its MST instances and claims to be the root for all of them. If the device receives superior MST root information (lower device ID, lower path cost, and so forth) than currently stored for the port, it relinquishes its claim as the CIST regional root.

During initialization, a region might have many subregions, each with its own CIST regional root. As switches receive superior IST information, they leave their old subregions and join the new subregion that contains the true CIST regional root. All subregions shrink except for the one that contains the true CIST regional root.

For correct operation, all switches in the MST region must agree on the same CIST regional root. Therefore, any two switches in the region only synchronize their port roles for an MST instance if they converge to a common CIST regional root.

Operations Between MST Regions

If there are multiple regions or legacy IEEE 802.1D devices within the network, MSTP establishes and maintains the CST, which includes all MST regions and all legacy STP devices in the network. The MST instances combine with the IST at the boundary of the region to become the CST.

The IST connects all the MSTP devices in the region and appears as a subtree in the CIST that encompasses the entire switched domain. The root of the subtree is the CIST regional root. The MST region appears as a virtual device to adjacent STP devices and MST regions.

Only the CST instance sends and receives BPDUs, and MST instances add their spanning-tree information into the BPDUs to interact with neighboring devices and compute the final spanning-tree topology. Because of this, the spanning-tree parameters related to BPDU transmission (for example, hello time, forward time, max-age, and max-hops) are configured only on the CST instance but affect all MST instances. Parameters related to the spanning-tree topology (for example, device priority, port VLAN cost, and port VLAN priority) can be configured on both the CST instance and the MST instance.

MSTP devices use Version 3 RSTP BPDUs or IEEE 802.1D STP BPDUs to communicate with legacy IEEE 802.1D devices. MSTP devices use MSTP BPDUs to communicate with MSTP devices.

IEEE 802.1s Terminology

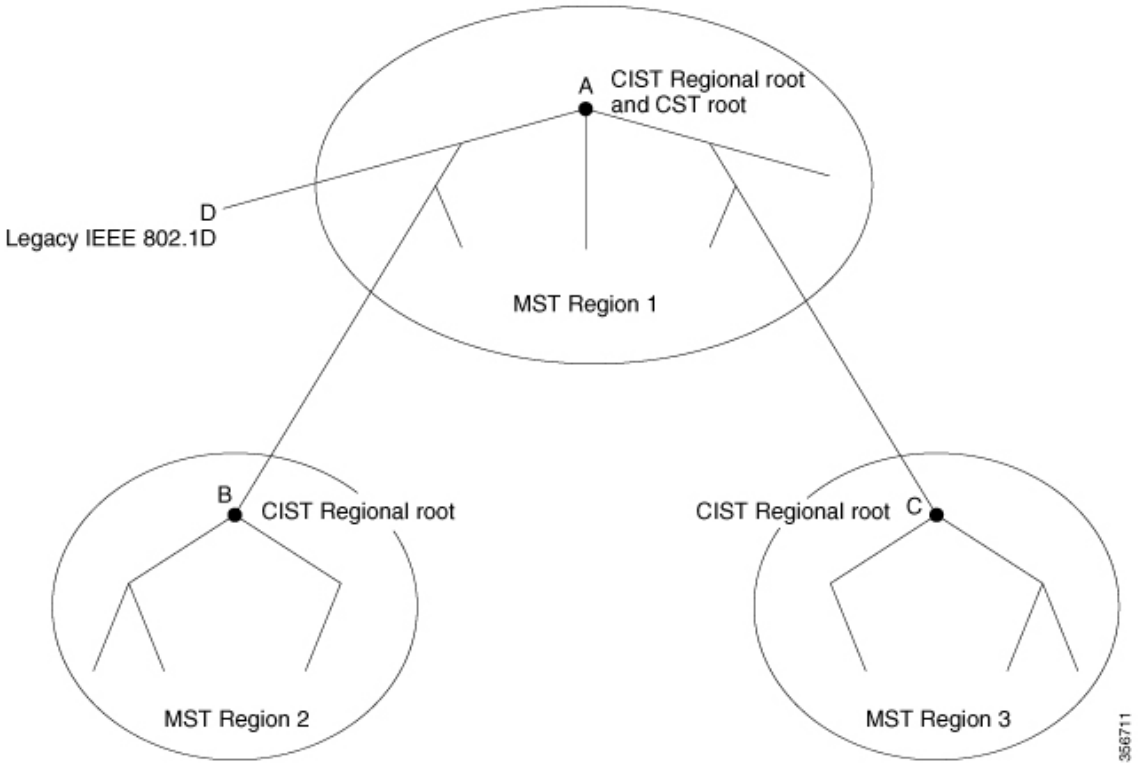
Some MST naming conventions used in Cisco's prestandard implementation have been changed to identify some *internal* or *regional* parameters. These parameters are significant only within an MST region, as opposed to external parameters that are relevant to the whole network. Because the CIST is the only spanning-tree instance that spans the whole network, only the CIST parameters require the external rather than the internal or regional qualifiers.

- The CIST root is the root device for the unique instance that spans the whole network, the CIST.
- The CIST external root path cost is the cost to the CIST root. This cost is left unchanged within an MST region. Remember that an MST region looks like a single device for the CIST. The CIST external root path cost is the root path cost calculated between these virtual devices and devices that do not belong to any region.
- If the CIST root is in the region, the CIST regional root is the CIST root. Otherwise, the CIST regional root is the closest device to the CIST root in the region. The CIST regional root acts as a root device for the IST.
- The CIST internal root path cost is the cost to the CIST regional root in a region. This cost is only relevant to the IST, instance 0.

Illustration of MST Regions

This figure displays three MST regions and a legacy IEEE 802.1D device (D). The CIST regional root for region 1 (A) is also the CIST root. The CIST regional root for region 2 (B) and the CIST regional root for region 3 (C) are the roots for their respective subtrees within the CIST. The RSTP runs in all regions.

Figure 5: MST Regions, CIST Regional Root, and CST Root



Hop Count

The IST and MST instances do not use the message-age and maximum-age information in the configuration BPDU to compute the spanning-tree topology. Instead, they use the path cost to the root and a hop-count mechanism similar to the IP time-to-live (TTL) mechanism.

By using the **spanning-tree mst max-hops** global configuration command, you can configure the maximum hops inside the region and apply it to the IST and all MST instances in that region. The hop count achieves the same result as the message-age information (triggers a reconfiguration). The root device of the instance always sends a BPDU (or M-record) with a cost of 0 and the hop count set to the maximum value. When a device receives this BPDU, it decrements the received remaining hop count by one and propagates this value as the remaining hop count in the BPDUs it generates. When the count reaches zero, the device discards the BPDU and ages the information held for the port.

The message-age and maximum-age information in the RSTP portion of the BPDU remain the same throughout the region, and the same values are propagated by the region designated ports at the boundary.

Boundary Ports

In the Cisco prestandard implementation, a boundary port connects an MST region to a single spanning-tree region running RSTP, to a single spanning-tree region running PVST+ or rapid PVST+, or to another MST region with a different MST configuration. A boundary port also connects to a LAN, the designated device of which is either a single spanning-tree device or a device with a different MST configuration.

There is no definition of a boundary port in the IEEE 802.1s standard. The IEEE 802.1Q-2002 standard identifies two kinds of messages that a port can receive:

- internal (coming from the same region)
- external (coming from another region)

When a message is internal, the CIST part is received by the CIST, and each MST instance receives its respective M-record.

When a message is external, it is received only by the CIST. If the CIST role is root or alternate, or if the external BPDU is a topology change, it could have an impact on the MST instances.

An MST region includes both devices and LANs. A segment belongs to the region of its designated port. Therefore, a port in a different region than the designated port for a segment is a boundary port. This definition allows two ports internal to a region to share a segment with a port belonging to a different region, creating the possibility of a port receiving both internal and external messages.

The primary change from the Cisco prestandard implementation is that a designated port is not defined as boundary, unless it is running in an STP-compatible mode.

Note If there is a legacy STP device on the segment, messages are always considered external.

The other change from the Cisco prestandard implementation is that the CIST regional root device ID field is now inserted where an RSTP or legacy IEEE 802.1Q device has the sender device ID. The whole region performs like a single virtual device by sending a consistent sender device ID to neighboring devices. In this example, device C would receive a BPDU with the same consistent sender device ID of root, whether or not A or B is designated for the segment.

IEEE 802.1s Implementation

The Cisco implementation of the IEEE MST standard includes features required to meet the standard, as well as some of the desirable prestandard functionality that is not yet incorporated into the published standard.

Port Role Naming Change

The boundary role is no longer in the final MST standard, but this boundary concept is maintained in Cisco's implementation. However, an MST instance port at a boundary of the region might not follow the state of the corresponding CIST port. Two boundary roles currently exist:

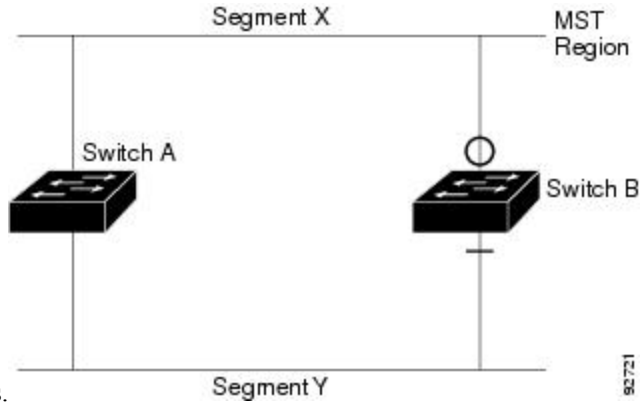
- The boundary port is the root port of the CIST regional root—When the CIST instance port is proposed and is in sync, it can send back an agreement and move to the forwarding state only after all the corresponding MSTI ports are in sync (and thus forwarding). The MSTI ports now have a special *primary* role.
- The boundary port is not the root port of the CIST regional root—The MSTI ports follow the state and role of the CIST port. The standard provides less information, and it might be difficult to understand why an MSTI port can be alternately blocking when it receives no BPDUs (MRecords). In this case, although the boundary role no longer exists, the **show** commands identify a port as boundary in the *type* column of the output.

Interoperation Between Legacy and Standard Devices

Because automatic detection of prestandard devices can fail, you can use an interface configuration command to identify prestandard ports. A region cannot be formed between a standard and a prestandard device, but they can interoperate by using the CIST. Only the capability of load-balancing over different instances is lost in that particular case. The CLI displays different flags depending on the port configuration when a port receives prestandard BPDUs. A syslog message also appears the first time a device receives a prestandard BPDU on a port that has not been configured for prestandard BPDU transmission.

Figure 6: Standard and Prestandard Device Interoperation

Assume that A is a standard device and B a prestandard device, both configured to be in the same region. A is the root device for the CIST, and B has a root port (BX) on segment X and an alternate port (BY) on segment Y. If segment Y flaps, and the port on BY becomes the alternate before sending out a single prestandard BPDU, AY cannot detect that a prestandard device is connected to Y and continues to send standard BPDUs. The port BY is fixed in a boundary, and no load balancing is possible between A and B. The same problem exists on segment X, but B might transmit topology



changes.

Note We recommend that you minimize the interaction between standard and prestandard MST implementations.

Detecting Unidirectional Link Failure

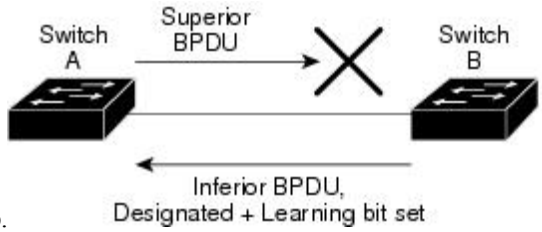
This feature is not yet present in the IEEE MST standard, but it is included in this Cisco IOS release. The software checks the consistency of the port role and state in the received BPDUs to detect unidirectional link failures that could cause bridging loops.

When a designated port detects a conflict, it keeps its role, but reverts to the discarding state because disrupting connectivity in case of inconsistency is preferable to opening a bridging loop.

Figure 7: Detecting Unidirectional Link Failure

This figure illustrates a unidirectional link failure that typically creates a bridging loop. Device A is the root device, and its BPDUs are lost on the link leading to device B. RSTP and MST BPDUs include the role and state of the sending port. With this information, device A can detect that device B does not react to the superior

BPDUs it sends and that device B is the designated, not root device. As a result, device A blocks (or keeps blocking) its port, which prevents the bridging loop.



MSTP and Device Stacks

A device stack appears as a single spanning-tree node to the rest of the network, and all stack members use the same bridge ID for a given spanning tree. The bridge ID is derived from the MAC address of the active stack.

If a device that does not support MSTP is added to a device stack that does support MSTP or the reverse, the device is put into a version mismatch state. If possible, the device is automatically upgraded or downgraded to the same version of software that is running on the device stack.

Interoperability with IEEE 802.1D STP

A device running MSTP supports a built-in protocol migration mechanism that enables it to interoperate with legacy IEEE 802.1D devices. If this device receives a legacy IEEE 802.1D configuration BPDU (a BPDU with the protocol version set to 0), it sends only IEEE 802.1D BPDUs on that port. An MSTP device also can detect that a port is at the boundary of a region when it receives a legacy BPDU, an MSTP BPDU (Version 3) associated with a different region, or an RSTP BPDU (Version 2).

However, the device does not automatically revert to the MSTP mode if it no longer receives IEEE 802.1D BPDUs because it cannot detect whether the legacy device has been removed from the link unless the legacy device is the designated device. A device might also continue to assign a boundary role to a port when the device to which this device is connected has joined the region. To restart the protocol migration process (force the renegotiation with neighboring devices), use the **clear spanning-tree detected-protocols** privileged EXEC command.

If all the legacy devices on the link are RSTP devices, they can process MSTP BPDUs as if they are RSTP BPDUs. Therefore, MSTP devices send either a Version 0 configuration and TCN BPDUs or Version 3 MSTP BPDUs on a boundary port. A boundary port connects to a LAN, the designated device of which is either a single spanning-tree device or a device with a different MST configuration.

RSTP Overview

The RSTP takes advantage of point-to-point wiring and provides rapid convergence of the spanning tree. Reconfiguration of the spanning tree can occur in less than 1 second (in contrast to 50 seconds with the default settings in the IEEE 802.1D spanning tree).

Port Roles and the Active Topology

The RSTP provides rapid convergence of the spanning tree by assigning port roles and by learning the active topology. The RSTP builds upon the IEEE 802.1D STP to select the device with the highest device priority (lowest numerical priority value) as the root device. The RSTP then assigns one of these port roles to individual ports:

- Root port—Provides the best path (lowest cost) when the device forwards packets to the root device.
- Designated port—Connects to the designated device, which incurs the lowest path cost when forwarding packets from that LAN to the root device. The port through which the designated device is attached to the LAN is called the designated port.
- Alternate port—Offers an alternate path toward the root device to that provided by the current root port.
- Backup port—Acts as a backup for the path provided by a designated port toward the leaves of the spanning tree. A backup port can exist only when two ports are connected in a loopback by a point-to-point link or when a device has two or more connections to a shared LAN segment.
- Disabled port—Has no role within the operation of the spanning tree.

A port with the root or a designated port role is included in the active topology. A port with the alternate or backup port role is excluded from the active topology.

In a stable topology with consistent port roles throughout the network, the RSTP ensures that every root port and designated port immediately transition to the forwarding state while all alternate and backup ports are always in the discarding state (equivalent to blocking in IEEE 802.1D). The port state controls the operation of the forwarding and learning processes.

Table 6: Port State Comparison

Operational Status	STP Port State (IEEE 802.1D)	RSTP Port State	Is Port Included in the Active Topology?
Enabled	Blocking	Discarding	No
Enabled	Listening	Discarding	No
Enabled	Learning	Learning	Yes
Enabled	Forwarding	Forwarding	Yes
Disabled	Disabled	Discarding	No

To be consistent with Cisco STP implementations, this guide defines the port state as *blocking* instead of *discarding*. Designated ports start in the listening state.

Rapid Convergence

The RSTP provides for rapid recovery of connectivity following the failure of a device, a device port, or a LAN. It provides rapid convergence for edge ports, new root ports, and ports connected through point-to-point links as follows:

- Edge ports—If you configure a port as an edge port on an RSTP device by using the **spanning-tree portfast** interface configuration command, the edge port immediately transitions to the forwarding state. An edge port is the same as a Port Fast-enabled port, and you should enable it only on ports that connect to a single end station.
- Root ports—If the RSTP selects a new root port, it blocks the old root port and immediately transitions the new root port to the forwarding state.

- Point-to-point links—If you connect a port to another port through a point-to-point link and the local port becomes a designated port, it negotiates a rapid transition with the other port by using the proposal-agreement handshake to ensure a loop-free topology.

Figure 8: Proposal and Agreement Handshaking for Rapid Convergence

Device A is connected to Device B through a point-to-point link, and all of the ports are in the blocking state. Assume that the priority of Device A is a smaller numerical value than the priority of Device B. Device A sends a proposal message (a configuration BPDU with the proposal flag set) to Device B, proposing itself as the designated device.

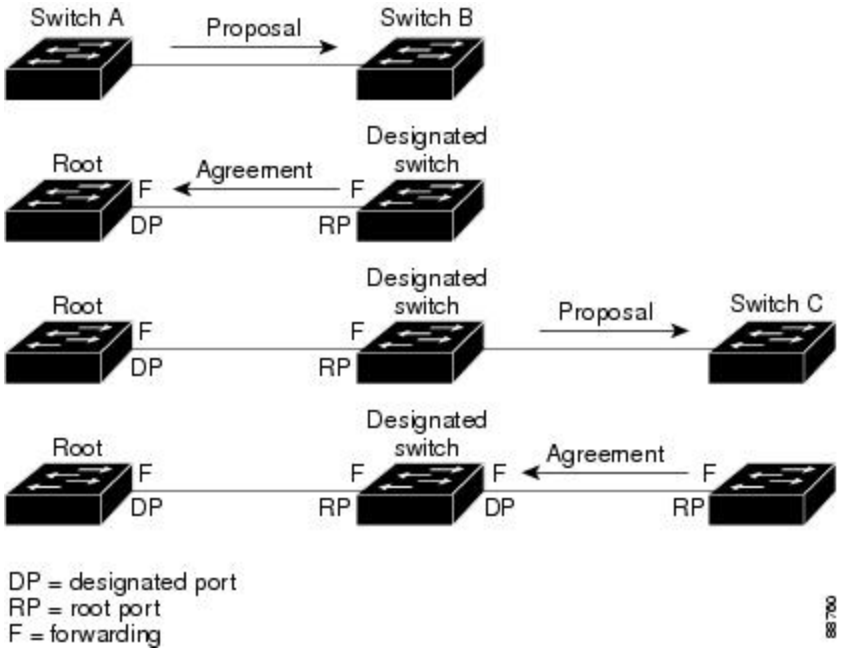
After receiving the proposal message, Device B selects as its new root port the port from which the proposal message was received, forces all nonedge ports to the blocking state, and sends an agreement message (a BPDU with the agreement flag set) through its new root port.

After receiving Device B's agreement message, Device A also immediately transitions its designated port to the forwarding state. No loops in the network are formed because Device B blocked all of its nonedge ports and because there is a point-to-point link between Devices A and B.

When Device C is connected to Device B, a similar set of handshaking messages are exchanged. Device C selects the port connected to Device B as its root port, and both ends immediately transition to the forwarding state. With each iteration of this handshaking process, one more device joins the active topology. As the network converges, this proposal-agreement handshaking progresses from the root toward the leaves of the spanning tree.

In a device stack, the cross-stack rapid transition (CSRT) feature ensures that a stack member receives acknowledgments from all stack members during the proposal-agreement handshaking before moving the port to the forwarding state. CSRT is automatically enabled when the device is in MST mode.

The device learns the link type from the port duplex mode: a full-duplex port is considered to have a point-to-point connection; a half-duplex port is considered to have a shared connection. You can override the default setting that is controlled by the duplex setting by using the **spanning-tree link-type** interface configuration command.



Synchronization of Port Roles

When the device receives a proposal message on one of its ports and that port is selected as the new root port, the RSTP forces all other ports to synchronize with the new root information.

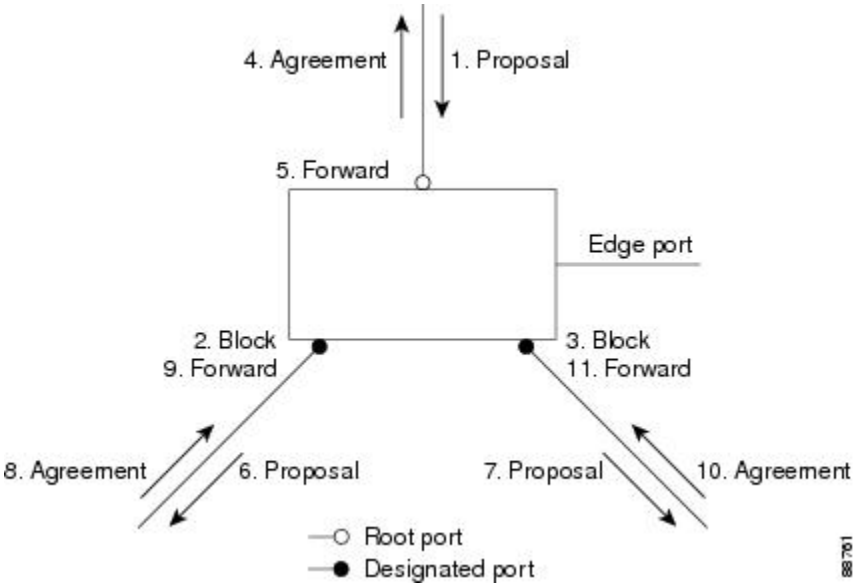
The device is synchronized with superior root information received on the root port if all other ports are synchronized. An individual port on the device is synchronized if

- That port is in the blocking state.
- It is an edge port (a port configured to be at the edge of the network).

If a designated port is in the forwarding state and is not configured as an edge port, it transitions to the blocking state when the RSTP forces it to synchronize with new root information. In general, when the RSTP forces a port to synchronize with root information and the port does not satisfy any of the above conditions, its port state is set to blocking.

Figure 9: Sequence of Events During Rapid Convergence

After ensuring that all of the ports are synchronized, the device sends an agreement message to the designated device corresponding to its root port. When the devices connected by a point-to-point link are in agreement about their port roles, the RSTP immediately transitions the port states to forwarding.



Bridge Protocol Data Unit Format and Processing

The RSTP BPDU format is the same as the IEEE 802.1D BPDU format except that the protocol version is set to 2. A new 1-byte Version 1 Length field is set to zero, which means that no version 1 protocol information is present.

Table 7: RSTP BPDU Flags

Bit	Function
0	Topology change (TC)

Bit	Function
1	Proposal
2–3: 00 01 10 11	Port role: Unknown Alternate port Root port Designated port
4	Learning
5	Forwarding
6	Agreement
7	Topology change acknowledgement (TCA)

The sending device sets the proposal flag in the RSTP BPDU to propose itself as the designated device on that LAN. The port role in the proposal message is always set to the designated port.

The sending device sets the agreement flag in the RSTP BPDU to accept the previous proposal. The port role in the agreement message is always set to the root port.

The RSTP does not have a separate topology change notification (TCN) BPDU. It uses the topology change (TC) flag to show the topology changes. However, for interoperability with IEEE 802.1D devices, the RSTP device processes and generates TCN BPDUs.

The learning and forwarding flags are set according to the state of the sending port.

Processing Superior BPDU Information

If a port receives superior root information (lower device ID, lower path cost, and so forth) than currently stored for the port, the RSTP triggers a reconfiguration. If the port is proposed and is selected as the new root port, RSTP forces all the other ports to synchronize.

If the BPDU received is an RSTP BPDU with the proposal flag set, the device sends an agreement message after all of the other ports are synchronized. If the BPDU is an IEEE 802.1D BPDU, the device does not set the proposal flag and starts the forward-delay timer for the port. The new root port requires twice the forward-delay time to transition to the forwarding state.

If the superior information received on the port causes the port to become a backup or alternate port, RSTP sets the port to the blocking state but does not send the agreement message. The designated port continues sending BPDUs with the proposal flag set until the forward-delay timer expires, at which time the port transitions to the forwarding state.

Processing Inferior BPDU Information

If a designated port receives an inferior BPDU (such as a higher device ID or a higher path cost than currently stored for the port) with a designated port role, it immediately replies with its own information.

Topology Changes

This section describes the differences between the RSTP and the IEEE 802.1D in handling spanning-tree topology changes.

- Detection—Unlike IEEE 802.1D in which *any* transition between the blocking and the forwarding state causes a topology change, *only* transitions from the blocking to the forwarding state cause a topology change with RSTP (only an increase in connectivity is considered a topology change). State changes on an edge port do not cause a topology change. When an RSTP device detects a topology change, it deletes the learned information on all of its nonedge ports except on those from which it received the TC notification.
- Notification—Unlike IEEE 802.1D, which uses TCN BPDUs, the RSTP does not use them. However, for IEEE 802.1D interoperability, an RSTP device processes and generates TCN BPDUs.
- Acknowledgement—When an RSTP device receives a TCN message on a designated port from an IEEE 802.1D device, it replies with an IEEE 802.1D configuration BPDU with the TCA bit set. However, if the TC-while timer (the same as the topology-change timer in IEEE 802.1D) is active on a root port connected to an IEEE 802.1D device and a configuration BPDU with the TCA bit set is received, the TC-while timer is reset.

  This behavior is only required to support IEEE 802.1D devices. The RSTP BPDUs never have the TCA bit set.
- Propagation—When an RSTP device receives a TC message from another device through a designated or root port, it propagates the change to all of its nonedge, designated ports and to the root port (excluding the port on which it is received). The device starts the TC-while timer for all such ports and flushes the information learned on them.
- Protocol migration—For backward compatibility with IEEE 802.1D devices, RSTP selectively sends IEEE 802.1D configuration BPDUs and TCN BPDUs on a per-port basis.

  When a port is initialized, the migrate-delay timer is started (specifies the minimum time during which RSTP BPDUs are sent), and RSTP BPDUs are sent. While this timer is active, the device processes all BPDUs received on that port and ignores the protocol type.

  If the device receives an IEEE 802.1D BPDU after the port migration-delay timer has expired, it assumes that it is connected to an IEEE 802.1D device and starts using only IEEE 802.1D BPDUs. However, if the RSTP device is using IEEE 802.1D BPDUs on a port and receives an RSTP BPDU after the timer has expired, it restarts the timer and starts using RSTP BPDUs on that port.

Protocol Migration Process

A device running MSTP supports a built-in protocol migration mechanism that enables it to interoperate with legacy IEEE 802.1D devices. If this device receives a legacy IEEE 802.1D configuration BPDU (a BPDU with the protocol version set to 0), it sends only IEEE 802.1D BPDUs on that port. An MSTP device also can detect that a port is at the boundary of a region when it receives a legacy BPDU, an MST BPDU (Version 3) associated with a different region, or an RST BPDU (Version 2).

However, the device does not automatically revert to the MSTP mode if it no longer receives IEEE 802.1D BPDUs because it cannot detect whether the legacy device has been removed from the link unless the legacy device is the designated device. A device also might continue to assign a boundary role to a port when the device to which it is connected has joined the region.

Default MSTP Configuration

Table 8: Default MSTP Configuration

Feature	Default Setting
Spanning-tree mode	
Device priority (configurable on a per-CIST port basis)	32768
Spanning-tree port priority (configurable on a per-CIST port basis)	128
Spanning-tree port cost (configurable on a per-CIST port basis)	
Hello time	
Forward-delay time	
Maximum-aging time	20 seconds
Maximum hop count	20 hops

How to Configure MSTP Features

Specifying the MST Region Configuration and Enabling MSTP

For two or more switches to be in the same MST region, they must have the same VLAN-to-instance mapping, the same configuration revision number, and the same name.

A region can have one member or multiple members with the same MST configuration; each member must be capable of processing RSTP BPDUs. There is no limit to the number of MST regions in a network, but each region can only support up to 65 spanning-tree instances. You can assign a VLAN to only one spanning-tree instance at a time.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spanning-tree mst configuration**
4. **instance** *instance-id* **vlan** *vlan-range*
5. **name** *name*
6. **revision** *version*
7. **show pending**
8. **exit**
9. **spanning-tree mode mst**
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree mst configuration** **Example:** `Device(config)# spanning-tree mst configuration`	Enters MST configuration mode.
Step 4	**instance** *instance-id* **vlan** *vlan-range* **Example:** `Device(config-mst)# instance 1 vlan 10-20`	Maps VLANs to an MST instance. • For *instance-id*, the range is 0 to 4094. • For **vlan** *vlan-range*, the range is 1 to 4094. When you map VLANs to an MST instance, the mapping is incremental, and the VLANs specified in the command are added to or removed from the VLANs that were previously mapped. To specify a VLAN range, use a hyphen; for example, **instance 1 vlan 1-63** maps VLANs 1 through 63 to MST instance 1. To specify a VLAN series, use a comma; for example, **instance 1 vlan 10, 20, 30** maps VLANs 10, 20, and 30 to MST instance 1.
Step 5	**name** *name* **Example:** `Device(config-mst)# name region1`	Specifies the configuration name. The *name* string has a maximum length of 32 characters and is case sensitive.
Step 6	**revision** *version* **Example:** `Device(config-mst)# revision 1`	Specifies the configuration revision number. The range is 0 to 65535.
Step 7	**show pending** **Example:**	Verifies your configuration by displaying the pending configuration.

	Command or Action	Purpose
	`Device(config-mst)# show pending`	
Step 8	**exit** **Example:** `Device(config-mst)# exit`	Applies all changes, and returns to global configuration mode.
Step 9	**spanning-tree mode mst** **Example:** `Device(config)# spanning-tree mode mst`	Enables MSTP. RSTP is also enabled. Changing spanning-tree modes can disrupt traffic because all spanning-tree instances are stopped for the previous mode and restarted in the new mode. You cannot run both MSTP and PVST+ or both MSTP and Rapid PVST+ at the same time.
Step 10	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring the Root Device

This procedure is optional.

Before you begin

A multiple spanning tree (MST) must be specified and enabled on the device. For instructions, see Related Topics.

You must also know the specified MST instance ID. Step 2 in the example uses 0 as the instance ID because that was the instance ID set up by the instructions listed under Related Topics.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spanning-tree mst** *instance-id* **root primary**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree mst** *instance-id* **root primary** **Example:** `Device(config)# spanning-tree mst 0 root primary`	Configures a device as the root device. • For *instance-id*, you can specify a single instance, a range of instances separated by a hyphen, or a series of instances separated by a comma. The range is 0 to 4094.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring a Secondary Root Device

When you configure a device with the extended system ID support as the secondary root, the device priority is modified from the default value (32768) to 28672. The device is then likely to become the root device for the specified instance if the primary root device fails. This is assuming that the other network devices use the default device priority of 32768 and therefore are unlikely to become the root device.

You can execute this command on more than one device to configure multiple backup root devices. Use the same network diameter and hello-time values that you used when you configured the primary root device with the **spanning-tree mst** *instance-id* **root primary** global configuration command.

This procedure is optional.

Before you begin

A multiple spanning tree (MST) must be specified and enabled on the device. For instructions, see Related Topics.

You must also know the specified MST instance ID. This example uses 0 as the instance ID because that was the instance ID set up by the instructions listed under Related Topics.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spanning-tree mst** *instance-id* **root secondary**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree mst** *instance-id* **root secondary** **Example:** `Device(config)# spanning-tree mst 0 root secondary`	Configures a device as the secondary root device. • For *instance-id*, you can specify a single instance, a range of instances separated by a hyphen, or a series of instances separated by a comma. The range is 0 to 4094.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring Port Priority

If a loop occurs, the MSTP uses the port priority when selecting an interface to put into the forwarding state. You can assign higher priority values (lower numerical values) to interfaces that you want selected first and lower priority values (higher numerical values) that you want selected last. If all interfaces have the same priority value, the MSTP puts the interface with the lowest interface number in the forwarding state and blocks the other interfaces.

Note If the device is a member of a device stack, you must use the **spanning-tree mst** [*instance-id*] **cost** *cost* interface configuration command instead of the **spanning-tree mst** [*instance-id*] **port-priority** *priority* interface configuration command to select a port to put in the forwarding state. Assign lower cost values to ports that you want selected first and higher cost values to ports that you want selected last. For more information, see the path costs topic listed under Related Topics.

This procedure is optional.

Before you begin

A multiple spanning tree (MST) must be specified and enabled on the device. For instructions, see Related Topics.

You must also know the specified MST instance ID and the interface used. This example uses 0 as the instance ID and GigabitEthernet0/1 as the interface because that was the instance ID and interface set up by the instructions listed under Related Topics.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *interface-id*
4. **spanning-tree mst** *instance-id* **port-priority** *priority*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/0/1`	Specifies an interface to configure, and enters interface configuration mode.
Step 4	**spanning-tree mst** *instance-id* **port-priority** *priority* **Example:** `Device(config-if)# spanning-tree mst 0 port-priority 64`	Configures port priority. • For *instance-id*, you can specify a single instance, a range of instances separated by a hyphen, or a series of instances separated by a comma. The range is 0 to 4094. • For *priority*, the range is 0 to 240 in increments of 16. The default is 128. The lower the number, the higher the priority. The priority values are 0, 16, 32, 48, 64, 80, 96, 112, 128, 144, 160, 176, 192, 208, 224, and 240. All other values are rejected.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

The **show spanning-tree mst interface** *interface-id* privileged EXEC command displays information only if the port is in a link-up operative state. Otherwise, you can use the **show running-config interface** privileged EXEC command to confirm the configuration.

Configuring Path Cost

The MSTP path cost default value is derived from the media speed of an interface. If a loop occurs, the MSTP uses cost when selecting an interface to put in the forwarding state. You can assign lower cost values to interfaces that you want selected first and higher cost values that you want selected last. If all interfaces have the same cost value, the MSTP puts the interface with the lowest interface number in the forwarding state and blocks the other interfaces.

This procedure is optional.

Before you begin

A multiple spanning tree (MST) must be specified and enabled on the device. For instructions, see Related Topics.

You must also know the specified MST instance ID and the interface used. This example uses 0 as the instance ID and GigabitEthernet1/0/1 as the interface because that was the instance ID and interface set up by the instructions listed under Related Topics.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *interface-id*
4. **spanning-tree mst** *instance-id* **cost** *cost*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/0/1`	Specifies an interface to configure, and enters interface configuration mode. Valid interfaces include physical ports and port-channel logical interfaces. The port-channel range is 1 to 48.

	Command or Action	Purpose
Step 4	**spanning-tree mst** *instance-id* **cost** *cost* **Example:** `Device(config-if)# spanning-tree mst 0 cost 17031970`	Configures the cost. If a loop occurs, the MSTP uses the path cost when selecting an interface to place into the forwarding state. A lower path cost represents higher-speed transmission. • For *instance-id*, you can specify a single instance, a range of instances separated by a hyphen, or a series of instances separated by a comma. The range is 0 to 4094. • For *cost*, the range is 1 to 200000000; the default value is derived from the media speed of the interface.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

The **show spanning-tree mst interface** *interface-id* privileged EXEC command displays information only for ports that are in a link-up operative state. Otherwise, you can use the **show running-config** privileged EXEC command to confirm the configuration.

Configuring the Device Priority

Changing the priority of a device makes it more likely to be chosen as the root device whether it is a standalone device or a device in the stack.

Note Exercise care when using this command. For normal network configurations, we recommend that you use the **spanning-tree mst** i*nstance-id* **root primary** and the **spanning-tree mst** *instance-id* **root secondary** global configuration commands to specify a device as the root or secondary root device. You should modify the device priority only in circumstances where these commands do not work.

This procedure is optional.

Before you begin

A multiple spanning tree (MST) must be specified and enabled on the device. For instructions, see Related Topics.

You must also know the specified MST instance ID used. This example uses 0 as the instance ID because that was the instance ID set up by the instructions listed under Related Topics.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spanning-tree mst** *instance-id* **priority** *priority*

4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree mst** *instance-id* **priority** *priority* **Example:** `Device(config)# spanning-tree mst 0 priority 40960`	Configures the device priority. • For *instance-id*, you can specify a single instance, a range of instances separated by a hyphen, or a series of instances separated by a comma. The range is 0 to 4094. • For *priority*, the range is 0 to 61440 in increments of 4096; the default is 32768. The lower the number, the more likely the device will be chosen as the root device. Priority values are 0, 4096, 8192, 12288, 16384, 20480, 24576, 28672, 32768, 36864, 40960, 45056, 49152, 53248, 57344, and 61440. These are the only acceptable values.
Step 4	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Configuring the Hello Time

The hello time is the time interval between configuration messages generated and sent by the root device.

This procedure is optional.

Before you begin

A multiple spanning tree (MST) must be specified and enabled on the device. For instructions, see Related Topics.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spanning-tree mst hello-time** *seconds*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree mst hello-time** *seconds* **Example:** `Device(config)# spanning-tree mst hello-time 4`	Configures the hello time for all MST instances. The hello time is the time interval between configuration messages generated and sent by the root device. These messages indicate that the device is alive. For *seconds*, the range is 1 to 10; the default is 3.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring the Forwarding-Delay Time

Before you begin

A multiple spanning tree (MST) must be specified and enabled on the device. For instructions, see Related Topics.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spanning-tree mst forward-time** *seconds*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree mst forward-time** *seconds* **Example:** `Device(config)# spanning-tree mst forward-time 25`	Configures the forward time for all MST instances. The forwarding delay is the number of seconds a port waits before changing from its spanning-tree learning and listening states to the forwarding state. For *seconds*, the range is 4 to 30; the default is 20.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring the Maximum-Aging Time

Before you begin

A multiple spanning tree (MST) must be specified and enabled on the device. For instructions, see Related Topics.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spanning-tree mst max-age** *seconds*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree mst max-age** *seconds* **Example:** `Device(config)# spanning-tree mst max-age 40`	Configures the maximum-aging time for all MST instances. The maximum-aging time is the number of seconds a device waits without receiving spanning-tree configuration messages before attempting a reconfiguration. For *seconds*, the range is 6 to 40; the default is 20.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring the Maximum-Hop Count

This procedure is optional.

Before you begin

A multiple spanning tree (MST) must be specified and enabled on the device. For instructions, see Related Topics.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spanning-tree mst max-hops** *hop-count*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree mst max-hops** *hop-count* **Example:** `Device(config)# spanning-tree mst max-hops 25`	Specifies the number of hops in a region before the BPDU is discarded, and the information held for a port is aged. For *hop-count*, the range is 1 to 255; the default is 20.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Specifying the Link Type to Ensure Rapid Transitions

If you connect a port to another port through a point-to-point link and the local port becomes a designated port, the RSTP negotiates a rapid transition with the other port by using the proposal-agreement handshake to ensure a loop-free topology.

By default, the link type is controlled from the duplex mode of the interface: a full-duplex port is considered to have a point-to-point connection; a half-duplex port is considered to have a shared connection. If you have a half-duplex link physically connected point-to-point to a single port on a remote device running MSTP, you can override the default setting of the link type and enable rapid transitions to the forwarding state.

This procedure is optional.

Before you begin

A multiple spanning tree (MST) must be specified and enabled on the device. For instructions, see Related Topics.

You must also know the specified MST instance ID and the interface used. This example uses 0 as the instance ID and GigabitEthernet1/0/1 as the interface because that was the instance ID and interface set up by the instructions listed under Related Topics.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *interface-id*
4. **spanning-tree link-type point-to-point**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/0/1`	Specifies an interface to configure, and enters interface configuration mode. Valid interfaces include physical ports, VLANs, and port-channel logical interfaces. The VLAN ID range is 1 to 4094. The port-channel range is 1 to 48.
Step 4	**spanning-tree link-type point-to-point** **Example:** `Device(config-if)# spanning-tree link-type point-to-point`	Specifies that the link type of a port is point-to-point.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Designating the Neighbor Type

A topology could contain both prestandard and IEEE 802.1s standard compliant devices. By default, ports can automatically detect prestandard devices, but they can still receive both standard and prestandard BPDUs. When there is a mismatch between a device and its neighbor, only the CIST runs on the interface.

You can choose to set a port to send only prestandard BPDUs. The prestandard flag appears in all the **show** commands, even if the port is in STP compatibility mode.

This procedure is optional.

Before you begin

A multiple spanning tree (MST) must be specified and enabled on the device. For instructions, see Related Topics.

SUMMARY STEPS

1. **enable**

2. **configure terminal**
3. **interface** *interface-id*
4. **spanning-tree mst pre-standard**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/0/1`	Specifies an interface to configure, and enters interface configuration mode. Valid interfaces include physical ports.
Step 4	**spanning-tree mst pre-standard** **Example:** `Device(config-if)# spanning-tree mst pre-standard`	Specifies that the port can send only prestandard BPDUs.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Restarting the Protocol Migration Process

This procedure restarts the protocol migration process and forces renegotiation with neighboring devices. It reverts the device to MST mode. It is needed when the device no longer receives IEEE 802.1D BPDUs after it has been receiving them.

Follow these steps to restart the protocol migration process (force the renegotiation with neighboring devices) on the device.

Before you begin

A multiple spanning tree (MST) must be specified and enabled on the device. For instructions, see Related Topics.

If you want to use the interface version of the command, you must also know the MST interface used. This example uses GigabitEthernet1/0/1 as the interface because that was the interface set up by the instructions listed under Related Topics.

SUMMARY STEPS

1. **enable**
2. Enter one of the following commands:
   - **clear spanning-tree detected-protocols**
   - **clear spanning-tree detected-protocols interface** *interface-id*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

Step 2	Enter one of the following commands:
	• **clear spanning-tree detected-protocols**
	• **clear spanning-tree detected-protocols interface** *interface-id*
	Example:
	`Device# clear spanning-tree detected-protocols`
	or
	`Device# clear spanning-tree detected-protocols interface gigabitethernet 1/0/1`
		The device reverts to the MSTP mode, and the protocol migration process restarts.

What to do next

This procedure may need to be repeated if the device receives more legacy IEEE 802.1D configuration BPDUs (BPDUs with the protocol version set to 0).

Additional References for MSTP

Related Documents

Related Topic	Document Title
For complete syntax and usage information for the commands used in this chapter.	See the Layer 2/3 Commands section of the*Command Reference (Catalyst 9400 Series Switches)*

Standards and RFCs

Standard/RFC	Title
None	—

MIBs

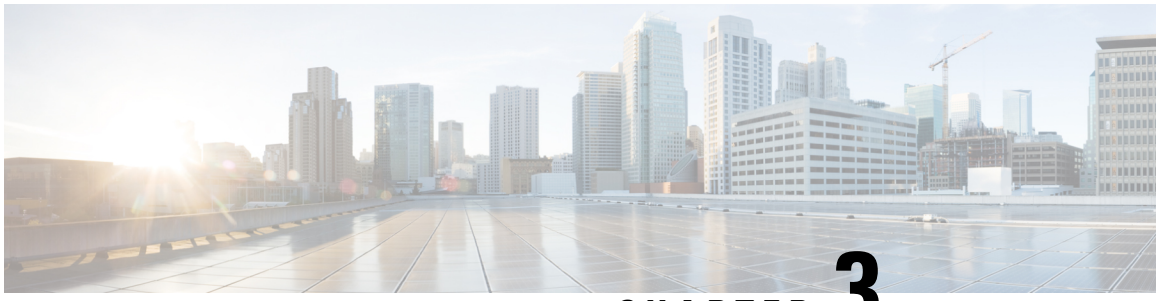
MIB	MIBs Link
All the supported MIBs for this release.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for MSTP

Release	Modification
Cisco IOS XE Everest 16.6.1	This feature was introduced.



CHAPTER 3

Configuring Optional Spanning-Tree Features

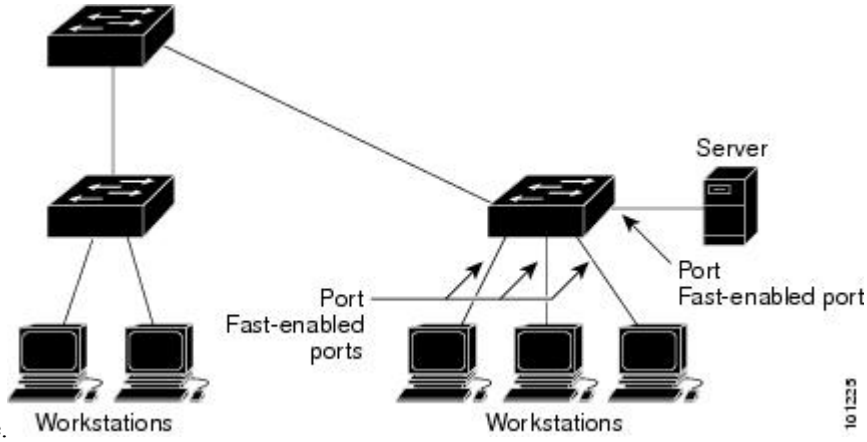
Information About Optional Spanning-Tree Features

PortFast

PortFast immediately brings an interface configured as an access or trunk port to the forwarding state from a blocking state, bypassing the listening and learning states.

Figure 10: PortFast-Enabled Interfaces

You can use PortFast on interfaces connected to a single workstation or server to allow those devices to immediately connect to the network, rather than waiting for the spanning tree to converge.



Interfaces connected to a single workstation or server should not receive bridge protocol data units (BPDUs). An interface with PortFast enabled goes through the normal cycle of spanning-tree status changes when the switch is restarted.

You can enable this feature by enabling it on either the interface or on all nontrunking ports.

BPDU Guard

The Bridge Protocol Data Unit (BPDU) guard feature can be globally enabled on the switch or can be enabled per port, but the feature operates with some differences.

When you enable BPDU guard at the global level on PortFast edge-enabled ports, spanning tree shuts down ports that are in a PortFast edge-operational state if any BPDU is received on them. In a valid configuration, PortFast edge-enabled ports do not receive BPDUs. Receiving a BPDU on a Port Fast edge-enabled port means an invalid configuration, such as the connection of an unauthorized device, and the BPDU guard feature puts the port in the error-disabled state. When this happens, the switch shuts down the entire port on which the violation occurred.

When you enable BPDU guard at the interface level on any port without also enabling the PortFast edge feature, and the port receives a BPDU, it is put in the error-disabled state.

The BPDU guard feature provides a secure response to invalid configurations because you must manually put the interface back in service. Use the BPDU guard feature in a service-provider network to prevent an access port from participating in the spanning tree.

BPDU Filtering

The BPDU filtering feature can be globally enabled on the switch or can be enabled per interface, but the feature operates with some differences.

Enabling BPDU filtering on PortFast edge-enabled interfaces at the global level keeps those interfaces that are in a PortFast edge-operational state from sending or receiving BPDUs. The interfaces still send a few BPDUs at link-up before the switch begins to filter outbound BPDUs. You should globally enable BPDU filtering on a switch so that hosts connected to these interfaces do not receive BPDUs. If a BPDU is received on a PortFast edge-enabled interface, the interface loses its PortFast edge-operational status, and BPDU filtering is disabled.

Enabling BPDU filtering on an interface without also enabling the PortFast edge feature keeps the interface from sending or receiving BPDUs.

Caution Enabling BPDU filtering on an interface is the same as disabling spanning tree on it and can result in spanning-tree loops.

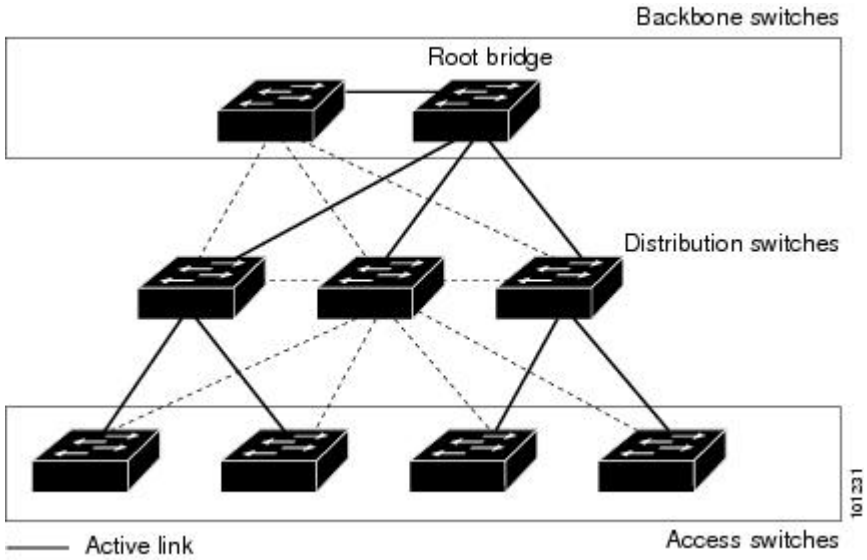
You can enable the BPDU filtering feature for the entire switch or for an interface.

UplinkFast

Figure 11: Switches in a Hierarchical Network

Switches in hierarchical networks can be grouped into backbone switches, distribution switches, and access switches. This complex network has distribution switches and access switches that each have at least one

redundant link that spanning tree blocks to prevent



loops.

If a switch loses connectivity, it begins using the alternate paths as soon as the spanning tree selects a new root port. You can accelerate the choice of a new root port when a link or switch fails or when the spanning tree reconfigures itself by enabling UplinkFast. The root port transitions to the forwarding state immediately without going through the listening and learning states, as it would with the normal spanning-tree procedures.

When the spanning tree reconfigures the new root port, other interfaces flood the network with multicast packets, one for each address that was learned on the interface. You can limit these bursts of multicast traffic by reducing the max-update-rate parameter (the default for this parameter is 150 packets per second). However, if you enter zero, station-learning frames are not generated, so the spanning-tree topology converges more slowly after a loss of connectivity.

Note UplinkFast is most useful in wiring-closet switches at the access or edge of the network. It is not appropriate for backbone devices. This feature might not be useful for other types of applications.

UplinkFast provides fast convergence after a direct link failure and achieves load-balancing between redundant Layer 2 links using uplink groups. An uplink group is a set of Layer 2 interfaces (per VLAN), only one of which is forwarding at any given time. Specifically, an uplink group consists of the root port (which is forwarding) and a set of blocked ports, except for self-looping ports. The uplink group provides an alternate path in case the currently forwarding link fails.

Figure 12: UplinkFast Example Before Direct Link Failure

This topology has no link failures. Switch A, the root switch, is connected directly to Switch B over link L1 and to Switch C over link L2. The Layer 2 interface on Switch C that is connected directly to Switch B is in a blocking state.

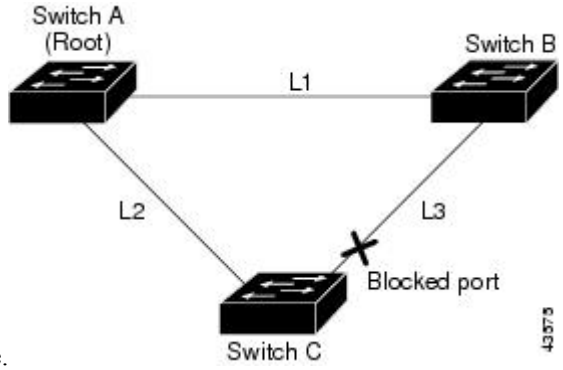
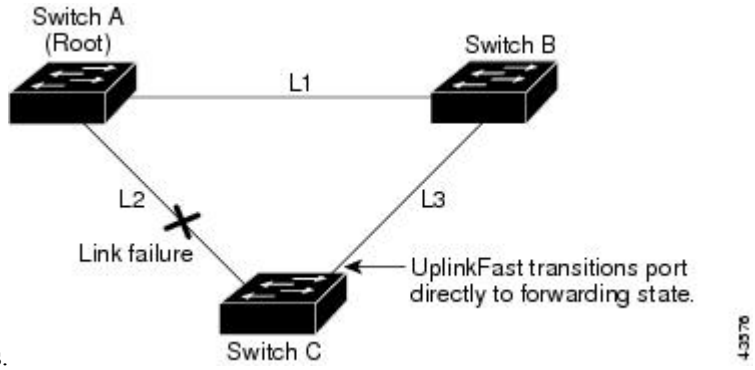


Figure 13: UplinkFast Example After Direct Link Failure

If Switch C detects a link failure on the currently active link L2 on the root port (a direct link failure), UplinkFast unblocks the blocked interface on Switch C and transitions it to the forwarding state without going through the listening and learning states. This change takes approximately 1 to 5 seconds.



Cross-Stack UplinkFast

Cross-Stack UplinkFast (CSUF) provides a fast spanning-tree transition (fast convergence in less than 1 second under normal network conditions) across a switch stack. During the fast transition, an alternate redundant link on the switch stack is placed in the forwarding state without causing temporary spanning-tree loops or loss of connectivity to the backbone. With this feature, you can have a redundant and resilient network in some configurations. CSUF is automatically enabled when you enable the UplinkFast feature.

CSUF might not provide a fast transition all the time; in these cases, the normal spanning-tree transition occurs, completing in 30 to 40 seconds. For more information, see Related Topics.

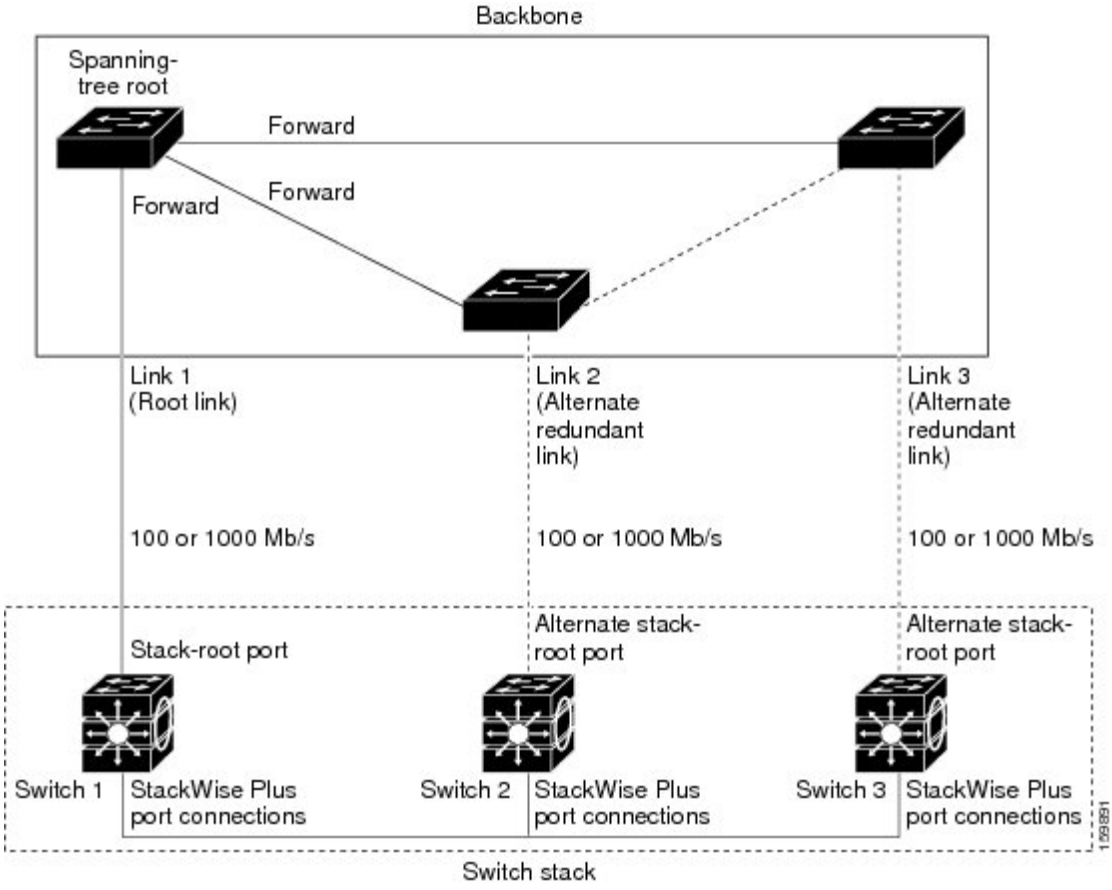
How Cross-Stack UplinkFast Works

Cross-Stack UplinkFast (CSUF) ensures that one link in the stack is elected as the path to the root.

Figure 14: Cross-Stack UplinkFast Topology

The stack-root port on Switch 1 provides the path to the root of the spanning tree. The alternate stack-root ports on Switches 2 and 3 can provide an alternate path to the spanning-tree root if the current stack-root switch fails or if its link to the spanning-tree root fails.

Link 1, the root link, is in the spanning-tree forwarding state. Links 2 and 3 are alternate redundant links that are in the spanning-tree blocking state. If Switch 1 fails, if its stack-root port fails, or if Link 1 fails, CSUF selects either the alternate stack-root port on Switch 2 or Switch 3 and puts it into the forwarding state in less than 1 second.



When certain link loss or spanning-tree events occur (described in the following topic), the Fast Uplink Transition Protocol uses the neighbor list to send fast-transition requests to stack members.

The switch sending the fast-transition request needs to do a fast transition to the forwarding state of a port that it has chosen as the root port, and it must obtain an acknowledgment from each stack switch before performing the fast transition.

Each switch in the stack decides if the sending switch is a better choice than itself to be the stack root of this spanning-tree instance by comparing the root, cost, and bridge ID. If the sending switch is the best choice as the stack root, each switch in the stack returns an acknowledgment; otherwise, it sends a fast-transition request. The sending switch then has not received acknowledgments from all stack switches.

When acknowledgments are received from all stack switches, the Fast Uplink Transition Protocol on the sending switch immediately transitions its alternate stack-root port to the forwarding state. If acknowledgments

from all stack switches are not obtained by the sending switch, the normal spanning-tree transitions (blocking, listening, learning, and forwarding) take place, and the spanning-tree topology converges at its normal rate (2 * forward-delay time + max-age time).

The Fast Uplink Transition Protocol is implemented on a per-VLAN basis and affects only one spanning-tree instance at a time.

Events That Cause Fast Convergence

Depending on the network event or failure, the CSUF fast convergence might or might not occur.

Fast convergence (less than 1 second under normal network conditions) occurs under these circumstances:

- The stack-root port link fails.

  If two switches in the stack have alternate paths to the root, only one of the switches performs the fast transition.

- The failed link, which connects the stack root to the spanning-tree root, recovers.
- A network reconfiguration causes a new stack-root switch to be selected.
- A network reconfiguration causes a new port on the current stack-root switch to be chosen as the stack-root port.

Note The fast transition might not occur if multiple events occur simultaneously. For example, if a stack member is powered off, and at the same time, the link connecting the stack root to the spanning-tree root comes back up, the normal spanning-tree convergence occurs.

Normal spanning-tree convergence (30 to 40 seconds) occurs under these conditions:

- The stack-root switch is powered off, or the software failed.
- The stack-root switch, which was powered off or failed, is powered on.
- A new switch, which might become the stack root, is added to the stack.

BackboneFast

BackboneFast detects indirect failures in the core of the backbone. BackboneFast is a complementary technology to the UplinkFast feature, which responds to failures on links directly connected to access switches. BackboneFast optimizes the maximum-age timer, which controls the amount of time the switch stores protocol information received on an interface. When a switch receives an inferior BPDU from the designated port of another switch, the BPDU is a signal that the other switch might have lost its path to the root, and BackboneFast tries to find an alternate path to the root.

BackboneFast starts when a root port or blocked interface on a switch receives inferior BPDUs from its designated switch. An inferior BPDU identifies a switch that declares itself as both the root bridge and the designated switch. When a switch receives an inferior BPDU, it means that a link to which the switch is not directly connected (an indirect link) has failed (that is, the designated switch has lost its connection to the root switch). Under spanning-tree rules, the switch ignores inferior BPDUs for the maximum aging time (default is 20 seconds).

The switch tries to find if it has an alternate path to the root switch. If the inferior BPDU arrives on a blocked interface, the root port and other blocked interfaces on the switch become alternate paths to the root switch. (Self-looped ports are not considered alternate paths to the root switch.) If the inferior BPDU arrives on the root port, all blocked interfaces become alternate paths to the root switch. If the inferior BPDU arrives on the root port and there are no blocked interfaces, the switch assumes that it has lost connectivity to the root switch, causes the maximum aging time on the root port to expire, and becomes the root switch according to normal spanning-tree rules.

If the switch has alternate paths to the root switch, it uses these alternate paths to send a root link query (RLQ) request. The switch sends the RLQ request on all alternate paths to learn if any stack member has an alternate root to the root switch and waits for an RLQ reply from other switches in the network and in the stack. The switch sends the RLQ request on all alternate paths and waits for an RLQ reply from other switches in the network.

When a stack member receives an RLQ reply from a nonstack member on a blocked interface and the reply is destined for another nonstacked switch, it forwards the reply packet, regardless of the spanning-tree interface state.

When a stack member receives an RLQ reply from a nonstack member and the response is destined for the stack, the stack member forwards the reply so that all the other stack members receive it.

If the switch discovers that it still has an alternate path to the root, it expires the maximum aging time on the interface that received the inferior BPDU. If all the alternate paths to the root switch indicate that the switch has lost connectivity to the root switch, the switch expires the maximum aging time on the interface that received the RLQ reply. If one or more alternate paths can still connect to the root switch, the switch makes all interfaces on which it received an inferior BPDU its designated ports and moves them from the blocking state (if they were in the blocking state), through the listening and learning states, and into the forwarding state.

Figure 15: BackboneFast Example Before Indirect Link Failure

This is an example topology with no link failures. Switch A, the root switch, connects directly to Switch B over link L1 and to Switch C over link L2. The Layer 2 interface on Switch C that connects directly to Switch B is in the blocking state.

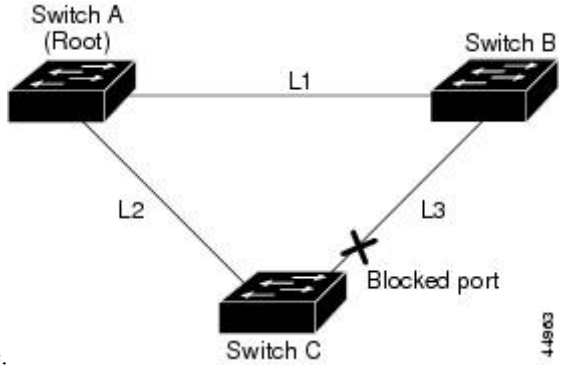
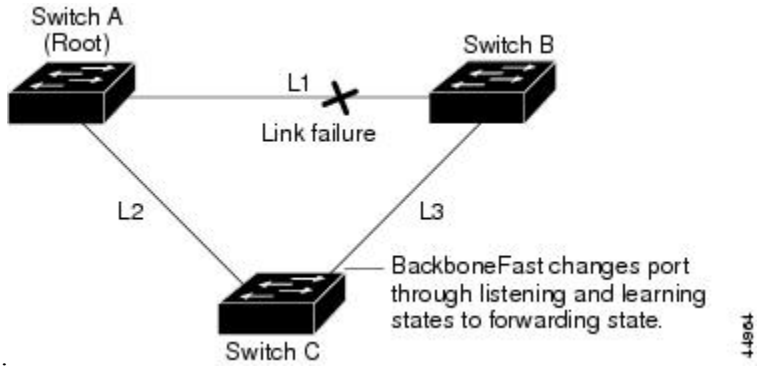


Figure 16: BackboneFast Example After Indirect Link Failure

If link L1 fails, Switch C cannot detect this failure because it is not connected directly to link L1. However, because Switch B is directly connected to the root switch over L1, it detects the failure, elects itself the root, and begins sending BPDUs to Switch C, identifying itself as the root. When Switch C receives the inferior BPDUs from Switch B, Switch C assumes that an indirect failure has occurred. At that point, BackboneFast allows the blocked interface on Switch C to move immediately to the listening state without waiting for the maximum aging time for the interface to expire. BackboneFast then transitions the Layer 2 interface on Switch C to the forwarding state, providing a path from Switch B to Switch A. The root-switch election takes

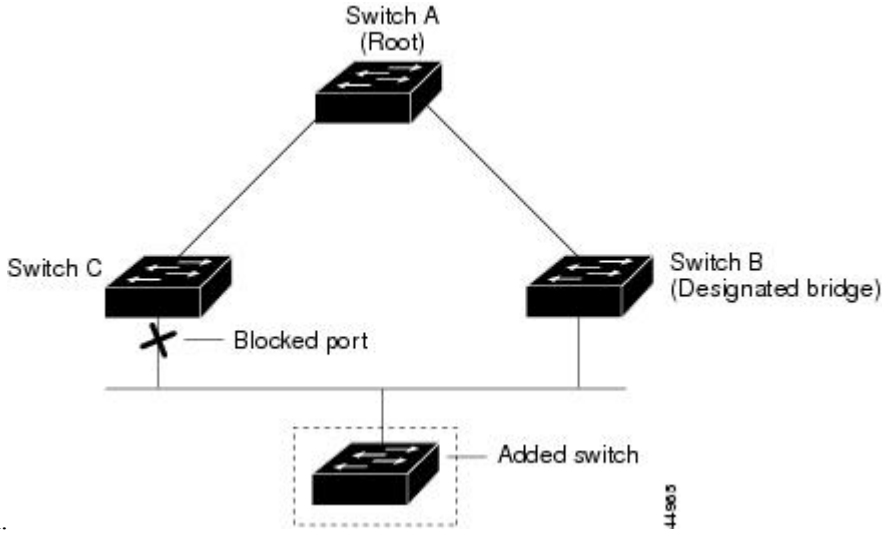
approximately 30 seconds, twice the Forward Delay time if the default Forward Delay time of 15 seconds is set. BackboneFast reconfigures the topology to account for the failure of link



L1.

Figure 17: Adding a Switch in a Shared-Medium Topology

If a new switch is introduced into a shared-medium topology, BackboneFast is not activated because the inferior BPDUs did not come from the recognized designated switch (Switch B). The new switch begins sending inferior BPDUs that indicate it is the root switch. However, the other switches ignore these inferior BPDUs, and the new switch learns that Switch B is the designated switch to Switch A, the root



switch.

EtherChannel Guard

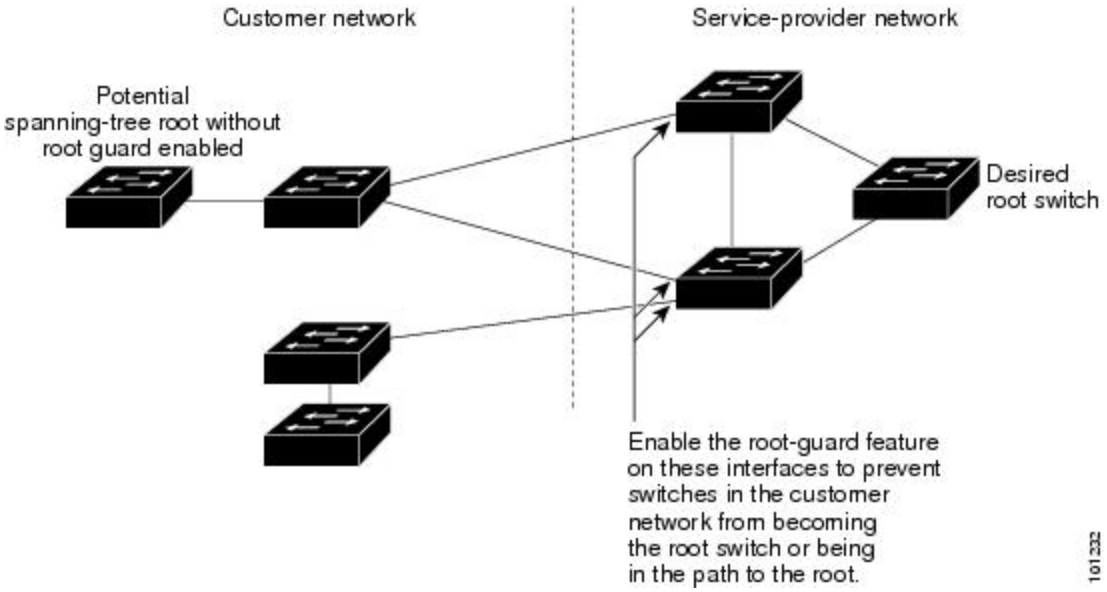
You can use EtherChannel guard to detect an EtherChannel misconfiguration between the switch and a connected device. A misconfiguration can occur if the switch interfaces are configured in an EtherChannel, but the interfaces on the other device are not. A misconfiguration can also occur if the channel parameters are not the same at both ends of the EtherChannel.

If the switch detects a misconfiguration on the other device, EtherChannel guard places the switch interfaces in the error-disabled state, and displays an error message.

Root Guard

Figure 18: Root Guard in a Service-Provider Network

The Layer 2 network of a service provider (SP) can include many connections to switches that are not owned by the SP. In such a topology, the spanning tree can reconfigure itself and select a customer switch as the root switch. You can avoid this situation by enabling root guard on SP switch interfaces that connect to switches in your customer's network. If spanning-tree calculations cause an interface in the customer network to be selected as the root port, root guard then places the interface in the root-inconsistent (blocked) state to prevent the customer's switch from becoming the root switch or being in the path to the root.



If a switch outside the SP network becomes the root switch, the interface is blocked (root-inconsistent state), and spanning tree selects a new root switch. The customer's switch does not become the root switch and is not in the path to the root.

If the switch is operating in multiple spanning-tree (MST) mode, root guard forces the interface to be a designated port. If a boundary port is blocked in an internal spanning-tree (IST) instance because of root guard, the interface also is blocked in all MST instances. A boundary port is an interface that connects to a LAN, the designated switch of which is either an IEEE 802.1D switch or a switch with a different MST region configuration.

Root guard enabled on an interface applies to all the VLANs to which the interface belongs. VLANs can be grouped and mapped to an MST instance.

Caution Misuse of the root guard feature can cause a loss of connectivity.

Loop Guard

You can use loop guard to prevent alternate or root ports from becoming designated ports because of a failure that leads to a unidirectional link. This feature is most effective when it is enabled on the entire switched

network. Loop guard prevents alternate and root ports from becoming designated ports, and spanning tree does not send BPDUs on root or alternate ports.

When the switch is operating in PVST+ or rapid-PVST+ mode, loop guard prevents alternate and root ports from becoming designated ports, and spanning tree does not send BPDUs on root or alternate ports.

When the switch is operating in MST mode, BPDUs are not sent on nonboundary ports only if the interface is blocked by loop guard in all MST instances. On a boundary port, loop guard blocks the interface in all MST instances.

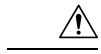
How to Configure Optional Spanning-Tree Features

Enabling PortFast

An interface with the PortFast feature enabled is moved directly to the spanning-tree forwarding state without waiting for the standard forward-time delay.

If you enable the voice VLAN feature, the PortFast feature is automatically enabled. When you disable voice VLAN, the PortFast feature is not automatically disabled.

You can enable this feature if your switch is running PVST+, Rapid PVST+, or MSTP.



Caution Use PortFast only when connecting a single end station to an access or trunk port. Enabling this feature on an interface connected to a switch or hub could prevent spanning tree from detecting and disabling loops in your network, which could cause broadcast storms and address-learning problems.

This procedure is optional.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *interface-id*
4. **spanning-tree portfast** [**trunk**]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

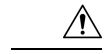
	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/0/2`	Specifies an interface to configure, and enters interface configuration mode.
Step 4	**spanning-tree portfast** [**trunk**] **Example:** `Device(config-if)# spanning-tree portfast trunk`	Enables PortFast on an access port connected to a single workstation or server. By specifying the **trunk** keyword, you can enable PortFast on a trunk port. **Note** To enable PortFast on trunk ports, you must use the **spanning-tree portfast trunk** interface configuration command. The **spanning-tree portfast** command will not work on trunk ports. Make sure that there are no loops in the network between the trunk port and the workstation or server before you enable PortFast on a trunk port. By default, PortFast is disabled on all interfaces.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

What to do next

You can use the **spanning-tree portfast default** global configuration command to globally enable the PortFast feature on all nontrunking ports.

Enabling BPDU Guard

You can enable the BPDU guard feature if your switch is running PVST+, Rapid PVST+, or MSTP.



Caution Configure PortFast edge only on ports that connect to end stations; otherwise, an accidental topology loop could cause a data packet loop and disrupt switch and network operation.

This procedure is optional.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spanning-tree portfast edge bpduguard default**
4. **interface** *interface-id*
5. **spanning-tree portfast edge**
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree portfast edge bpduguard default** **Example:** `Device(config)# spanning-tree portfast edge bpduguard default`	Globally enables BPDU guard.
Step 4	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/0/2`	Specifies the interface connected to an end station, and enters interface configuration mode.
Step 5	**spanning-tree portfast edge** **Example:** `Device(config-if)# spanning-tree portfast edge`	Enables the PortFast edge feature.
Step 6	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

What to do next

To prevent the port from shutting down, you can use the **errdisable detect cause bpduguard shutdown vlan** global configuration command to shut down just the offending VLAN on the port where the violation occurred.

You also can use the **spanning-tree bpduguard enable** interface configuration command to enable BPDU guard on any port without also enabling the PortFast edge feature. When the port receives a BPDU, it is put it in the error-disabled state.

Enabling BPDU Filtering

You can also use the **spanning-tree bpdufilter enable** interface configuration command to enable BPDU filtering on any interface without also enabling the PortFast edge feature. This command prevents the interface from sending or receiving BPDUs.

Caution Enabling BPDU filtering on an interface is the same as disabling spanning tree on it and can result in spanning-tree loops.

You can enable the BPDU filtering feature if your switch is running PVST+, Rapid PVST+, or MSTP.

Caution Configure PortFast edge only on interfaces that connect to end stations; otherwise, an accidental topology loop could cause a data packet loop and disrupt switch and network operation.

This procedure is optional.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spanning-tree portfast edge bpdufilter default**
4. **interface** *interface-id*
5. **spanning-tree portfast edge**
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree portfast edge bpdufilter default** **Example:**	Globally enables BPDU filtering. By default, BPDU filtering is disabled.

	Command or Action	Purpose
	`Device(config)# spanning-tree portfast edge bpdufilter default`	
Step 4	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/0/2`	Specifies the interface connected to an end station, and enters interface configuration mode.
Step 5	**spanning-tree portfast edge** **Example:** `Device(config-if)# spanning-tree portfast edge`	Enables the PortFast edge feature on the specified interface.
Step 6	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Enabling UplinkFast for Use with Redundant Links

Note When you enable UplinkFast, it affects all VLANs on the switch or switch stack. You cannot configure UplinkFast on an individual VLAN.

You can configure the UplinkFast or the Cross-Stack UplinkFast (CSUF) feature for Rapid PVST+ or for the MSTP, but the feature remains disabled (inactive) until you change the spanning-tree mode to PVST+.

This procedure is optional. Follow these steps to enable UplinkFast and CSUF.

Before you begin

UplinkFast cannot be enabled on VLANs that have been configured with a switch priority. To enable UplinkFast on a VLAN with switch priority configured, first restore the switch priority on the VLAN to the default value using the **no spanning-tree vlan** *vlan-id* **priority** global configuration command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spanning-tree uplinkfast** [**max-update-rate** *pkts-per-second*]
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree uplinkfast** [**max-update-rate** *pkts-per-second*] **Example:** `Device(config)# spanning-tree uplinkfast max-update-rate 200`	Enables UplinkFast. (Optional) For *pkts-per-second*, the range is 0 to 32000 packets per second; the default is 150. If you set the rate to 0, station-learning frames are not generated, and the spanning-tree topology converges more slowly after a loss of connectivity. When you enter this command, CSUF also is enabled on all nonstack port interfaces.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

When UplinkFast is enabled, the switch priority of all VLANs is set to 49152. If you change the path cost to a value less than 3000 and you enable UplinkFast or UplinkFast is already enabled, the path cost of all interfaces and VLAN trunks is increased by 3000 (if you change the path cost to 3000 or above, the path cost is not altered). The changes to the switch priority and the path cost reduce the chance that a switch will become the root switch.

When UplinkFast is disabled, the switch priorities of all VLANs and path costs of all interfaces are set to default values if you did not modify them from their defaults.

When you enable the UplinkFast feature using these instructions, CSUF is automatically globally enabled on nonstack port interfaces.

Disabling UplinkFast

This procedure is optional.

Follow these steps to disable UplinkFast and Cross-Stack UplinkFast (CSUF).

Before you begin

UplinkFast must be enabled.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no spanning-tree uplinkfast**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no spanning-tree uplinkfast** **Example:** `Device(config)# no spanning-tree uplinkfast`	Disables UplinkFast and CSUF on the switch and all of its VLANs.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

When UplinkFast is disabled, the switch priorities of all VLANs and path costs of all interfaces are set to default values if you did not modify them from their defaults.

When you disable the UplinkFast feature using these instructions, CSUF is automatically globally disabled on nonstack port interfaces.

Enabling BackboneFast

You can enable BackboneFast to detect indirect link failures and to start the spanning-tree reconfiguration sooner.

You can configure the BackboneFast feature for Rapid PVST+ or for the MSTP, but the feature remains disabled (inactive) until you change the spanning-tree mode to PVST+.

This procedure is optional. Follow these steps to enable BackboneFast on the switch.

Before you begin

If you use BackboneFast, you must enable it on all switches in the network. BackboneFast is not supported on Token Ring VLANs. This feature is supported for use with third-party switches.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spanning-tree backbonefast**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree backbonefast** **Example:** `Device(config)# spanning-tree backbonefast`	Enables BackboneFast.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Enabling EtherChannel Guard

You can enable EtherChannel guard to detect an EtherChannel misconfiguration if your device is running PVST+, Rapid PVST+, or MSTP.

This procedure is optional.

Follow these steps to enable EtherChannel Guard on the device.

SUMMARY STEPS

1. **enable**

2. **configure terminal**
3. **spanning-tree etherchannel guard misconfig**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree etherchannel guard misconfig** **Example:** `Device(config)# spanning-tree etherchannel guard misconfig`	Enables EtherChannel guard.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

What to do next

You can use the **show interfaces status err-disabled** privileged EXEC command to show which device ports are disabled because of an EtherChannel misconfiguration. On the remote device, you can enter the **show etherchannel summary** privileged EXEC command to verify the EtherChannel configuration.

After the configuration is corrected, enter the **shutdown** and **no shutdown** interface configuration commands on the port-channel interfaces that were misconfigured.

Enabling Root Guard

Root guard enabled on an interface applies to all the VLANs to which the interface belongs. Do not enable the root guard on interfaces to be used by the UplinkFast feature. With UplinkFast, the backup interfaces (in the blocked state) replace the root port in the case of a failure. However, if root guard is also enabled, all the backup interfaces used by the UplinkFast feature are placed in the root-inconsistent state (blocked) and are prevented from reaching the forwarding state.

Note You cannot enable both root guard and loop guard at the same time.

You can enable this feature if your switch is running PVST+, Rapid PVST+, or MSTP.

This procedure is optional.

Follow these steps to enable root guard on the switch.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *interface-id*
4. **spanning-tree guard root**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/0/2`	Specifies an interface to configure, and enters interface configuration mode.
Step 4	**spanning-tree guard root** **Example:** `Device(config-if)# spanning-tree guard root`	Enables root guard on the interface. By default, root guard is disabled on all interfaces.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Enabling Loop Guard

You can use loop guard to prevent alternate or root ports from becoming designated ports because of a failure that leads to a unidirectional link. This feature is most effective when it is configured on the entire switched network. Loop guard operates only on interfaces that are considered point-to-point by the spanning tree.

Note You cannot enable both loop guard and root guard at the same time.

You can enable this feature if your device is running PVST+, Rapid PVST+, or MSTP.

This procedure is optional. Follow these steps to enable loop guard on the device.

SUMMARY STEPS

1. Enter one of the following commands:
   - **show spanning-tree active**
   - **show spanning-tree mst**
2. **configure terminal**
3. **spanning-tree loopguard default**
4. **end**

DETAILED STEPS

	Command or Action	Purpose

Step 1	Enter one of the following commands:
	• **show spanning-tree active**
	• **show spanning-tree mst**
	Example:
	`Device# show spanning-tree active`
	or
	`Device# show spanning-tree mst`
		Verifies which interfaces are alternate or root ports.

Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree loopguard default** **Example:** `Device(config)# spanning-tree loopguard default`	Enables loop guard. By default, loop guard is disabled.

	Command or Action	Purpose
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Monitoring the Spanning-Tree Status

Table 9: Commands for Monitoring the Spanning-Tree Status

Command	Purpose
show spanning-tree active	Displays spanning-tree information on active interfaces only.
show spanning-tree detail	Displays a detailed summary of interface information.
show spanning-tree interface *interface-id*	Displays spanning-tree information for the specified interface.
show spanning-tree mst interface *interface-id*	Displays MST information for the specified interface.
show spanning-tree summary [**totals**]	Displays a summary of interface states or displays the total lin spanning-tree state section.
show spanning-tree mst interface *interface-id* **portfast edge**	Displays spanning-tree portfast information for the specified i

Additional References for Optional Spanning Tree Features

Related Documents

Related Topic	Document Title

Standards and RFCs

Standard/RFC	Title
None	—

MIBs

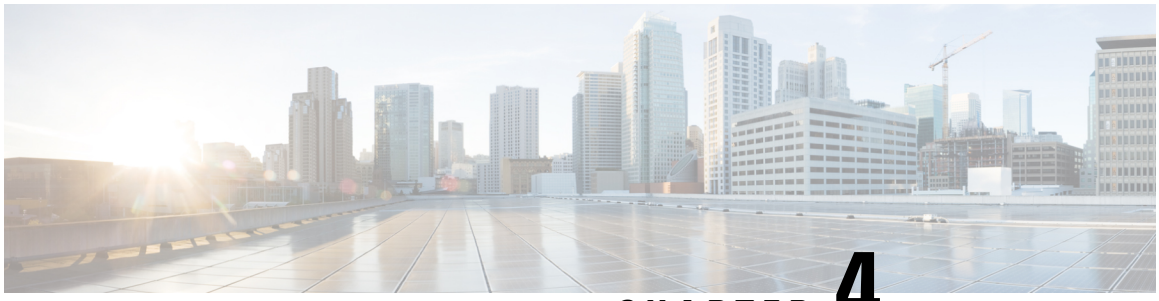
MIB	MIBs Link
All the supported MIBs for this release.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for Optional Spanning-Tree Features

Release	Modification
Cisco IOS XE Everest 16.6.1	This feature was introduced.



CHAPTER 4

Configuring EtherChannels

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/. An account on Cisco.com is not required.

Restrictions for EtherChannels

The following are restrictions for EtherChannels:

- All ports in an EtherChannel must be assigned to the same VLAN or they must be configured as trunk port.

Information About EtherChannels

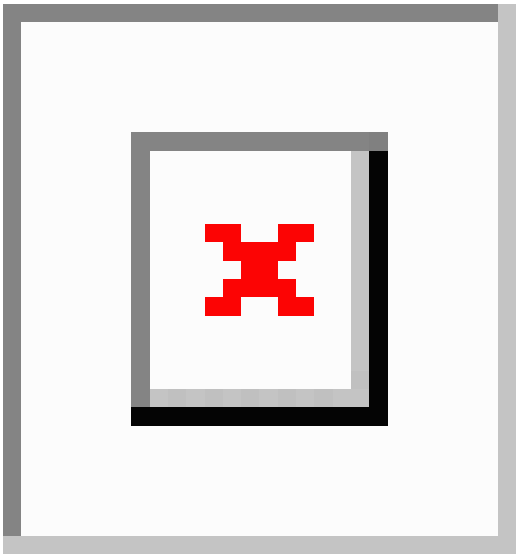
EtherChannel Overview

EtherChannel provides fault-tolerant high-speed links between switches, routers, and servers. You can use the EtherChannel to increase the bandwidth between the wiring closets and the data center, and you can deploy

it anywhere in the network where bottlenecks are likely to occur. EtherChannel provides automatic recovery for the loss of a link by redistributing the load across the remaining links. If a link fails, EtherChannel redirects traffic from the failed link to the remaining links in the channel without intervention.

An EtherChannel consists of individual Ethernet links bundled into a single logical link.

Figure 19: Typical EtherChannel Configuration



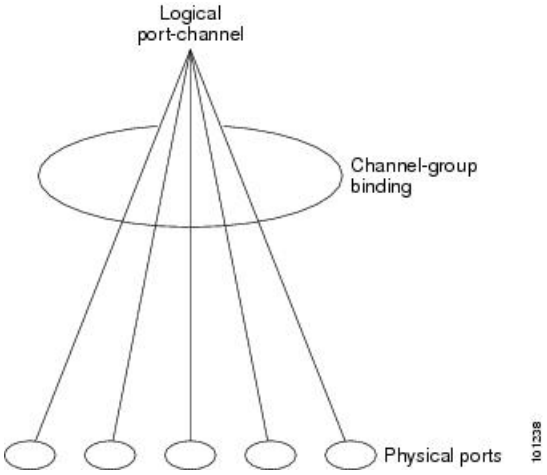
Each EtherChannel can consist of up to eight compatibly configured Ethernet ports.

Channel Groups and Port-Channel Interfaces

An EtherChannel comprises a channel group and a port-channel interface. The channel group binds physical ports to the port-channel interface. Configuration changes applied to the port-channel interface apply to all the physical ports bound together in the channel group.

Figure 20: Relationship of Physical Ports, Channel Group and Port-Channel Interface

The **channel-group** command binds the physical port and the port-channel interface together. Each EtherChannel has a port-channel logical interface numbered from 1 to 252. This port-channel interface number corresponds to the one specified with the **channel-group** interface configuration command.



- With Layer 2 ports, use the **channel-group** interface configuration command to dynamically create the port-channel interface.

  You also can use the **interface port-channel** *port-channel-number* global configuration command to manually create the port-channel interface, but then you must use the **channel-group** *channel-group-number* command to bind the logical interface to a physical port. The *channel-group-number* can be the same as the *port-channel-number*, or you can use a new number. If you use a new number, the **channel-group** command dynamically creates a new port channel.

- With Layer 3 ports, you should manually create the logical interface by using the **interface port-channel** global configuration command followed by the **no switchport** interface configuration command. You then manually assign an interface to the EtherChannel by using the **channel-group** interface configuration command.

Port Aggregation Protocol

The Port Aggregation Protocol (PAgP) is a Cisco-proprietary protocol that can be run only on Cisco devices and on those devices licensed by vendors to support PAgP. PAgP facilitates the automatic creation of EtherChannels by exchanging PAgP packets between Ethernet ports.

By using PAgP, the device or device stack learns the identity of partners capable of supporting PAgP and the capabilities of each port. It then dynamically groups similarly configured ports (on a single device in the stack) into a single logical link (channel or aggregate port). Similarly configured ports are grouped based on hardware, administrative, and port parameter constraints. For example, PAgP groups the ports with the same speed, duplex mode, native VLAN, VLAN range, and trunking status and type. After grouping the links into an EtherChannel, PAgP adds the group to the spanning tree as a single device port.

PAgP Modes

PAgP modes specify whether a port can send PAgP packets, which start PAgP negotiations, or only respond to PAgP packets received.

Table 10: EtherChannel PAgP Modes

Mode	Description
auto	Places a port into a passive negotiating state, in which the port responds to PAgP packets it receives but does not start PAgP packet negotiation. This setting minimizes the transmission of PAgP packets.
desirable	Places a port into an active negotiating state, in which the port starts negotiations with other ports by sending PAgP packets.

Switch ports exchange PAgP packets only with partner ports configured in the **auto** or **desirable** modes. Ports configured in the **on** mode do not exchange PAgP packets.

Both the **auto** and **desirable** modes enable ports to negotiate with partner ports to form an EtherChannel based on criteria such as port speed. and for Layer 2 EtherChannels, based on trunk state and VLAN numbers.

Ports can form an EtherChannel when they are in different PAgP modes as long as the modes are compatible. For example:

- A port in the **desirable** mode can form an EtherChannel with another port that is in the **desirable** or **auto** mode.
- A port in the **auto** mode can form an EtherChannel with another port in the **desirable** mode.

A port in the **auto** mode cannot form an EtherChannel with another port that is also in the **auto** mode because neither port starts PAgP negotiation.

Silent Mode

If your switch is connected to a partner that is PAgP-capable, you can configure the switch port for nonsilent operation by using the **non-silent** keyword. If you do not specify **non-silent** with the **auto** or **desirable** mode, silent mode is assumed.

Use the silent mode when the switch is connected to a device that is not PAgP-capable and seldom, if ever, sends packets. An example of a silent partner is a file server or a packet analyzer that is not generating traffic. In this case, running PAgP on a physical port connected to a silent partner prevents that switch port from ever becoming operational. However, the silent setting allows PAgP to operate, to attach the port to a channel group, and to use the port for transmission.

PAgP Learn Method and Priority

Network devices are classified as PAgP physical learners or aggregate-port learners. A device is a physical learner if it learns addresses by physical ports and directs transmissions based on that knowledge. A device is an aggregate-port learner if it learns addresses by aggregate (logical) ports. The learn method must be configured the same at both ends of the link.

When a device and its partner are both aggregate-port learners, they learn the address on the logical port-channel. The device sends packets to the source by using any of the ports in the EtherChannel. With aggregate-port learning, it is not important on which physical port the packet arrives.

PAgP cannot automatically detect when the partner device is a physical learner and when the local device is an aggregate-port learner. Therefore, you must manually set the learning method on the local device to learn addresses by physical ports. You also must set the load-distribution method to source-based distribution, so that any given source MAC address is always sent on the same physical port.

You also can configure a single port within the group for all transmissions and use other ports for hot-standby. The unused ports in the group can be swapped into operation in just a few seconds if the selected single port loses hardware-signal detection. You can configure which port is always selected for packet transmission by changing its priority with the **pagp port-priority** interface configuration command. The higher the priority, the more likely that the port will be selected.

Note The device supports address learning only on aggregate ports even though the **physical-port** keyword is provided in the CLI. The **pagp learn-method** command and the **pagp port-priority** command have no effect on the device hardware, but they are required for PAgP interoperability with devices that only support address learning by physical ports, such as the Catalyst 1900 switch.

When the link partner of the device is a physical learner, we recommend that you configure the device as a physical-port learner by using the **pagp learn-method physical-port** interface configuration command. Set the load-distribution method based on the source MAC address by using the **port-channel load-balance src-mac** global configuration command. The device then sends packets to the physcial learner using the same port in the EtherChannel from which it learned the source address. Only use the **pagp learn-method** command in this situation.

PAgP Interaction with Other Features

The Dynamic Trunking Protocol (DTP) and the Cisco Discovery Protocol (CDP) send and receive packets over the physical ports in the EtherChannel. Trunk ports send and receive PAgP protocol data units (PDUs) on the lowest numbered VLAN.

In Layer 2 EtherChannels, the first port in the channel that comes up provides its MAC address to the EtherChannel. If this port is removed from the bundle, one of the remaining ports in the bundle provides its MAC address to the EtherChannel. For Layer 3 EtherChannels, the MAC address is allocated by the active device as soon as the interface is created (through the **interface port-channel** global configuration command).

PAgP sends and receives PAgP PDUs only from ports that are up and have PAgP enabled for the auto or desirable mode.

Link Aggregation Control Protocol

The LACP is defined in IEEE 802.3ad and enables Cisco devices to manage Ethernet channels between devices that conform to the IEEE 802.3ad protocol. LACP facilitates the automatic creation of EtherChannels by exchanging LACP packets between Ethernet ports.

By using LACP, the device or device stack learns the identity of partners capable of supporting LACP and the capabilities of each port. It then dynamically groups similarly configured ports into a single logical link (channel or aggregate port). Similarly configured ports are grouped based on hardware, administrative, and port parameter constraints. For example, LACP groups the ports with the same speed, duplex mode, native VLAN, VLAN range, and trunking status and type. After grouping the links into an EtherChannel, LACP adds the group to the spanning tree as a single device port.

The independent mode behavior of ports in a port channel is changed. With CSCtn96950, by default, standalone mode is enabled. When no response is received from an LACP peer, ports in the port channel are moved to suspended state.

LACP Modes

LACP modes specify whether a port can send LACP packets or only receive LACP packets.

Table 11: EtherChannel LACP Modes

Mode	Description
active	Places a port into an active negotiating state in which the port starts negotiations with other ports by sending LACP packets.
passive	Places a port into a passive negotiating state in which the port responds to LACP packets that it receives, but does not start LACP packet negotiation. This setting minimizes the transmission of LACP packets.

Both the **active** and **passive LACP** modes enable ports to negotiate with partner ports to an EtherChannel based on criteria such as port speed, and for Layer 2 EtherChannels, based on trunk state and VLAN numbers.

Ports can form an EtherChannel when they are in different LACP modes as long as the modes are compatible. For example:

- A port in the **active** mode can form an EtherChannel with another port that is in the **active** or **passive** mode.
- A port in the **passive** mode cannot form an EtherChannel with another port that is also in the **passive** mode because neither port starts LACP negotiation.

LACP and Link Redundancy

LACP port-channel operation, bandwidth availability, and link redundancy can be further refined with the LACP port-channel min-links and the LACP max-bundle features.

The LACP port-channel min-links feature:

- Configures the minimum number of ports that must be linked up and bundled in the LACP port channel.
- Prevents a low-bandwidth LACP port channel from becoming active.
- Causes an LACP port channel to become inactive if there are too few active members ports to supply the required minimum bandwidth.

The LACP max-bundle feature:

- Defines an upper limit on the number of bundled ports in an LACP port channel.
- Allows hot-standby ports with fewer bundled ports. For example, in an LACP port channel with five ports, you can specify a max-bundle of three, and the two remaining ports are designated as hot-standby ports.

LACP Interaction with Other Features

The DTP and the CDP send and receive packets over the physical ports in the EtherChannel. Trunk ports send and receive LACP PDUs on the lowest numbered VLAN.

In Layer 2 EtherChannels, the first port in the channel that comes up provides its MAC address to the EtherChannel. If this port is removed from the bundle, one of the remaining ports in the bundle provides its MAC address to the EtherChannel. For Layer 3 EtherChannels, the MAC address is allocated by the active device as soon as the interface is created through the **interface port-channel** global configuration command.

LACP sends and receives LACP PDUs only from ports that are up and have LACP enabled for the active or passive mode.

EtherChannel On Mode

EtherChannel **on** mode can be used to manually configure an EtherChannel. The **on** mode forces a port to join an EtherChannel without negotiations. The **on** mode can be useful if the remote device does not support PAgP or LACP. In the **on** mode, a usable EtherChannel exists only when the devices at both ends of the link are configured in the **on** mode.

Ports that are configured in the **on** mode in the same channel group must have compatible port characteristics, such as speed and duplex. Ports that are not compatible are suspended, even though they are configured in the **on** mode.

Caution You should use care when using the **on** mode. This is a manual configuration, and ports on both ends of the EtherChannel must have the same configuration. If the group is misconfigured, packet loss or spanning-tree loops can occur.

Load-Balancing and Forwarding Methods

EtherChannel balances the traffic load across the links in a channel by reducing part of the binary pattern formed from the addresses in the frame to a numerical value that selects one of the links in the channel. You can specify one of several different load-balancing modes, including load distribution based on MAC addresses, IP addresses, source addresses, destination addresses, or both source and destination addresses. The selected mode applies to all EtherChannels configured on the device.

Note Layer 3 Equal-cost multi path (ECMP) load balancing is based on source IP address, destination IP address, source port, destination port, and layer 4 protocol. Fragmented packets will be treated on two different links based on the algorithm calculated using these parameters. Any changes in one of these parameters will result in load balancing.

MAC Address Forwarding

With source-MAC address forwarding, when packets are forwarded to an EtherChannel, they are distributed across the ports in the channel based on the source-MAC address of the incoming packet. Therefore, to provide load-balancing, packets from different hosts use different ports in the channel, but packets from the same host use the same port in the channel.

With destination-MAC address forwarding, when packets are forwarded to an EtherChannel, they are distributed across the ports in the channel based on the destination host's MAC address of the incoming packet. Therefore, packets to the same destination are forwarded over the same port, and packets to a different destination are sent on a different port in the channel.

With source-and-destination MAC address forwarding, when packets are forwarded to an EtherChannel, they are distributed across the ports in the channel based on both the source and destination MAC addresses. This forwarding method, a combination source-MAC and destination-MAC address forwarding methods of load distribution, can be used if it is not clear whether source-MAC or destination-MAC address forwarding is better suited on a particular device. With source-and-destination MAC-address forwarding, packets sent from host A to host B, host A to host C, and host C to host B could all use different ports in the channel.

IP Address Forwarding

With source-IP address-based forwarding, packets are distributed across the ports in the EtherChannel based on the source-IP address of the incoming packet. To provide load balancing, packets from different IP addresses use different ports in the channel, and packets from the same IP address use the same port in the channel.

With destination-IP address-based forwarding, packets are distributed across the ports in the EtherChannel based on the destination-IP address of the incoming packet. To provide load balancing, packets from the same IP source address sent to different IP destination addresses could be sent on different ports in the channel. Packets sent from different source IP addresses to the same destination IP address are always sent on the same port in the channel.

With source-and-destination IP address-based forwarding, packets are distributed across the ports in the EtherChannel based on both the source and destination IP addresses of the incoming packet. This forwarding method, a combination of source-IP and destination-IP address-based forwarding, can be used if it is not clear whether source-IP or destination-IP address-based forwarding is better suited on a particular device. In this method, packets sent from the IP address A to IP address B, from IP address A to IP address C, and from IP address C to IP address B could all use different ports in the channel.

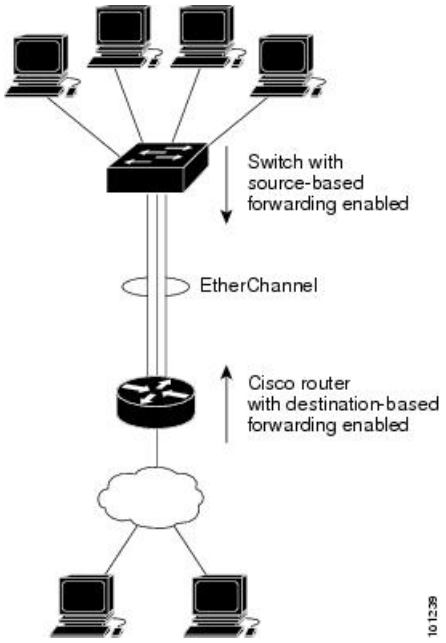
Load-Balancing Advantages

Different load-balancing methods have different advantages, and the choice of a particular load-balancing method should be based on the position of the device in the network and the kind of traffic that needs to be load-distributed.

Figure 21: Load Distribution and Forwarding Methods

In the following figure, an EtherChannel of four workstations communicates with a router. Because the router is a single MAC-address device, source-based forwarding on the device EtherChannel ensures that the device uses all available bandwidth to the router. The router is configured for destination-based forwarding because

the large number of workstations ensures that the traffic is evenly distributed from the router EtherChannel.



Use the option that provides the greatest variety in your configuration. For example, if the traffic on a channel is going only to a single MAC address, using the destination-MAC address always chooses the same link in the channel. Using source addresses or IP addresses might result in better load-balancing.

EtherChannel and Device Stacks

If a stack member that has ports participating in an EtherChannel fails or leaves the stack, the active device removes the failed stack member device ports from the EtherChannel. The remaining ports of the EtherChannel, if any, continue to provide connectivity.

When a device is added to an existing stack, the new device receives the running configuration from the active device and updates itself with the EtherChannel-related stack configuration. The stack member also receives the operational information (the list of ports that are up and are members of a channel).

When two stacks merge that have EtherChannels configured between them, self-looped ports result. Spanning tree detects this condition and acts accordingly. Any PAgP or LACP configuration on a winning device stack is not affected, but the PAgP or LACP configuration on the losing device stack is lost after the stack reboots.

Device Stack and PAgP

With PAgP, if the active device fails or leaves the stack, the standby device becomes the new active device. The new active device synchronizes the configuration of the stack members to that of the active device. The PAgP configuration is not affected after an active device change unless the EtherChannel has ports residing on the old active device.

Device Stacks and LACP

With LACP, the system ID uses the stack MAC address from the active device. When an active device fails or leaves the stack and the standby device becomes the new active device change, the LACP system ID is unchanged. By default, the LACP configuration is not affected after the active device changes.

Default EtherChannel Configuration

The default EtherChannel configuration is described in this table.

Table 12: Default EtherChannel Configuration

Feature	Default Setting
Channel groups	None assigned.
Port-channel logical interface	None defined.
PAgP mode	No default.
PAgP learn method	Aggregate-port learning on all ports.
PAgP priority	128 on all ports.
LACP mode	No default.
LACP learn method	Aggregate-port learning on all ports.
LACP port priority	32768 on all ports.
LACP system priority	32768.
LACP system ID	LACP system priority and the switch or stack MAC address.
Load-balancing	Load distribution on the switch is based on the source-MAC address of the incoming packet. The source-MAC address is **src-mac**.

EtherChannel Configuration Guidelines

If improperly configured, some EtherChannel ports are automatically disabled to avoid network loops and other problems. Follow these guidelines to avoid configuration problems:

- A maximum of 128 EtherChannels (non-StackWise Virtual setup) and 126 EtherChannels (StackWise Virtual setup) are supported on a switch or switch stack.
- Configure a PAgP EtherChannel with up to eight Ethernet ports of the same type.
- Configure a LACP EtherChannel with up to 16 Ethernet ports of the same type. Up to eight ports can be active, and up to eight ports can be in standby mode.
- Configure all ports in an EtherChannel to operate at the same speeds and duplex modes.
- Enable all ports in an EtherChannel. A port in an EtherChannel that is disabled by using the **shutdown** interface configuration command is treated as a link failure, and its traffic is transferred to one of the remaining ports in the EtherChannel.

- When a group is first created, all ports follow the parameters set for the first port to be added to the group. If you change the configuration of one of these parameters, you must also make the changes to all ports in the group:
  - Allowed-VLAN list
  - Spanning-tree path cost for each VLAN
  - Spanning-tree port priority for each VLAN
  - Spanning-tree Port Fast setting
- Do not configure a port to be a member of more than one EtherChannel group.
- Do not configure an EtherChannel in both the PAgP and LACP modes. EtherChannel groups running PAgP and LACP can coexist on the same switch or on different switches in the stack. Individual EtherChannel groups can run either PAgP or LACP, but they cannot interoperate.
- Do not configure a port that is an active or a not-yet-active member of an EtherChannel as an IEEE 802.1x port. If you try to enable IEEE 802.1x on an EtherChannel port, an error message appears, and IEEE 802.1x is not enabled.
- If EtherChannels are configured on device interfaces, remove the EtherChannel configuration from the interfaces before globally enabling IEEE 802.1x on a device by using the **dot1x system-auth-control** global configuration command.

Layer 2 EtherChannel Configuration Guidelines

When configuring Layer 2 EtherChannels, follow these guidelines:

- Assign all ports in the EtherChannel to the same VLAN, or configure them as trunks. Ports with different native VLANs cannot form an EtherChannel.
- An EtherChannel supports the same allowed range of VLANs on all the ports in a trunking Layer 2 EtherChannel. If the allowed range of VLANs is not the same, the ports do not form an EtherChannel even when PAgP is set to the **auto** or **desirable** mode.
- Ports with different spanning-tree path costs can form an EtherChannel if they are otherwise compatibly configured. Setting different spanning-tree path costs does not, by itself, make ports incompatible for the formation of an EtherChannel.

Layer 3 EtherChannel Configuration Guidelines

For Layer 3 EtherChannels, assign the Layer 3 address to the port-channel logical interface, not to the physical ports in the channel.

Auto-LAG

The auto-LAG feature provides the ability to auto create EtherChannels on ports connected to a switch. By default, auto-LAG is disabled globally and is enabled on all port interfaces. The auto-LAG applies to a switch only when it is enabled globally.

On enabling auto-LAG globally, the following scenarios are possible:

- All port interfaces participate in creation of auto EtherChannels provided the partner port interfaces have EtherChannel configured on them. For more information, see the *"The supported auto-LAG configurations between the actor and partner devices"* table below.
- Ports that are already part of manual EtherChannels cannot participate in creation of auto EtherChannels.
- When auto-LAG is disabled on a port interface that is already a part of an auto created EtherChannel, the port interface will unbundle from the auto EtherChannel.

The following table shows the supported auto-LAG configurations between the actor and partner devices:

Table 13: The supported auto-LAG configurations between the actor and partner devices

Actor/Partner	Active	Passive	Auto
Active	Yes	Yes	Yes
Passive	Yes	No	Yes
Auto	Yes	Yes	Yes

On disabling auto-LAG globally, all auto created Etherchannels become manual EtherChannels.

You cannot add any configurations in an existing auto created EtherChannel. To add, you should first convert it into a manual EtherChannel by executing the **port-channel**<*channel-number*>**persistent**.

Note Auto-LAG uses the LACP protocol to create auto EtherChannel. Only one EtherChannel can be automatically created with the unique partner devices.

Auto-LAG Configuration Guidelines

Follow these guidelines when configuring the auto-LAG feature.

- When auto-LAG is enabled globally and on the port interface , and if you do not want the port interface to become a member of the auto EtherChannel, disable the auto-LAG on the port interface.
- A port interface will not bundle to an auto EtherChannel when it is already a member of a manual EtherChannel. To allow it to bundle with the auto EtherChannel, first unbundle the manual EtherChannel on the port interface.
- When auto-LAG is enabled and auto EtherChannel is created, you can create multiple EtherChannels manually with the same partner device. But by default, the port tries to create auto EtherChannel with the partner device.
- The auto-LAG is supported only on Layer 2 EtherChannel. It is not supported on Layer 3 interface and Layer 3 EtherChannel.
- The auto-LAG is supported on cross-stack EtherChannel.

How to Configure EtherChannels

After you configure an EtherChannel, configuration changes applied to the port-channel interface apply to all the physical ports assigned to the port-channel interface, and configuration changes applied to the physical port affect only the port where you apply the configuration.

Configuring Layer 2 EtherChannels

You configure Layer 2 EtherChannels by assigning ports to a channel group with the **channel-group** interface configuration command. This command automatically creates the port-channel logical interface.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface-id*
3. **switchport mode** {**access** | **trunk**}
4. **switchport access vlan** *vlan-id*
5. **channel-group** *channel-group-number* **mode** {**auto** [**non-silent**] | **desirable** [**non-silent**] | **on** } | { **active** | **passive**}
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet`	Specifies a physical port, and enters interface configuration mode. Valid interfaces are physical ports. For a PAgP EtherChannel, you can configure up to eight ports of the same type and speed for the same group. For a LACP EtherChannel, you can configure up to 16 Ethernet ports of the same type. Up to eight ports can be active, and up to eight ports can be in standby mode.
Step 3	**switchport mode** {**access**	**trunk**} **Example:** `Device(config-if)# switchport mode access`	Assigns all ports as static-access ports in the same VLAN, or configure them as trunks. If you configure the port as a static-access port, assign it to only one VLAN. The range is 1 to 4094.

	Command or Action	Purpose
Step 4	**switchport access vlan** *vlan-id* **Example:** `Device(config-if)# switchport access vlan 22`	(Optional) If you configure the port as a static-access port, assign it to only one VLAN. The range is 1 to 4094.
Step 5	**channel-group** *channel-group-number* **mode** {**auto** [**non-silent**] | **desirable** [**non-silent**] | **on** } | { **active** | **passive**} **Example:** `Device(config-if)# channel-group 5 mode auto`	Assigns the port to a channel group, and specifies the PAgP or the LACP mode. For **mode**, select one of these keywords: • **auto** –Enables PAgP only if a PAgP device is detected. It places the port into a passive negotiating state, in which the port responds to PAgP packets it receives but does not start PAgP packet negotiation.. • **desirable** –Unconditionally enables PAgP. It places the port into an active negotiating state, in which the port starts negotiations with other ports by sending PAgP packets. . • **on** –Forces the port to channel without PAgP or LACP. In the **on** mode, an EtherChannel exists only when a port group in the **on** mode is connected to another port group in the **on** mode. • **non-silent** –(Optional) If your device is connected to a partner that is PAgP-capable, configures the device port for nonsilent operation when the port is in the **auto** or **desirable** mode. If you do not specify **non-silent**, silent is assumed. The silent setting is for connections to file servers or packet analyzers. This setting allows PAgP to operate, to attach the port to a channel group, and to use the port for transmission. • **active**—Enables LACP only if a LACP device is detected. It places the port into an active negotiating state in which the port starts negotiations with other ports by sending LACP packets. • **passive** –Enables LACP on the port and places it into a passive negotiating state in which the port responds to LACP packets that it receives, but does not start LACP packet negotiation.
Step 6	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Configuring Layer 3 EtherChannels

Follow these steps to assign an Ethernet port to a Layer 3 EtherChannel. This procedure is required.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *interface-id*
4. **no ip address**
5. **no switchport**
6. **channel-group** *channel-group-number* **mode** { **auto** [**non-silent**] | **desirable** [**non-silent**] | **on** } | { **active** | **passive** }
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/0/2`	Specifies a physical port, and enters interface configuration mode. Valid interfaces include physical ports. For a PAgP EtherChannel, you can configure up to eight ports of the same type and speed for the same group. For a LACP EtherChannel, you can configure up to 16 Ethernet ports of the same type. Up to eight ports can be active, and up to eight ports can be in standby mode.
Step 4	**no ip address** **Example:** `Device(config-if)# no ip address`	Ensures that there is no IP address assigned to the physical port.
Step 5	**no switchport** **Example:**	Puts the port into Layer 3 mode.

	Command or Action	Purpose
	`Device(config-if)# no switchport`	
Step 6	**channel-group** *channel-group-number* **mode** { **auto** [**non-silent**] | **desirable** [**non-silent**] | **on** } | { **active** | **passive** } **Example:** `Device(config-if)# channel-group 5 mode auto`	Assigns the port to a channel group, and specifies the PAgP or the LACP mode. For **mode**, select one of these keywords: • **auto**—Enables PAgP only if a PAgP device is detected. It places the port into a passive negotiating state, in which the port responds to PAgP packets it receives but does not start PAgP packet negotiation. This keyword is not supported when EtherChannel members are from different devices in the device stack. • **desirable**—Unconditionally enables PAgP. It places the port into an active negotiating state, in which the port starts negotiations with other ports by sending PAgP packets. This keyword is not supported when EtherChannel members are from different devices in the device stack. • **on**—Forces the port to channel without PAgP or LACP. In the **on** mode, an EtherChannel exists only when a port group in the **on** mode is connected to another port group in the **on** mode. • **non-silent**—(Optional) If your device is connected to a partner that is PAgP capable, configures the device port for nonsilent operation when the port is in the **auto** or **desirable** mode. If you do not specify **non-silent**, silent is assumed. The silent setting is for connections to file servers or packet analyzers. This setting allows PAgP to operate, to attach the port to a channel group, and to use the port for transmission. • **active**—Enables LACP only if a LACP device is detected. It places the port into an active negotiating state in which the port starts negotiations with other ports by sending LACP packets. • **passive** –Enables LACP on the port and places it into a passive negotiating state in which the port responds to LACP packets that it receives, but does not start LACP packet negotiation.
Step 7	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Configuring EtherChannel Load-Balancing

You can configure EtherChannel load-balancing to use one of several different forwarding methods.

This task is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**port-channel load-balance** {**dst-ip** | **dst-mac** | **dst-mixed-ip-port** | **dst-port** | **extended** | **src-dst-ip** | **src-dst-mac** | **src-dst-mixed-ip-port** | **src-dst-port** | **src-ip** | **src-mac** | **src-mixed-ip-port** | **src-port** } **Example:** `Device(config)# port-channel load-balance src-mac`	Configures an EtherChannel load-balancing method. The default is **src-mac**. Select one of these load-distribution methods: • **dst-ip**—Specifies destination-host IP address. • **dst-mac**—Specifies the destination-host MAC address of the incoming packet. • **dst-mixed-ip-port**—Specifies the host IP address and TCP/UDP port. • **dst-port**—Specifies the destination TCP/UDP port. • **src-dst-ip**—Specifies the source and destination host IP address. • **src-dst-mac**—Specifies the source and destination host MAC address. • **src-dst-mixed-ip-port**—Specifies the source and destination host IP address and TCP/UDP port. • **src-dst-port**—Specifies the source and destination TCP/UDP port. • **extended**—Specifies extended load balance methods--combinations of source and destination methods beyond those available with the standard command. • **src-ip**—Specifies the source host IP address. • **src-mac**—Specifies the source MAC address of the incoming packet.

	Command or Action	Purpose
		• **src-mixed-ip-port**—Specifies the source host IP address and TCP/UDP port. • **src-port**—Specifies the source TCP/UDP port.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring EtherChannel Extended Load-Balancing

Configure EtherChannel extended load-balancing when you want to use a combination of load-balancing methods.

This task is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**port-channel load-balance extended** [**dst-ip** | **dst-mac dst-port** | **ipv6-label** | **l3-proto** | **src-ip** | **src-mac** | **src-port**] **Example:** `Device(config)# port-channel load-balance extended dst-ip dst-mac src-ip`	Configures an EtherChannel extended load-balancing method. The default is **src-mac**. Select one of these load-distribution methods: • **dst-ip**—Specifies destination-host IP address. • **dst-mac**—Specifies the destination-host MAC address of the incoming packet. • **dst-port**—Specifies the destination TCP/UDP port. • **ipv6-label**—Specifies the IPv6 flow label. • **l3-proto**—Specifies the Layer 3 protocol. • **src-ip**—Specifies the source host IP address. • **src-mac**—Specifies the source MAC address of the incoming packet. • **src-port**—Specifies the source TCP/UDP port.

	Command or Action	Purpose
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring the PAgP Learn Method and Priority

This task is optional.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface-id*
3. **pagp learn-method physical-port**
4. **pagp port-priority** *priority*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet`	Specifies the port for transmission, and enters interface configuration mode.
Step 3	**pagp learn-method physical-port** **Example:** `Device(config-if)# pagp learn-method physical port`	Selects the PAgP learning method. By default, **aggregation-port learning** is selected, which means the device sends packets to the source by using any of the ports in the EtherChannel. With aggregate-port learning, it is not important on which physical port the packet arrives. Selects **physical-port** to connect with another device that `is` a physical learner. Make sure to configure the **port-channel load-balance** global configuration command to **src-mac**. The learning method must be configured the same at both ends of the link.

	Command or Action	Purpose
Step 4	**pagp port-priority** *priority* **Example:** `Device(config-if)# pagp port-priority 200`	Assigns a priority so that the selected port is chosen for packet transmission. For *priority*, the range is 0 to 255. The default is 128. The higher the priority, the more likely that the port will be used for PAgP transmission.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Configuring LACP Hot-Standby Ports

When LACP is enabled, the software, by default, tries to configure the maximum number of LACP-compatible ports in a channel, up to a maximum of 16 ports. Only eight LACP links can be active at one time; the remaining eight links are placed in hot-standby mode. If one of the active links becomes inactive, a link that is in the hot-standby mode becomes active in its place.

You can override the default behavior by specifying the maximum number of active ports in a channel, in which case, the remaining ports become hot-standby ports. For example, if you specify a maximum of five ports in a channel, up to 11 ports become hot-standby ports.

If you configure more than eight links for an EtherChannel group, the software automatically decides which of the hot-standby ports to make active based on the LACP priority. To every link between systems that operate LACP, the software assigns a unique priority made up of these elements (in priority order):

- LACP system priority
- System ID (the device MAC address)
- LACP port priority
- Port number

In priority comparisons, numerically lower values have higher priority. The priority decides which ports should be put in standby mode when there is a hardware limitation that prevents all compatible ports from aggregating.

Determining which ports are active and which are hot standby is a two-step procedure. First the system with a numerically lower system priority and system ID is placed in charge of the decision. Next, that system decides which ports are active and which are hot standby, based on its values for port priority and port number. The port priority and port number values for the other system are not used.

You can change the default values of the LACP system priority and the LACP port priority to affect how the software selects active and standby links.

Configuring the LACP Max Bundle Feature

When you specify the maximum number of bundled LACP ports allowed in a port channel, the remaining ports in the port channel are designated as hot-standby ports.

Beginning in privileged EXEC mode, follow these steps to configure the maximum number of LACP ports in a port channel. This procedure is optional.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *channel-number*
3. **lacp max-bundle** *max-bundle-number*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface port-channel** *channel-number* **Example:** `Device(config)# interface port-channel 2`	Enters interface configuration mode for a port channel. The range is 1 to 128.
Step 3	**lacp max-bundle** *max-bundle-number* **Example:** `Device(config-if)# lacp max-bundle 3`	Specifies the maximum number of LACP ports in the port-channel bundle. The range is 1 to 8.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring LACP Port-Channel Standalone Disable

To disable the standalone EtherChannel member port state on a port channel, perform this task on the port channel interface:

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *channel-group*
3. **port-channel standalone-disable**
4. **end**
5. **show etherchannel**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface port-channel** *channel-group* **Example:** `Device(config)# interface port-channel channel-group`	Selects a port channel interface to configure.
Step 3	**port-channel standalone-disable** **Example:** `Device(config-if)# port-channel standalone-disable`	Disables the standalone mode on the port-channel interface.
Step 4	**end** **Example:** `Device(config-if)# end`	Exits configuration mode.

Step 5	**show etherchannel**
	Example:
	`Device# show etherchannel channel-group port-channel`
	`Device# show etherchannel channel-group detail`
		Verifies the configuration.

Configuring the LACP Port Channel Min-Links

You can specify the minimum number of active ports that must be in the link-up state and bundled in an EtherChannel for the port channel interface to transition to the link-up state. Using EtherChannel min-links, you can prevent low-bandwidth LACP EtherChannels from becoming active. Port channel min-links also cause LACP EtherChannels to become inactive if they have too few active member ports to supply the requiredminimum bandwidth.

To configure the minimum number of links that are required for a port channel. Perform the following tasks.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *channel-number* **Example:** `Device(config)# interface port-channel 2`	Enters interface configuration mode for a port-channel. For *channel-number*, the range is 1 to 252.
Step 4	**port-channel min-links** *min-links-number* **Example:** `Device(config-if)# port-channel min-links 3`	Specifies the minimum number of member ports that must be in the link-up state and bundled in the EtherChannel for the port channel interface to transition to the link-up state. For *min-links-number*, the range is 2 to 8.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring the LACP System Priority

You can configure the system priority for all the EtherChannels that are enabled for LACP by using the **lacp system-priority** global configuration command. You cannot configure a system priority for each LACP-configured channel. By changing this value from the default, you can affect how the software selects active and standby links.

You can use the **show etherchannel summary** privileged EXEC command to see which ports are in the hot-standby mode (denoted with an H port-state flag).

Follow these steps to configure the LACP system priority. This procedure is optional.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **lacp system-priority** *priority*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**lacp system-priority** *priority* **Example:** `Device(config)# lacp system-priority 32000`	Configures the LACP system priority. The range is 1 to 65535. The default is 32768. The lower the value, the higher the system priority.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring the LACP Port Priority

By default, all ports use the same port priority. If the local system has a lower value for the system priority and the system ID than the remote system, you can affect which of the hot-standby links become active first by changing the port priority of LACP EtherChannel ports to a lower value than the default. The hot-standby ports that have lower port numbers become active in the channel first. You can use the **show etherchannel summary** privileged EXEC command to see which ports are in the hot-standby mode (denoted with an H port-state flag).

Note If LACP is not able to aggregate all the ports that are compatible (for example, the remote system might have more restrictive hardware limitations), all the ports that cannot be actively included in the EtherChannel are put in the hot-standby state and are used only if one of the channeled ports fails.

Follow these steps to configure the LACP port priority. This procedure is optional.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *interface-id*
4. **lacp port-priority** *priority*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet`	Specifies the port to be configured, and enters interface configuration mode.
Step 4	**lacp port-priority** *priority* **Example:** `Device(config-if)# lacp port-priority 32000`	Configures the LACP port priority. The range is 1 to 65535. The default is 32768. The lower the value, the more likely that the port will be used for LACP transmission.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Configuring LACP Fast Rate Timer

You can change the LACP timer rate to modify the duration of the LACP timeout. Use the **lacp rate** command to set the rate at which LACP control packets are received by an LACP-supported interface. You can change the timeout rate from the default rate (30 seconds) to the fast rate (1 second). This command is supported only on LACP-enabled interfaces.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** {**fastethernet** | **gigabitethernet** | **tengigabitethernet**} *slot/port*
4. **lacp rate** {**normal** | **fast**}
5. **end**
6. **show lacp internal**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** {**fastethernet** | **gigabitethernet** | **tengigabitethernet**} *slot/port* **Example:** `Device(config)# interface gigabitEthernet 2/1`	Configures an interface and enters interface configuration mode.
Step 4	**lacp rate** {**normal** | **fast**} **Example:** `Device(config-if)# lacp rate fast`	Configures the rate at which LACP control packets are received by an LACP-supported interface. • To reset the timeout rate to its default, use the **no lacp rate** command.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Step 6	**show lacp internal**
	Example:
	`Device# show lacp internal`
	`Device# show lacp counters`
		Verifies your configuration.

Configuring Auto-LAG Globally

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. [**no**] **port-channel auto**
4. **end**
5. **show etherchannel auto**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	[**no**] **port-channel auto** **Example:** `Device(config)# port-channel auto`	Enables the auto-LAG feature on a switch globally. Use the no form of this command to disable the auto-LAG feature on the switch globally. **Note** By default, the auto-LAG feature is enabled on the port.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show etherchannel auto** **Example:** `Device# show etherchannel auto`	Displays that EtherChannel is created automatically.

Configuring Auto-LAG on a Port Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *interface-id*
4. [**no**] **channel-group auto**
5. **end**
6. **show etherchannel auto**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet`	Specifies the port interface to be enabled for auto-LAG, and enters interface configuration mode.
Step 4	[**no**] **channel-group auto** **Example:** `Device(config-if)# channel-group auto`	(Optional) Enables auto-LAG feature on individual port interface. Use the no form of this command to disable the auto-LAG feature on individual port interface. **Note** By default, the auto-LAG feature is enabled on the port.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 6	**show etherchannel auto** **Example:** `Device# show etherchannel auto`	Displays that EtherChannel is created automatically.

What to do next

Configuring Persistence with Auto-LAG

You use the persistence command to convert the auto created EtherChannel into a manual one and allow you to add configuration on the existing EtherChannel.

SUMMARY STEPS

1. **enable**
2. **port-channel** *channel-number* **persistent**
3. **show etherchannel summary**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**port-channel** *channel-number* **persistent** **Example:** `Device# port-channel 1 persistent`	Converts the auto created EtherChannel into a manual one and allows you to add configuration on the EtherChannel.
Step 3	**show etherchannel summary** **Example:** `Device# show etherchannel summary`	Displays the EtherChannel information.

Monitoring EtherChannel, PAgP, and LACP Status

You can display EtherChannel, PAgP, and LACP status using the commands listed in this table.

Table 14: Commands for Monitoring EtherChannel, PAgP, and LACP Status

Command	Description
clear lacp { *channel-group-number* **counters** | **counters** }	Clears LACP channel-group information and traffic counters.
clear pagp { *channel-group-number* **counters** | **counters** }	Clears PAgP channel-group information and traffic counters.
show etherchannel [*channel-group-number* { **detail** | **load-balance** | **port** | **port-channel** | **protocol** | **summary** }] [**detail** | **load-balance** | **port** | **port-channel** | **protocol** | **auto** | **summary**]	Displays EtherChannel information in a brief, detailed, and one-line summary form. Also displays the load-balance or frame-distribution scheme, port, port-channel, protocol, and Auto-LAG information.
show pagp [*channel-group-number*] { **counters** | **internal** | **neighbor** }	Displays PAgP information such as traffic information, the internal PAgP configuration, and neighbor information.
show pagp [*channel-group-number*] **dual-active**	Displays the dual-active detection status.
show lacp [*channel-group-number*] { **counters** | **internal** | **neighbor** | **sys-id**}	Displays LACP information such as traffic information, the internal LACP configuration, and neighbor information.
show running-config	Verifies your configuration entries.
show etherchannel load-balance	Displays the load balance or frame distribution scheme among ports in the port channel.

Configuration Examples for Configuring EtherChannels

Configuring Layer 2 EtherChannels: Examples

This example shows how to configure an EtherChannel on a single device in the stack. It assigns two ports as static-access ports in VLAN 10 to channel 5 with the PAgP mode **desirable**:

```
Device# configure terminal
Device(config)# interface range gigabitethernet -2
Device(config-if-range)# switchport mode access
Device(config-if-range)# switchport access vlan 10
Device(config-if-range)# channel-group 5 mode desirable non-silent
Device(config-if-range)# end
```

This example shows how to configure an EtherChannel on a single device in the stack. It assigns two ports as static-access ports in VLAN 10 to channel 5 with the LACP mode **active :**

```
Device# configure terminal
Device(config)# interface range gigabitethernet -2
Device(config-if-range)# switchport mode access
Device(config-if-range)# switchport access vlan 10
Device(config-if-range)# channel-group 5 mode active
Device(config-if-range)# end
```

This example shows how to configure a cross-stack EtherChannel. It uses LACP passive mode and assigns two ports on stack member 1 and one port on stack member 2 as static-access ports in VLAN 10 to channel 5:

```
Device# configure terminal
Device(config)# interface range gigabitethernet -5
Device(config-if-range)# switchport mode access
Device(config-if-range)# switchport access vlan 10
Device(config-if-range)# channel-group 5 mode passive
Device(config-if-range)# exit
Device(config)# interface gigabitethernet
Device(config-if)# switchport mode access
Device(config-if)# switchport access vlan 10
Device(config-if)# channel-group 5 mode passive
Device(config-if)# exit
```

PoE or LACP negotiation errors may occur if you configure two ports from switch to the access point (AP). This scenario can be avoided if the port channel configuration is on the switch side. For more details, see the following example:

```
interface Port-channel1
  switchport access vlan 20
 switchport mode access
  switchport nonegotiate
  no port-channel standalone-disable   <--this one
  spanning-tree portfast
```

Note If the port reports LACP errors on port flap, you should include the following command as well: **no errdisable detect cause pagp-flap**

Configuring Layer 3 EtherChannels: Examples

This example shows how to configure a Layer 3 EtherChannel. It assigns two ports to channel 5 with the LACP mode **active**:

```
Device# configure terminal
Device(config)# interface range gigabitethernet2/0/1 -2
Device(config-if-range)# no ip address
Device(config-if-range)# no switchport
Device(config-if-range)# channel-group 5 mode active
Device(config-if-range)# end
```

This example shows how to configure a cross-stack Layer 3 EtherChannel. It assigns two ports on stack member 2 and one port on stack member 3 to channel 7 using LACP active mode:

```
Device# configure terminal
Device(config)# interface range gigabitethernet2/0/4 -5
Device(config-if-range)# no ip address
Device(config-if-range)# no switchport
Device(config-if-range)# channel-group 7 mode active
Device(config-if-range)# exit
Device(config)# interface gigabitethernet3/0/3
Device(config-if)# no ip address
Device(config-if)# no switchport
Device(config-if)# channel-group 7 mode active
Device(config-if)# exit
```

Configuring LACP Hot-Standby Ports: Example

This example shows how to configure an Etherchannel (port channel 2) that will be active when there are at least three active ports, will comprise up to seven active ports and the remaining ports (up to nine) as hot-standby ports :

```
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# port-channel min-links 3
Device(config-if)# lacp max-bundle 7
```

Configuring Auto LAG: Examples

This example shows how to configure Auto-LAG on a switch

```
device> enable
device# configure terminal
device (config)# port-channel auto
device (config-if)# end
device#  show etherchannel auto
```

The following example shows the summary of EtherChannel that was created automatically.

```
device# show etherchannel auto
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator
        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port
        A - formed by Auto LAG

Number of channel-groups in use: 1
Number of aggregators:           1

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
1      Po1(SUA)        LACP      Gi1/0/45(P) Gi2/0/21(P) Gi3/0/21(P)
```

The following example shows the summary of auto EtherChannel after executing the **port-channel** 1 **persistent** command.

```
device# port-channel 1 persistent

device# show etherchannel summary
Switch# show etherchannel summary
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator
        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port
        A - formed by Auto LAG

Number of channel-groups in use: 1
Number of aggregators:           1

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
1      Po1(SU)         LACP      Gi1/0/45(P) Gi2/0/21(P) Gi3/0/21(P)
```

Additional References for EtherChannels

Related Documents

Related Topic	Document Title
Layer 2 command reference	

Related Topic	Document Title
For complete syntax and usage information for the commands used in this chapter.	See the Layer 2/3 Commands section of the*Command Reference (Catalyst 9400 Series Switches)*

Standards and RFCs

Standard/RFC	Title
None	—

MIBs

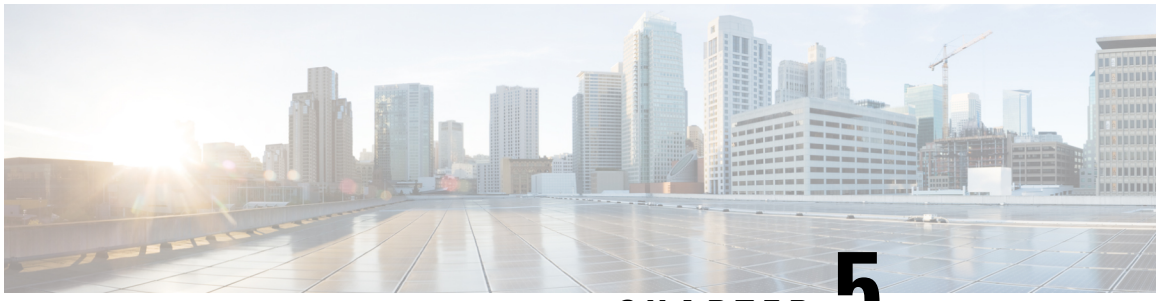
MIB	MIBs Link
All the supported MIBs for this release.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for EtherChannels

Release	Modification
Cisco IOS XE Everest 16.6.1	This feature was introduced.



CHAPTER 5

Configuring Resilient Ethernet Protocol

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/. An account on Cisco.com is not required.

Information About Resilient Ethernet Protocol

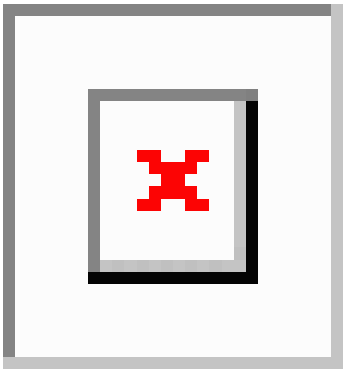
Resilient Ethernet Protocol (REP) is a Cisco proprietary protocol that provides an alternative to Spanning Tree Protocol (STP) to control network loops, handle link failures, and improve convergence time. REP controls a group of ports connected in a segment, ensures that the segment does not create any bridging loops, and responds to link failures within the segment. REP provides a basis for constructing more complex networks and supports VLAN load balancing.

Note The feature is supported on Cisco Catalyst Series Switches with the Network Essentials license.

REP segment is a chain of ports connected to each other and configured with a segment ID. Each segment consists of standard (non-edge) segment ports and two user-configured edge ports. A switch can have no more than two ports that belong to the same segment, and each segment port can have only one external neighbor. A segment can go through a shared medium, but on any link, only two ports can belong to the same segment. REP is supported only on Trunk ports.

The figure below shows an example of a segment consisting of six ports spread across four switches. Ports E1 and E2 are configured as edge ports. When all ports are operational (as in the segment on the left), a single port is blocked, shown by the diagonal line. This blocked port is also known as the Alternate port (ALT port). When there is a failure in the network, the blocked port returns to the forwarding state to minimize network disruption.

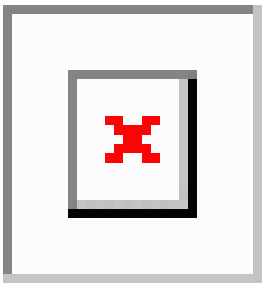
Figure 22: REP Open Segment



The segment shown in the figure above is an open segment; there is no connectivity between the two edge ports. The REP segment cannot cause a bridging loop, and you can safely connect the segment edges to any network. All hosts connected to switches inside the segment have two possible connections to the rest of the network through the edge ports, but only one connection is accessible at any time. If a failure occurs on any segment or on any port on a REP segment, REP unblocks the ALT port to ensure that connectivity is available through the other gateway.

The segment below is a closed segment, also known as Ring Segment, with both edge ports located on the same router. With this configuration, you can create a redundant connection between any two routers in the segment.

Figure 23: REP Ring Segment



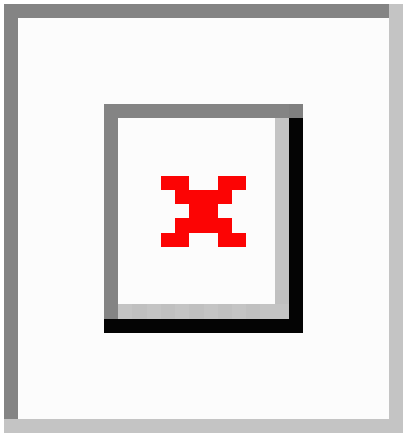
REP segments have the following characteristics:

- If all ports in a segment are operational, one port (referred to as the ALT port) is in the blocked state for each VLAN. If VLAN load balancing is configured, two ALT ports in the segment control the blocked state of VLANs.
- If a port is not operational, and cause a link failure, all ports forward traffic on all VLANs to ensure connectivity.
- In case of a link failure, alternate ports are unblocked as quickly as possible. When the failed link is restored, a logically blocked port per VLAN is selected with minimal disruption to the network.

You can construct almost any type of network based on REP segments.

In access ring topologies, the neighboring switch might not support REP as shown in the figure below. In this case, you can configure the non-REP facing ports (E1 and E2) as edge no-neighbor ports. The edge no-neighbor port can be configured to send an STP topology change notice (TCN) towards the aggregation switch.

Figure 24: Edge No-Neighbor Ports



REP has these limitations:

- You must configure each segment port; an incorrect configuration can cause forwarding loops in the networks.
- REP can manage only a single failed port within the segment; multiple port failures within the REP segment cause loss of network connectivity.
- You should configure REP only in networks with redundancy. Configuring REP in a network without redundancy causes loss of connectivity.

Link Integrity

REP does not use an end-to-end polling function between edge ports to verify link integrity. It implements local link failure detection. The REP Link Status Layer (LSL) detects its REP-aware neighbor and establishes connectivity within the segment. All the VLANs are blocked on an interface until the neighbor is detected. After the neighbor is identified, REP determines which neighbor port should become the alternate port and which ports should forward traffic.

Each port in a segment has a unique port ID. The port ID format is similar to that used by the spanning tree algorithm: a port number (unique on the bridge) associated to a MAC address (unique in the network). When a segment port is coming up, its LSL starts sending packets that include the segment ID and the port ID. The port is declared as operational after it performs a three-way handshake with a neighbor in the same segment.

A segment port does not become operational if:

- No neighbor has the same segment ID.
- More than one neighbor has the same segment ID.
- A neighbor does not acknowledge a local port as a peer.

Each port creates an adjacency with its immediate neighbor. After the neighbor adjacencies are created, the ports negotiate with each other to determine the blocked port for the segment, which will function as the alternate port. All the other ports become unblocked. By default, REP packets are sent to a bridge protocol data unit-class MAC address. The packets can also be sent to a Cisco multicast address, which is used only to send blocked port advertisement (BPA) messages when there is a failure in the segment. The packets are dropped by the devices not running REP.

Fast Convergence

REP runs on a physical link basis and not on a per-VLAN basis. Only one hello message is required for all the VLANs, and this reduces the load on the protocol. We recommend that you create VLANs consistently on all the switches in a given segment and configure the same allowed VLANs on the REP trunk ports. To avoid the delay introduced by relaying messages in software, REP also allows some packets to be flooded to a regular multicast address. These messages operate at the hardware flood layer (HFL) and are flooded to the entire network, not just the REP segment. Switches that do not belong to the segment treat them as data traffic. You can control flooding of these messages by configuring an administrative VLAN for the entire domain or for a particular segment.

VLAN Load Balancing

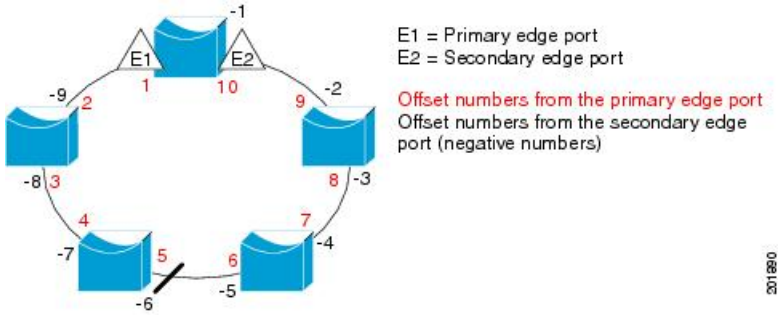
One edge port in the REP segment acts as the primary edge port; and another as the secondary edge port. It is the primary edge port that always participates in VLAN load balancing in the segment. REP VLAN balancing is achieved by blocking some VLANs at a configured alternate port and all the other VLANs at the primary edge port. When you configure VLAN load balancing, you can specify the alternate port in one of three ways:

- By entering the port ID of the interface. To identify the port ID of a port in the segment, enter the **show interface rep detail** interface configuration command for the port.
- By entering the **preferred** keyword to select the port that you previously configured as the preferred alternate port with the **rep segment** *segment-id* **preferred** interface configuration command.
- By entering the neighbor offset number of a port in the segment, which identifies the downstream neighbor port of an edge port. The neighbor offset number range is –256 to +256; a value of 0 is invalid. The primary edge port has an offset number of 1; positive numbers above 1 identify downstream neighbors of the primary edge port. Negative numbers indicate the secondary edge port (offset number -1) and its downstream neighbors.

Note Configure offset numbers on the primary edge port by identifying a port's downstream position from the primary (or secondary) edge port. Never enter an offset value of 1 because that is the offset number of the primary edge port.

The following figure shows neighbor offset numbers for a segment, where E1 is the primary edge port and E2 is the secondary edge port. The numbers inside the ring are numbers offset from the primary edge port; the numbers outside of the ring show the offset numbers from the secondary edge port. Note that you can identify all the ports (except the primary edge port) by either a positive offset number (downstream position from the primary edge port) or a negative offset number (downstream position from the secondary edge port). If E2 became the primary edge port, its offset number would then be 1 and E1 would be -1.

Figure 25: Neighbor Offset Numbers in a Segment



When the REP segment is complete, all the VLANs are blocked. When you configure VLAN load balancing, you must also configure triggers in one of two ways:

- Manually trigger VLAN load balancing at any time by entering the **rep preempt segment** *segment-id* privileged EXEC command on the switch that has the primary edge port.
- Configure a preempt delay time by entering the **rep preempt delay** *seconds* interface configuration command. After a link failure and recovery, VLAN load balancing begins after the configured preemption time period elapses. Note that the delay timer restarts if another port fails before the time has elapsed.

Note When VLAN load balancing is configured, it does not start working until triggered by either manual intervention or a link failure and recovery.

When VLAN load balancing is triggered, the primary edge port sends out a message to alert all the interfaces in the segment about the preemption. When the secondary port receives the message, the message is sent to the network to notify the alternate port to block the set of VLANs specified in the message and to notify the primary edge port to block the remaining VLANs.

You can also configure a particular port in the segment to block all the VLANs. Only the primary edge port initiates VLAN load balancing, which is not possible if the segment is not terminated by an edge port on each end. The primary edge port determines the local VLAN load-balancing configuration.

Reconfigure the primary edge port to reconfigure load balancing. When you change the load-balancing configuration, the primary edge port waits for the **rep preempt segment** command or for the configured preempt delay period after a port failure and recovery, before executing the new configuration. If you change an edge port to a regular segment port, the existing VLAN load-balancing status does not change. Configuring a new edge port might cause a new topology configuration.

Spanning Tree Interaction

REP does not interact with STP, but it can coexist. A port that belongs to a segment is removed from spanning tree control and STP BPDUs are not accepted or sent from segment ports. Therefore, STP cannot run on a segment.

To migrate from an STP ring configuration to REP segment configuration, begin by configuring a single port in the ring as part of the segment and continue by configuring contiguous ports to minimize the number of segments. Each segment always contains a blocked port, so multiple segments means multiple blocked ports

and a potential loss of connectivity. When the segment has been configured in both directions up to the location of the edge ports, you then configure the edge ports.

REP Ports

REP segments consist of Failed, Open, or Alternate ports:

- A port configured as a regular segment port starts as a failed port.
- After the neighbor adjacencies are determined, the port transitions to alternate port state, blocking all the VLANs on the interface. Blocked-port negotiations occur, and when the segment settles, one blocked port remains in the alternate role and all the other ports become open ports.
- When a failure occurs in a link, all the ports move to the Failed state. When the Alternate port receives the failure notification, it changes to the Open state, forwarding all the VLANs.

A regular segment port converted to an edge port, or an edge port converted to a regular segment port, does not always result in a topology change. If you convert an edge port into a regular segment port, VLAN load balancing is not implemented unless it has been configured. For VLAN load balancing, you must configure two edge ports in the segment.

A segment port that is reconfigured as a spanning tree port restarts according to the spanning tree configuration. By default, this is a designated blocking port. If PortFast is configured or if STP is disabled, the port goes into the forwarding state.

How to Configure Resilient Ethernet Protocol

A segment is a collection of ports connected to one another in a chain and configured with a segment ID. To configure REP segments, configure the REP administrative VLAN (or use the default VLAN 1) and then add the ports to the segment, using interface configuration mode. You should configure two edge ports in a segment, with one of them being the primary edge port and the other the secondary edge port by default. A segment should have only one primary edge port. If you configure two ports in a segment as primary edge ports, for example, ports on different switches, the REP selects one of them to serve as the segment's primary edge port. If required, you can configure the location to which segment topology change notices (STCNs) and VLAN load balancing are to be sent.

Default REP Configuration

REP is disabled on all the interfaces. When enabled, the interface is a regular segment port unless it is configured as an edge port.

When REP is enabled, the task of sending segment topology change notices (STCNs) is disabled, all the VLANs are blocked, and the administrative VLAN is VLAN 1.

When VLAN load balancing is enabled, the default is manual preemption with the delay timer disabled. If VLAN load balancing is not configured, the default after manual preemption is to block all the VLANs in the primary edge port.

REP Configuration Guidelines

Follow these guidelines when configuring REP:

- REP is supported on 1-Gigabit Ethernet and 10-Gigabit Ethernet interfaces.
- We recommend that you begin by configuring one port and then configure contiguous ports to minimize the number of segments and the number of blocked ports.
- If more than two ports in a segment fail when no external neighbors are configured, one port goes into a forwarding state for the data path to help maintain connectivity during configuration. In the **show rep interface** command output, the Port Role for this port shows as "Fail Logical Open"; the Port Role for the other failed port shows as "Fail No Ext Neighbor". When the external neighbors for the failed ports are configured, the ports go through the alternate port state transitions and eventually go to an open state or remain as the alternate port, based on the alternate port selection mechanism.
- REP ports must be Layer 2 IEEE 802.1Q or Trunk ports.
- We recommend that you configure all trunk ports in the segment with the same set of allowed VLANs.
- Be careful when configuring REP through a Telnet connection. Because REP blocks all VLANs until another REP interface sends a message to unblock it. You might lose connectivity to the router if you enable REP in a Telnet session that accesses the router through the same interface.
- You cannot run REP and STP on the same segment or interface.
- If you connect an STP network to a REP segment, be sure that the connection is at the segment edge. An STP connection that is not at the edge could cause a bridging loop because STP does not run on REP segments. All STP BPDUs are dropped at REP interfaces.
- You must configure all trunk ports in the segment with the same set of allowed VLANs, or a misconfiguration occurs.
- If REP is enabled on two ports on a switch, both ports must be either regular segment ports or edge ports. REP ports follow these rules:
  - There is no limit to the number of REP ports on a switch; however, only two ports on a switch can belong to the same REP segment.
  - If only one port on a switch is configured in a segment, the port should be an edge port.
  - If two ports on a switch belong to the same segment, they must be both edge ports, both regular segment ports, or one regular port and one edge no-neighbor port. An edge port and regular segment port on a switch cannot belong to the same segment.
  - If two ports on a switch belong to the same segment and one is configured as an edge port and one as a regular segment port (a misconfiguration), the edge port is treated as a regular segment port.
- REP interfaces come up in a blocked state and remain in a blocked state until they are safe to be unblocked. You need to be aware of this status to avoid sudden connection losses.
- REP sends all LSL PDUs in untagged frames on the native VLAN. The BPA message sent to the Cisco multicast address is sent on the administration VLAN, which is VLAN 1 by default.
- You can configure how long a REP interface remains up without receiving a hello from a neighbor. You can use the **rep lsl-age-timer** value interface configuration command to set the time from 120 ms to 10000 ms. The LSL hello timer is then set to the age-timer value divided by 3. In normal operation, three LSL hellos are sent before the age timer on the peer switch expires and checks for hello messages.

- EtherChannel port channel interfaces do not support LSL age-timer values less than 1000 ms. If you try to configure a value less than 1000 ms on a port channel, you receive an error message and the command is rejected.

- REP ports cannot be configured as one of the following port types:
  - Switched Port Analyzer (SPAN) destination port
  - Tunnel port
  - Access port
- REP is supported on EtherChannels, but not on an individual port that belongs to an EtherChannel.
- There can be a maximum of 26 REP segments per switch.

Configuring REP Administrative VLAN

To avoid the delay created by link-failure messages, and VLAN-blocking notifications during load balancing, REP floods packets to a regular multicast address at the hardware flood layer (HFL). These messages are flooded to the whole network, and not just the REP segment. You can control the flooding of these messages by configuring an administrative VLAN.

Follow these guidelines when configuring the REP administrative VLAN:

- If you do not configure an administrative VLAN, the default is VLAN 1.
- You can configure one admin VLAN on the switch for all segments.
- The administrative VLAN cannot be the RSPAN VLAN.

To configure the REP administrative VLAN, follow these steps, beginning in privileged EXEC mode:

SUMMARY STEPS

1. **configure terminal**
2. **rep admin vlan** *vlan-id*
3. **end**
4. **show interface** [*interface-id*] **rep detail**
5. **copy running-config startup config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**rep admin vlan** *vlan-id* **Example:**	Specifies the administrative VLAN. The range is from 2 to 4094.

	Command or Action	Purpose
	`Device(config)# rep admin vlan 2`	To set the admin VLAN to 1, which is the default, enter the **no rep admin vlan** global configuration command.
Step 3	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.
Step 4	**show interface** [*interface-id*] **rep detail** **Example:** `Device# show interface gigabitethernet1/1 rep detail`	(Optional) Verifies the configuration on a REP interface.
Step 5	**copy running-config startup config** **Example:** `Device# copy running-config startup config`	(Optional) Saves your entries in the switch startup configuration file.

Configuring a REP Interface

To configure REP, enable REP on each segment interface and identify the segment ID. This task is mandatory, and must be done before other REP configurations. You must also configure a primary and secondary edge port on each segment. All the other steps are optional.

Follow these steps to enable and configure REP on an interface:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *interface-id*
4. **switchport mode trunk**
5. **rep segment** *segment-id* [**edge** [**no-neighbor**] [**primary**]] [**preferred**]
6. **rep stcn** {**interface** *interface id* | **segment** *id-list* | **stp**}
7. **rep block port** {**id** *port-id* | *neighbor-offset* | **preferred**} **vlan** {*vlan-list* | **all**}
8. **rep preempt delay** *seconds*
9. **rep lsl-age-timer** *value*
10. **end**
11. **show interface** [*interface-id*] **rep** [**detail**]
12. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device# interface gigabitethernet1/1`	Specifies the interface, and enters interface configuration mode. The interface can be a physical Layer 2 interface or a port channel (logical interface).
Step 4	**switchport mode trunk** **Example:** `Device# switchport mode trunk`	Configures the interface as a Layer 2 trunk port.
Step 5	**rep segment** *segment-id* [**edge** [**no-neighbor**] [**primary**]] [**preferred**] **Example:** `Device# rep segment 1 edge no-neighbor primary`	Enables REP on the interface and identifies a segment number. The segment ID range is from 1 to 1024. **Note** You must configure two edge ports, including one primary edge port, for each segment. These optional keywords are available: • (Optional) **edge**—Configures the port as an edge port. Each segment has only two edge ports. Entering the keyword **edge** without the keyword **primary** configures the port as the secondary edge port. • (Optional) **primary**—Configures the port as the primary edge port, the port on which you can configure VLAN load balancing. • (Optional) **no-neighbor**—Configures a port with no external REP neighbors as an edge port. The port inherits all the properties of an edge port, and you can configure the properties the same way you would for an edge port. **Note** Although each segment can have only one primary edge port, if you configure edge ports on two different switches and enter the keyword **primary** on both the switches, the configuration is valid. However, REP selects only one of these ports as the segment primary edge port. You can identify the primary edge port for a segment by entering the **show rep topology** privileged EXEC command.

	Command or Action	Purpose
		• (Optional) **preferred**—Indicates that the port is the preferred alternate port or the preferred port for VLAN load balancing. **Note** Configuring a port as preferred does not guarantee that it becomes the alternate port; it merely gives the port a slight edge over equal contenders. The alternate port is usually a previously failed port.
Step 6	**rep stcn** {**interface** *interface id* | **segment** *id-list* | **stp**} **Example:** `Device# rep stcn segment 25-50`	(Optional) Configures the edge port to send segment topology change notices (STCNs). • **interface** *interface-id*—Designates a physical interface or port channel to receive STCNs. • **segment** *id-list*—Identifies one or more segments to receive STCNs. The range is from 1 to 1024. • **stp**—Sends STCNs to STP networks. **Note** Spanning Tree (MST) mode is required on edge no-neighbor nodes when **rep stcn stp** command is configured for sending STCNs to STP networks.
Step 7	**rep block port** {**id** *port-id* | *neighbor-offset* | **preferred**} **vlan** {*vlan-list* | **all**} **Example:** `Device# rep block port id 0009001818D68700 vlan 1-100`	(Optional) Configures VLAN load balancing on the primary edge port, identifies the REP alternate port in one of three ways (**id** *port-id*, *neighbor_offset*, **preferred**), and configures the VLANs to be blocked on the alternate port. • **id** *port-id*—Identifies the alternate port by port ID. The port ID is automatically generated for each port in the segment. You can view interface port IDs by entering the **show interface** *type number* **rep** [**detail**] privileged EXEC command. • *neighbor_offset*—Number to identify the alternate port as a downstream neighbor from an edge port. The range is from -256 to 256, with negative numbers indicating the downstream neighbor from the secondary edge port. A value of **0** is invalid. Enter **-1** to identify the secondary edge port as the alternate port. **Note** Because you enter the **rep block port** command at the primary edge port (offset number 1), you cannot enter an offset value of 1 to identify an alternate port.

	Command or Action	Purpose
		• **preferred**—Selects the regular segment port previously identified as the preferred alternate port for VLAN load balancing. • **vlan** *vlan-list*—Blocks one VLAN or a range of VLANs. • **vlan all**—Blocks all the VLANs. **Note** Enter this command only on the REP primary edge port.
Step 8	**rep preempt delay** *seconds* **Example:** `Device# rep preempt delay 100`	(Optional) Configures a preempt time delay. • Use this command if you want VLAN load balancing to be automatically triggered after a link failure and recovery. • The time delay range is between15 to 300 seconds. The default is manual preemption with no time delay. **Note** Enter this command only on the REP primary edge port.
Step 9	**rep lsl-age-timer** *value* **Example:** `Device# rep lsl-age-timer 2000`	(Optional) Configures a time (in milliseconds) for which the REP interface remains up without receiving a hello from a neighbor. The range is from 120 to 10000 ms in 40-ms increments. The default is 5000 ms (5 seconds). **Note** • EtherChannel port channel interfaces do not support LSL age-timer values that are less than 1000 ms. • Both the ports on the link should have the same LSL age configured in order to avoid link flaps.
Step 10	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.
Step 11	**show interface** [*interface-id*] **rep** [**detail**] **Example:** `Device(config)# show interface gigabitethernet1/1 rep detail`	(Optional) Displays the REP interface configuration.
Step 12	**copy running-config startup-config** **Example:**	(Optional) Saves your entries in the router startup configuration file.

	Command or Action	Purpose
	`Device(config)# copy running-config startup-config`	

Setting Manual Preemption for VLAN Load Balancing

If you do not enter the **rep preempt delay** *seconds* interface configuration command on the primary edge port to configure a preemption time delay, the default is to manually trigger VLAN load balancing on the segment. Be sure that all the other segment configurations have been completed before manually preempting VLAN load balancing. When you enter the **rep preempt delay segment** *segment-id* command, a confirmation message is displayed before the command is executed because preemption might cause network disruption.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **rep preempt segment** *segment-id*
4. **show rep topology segment** *segment-id*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 3	**rep preempt segment** *segment-id*
	Example:
	`Device# rep preempt segment 100`
	`The command will cause a momentary traffic disruption.`
	`Do you still want to continue? [confirm]`
		Manually triggers VLAN load balancing on the segment.
		You need to confirm the command before it is executed.

Step 4	**show rep topology segment** *segment-id* **Example:** `Device# show rep topology segment 100`	(Optional) Displays REP topology information.

	Command or Action	Purpose
Step 5	**end** **Example:** `Device# end`	Exits privileged EXEC mode.

Configuring SNMP Traps for REP

You can configure a router to send REP-specific traps to notify the Simple Network Management Protocol (SNMP) server of link-operational status changes and port role changes.

SUMMARY STEPS

1. **configure terminal**
2. **snmp mib rep trap-rate** *value*
3. **end**
4. **show running-config**
5. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**snmp mib rep trap-rate** *value* **Example:** `Device(config)# snmp mib rep trap-rate 500`	Enables the switch to send REP traps, and sets the number of traps sent per second. • Enter the number of traps sent per second. The range is from 0 to 1000. The default is 0 (no limit is imposed; a trap is sent at every occurrence).
Step 3	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 4	**show running-config** **Example:** `Device# show running-config`	(Optional) Displays the running configuration, which can be used to verify the REP trap configuration.
Step 5	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the switch startup configuration file.

Monitoring Resilient Ethernet Protocol Configurations

-

This is an example of the output for the **show interface** [*interface-id*] **rep** [**detail**] command. For this display, the REP configuration and status on an uplink port is shown.

```
Device# show interfaces TenGigabitEthernet4/1 rep detail

TenGigabitEthernet4/1 REP enabled
Segment-id: 3 (Primary Edge)
PortID: 03010015FA66FF80
Preferred flag: No
Operational Link Status: TWO_WAY
Current Key: 02040015FA66FF804050
Port Role: Open
Blocked VLAN: <empty>
Admin-vlan: 1
Preempt Delay Timer: disabled
Configured Load-balancing Block Port: none
Configured Load-balancing Block VLAN: none
STCN Propagate to: none
LSL PDU rx: 999, tx: 652
HFL PDU rx: 0, tx: 0
BPA TLV rx: 500, tx: 4
BPA (STCN, LSL) TLV rx: 0, tx: 0
BPA (STCN, HFL) TLV rx: 0, tx: 0
EPA-ELECTION TLV rx: 6, tx: 5
EPA-COMMAND TLV rx: 0, tx: 0
EPA-INFO TLV rx: 135, tx: 136
```

This is an example of the output for the **show interface** [*interface-id*] **rep** [**detail**] command. For this display, the REP configuration and status on a downlink port is shown.

```
Device#show interface TenGigabitEthernet5/0/27 rep detail
TenGigabitEthernet5/0/27   REP enabled
Segment-id: 1 (Segment)
PortID: 019B380E4D9ACAC0
Preferred flag: No
Operational Link Status: NO_NEIGHBOR
Current Key: 019B380E4D9ACAC0696B
Port Role: Fail No Ext Neighbor
Blocked VLAN: 1-4094
Admin-vlan: 1
Preempt Delay Timer: 100 sec
LSL Ageout Timer: 2000 ms
LSL Ageout Retries: 5
Configured Load-balancing Block Port: 09E9380E4D9ACAC0
Configured Load-balancing Block VLAN: 1-100
STCN Propagate to: segment 25
LSL PDU rx: 292, tx: 340
HFL PDU rx: 0, tx: 0
BPA TLV rx: 0, tx: 0
BPA (STCN, LSL) TLV rx: 0, tx: 0
BPA (STCN, HFL) TLV rx: 0, tx: 0
EPA-ELECTION TLV rx: 0, tx: 0
EPA-COMMAND TLV rx: 0, tx: 0
EPA-INFO TLV rx: 0, tx: 0
```

This is an example for the **show rep topology** [**segment** *segment-id*] [**archive**] [**detail**] command. For this display, the REP topology information for all the segments is shown.

```
Device# show rep topology

REP Segment 1
BridgeName       PortName   Edge Role
---------------- ---------- ---- ----
10.64.106.63     Te5/4      Pri  Open
10.64.106.228    Te3/4           Open
10.64.106.228    Te3/3           Open
10.64.106.67     Te4/3           Open
10.64.106.67     Te4/4           Alt
10.64.106.63     Te4/4      Sec  Open

REP Segment 3
BridgeName       PortName   Edge Role
---------------- ---------- ---- ----
10.64.106.63     Gi50/1     Pri  Open
SVT_3400_2       Gi0/3           Open
SVT_3400_2       Gi0/4           Open
10.64.106.68     Gi40/2          Open
10.64.106.68     Gi40/1          Open
10.64.106.63     Gi50/2     Sec  Alt
```

Additional References for Resilient Ethernet Protocol

Related Documents

Related Topic	Document Title
REP commands	See the Layer 2/3 Commands section of the *Command Reference (Catalyst 9400 Series Switches)*

MIBs

MIB	MIBs Link
All the supported MIBs for this release.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use the Cisco MIB Locator found at: http://www.cisco.com/go/mibs.

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

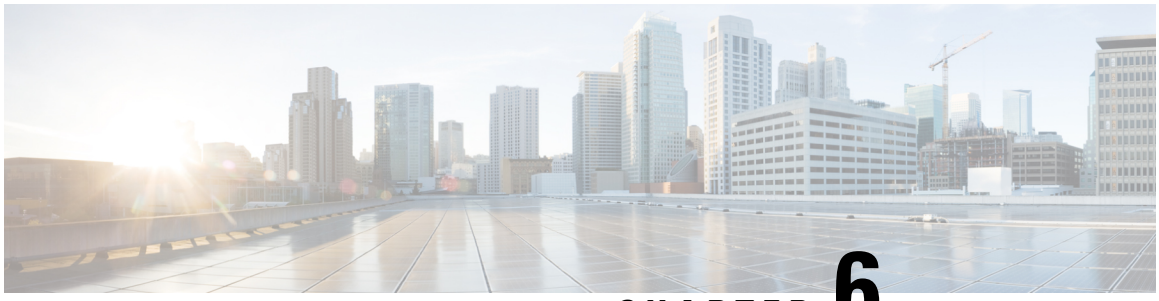
Feature History for Resilient Ethernet Protocol

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE Fuji 16.9.1	Resilient Ethernet Protocol	REP is a Cisco proprietary protocol that provides an alternative to STP to control network loops, handle link failures, and improve convergence time. You can configure the feature on uplink and downlink ports.

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.



CHAPTER 6

Configuring UniDirectional Link Detection

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/. An account on Cisco.com is not required.

Restrictions for Configuring UDLD

The following are restrictions for configuring UniDirectional Link Detection (UDLD):

- A UDLD-capable port cannot detect a unidirectional link if it is connected to a UDLD-incapable port of another device.
- When configuring the mode (normal or aggressive), make sure that the same mode is configured on both sides of the link.

Caution Loop guard works only on point-to-point links. We recommend that each end of the link has a directly connected device that is running STP.

Information About UDLD

UniDirectional Link Detection (UDLD) is a Layer 2 protocol that enables devices connected through fiber-optic or twisted-pair Ethernet cables to monitor the physical configuration of the cables and detect when a unidirectional link exists. All connected devices must support UDLD for the protocol to successfully identify and disable unidirectional links. When UDLD detects a unidirectional link, it disables the affected port and alerts you. Unidirectional links can cause a variety of problems, including spanning-tree topology loops.

Modes of Operation

UDLD supports two modes of operation: normal (the default) and aggressive. In normal mode, UDLD can detect unidirectional links due to misconnected ports on fiber-optic connections. In aggressive mode, UDLD can also detect unidirectional links due to one-way traffic on fiber-optic and twisted-pair links and to misconnected ports on fiber-optic links.

In normal and aggressive modes, UDLD works with the Layer 1 mechanisms to learn the physical status of a link. At Layer 1, autonegotiation takes care of physical signaling and fault detection. UDLD performs tasks that autonegotiation cannot perform, such as detecting the identities of neighbors and shutting down misconnected ports. When you enable both autonegotiation and UDLD, the Layer 1 and Layer 2 detections work together to prevent physical and logical unidirectional connections and the malfunctioning of other protocols.

A unidirectional link occurs whenever traffic sent by a local device is received by its neighbor but traffic from the neighbor is not received by the local device.

Normal Mode

In normal mode, UDLD detects a unidirectional link when fiber strands in a fiber-optic port are misconnected and the Layer 1 mechanisms do not detect this misconnection. If the ports are connected correctly but the traffic is one way, UDLD does not detect the unidirectional link because the Layer 1 mechanism, which is supposed to detect this condition, does not do so. In this case, the logical link is considered undetermined, and UDLD does not disable the port.

When UDLD is in normal mode, if one of the fiber strands in a pair is disconnected, as long as autonegotiation is active, the link does not stay up because the Layer 1 mechanisms detects a physical problem with the link. In this case, UDLD does not take any action and the logical link is considered undetermined.

Aggressive Mode

In aggressive mode, UDLD detects a unidirectional link by using the previous detection methods. UDLD in aggressive mode can also detect a unidirectional link on a point-to-point link on which no failure between the two devices is allowed. It can also detect a unidirectional link when one of these problems exists:

- On fiber-optic or twisted-pair links, one of the ports cannot send or receive traffic.
- On fiber-optic or twisted-pair links, one of the ports is down while the other is up.
- One of the fiber strands in the cable is disconnected.

In these cases, UDLD disables the affected port.

In a point-to-point link, UDLD hello packets can be considered as a heart beat whose presence guarantees the health of the link. Conversely, the loss of the heart beat means that the link must be shut down if it is not possible to reestablish a bidirectional link.

If both fiber strands in a cable are working normally from a Layer 1 perspective, UDLD in aggressive mode detects whether those fiber strands are connected correctly and whether traffic is flowing bidirectionally between the correct neighbors. This check cannot be performed by autonegotiation because autonegotiation operates at Layer 1.

Methods to Detect Unidirectional Links

UDLD operates by using two methods:

- Neighbor database maintenance
- Event-driven detection and echoing

Neighbor Database Maintenance

UDLD learns about other UDLD-capable neighbors by periodically sending a hello packet (also called an advertisement or probe) on every active port to keep each device informed about its neighbors.

When the device receives a hello message, it caches the information until the age time (hold time or time-to-live) expires. If the device receives a new hello message before an older cache entry ages, the device replaces the older entry with the new one.

Whenever a port is disabled and UDLD is running, whenever UDLD is disabled on a port, or whenever the device is reset, UDLD clears all existing cache entries for the ports affected by the configuration change. UDLD sends at least one message to inform the neighbors to flush the part of their caches affected by the status change. The message is intended to keep the caches synchronized.

Event-Driven Detection and Echoing

UDLD relies on echoing as its detection operation. Whenever a UDLD device learns about a new neighbor or receives a resynchronization request from an out-of-sync neighbor, it restarts the detection window on its side of the connection and sends echo messages in reply. Because this behavior is the same on all UDLD neighbors, the sender of the echoes expects to receive an echo in reply.

If the detection window ends and no valid reply message is received, the link might shut down, depending on the UDLD mode. When UDLD is in normal mode, the link might be considered undetermined and might not be shut down. When UDLD is in aggressive mode, the link is considered unidirectional, and the port is disabled.

UDLD Reset Options

If an interface becomes disabled by UDLD, you can use one of the following options to reset UDLD:

- The **udld reset** interface configuration command.
- The **shutdown** interface configuration command followed by the **no shutdown** interface configuration command restarts the disabled port.
- The **no udld** {**aggressive** | **enable**} global configuration command followed by the **udld** {**aggressive** | **enable**} global configuration command reenables the disabled ports.

- The **no udld port** interface configuration command followed by the **udld port** [**aggressive**] interface configuration command reenables the disabled fiber-optic port.
- The **errdisable recovery cause udld** global configuration command enables the timer to automatically recover from the UDLD error-disabled state, and the **errdisable recovery interval** *interval* global configuration command specifies the time to recover from the UDLD error-disabled state.

Default UDLD Configuration

Table 15: Default UDLD Configuration

Feature	Default Setting
UDLD global enable state	Globally disabled
UDLD per-port enable state for fiber-optic media	Disabled on all Ethernet fiber-optic ports
UDLD per-port enable state for twisted-pair (copper) media	Disabled on all Ethernet 10/100 and 1000BASE-TX p
UDLD aggressive mode	Disabled

How to Configure UDLD

Enabling UDLD Globally

Follow these steps to enable UDLD in the aggressive or normal mode and to set the configurable message timer on all fiber-optic ports on the device.

SUMMARY STEPS

1. **configure terminal**
2. **udld** {**aggressive** | **enable** | **message time** *message-timer-interval*}
3. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**udld** {**aggressive**	**enable**	**message time** *message-timer-interval*} **Example:**	Specifies the UDLD mode of operation: • **aggressive**—Enables UDLD in aggressive mode on all fiber-optic ports.

	Command or Action	Purpose
	`Device(config)# udld enable message time 10`	• **enable**—Enables UDLD in normal mode on all fiber-optic ports on the device. UDLD is disabled by default. An individual interface configuration overrides the setting of the **udld enable** global configuration command. • **message time** *message-timer-interval*—Configures the period of time between UDLD probe messages on ports that are in the advertisement phase and are detected to be bidirectional. The range is from 1 to 90 seconds; the default value is 15. **Note** This command affects fiber-optic ports only. Use the **udld** interface configuration command to enable UDLD on other port types. Use the **no** form of this command, to disable UDLD.
Step 3	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Enabling UDLD on an Interface

Follow these steps either to enable UDLD in the aggressive or normal mode or to disable UDLD on a port.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface-id*
3. **udld port** [**aggressive**]
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 2	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet`	Specifies the port to be enabled for UDLD, and enters interface configuration mode.
Step 3	**udld port** [**aggressive**] **Example:** `Device(config-if)# udld port aggressive`	UDLD is disabled by default. • **udld port**—Enables UDLD in normal mode on the specified port. • **udld port aggressive**—(Optional) Enables UDLD in aggressive mode on the specified port. **Note** Use the **no udld port** interface configuration command to disable UDLD on a specified fiber-optic port.
Step 4	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Monitoring and Maintaining UDLD

Command	Purpose
show udld [*interface-id* | **neighbors**]	Displays the UDLD status for the specified port or for all ports.

Additional References for UDLD

Related Documents

Related Topic	Document Title
For complete syntax and usage information for the commands used in this chapter.	

Standards and RFCs

Standard/RFC	Title
None	—

MIBs

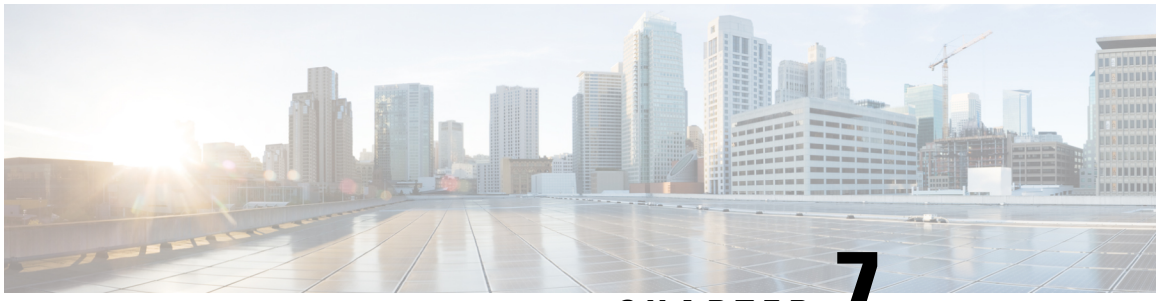
MIB	MIBs Link
All the supported MIBs for this release.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for UDLD

Release	Modification
Cisco IOS XE Everest 16.6.1	This feature was introduced.



CHAPTER 7

Configuring IEEE 802.1Q Tunneling

Information About IEEE 802.1Q Tunneling

The IEEE 802.1Q Tunneling feature is designed for service providers who carry traffic of multiple customers across their networks and are required to maintain the VLAN and Layer 2 protocol configurations of each customer without impacting the traffic of other customers.

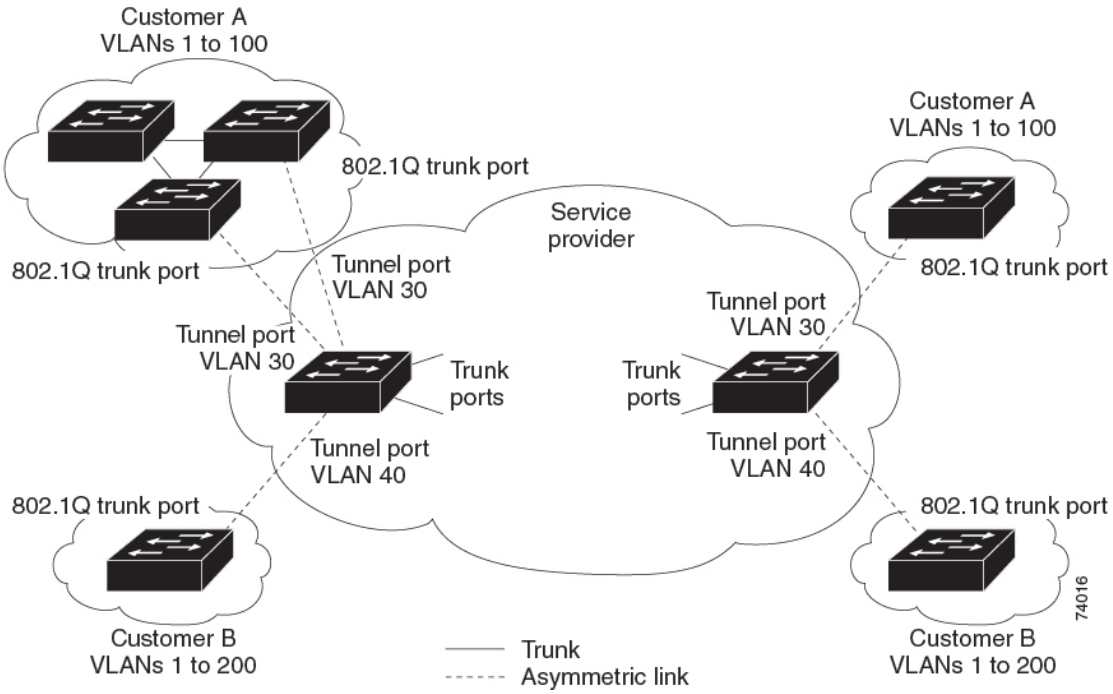
IEEE 802.1Q Tunnel Ports in a Service Provider Network

Business customers of service providers often have specific requirements for VLAN IDs and the number of VLANs to be supported. The VLAN ranges required by different customers in the same service-provider network might overlap, and traffic of customers through the infrastructure might be mixed. Assigning a unique range of VLAN IDs to each customer would restrict customer configurations and could easily exceed the VLAN limit (4096) of the IEEE 802.1Q specification.

Using the IEEE 802.1Q tunneling feature, service providers can use a single VLAN to support customers who have multiple VLANs. Customer VLAN IDs are preserved, and traffic from different customers is segregated within the service-provider network, even when they appear to be in the same VLAN. Using IEEE 802.1Q tunneling expands VLAN space by using a VLAN-in-VLAN hierarchy and retagging the tagged packets. A port configured to support IEEE 802.1Q tunneling is called a tunnel port. When you configure tunneling, you assign a tunnel port to a VLAN ID that is dedicated to tunneling. Each customer requires a separate service-provider VLAN ID, but that VLAN ID supports all of the customer's VLANs.

Customer traffic that is tagged in the normal way with appropriate VLAN IDs comes from an IEEE 802.1Q trunk port on the customer device and into a tunnel port on the service-provider edge device. The link between the customer device and the edge device is asymmetric because one end is configured as an IEEE 802.1Q trunk port, and the other end is configured as a tunnel port. You assign the tunnel port interface to an access VLAN ID that is unique to each customer.

Figure 26: IEEE 802.1Q Tunnel Ports in a Service-Provider Network

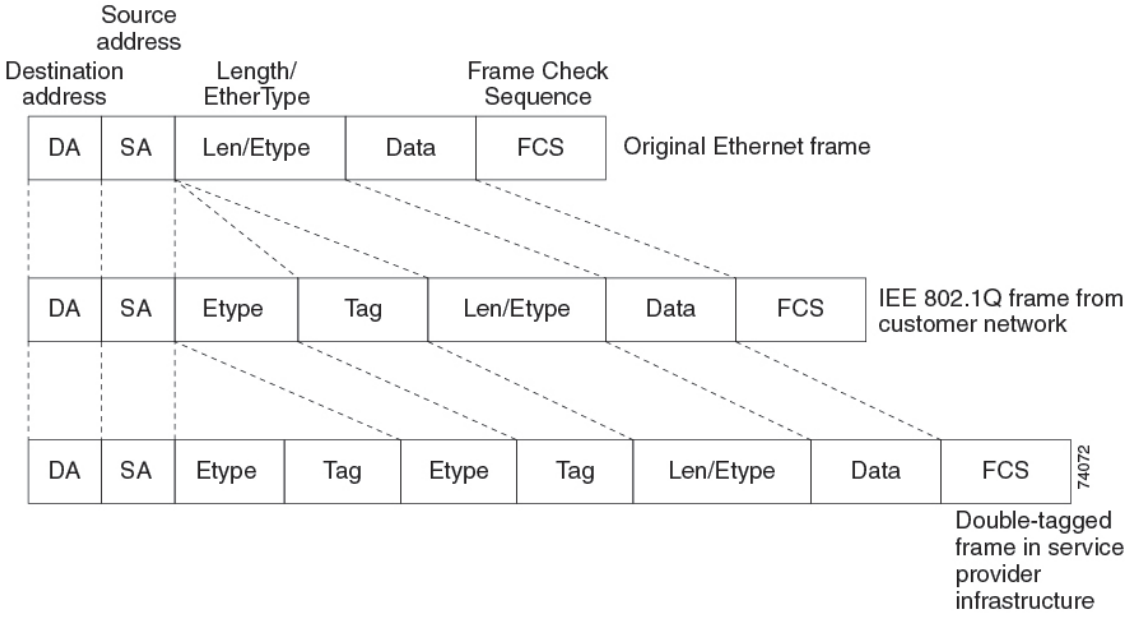


Packets coming from the customer trunk port into the tunnel port on the service-provider edge device are normally IEEE 802.1Q-tagged with the appropriate VLAN ID. The tagged packets remain intact inside the device and when they exit the trunk port into the service-provider network, they are encapsulated with another layer of an IEEE 802.1Q tag (called the metro tag) that contains the VLAN ID that is unique to the customer. The original customer IEEE 802.1Q tag is preserved in the encapsulated packet. Therefore, packets entering the service-provider network are double-tagged, with the outer (metro) tag containing the customer's access VLAN ID, and the inner VLAN ID being that of the incoming traffic.

When the double-tagged packet enters another trunk port in a service-provider core device, the outer tag is stripped as the device processes the packet. When the packet exits another trunk port on the same core device, the same metro tag is again added to the packet.

Figure 27: Original (Normal), IEEE 802.1Q, and Double-Tagged Ethernet Packet Formats

This figure shows the tag structures of the double-tagged packets.



When the packet enters the trunk port of the service-provider egress device, the outer tag is again stripped as the device internally processes the packet. However, the metro tag is not added when the packet is sent out the tunnel port on the edge device into the customer network. The packet is sent as a normal IEEE 802.1Q-tagged frame to preserve the original VLAN numbers in the customer network.

In the above network figure, Customer A was assigned VLAN 30, and Customer B was assigned VLAN 40. Packets entering the edge device tunnel ports with IEEE 802.1Q tags are double-tagged when they enter the service-provider network, with the outer tag containing VLAN ID 30 or 40, appropriately, and the inner tag containing the original VLAN number, for example, VLAN 100. Even if both Customers A and B have VLAN 100 in their networks, the traffic remains segregated within the service-provider network because the outer tag is different. Each customer controls its own VLAN numbering space, which is independent of the VLAN numbering space that is used by other customers and the VLAN numbering space that is used by the service-provider network.

At the outbound tunnel port, the original VLAN numbers on the customer's network are recovered. It is possible to have multiple levels of tunneling and tagging, but the device supports only one level in this release.

If traffic coming from a customer network is not tagged (native VLAN frames), these packets are bridged or routed as normal packets. All packets entering the service-provider network through a tunnel port on an edge device are treated as untagged packets, whether they are untagged or already tagged with IEEE 802.1Q headers. The packets are encapsulated with the metro tag VLAN ID (set to the access VLAN of the tunnel port) when they are sent through the service-provider network on an IEEE 802.1Q trunk port. The priority field on the metro tag is set to the interface class of service (CoS) priority configured on the tunnel port. (The default is zero if none is configured.)

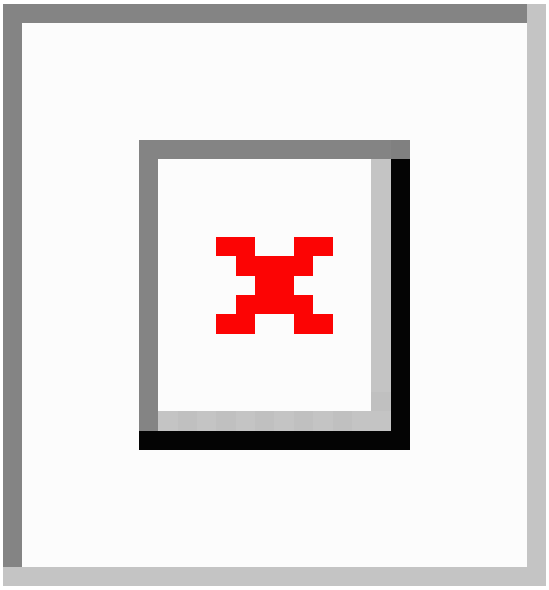
On switches, because 802.1Q tunneling is configured on a per-port basis, it does not matter whether the switch is a standalone device or a member switch. All configuration is done on the active switch.

Native VLANs

When configuring IEEE 802.1Q tunneling on an edge device, you must use IEEE 802.1Q trunk ports for sending packets into the service-provider network. However, packets going through the core of the service-provider network can be carried through IEEE 802.1Q trunks, ISL trunks, or nontrunking links. When IEEE 802.1Q trunks are used in these core devices, the native VLANs of the IEEE 802.1Q trunks must not match any native VLAN of the nontrunking (tunneling) port on the same device because traffic on the native VLAN would not be tagged on the IEEE 802.1Q sending trunk port.

In the following network figure, VLAN 40 is configured as the native VLAN for the IEEE 802.1Q trunk port from Customer X at the ingress edge device in the service-provider network (Device B). Device A of Customer X sends a tagged packet on VLAN 30 to the ingress tunnel port of Device B in the service-provider network, which belongs to access VLAN 40. Because the access VLAN of the tunnel port (VLAN 40) is the same as the native VLAN of the edge device trunk port (VLAN 40), the metro tag is not added to tagged packets received from the tunnel port. The packet carries only the VLAN 30 tag through the service-provider network to the trunk port of the egress-edgedevice (Device C) and is misdirected through the egress device tunnel port to Customer Y.

Figure 28: Potential Problems with IEEE 802.1Q Tunneling and Native VLANs



These are some ways to solve this problem:

- Use the **vlan dot1q tag native** global configuration command to configure the edge devices so that all packets going out an IEEE 802.1Q trunk, including the native VLAN, are tagged. If the devices is configured to tag native VLAN packets on all IEEE 802.1Q trunks, the devices accepts untagged packets, but sends only tagged packets.
- Ensure that the native VLAN ID on the edge devices trunk port is not within the customer VLAN range. For example, if the trunk port carries traffic of VLANs 100 to 200, assign the native VLAN a number outside that range.

System MTU

The default system MTU for traffic on the device is 1500 bytes.

You can configure 10-Gigabit and Gigabit Ethernet ports to support frames larger than 1500 bytes by using the **system mtu** *bytes* global configuration command.

The system MTU and system jumbo MTU values do not include the IEEE 802.1Q header. Because the IEEE 802.1Q tunneling feature increases the frame size by 4 bytes when the metro tag is added, you must configure all devices in the service-provider network to be able to process maximum frames by adding 4 bytes to the system MTU size.

For example, the device supports a maximum frame size of 1496 bytes with this configuration: The device has a system MTU value of 1500 bytes, and the **switchport mode dot1q tunnel** interface configuration command is configured on a 10-Gigabit or Gigabit Ethernet device port.

IEEE 802.1Q Tunneling and Other Features

Although IEEE 802.1Q tunneling works well for Layer 2 packet switching, there are incompatibilities between some Layer 2 features and Layer 3 switching.

- A tunnel port cannot be a routed port.
- IP routing is not supported on a VLAN that includes IEEE 802.1Q tunnel ports. Packets that are received from a tunnel port are forwarded based only on Layer 2 information. If routing is enabled on a switch virtual interface (SVI) that includes tunnel ports, untagged IP packets received from the tunnel port are recognized and routed by the switch. Customers can access the Internet through its native VLAN. If this access is not needed, you should not configure SVIs on VLANs that include tunnel ports.
- Fallback bridging is not supported on tunnel ports. Because all IEEE 802.1Q-tagged packets that are received from a tunnel port are treated as non-IP packets, if fallback bridging is enabled on VLANs that have tunnel ports that are configured, IP packets would be improperly bridged across VLANs. Therefore, you must not enable fallback bridging on VLANs with tunnel ports.
- Tunnel ports do not support IP access control lists (ACLs).
- Layer 3 quality of service (QoS) ACLs and other QoS features related to Layer 3 information are not supported on tunnel ports. MAC-based QoS is supported on tunnel ports.
- EtherChannel port groups are compatible with tunnel ports as long as the IEEE 802.1Q configuration is consistent within an EtherChannel port group.
- Port Aggregation Protocol (PAgP), Link Aggregation Control Protocol (LACP), and UniDirectional Link Detection (UDLD) are supported on IEEE 802.1Q tunnel ports.
- Dynamic Trunking Protocol (DTP) is not compatible with IEEE 802.1Q tunneling because you must manually configure asymmetric links with tunnel ports and trunk ports.
- VLAN Trunking Protocol (VTP) does not work between devices that are connected by an asymmetrical link or devices that communicate through a tunnel.
- Loopback detection is supported on IEEE 802.1Q tunnel ports.
- When a port is configured as an IEEE 802.1Q tunnel port, spanning-tree bridge protocol data unit (BPDU) filtering is automatically enabled on the interface. Cisco Discovery Protocol (CDP) is automatically disabled on the interface.

Note When you are configuring IEEE 802.1Q tunneling, the BPDU filtering configuration information is not displayed as spanning-tree BPDU filter is automatically enabled. You can verify the BPDU filter information using the **show spanning tree interface** command.

- When an IEEE 802.1Q tunnel port is configured as SPAN source, span filter must be applied for SVLAN to avoid packet loss.
- IGMP/MLD packet forwarding can be enabled on IEEE 802.1Q tunnels. This can be done by disabling IGMP/MLD snooping on the service provider network.

Default IEEE 802.1Q Tunneling Configuration

By default, IEEE 802.1Q tunneling is disabled because the default switchport mode is dynamic auto. Tagging of IEEE 802.1Q native VLAN packets on all IEEE 802.1Q trunk ports is also disabled.

How to Configure IEEE 802.1Q Tunneling

Follow these steps to configure a port as an IEEE 802.1Q tunnel port:

Before you begin

- Always use an asymmetrical link between the customer device and the edge device, with the customer device port configured as an IEEE 802.1Q trunk port and the edge device port configured as a tunnel port.
- Assign tunnel ports only to VLANs that are used for tunneling.
- Observe configuration requirements for native VLANs and for and maximum transmission units (MTUs).

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *interface-id*
4. **switchport access vlan** *vlan-id*
5. **switchport mode dot1q-tunnel**
6. **exit**
7. **vlan dot1q tag native**
8. **end**
9. Use one of the following:
   - **show dot1q-tunnel**
   - **show running-config interface**
10. **show vlan dot1q tag native**
11. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet2/0/1`	Enters interface configuration mode for the interface to be configured as a tunnel port. This should be the edge port in the service-provider network that connects to the customer device. Valid interfaces include physical interfaces and port-channel logical interfaces (port channels 1 to 48).
Step 4	**switchport access vlan** *vlan-id* **Example:** `Device(config-if)# switchport access vlan 2`	Specifies the default VLAN, which is used if the interface stops trunking. This VLAN ID is specific to the particular customer.
Step 5	**switchport mode dot1q-tunnel** **Example:** `Device(config-if)# switchport mode dot1q-tunnel`	Sets the interface as an IEEE 802.1Q tunnel port. **Note** Use the **no switchport mode dot1q-tunnel** interface configuration command to return the port to the default state of dynamic desirable.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Returns to global configuration mode.
Step 7	**vlan dot1q tag native** **Example:** `Device(config)# vlan dot1q tag native`	(Optional) Sets the device to enable tagging of native VLAN packets on all IEEE 802.1Q trunk ports. When not set, and a customer VLAN ID is the same as the native VLAN, the trunk port does not apply a metro tag, and packets could be sent to the wrong destination. **Note** Use the**no vlan dot1q tag native** global configuration command to disable tagging of native VLAN packets.

	Command or Action	Purpose
Step 8	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Step 9	Use one of the following:
	• **show dot1q-tunnel**
	• **show running-config interface**
	Example:
	`Device# show dot1q-tunnel`
	or
	`Device# show running-config interface`
		Displays the ports configured for IEEE 802.1Q tunneling.
		Displays the ports that are in tunnel mode.

Step 10	**show vlan dot1q tag native** **Example:** `Device# show vlan dot1q native`	Displays IEEE 802.1Q native VLAN tagging status.
Step 11	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Monitoring Tunneling Status

The following table describes the commands used to monitor tunneling status.

Table 16: Commands for Monitoring Tunneling

Command	Purpose
show dot1q-tunnel	Displays IEEE 802.1Q tunnel ports on the device.
show dot1q-tunnel interface *interface-id*	Verifies if a specific interface is a tunnel port.
show vlan dot1q tag native	Displays the status of native VLAN tagging on the device.

Example: Configuring an IEEE 802.1Q Tunneling Port

The following example shows how to configure an interface as a tunnel port, enable tagging of native VLAN packets, and verify the configuration. In this configuration, the VLAN ID for the customer connected to Gigabit Ethernet interface 7 on stack member 1 is VLAN 22.

```
Switch(config)# interface gigabitethernet1/0/7
Switch(config-if)# switchport access vlan 22
% Access VLAN does not exist. Creating vlan 22
Switch(config-if)# switchport mode dot1q-tunnel
Switch(config-if)# exit
Switch(config)# vlan dot1q tag native
Switch(config)# end
Switch# show dot1q-tunnel interface gigabitethernet1/0/7
Port
-----
Gi1/0/1Port
-----
Switch# show vlan dot1q tag native
dot1q native vlan tagging is enabled
```

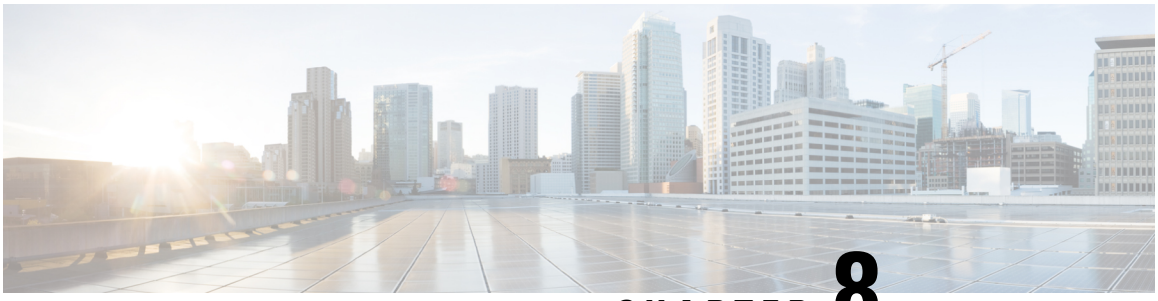
Feature History for IEEE 802.1Q Tunneling

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE Everest 16.6.1	IEEE 802.1Q Tunneling	The IEEE 802.1Q tunneling feature is designed for service providers who carry traffic of multiple customers across their networks and are required to maintain the VLAN and Layer 2 protocol configurations of each customer without impacting the traffic of other customers.

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.



CHAPTER 8

Configuring VXLAN BGP EVPN

Information About VXLAN BGP EVPN

VXLAN is a MAC in IP/UDP overlay that allows layer 2 segments to be stretched across an IP core. All the benefits of layer 3 topologies are thereby available with VXLAN. The encapsulation and decapsulation of VXLAN headers is handled by a functionality embedded in VXLAN Tunnel End Points (VTEPs). VTEPs themselves could be implemented in software or a hardware form-factor.

VXLAN natively operates on a flood-n-learn mechanism where BU (Broadcast, Unknown Unicast) traffic and Layer 2 Multicast traffic in a given VXLAN network is sent over the IP core to every VTEP that has membership in that network. IP multicast is used to send traffic over the network.The receiving VTEPs decapsulate the packet, and based on the inner frame perform layer-2 MAC learning. The inner SMAC is learnt against the outer Source IP Address (SIP) corresponding to the source VTEP. In this way, reverse traffic can be unicasted toward the previously learnt end host.

Motivations for using an overlay architecture include:

- Scalability — VXLAN provides Layer-2 connectivity that allows the infrastructure that can scale to 16 million tenant networks. It overcomes the 4094-segment limitation of VLANs. This is necessary to address today's multi-tenant cloud requirements.
- Flexibility — VXLAN allows workloads to be placed anywhere, along with the traffic separation required in a multi-tenant environment. The traffic separation is done using network segmentation (segment IDs or virtual network identifiers [VNIs]).Workloads for a tenant can be distributed across different physical devices (since workloads are added as the need arises, into available server space) but the workloads are identified by the same layer 2 or layer 3 VNI as the case may be.
- Mobility — VMs can be moved from one data center location to another without updating spine switch tables. This is because entities within the same tenant network in a VXLAN/EVPN fabric setup retain the same segment ID, regardless of their location.

One of the biggest limitations of VXLAN flood-n-learn is the inherent flooding that is required ensuring that learning happens at the VTEPs. In a traditional deployment, a layer-2 segment is represented with a VLAN that comprises a broadcast domain, which also scopes BU traffic. With VXLAN, now the layer-2 segment spans a much larger boundary across an IP core where floods are translated to IP multicast (or HER). Consequently, the flood-n-learn based scheme presents serious scale challenges especially as the number of end hosts go up. This is addressed via learning using a control-plane for distribution of end host addresses. The control plane of choice is BGP EVPN.

Guidelines and Limitations for VXLAN BGP EVPN

The following are the limitations for Virtual Extensible LAN (VXLAN) Border Gateway Protocol (BGP) Ethernet VPN (EVPN) has the following:

- Multicast over VXLAN is currently not supported.
- show commands with the keyword **internal** are not supported.
- For EBGP, it is recommended to use a single overlay EBGP EVPN session between loopbacks.
- Bind NVE to a loopback address that is separate from other loopback addresses that are required by Layer 3 protocols. A best practice is to use a dedicated loopback address for VXLAN.
- VXLAN BGP EVPN does not support an NVE interface in a non-default VRF.
- It is recommended to configure a single BGP session over the loopback for an overlay BGP session.
- The VXLAN UDP port number is used for VXLAN encapsulation. It complies with IETF standards and is not configurable.
- VXLAN BGP EVPN currently supports only leaf switch functionality. Spine switch functionality is not supported.
- Support is not available for any integrated underlay technologies such as route-reflector, or anycast rendezvous point , or Multicast Source Discovery Protocol (MSDP) endezvous point.
- Border leaf functionality and interworking between BGP EVPN and traditional Layer 3 and Layer 2 overlay networks are not supported.
- Auto route-distinguisher and auto route-target for IP VRF is not supported
- Centralized Gateway for Layer 2 VXLAN network identifier (L2VNI) is not supported.
- BGP EVPN Network Virtualization Overlay MIB is not supported.
- In EVPN deployments, once a VLAN is used for a core-facing SVI, it should not be allowed in any trunk. For a core-facing SVI to function properly, the **no autostate** command must be configured under the SVI.

Considerations for VXLAN BGP EVPN deployment

The following considerations need to be taken into account for VXLAN BGP EVPN deployment:

- A loopback address is required when using the source-interface config command. The loopback address represents the local VTEP IP.

- To establish IP multicast routing in the core, IP multicast configuration, PIM configuration, and RP configuration are required.
- VTEP to VTEP unicast reachability can be configured through any IGP/BGP protocol.
- If the anycast gateway feature is enabled for a specific VNI, then the anycast gateway feature must be enabled on all VTEPs that have that VNI configured. Having the anycast gateway feature configured on only some of the VTEPs enabled for a specific VNI is not supported.
- It is a requirement when changing the primary or secondary IP address of the NVE source interfaces to shut the NVE interface before changing the IP address.
- As a best practice, the RP for the multicast group should be configured only on the spine layer. Usethe anycast RP for RP load balancing and redundancy.
- Every tenant VRF needs a VRF overlay, VLAN and SVI for VXLAN routing.
- The following considerations need to be taken into account with eBGP use case:
  - Manual configuration of the Route Targets (RT) is required. RT must be matching between the VTEPs for a given EVPN instance (EVI).
  - The **retain route-target all** BGP knob must be enabled on the Spine nodes under BGP routing process
  - The **set ip next-hop unchanged** BGP knob must be enabled on Spine nodes to set next hop for EVPN routes to the proper VTEP node.
  - Peering between VTEPs can be achieved to multiple Spine nodes to achieve redundancy.
- Ensure the following to create a proper VLAN database:
  - The route targets with eBGP EVPN VxLAN design model cannot be auto generated like in iBGP/IGP model, hence they need to be manually configured for each EVPN instance (EVI) and should be matching for a given EVI. Failure to manually configure route target will result in loss of connectivity and improper operation due to routes not being installed.
  - To ensure proper operation of EVPN VXLAN, assign the vlan first as an access interface to create the vlan and store it in the vlan.dat file. For a trunk interface, trying to create a SVI before creating the vlan in VLAN.dat will put the SVI in a down state.
- In case of a scoped configuration, not all L2 VNIs need to be enabled on all VTEP switches. They will only be enabled as needed on a given VTEP.
- Route Distinguishers (RD) need to be unique per IP VRF (L3 VNI). Route Targets (RT) must match for a given IP VRF (L3 VNI) . There is no auto-generation neither for RD or RT for the case of IP VRF (L3 VNI).
- All VTEP switches need not be configured with same L2 VNIs unless in the scoped configuration. Access VLANs are the VLANs connected to hosts. Access SVIs must have an IP address with the same subnet as the hosts the VLAN is connected to. For AnyCast Gateway support, Access SVIs of the same VLAN should have the same IP and MAC addresses in all VTEPs.
- It is important to configure additional L3 VNIs on all VTEP nodes where Inter-VxLAN communication is needed.

Network considerations for VXLAN deployments

The following network consideration need to be taken into account for VXLAN deployments:

MTU Size in the Transport Network

Due to the MAC-to-UDP encapsulation, VXLAN introduces 50-byte overhead to the original frames. Therefore, the maximum transmission unit (MTU) in the transport network needs to be increased by50 bytes. If the overlays use a 1500-byteMTU, the transport network needs to be configured to accommodate 1550-byte packets at a minimum. Jumbo-frame support in the transport network is required if the overlay applications tend to use larger frame sizes than 1500 bytes.

ECMP and LACP Hashing Algorithms in the Transport Network

Switches introduce a level of entropy in the source UDP port for ECMP and LACP hashing in the transport network. As a way to augment this implementation, the transport network uses an ECMP or LACP hashing algorithm that takes the UDP source port as an input for hashing, which achieves the best load-sharing results for VXLANencapsulated traffic.

Multicast Group Scaling

The VXLAN implementation uses multicast tunnels for broadcast, unknown unicast, and multicast traffic forwarding. Ideally, one VXLAN segment mapping to one IP multicast group is the way to provide the optimal multicast forwarding. It is possible, however, to have multiple VXLAN segments share a single IP multicast group in the core network. VXLAN can support up to 16 million logical Layer 2 segments, using the 24-bit VNID field in the header. With one-to-one mapping between VXLAN segments and IP multicast groups, an increase in the number of VXLAN segments causes a parallel increase in the required multicast address space and the amount of forwarding states on the core network devices. At some point, multicast scalability in the transport network can become a concern. In this case, mapping multiple VXLAN segments to a single multicast group can help conserve multicast control plane resources on the core devices and achieve the desired VXLAN scalability. However, this mapping comes at the cost of suboptimal multicast forwarding. Packets forwarded to the multicast group for one tenant are now sent to the VTEPs of other tenants that are sharing the same multicast group. This causes inefficient utilization of multicast data plane resources. Therefore, this solution is a trade-off between control plane scalability and data plane efficiency.

Despite the suboptimal multicast replication and forwarding, having multiple-tenant VXLAN networks to share a multicast group does not bring any implications to the Layer 2 isolation between the tenant networks. After receiving an encapsulated packet fromthe multicast group, a VTEP checks and validates the VNID in the VXLAN header of the packet. The VTEP discards the packet if the VNID is unknown to it. Only when the VNID matches one of the VTEP's local VXLAN VNIDs, does it forward the packet to that VXLAN segment.Other tenant networks will not receive the packet. Thus, the segregation between VXLAN segments is not compromised.

Considerations for the Transport Network

The following considerations need to be taken into account for the configuration of the transport network:

- On the VTEP device:
  - Enable and configure IP multicast.
  - Create and configure a loopback interface with a /32 IP address.
  - Enable IP multicast on the loopback interface.

- Advertise the loopback interface /32 addresses through the routing protocol (static route) that runs in the transport network.
- Enable IP multicast on the uplink outgoing physical interface.

- Throughout the transport network:
  - Enable and configure IP multicast.

Configuring VXLAN BGP EVPN

Configuring Underlay Transport (Unicast and Multicast) between the VTEPs and the Spines

Follow these steps to configure underlay transport on the Spine:

Note This configuration is applicable to Cisco Nexus Series Switches and is not applicable to Cisco Catalyst 9000 Family Switches.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip pim rp-address** *rp-address* **group-list** *prefix*
4. **ip pim rp-candidate loopback** *if_number* **group-list** *prefix*
5. **ip pim ssm range** *groups*
6. **ip pim anycast-rp** *rp-address anycast-rp-peer-address*
7. **interface loopback** *number*
8. **ip address** *ip address*
9. **ip pim sparse-mode**
10. **exit**
11. **interface port-channel** *channel-number*
12. **mtu** *bytes*
13. **medium p2p**
14. **ip address** *ip-address mask*
15. **ip pim sparse-mode**
16. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip pim rp-address** *rp-address* **group-list** *prefix* **Example:** `Device(config)# ip pim rp-address 100.1.1.1 group-list 239.0.0.0/8`	Configures a PIM static route processor (RP) address for a multicast group range and specifies a group range for a static RP.
Step 4	**ip pim rp-candidate loopback** *if_number* **group-list** *prefix* **Example:** `Device(config)# ip pim rp-candidate loopback1 group-list 239.0.0.0/8`	Configures a PIM address as a RP candidate. Specifies the loopkback interface. Specifies a group range handled by the RP.
Step 5	**ip pim ssm range** *groups* **Example:** `Device(config)# ip pim ssm range 232.0.0.0/8`	Configures a group range for SSM.
Step 6	**ip pim anycast-rp** *rp-address anycast-rp-peer-address* **Example:** `Device(config)# ip pim anycast-rp 100.1.1.1 10.1.1.1`	Configures PIM Anycast-RP peer for the specified Anycast-RP address.
Step 7	**interface loopback** *number* **Example:** `Device(config)# interface loopback0`	Creates a loopback interface and enters interface configuration mode.
Step 8	**ip address** *ip address* **Example:** `Device(config-if)# ip address 10.1.1.1/32`	Defines the IP address for an interface.
Step 9	**ip pim sparse-mode** **Example:** `Device(config-if)# ip pim sparse-mode`	Enables Protocol Independent Multicast (PIM) sparse mode on an interface.
Step 10	**exit** **Example:** `Device(config-if)# exit`	Exits the interface configuration mode
Step 11	**interface port-channel** *channel-number* **Example:**	Specifies the port-channel interface to configure, and enters the interface configuration mode.

	Command or Action	Purpose
	`Device(config)# interface port-channel1`	
Step 12	**mtu** *bytes* **Example:** `Device(config-if)# mtu 9198`	Sets the interface MTU size.
Step 13	**medium p2p** **Example:** `Device(config-if)# medium p2p`	Configures the interface medium as point to point.
Step 14	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 10.10.1.1/30`	Defines the IP address for an interface.
Step 15	**ip pim sparse-mode** **Example:** `Device(config-if)# ip pim sparse-mode`	Enables Protocol Independent Multicast (PIM) sparse mode on an interface.
Step 16	**exit** **Example:** `Device(config-if)# exit`	Exits the interface configuration mode.

Configuring the VTEP

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip multicast-routing**
4. **ip pim rp-address** *rp-address*
5. **ip routing**
6. **interface loopback** *number*
7. **ip address** *ip address*
8. **ip pim sparse-mode**
9. **exit**
10. **interface loopback** *number*
11. **ip vrf forwarding** *vrf name*
12. **ip address** *ip address*
13. **exit**
14. **interface tengigabitethernet** *slot/port*
15. **no switchport**
16. **no ip address**
17. **channel-group** *number*
18. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip multicast-routing** **Example:** `Device(config)# ip multicast-routing`	Enables IP multicast routing.
Step 4	**ip pim rp-address** *rp-address* **Example:** `Device(config)# ip pim rp-address 100.1.1.1`	Configures a PIM static route processor (RP) address for a multicast group range. The rp address used in this step should be the same one used on the spine.
Step 5	**ip routing** **Example:** `Device(config)# ip routing`	Enables routing on the switch. Even if IP routing was previously enabled, this step ensures that it is activated.
Step 6	**interface loopback** *number* **Example:** `Device(config)# interface Loopback0`	Creates a loopback interface and enters interface configuration mode. This loopback interface is assinged to the NVE interface.
Step 7	**ip address** *ip address* **Example:** `Device(config-if)# ip address 10.11.11.11 255.255.255.255`	Defines the IP address for an interface.
Step 8	**ip pim sparse-mode** **Example:** `Device(config-if)# ip pim sparse-mode`	Enables Protocol Independent Multicast (PIM) sparse mode on an interface.
Step 9	**exit** **Example:** `Device(config-if)# exit`	Exits the interface configuration mode
Step 10	**interface loopback** *number* **Example:** `Device(config)# interface Loopback2`	Creates a loopback interface and enters interface configuration mode. This loopback interface is assinged to the L3 VNI.
Step 11	**ip vrf forwarding** *vrf name* **Example:**	Associates the VRF with the Layer 3 interface.

	Command or Action	Purpose
	`Device(config-if)# vrf forwarding tenant_1`	
Step 12	**ip address** *ip address* **Example:** `Device(config-if)# ip address 11.11.11.11 255.255.255.255`	Defines the IP address for an interface.
Step 13	**exit** **Example:** `Device(config-if)# exit`	Exits the interface configuration mode
Step 14	**interface tengigabitethernet** *slot/port* **Example:** `Device(config)# interface TenGigabitEthernet1/1/2`	Selects the port to configure.
Step 15	**no switchport** **Example:** `Device(config-if)# no switchport`	Makes the interface Layer 3 capable.
Step 16	**no ip address** **Example:** `Device(config-if)# no ip address`	Disables IP processing on a particular interface.
Step 17	**channel-group** *number* **Example:** `Device(config-if)# channel-group 1 mode active`	Assigns and configure a physical interface to an EtherChannel.
Step 18	**exit** **Example:** `Device(config-if)# exit`	Exits the interface configuration mode

Configuring eBGP on the Spine:

Follow these steps to configure eBGP with EVPN address family on the Spine:

Note This configuration is applicable to Cisco Nexus Series Switches and is not applicable to Cisco Catalyst 9000 Family Switches.

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **ip prefix-list** *name* [**seq** *number*] {**permit** | **deny**} *prefix* [**eq** *length*] | [**ge** *length*] | [**le** *length*]]
4. **route-map name**{**permit**|**deny**}[*sequence-number*]
5. **set ip next-hop unchanged**
6. **exit**
7. **route-map name** {**permit** | **deny**} [**sequence number**]
8. **match ip address prefix-list** *name* [*name*]
9. **exit**
10. **router bgp** *number*
11. **router id** {*router id*}
12. **bgp log-neighbor-changes**
13. **address-family ipv4 unicast**
14. **redistribute direct** [**route-map** *map-name*]
15. **exit**
16. **address-family l2vpn evpn**
17. **nexthop route-map** *name*
18. **retain route-target all**
19. **exit**
20. **neighbor** *vtep1 loopback address* **remote-as** *number*
21. **neighbor** *ip-address* **update-source** *interface-type interface-number*
22. **neighbor** {*ip address* | *peer-group-name*} **ebgp-multihop** [**ttl**]
23. **address-family ipv4 unicast**
24. **neighbor** {*ip address* | *peer-group-name*} **send-community both**
25. **soft-reconfiguration inbound**
26. **exit**
27. **address-family l2vpn evpn**
28. **neighbor** {*ip address* | *peer-group-name*} **send-community both**
29. **neighbor** {*ip address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}
30. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip prefix-list** *name* [**seq** *number*] {**permit**	**deny**} *prefix* [**eq** *length*]	[**ge** *length*]	[**le** *length*]] **Example:**	Creates a prefix list to match IP packets or routes against.

	Command or Action	Purpose
	`Device(config)# ip prefix-list lo_prefix seq 5 permit 0.0.0.0/0 le 32`	
Step 4	**route-map name** {**permit** | **deny**} [*sequence-number*] **Example:** `Device(config)# route-map NH-UNCHANGED permit 10`	Creates the route map entry. Enters route-map configuration mode.
Step 5	**set ip next-hop unchanged** **Example:** `Device(config-route-map)# set ip next-hop unchanged`	Defines the route-map and applies outbound policy for neighbour.
Step 6	**exit** **Example:** `Device(config-route-map)# exit`	Exits the route-map configuration mode
Step 7	**route-map name** {**permit** | **deny**} [**sequence number**] **Example:** `Device(config)# route-map any_prefix permit 10`	Creates the route map entry. Enters route-map configuration mode.
Step 8	**match ip address prefix-list** *name* [*name*] **Example:** `Device(config-route-map)# match ip address prefix-list lo_prefix`	Matches against one or more ip address prefix lists.
Step 9	**exit** **Example:** `Device(config-route-map)# exit`	Exits the route-map configuration mode
Step 10	**router bgp** *number* **Example:** `Device(config)# router bgp 1`	Configures BGP.
Step 11	**router id** {*router id*} **Example:** `Device(config-router)# router-id 10.1.1.1`	Specifies a fixed router ID in the router configuration mode.
Step 12	**bgp log-neighbor-changes** **Example:** `Device(config-router)# log-neighbor-changes`	Enables the generation of logging messages generated when the status of a BGP neighbor changes.
Step 13	**address-family ipv4 unicast** **Example:** `Device(config-router)# address-family ipv4 unicast`	Enters address family configuration mode and Specifies IP Version 4 unicast address prefixes.

	Command or Action	Purpose
Step 14	**redistribute direct** [**route-map** *map-name*] **Example:** `Device(config-router-af)# redistribute direct route-map any_prefix`	Distributes routes that are directly connected on an interface.
Step 15	**exit** **Example:** `Device(config-router-af)# exit`	Exits the address family configuration mode
Step 16	**address-family l2vpn evpn** **Example:** `Device(config-router)# address-family l2vpn evpn`	Specifies the L2VPN address family and enters address family configuration mode. The evpn keyword specifies that EVPN endpoint provisioning information is to be distributed to BGP peers.
Step 17	**nexthop route-map** *name* **Example:** `Device(config-router-af)# nexthop route-map NH-UNCHANGED`	Specifies that Border Gateway Protocol (BGP) routes are resolved using only the next hops that have routes that match specific characteristics.
Step 18	**retain route-target all** **Example:** `Device(config-router-af)# retain route-target all`	Accepts received updates with specified route targets.
Step 19	**exit** **Example:** `Device(config-router-af)# exit`	Exits the address family configuration mode
Step 20	**neighbor** *vtep1 loopback address* **remote-as** *number* **Example:** `Device(config-router)# neighbor 10.11.11.11 remote-as 2`	Adds an entry to the BGP or multiprotocol BGP neighbor table in the router configuration mode.
Step 21	**neighbor** *ip-address* **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 10.11.11.11 update-source loopback0`	Allows BGP sessions to use any operational interface for TCP connections.
Step 22	**neighbor** {*ip address* | *peer-group-name*} **ebgp-multihop** [**ttl**] **Example:** `Device(config-router)# neighbor 10.11.11.11 ebgp-multihop 10`	Allows BGP connections to external peers on networks that are not directly connected.
Step 23	**address-family ipv4 unicast** **Example:**	Enters address family configuration mode and Specifies IP Version 4 unicast address prefixes.

	Command or Action	Purpose
	`Device(config-router)# address-family ipv4 unicast`	
Step 24	**neighbor** {*ip address* | *peer-group-name*} **send-community both** **Example:** `Device(config-router-af)# neighbor 10.11.11.11 send-community both`	Specifies both standard and extended communities attribute should be sent to a BGP neighbour.
Step 25	**soft-reconfiguration inbound** **Example:** `Device(config-router-af)# soft-reconfiguration inbound`	Configures the switch software to start storing BGP peer updates.
Step 26	**exit** **Example:** `Device(config-router-af)# exit`	Exits the address family configuration mode
Step 27	**address-family l2vpn evpn** **Example:** `Device(config-router)# address-family l2vpn evpn`	Specifies the L2VPN address family and enters address family configuration mode. The evpn keyword specifies that EVPN endpoint provisioning information is to be distributed to BGP peers.
Step 28	**neighbor** {*ip address* | *peer-group-name*} **send-community both** **Example:** `Device(config-router-af)# neighbor 10.11.11.11 send-community both`	Specifies both standard and extended communities attribute should be sent to a BGP neighbour.
Step 29	**neighbor** {*ip address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**} **Example:** `Device(config-router-af)# neighbor 10.11.11.11 route-map NH-UNCHANGED out`	Applies the inbound route map to routes received from the specified neighbor, or applies an outbound route map to routes advertised to the specified neighbor.
Step 30	**exit** **Example:** `Device(config-router-af)# exit`	Exits the address family configuration mode

Configuring eBGP on the VTEP

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *number*
4. **bgp router-id interface** *loopback address*

5. **bgp log-neighbor-changes**
6. **bgp graceful-restart**
7. **neighbor** *spine 1 loopback address* **remote-as***number*
8. **neighbor** {*ip address* | *peer-group-name*} **ebgp-multihop** [**ttl**]
9. **neighbor** {*ip address* | *group-name*} **update-source** *interface*
10. **address-family ipv4**
11. **redistribute connected**
12. **neighbor** *ip-address* **activate**
13. **exit**
14. **address-family l2vpn evpn**
15. **neighbor** *ip-address* **activate**
16. **neighbor** *ip-address* **send-community both**
17. **maximum-paths** *number-of-paths*
18. **exit**
19. **address-family ipv4 vrf** *vrf-name*
20. **advertise l2vpn evpn**
21. **redistribute connected**
22. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *number* **Example:** `Device(config)# router bgp 2`	Configures BGP.
Step 4	**bgp router-id interface** *loopback address* **Example:** `Device(config-router)# bgp router-id interface Loopback0`	Specifies loopback address as router address.
Step 5	**bgp log-neighbor-changes** **Example:** `Device(config-router)# bgp log-neighbor-changes`	Enables the generation of logging messages generated when the status of a BGP neighbor changes.
Step 6	**bgp graceful-restart** **Example:**	Enables the BGP graceful restart capability for a BGP neighbor.

	Command or Action	Purpose
	`Device(config-router)# bgp graceful-restart`	
Step 7	**neighbor** *spine 1 loopback address* **remote-as***number* **Example:** `Device(config-router)# neighbor 10.1.1.1 remote-as 1`	Defines MP-BGP neighbors. Under each neighbor define l2vpn evpn.
Step 8	**neighbor** {*ip address* | *peer-group-name*} **ebgp-multihop** [**ttl**] **Example:** `Device(config-router)# neighbor 10.1.1.1 ebgp-multihop 10`	Allows BGP connections to external peers on networks that are not directly connected.
Step 9	**neighbor** {*ip address* | *group-name*} **update-source** *interface* **Example:** `Device(config-router)# neighbor 10.1.1.1 update-source Loopback0`	Configures update source. Update source can be configured per neighbor or per peer-group
Step 10	**address-family ipv4** **Example:** `Device(config-router)# address-family ipv4`	Enters address family configuration mode.
Step 11	**redistribute connected** **Example:** `Device(config-router-af)# redistribute connected`	Redistributes connected routes from another routing protocol.
Step 12	**neighbor** *ip-address* **activate** **Example:** `Device(config-router-af)# neighbor 10.1.1.1 activate`	Enables the exchange information from a bgp neighbor
Step 13	**exit** **Example:** `Device(config-router-af)# exit-address-family`	Exits the address family configuration mode
Step 14	**address-family l2vpn evpn** **Example:** `Device(config-router)# address-family l2vpn evpn`	Specifies the L2VPN address family and enters address family configuration mode.
Step 15	**neighbor** *ip-address* **activate** **Example:** `Device(config-router-af)# neighbor 10.1.1.1 activate`	Enables the exchange information from a bgp neighbor

	Command or Action	Purpose
Step 16	**neighbor** *ip-address* **send-community both** **Example:** `Device(config-router-af)# neighbor 10.1.1.1 send-community both`	Specifies the communities attribute sent to a bgp neighbor
Step 17	**maximum-paths** *number-of-paths* **Example:** `Device(config-router-af)# maximum-paths 2`	Controls the maximum number of parallel routes an IP routing protocol can support.
Step 18	**exit** **Example:** `Device(config-router-af)# exit-address-family`	Exits the address family configuration mode
Step 19	**address-family ipv4 vrf** *vrf-name* **Example:** `Device(config-router)# address-family ipv4 vrf tenant_1`	Specifies the name of the VRF instance to associate with subsequent address family configuration mode commands.
Step 20	**advertise l2vpn evpn** **Example:** `Device(config-router-af)# advertise l2vpn evpn`	Advertises (L2VPN) EVPN routes within a tenant VRF in a VXLAN EVPN fabric.
Step 21	**redistribute connected** **Example:** `Device(config-router-af)# redistribute connected`	Redistributes connected routes from another routing protocol.
Step 22	**exit** **Example:** `Device(config-router-af)# exit-address-family`	Exits the address family configuration mode

Configuring the NVE Interface and VNIs

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface***nve-interface*
4. **no ip address**
5. **source-interface loopback***number*
6. **host-reachability protocol bgp**
7. **member vni***vni***associate-vrf**
8. **member vni***vni***mcast-group***address*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface***nve-interface* **Example:** `Device(config)# interface nve1`	Configures the NVE interface.
Step 4	**no ip address** **Example:** `Device(config-if)# no ip address`	Disables IP processing on the interface.
Step 5	**source-interface loopback***number* **Example:** `Device(config-if)# source-interface Loopback1`	Creates a loopback interface. **Note** This interface will be a different loopback from the loopback interface used for underlay.
Step 6	**host-reachability protocol bgp** **Example:** `Device(config-if)# host-reachability protocol bgp`	Defines BGP as the mechanism for host reachability advertisement.
Step 7	**member vni***vni***associate-vrf** **Example:** `Device(config-if)# member vni 11001 mcast-group 239.0.1.1`	Adds Layer-3 VNIs, one per tenant VRF, to the overlay. **Note** Required for VXLAN routing only.
Step 8	**member vni***vni***mcast-group***address* **Example:** `Device(config-if)# member vni 900001 vrf tenant_1`	Adds Layer 2 VNIs to the tunnel interface and assigns a multicast group to the VNIs.

Configuring L2VPN EVPN on all VTEPs

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2vpn evpn**
4. **replication-type static**
5. **router-id loopback***number*

6. **exit**
7. **l2vpn evpn instance***instance-number***vlan-based**
8. **encapsulation vxlan**
9. **route-target export***route-target-id*
10. **route-target import***route-target-id*
11. **no auto-route-target**
12. **exit**
13. **vlan configuration***vlan-id*
14. **member evpn-instance***evpn-instance-number***vni***vni-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**l2vpn evpn** **Example:** `Device(config)# l2vpn evpn`	Enters L2VPN configuration mode
Step 4	**replication-type static** **Example:** `Device(config-l2vpn)# replication-type static`	Suppresses use of Inclusive Multicast Ethernet Tag (IMET) routes. IP Multicast is used for BUM traffic.
Step 5	**router-id loopback***number* **Example:** `Device(config-l2vpn)# router-id Loopback1`	Specifies the interface that will supply the IP addresses to be used in auto-generating route distinguishers.
Step 6	**exit** **Example:** `Device(config-l2vpn)# exit`	Exits the L2VPN configuration.
Step 7	**l2vpn evpn instance***instance-number***vlan-based** **Example:** `Device(config)# l2vpn evpn instance 1 vlan-based`	Configures VLAN based EVI in the L2VPN configuration mode. This command is optional if the route targets or the route distinguishers are not needed to be configured manually.
Step 8	**encapsulation vxlan** **Example:** `Device(config-l2vpn)# encapsulation vxlan`	Defines the encapsulation format as VXLAN

	Command or Action	Purpose
Step 9	**route-target export***route-target-id* **Example:** `Device(config-l2vpn)# route-target export 2:1`	Configures BGP route exchange.
Step 10	**route-target import***route-target-id* **Example:** `Device(config-l2vpn)# route-target import 2:1`	Configures BGP route exchange.
Step 11	**no auto-route-target** **Example:** `Device(config-l2vpn)# no auto-route-target`	Removes the automatically generated route-targets.
Step 12	**exit** **Example:** `Device(config-l2vpn)# exit`	Exits the L2VPN configuration.
Step 13	**vlan configuration***vlan-id* **Example:** `Device(config)# vlan configuration 11`	Enters the vlan feature configuration mode.
Step 14	**member evpn-instance***evpn-instance-number***vni***vni-number* **Example:** `Device(config-vlan)# member evpn-instance 1 vni 11001`	Configures the evpn vxlan vni instance.

Configuring access customer facing VLAN VTEP

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet***slot/port*
4. **switchport access vlan***vlan-id*
5. **switchport mode access**
6. **exit**
7. **interface gigabitethernet***slot/port*
8. **switchport trunk allowed vlan***vlan_list*
9. **switchport mode trunk**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface gigabitethernet***slot/port* **Example:** `Device(config)# interface GigabitEthernet1/0/11`	Enters the interface configuration mode on the Gigabit Ethernet interface.
Step 4	**switchport access vlan***vlan-id* **Example:** `Device(config-if)# switchport access vlan 11`	Sets the access VLAN when the interface is in access mode.
Step 5	**switchport mode access** **Example:** `Device(config-if)# switchport mode access`	Sets the interface as a nontrunking nontagged single-VLAN Ethernet interface.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Exits the interface configuration mode.
Step 7	**interface gigabitethernet***slot/port* **Example:** `Device(config)# interface TenGigabitEthernet1/1/7`	Enters the interface configuration mode on the Gigabit Ethernet interface.
Step 8	**switchport trunk allowed vlan***vlan_list* **Example:** `Device(config-if)# switchport trunk allowed vlan 11-210,901-905`	Configures the VLAN ids of the allowed VLANs for the interface.
Step 9	**switchport mode trunk** **Example:** `Device(config-if)# switchport mode trunk`	Sets the interface as an Ethernet trunk port.

Configuring IP VRF on VTEPs for Inter-VxLAN routing

SUMMARY STEPS

1. **enable**

2. **configure terminal**
3. **vrf definition***vrf-name*
4. **rd***route-distinguisher*
5. **address-family ipv4**
6. **route-target export***route-target-id*
7. **route-target import***route-target-id*
8. **route-target import***route-target-id***stitching**
9. **route-target export***route-target-id***stitching**
10. **exit-address-family**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vrf definition***vrf-name* **Example:** `Device(config)# vrf definition tenant_1`	Configures a virtual routing and forwarding (VRF) routing-table instance and enters VRF configuration mode.
Step 4	**rd***route-distinguisher* **Example:** `Device(config-vrf)# rd 1:1`	Creates routing and forwarding tables for a VRF.
Step 5	**address-family ipv4** **Example:** `Device(config-vrf)# address-family ipv4`	Enters address family configuration mode.
Step 6	**route-target export***route-target-id* **Example:** `Device(config-vrf-af)# route-target export 1:1`	Creates a list of export RTs for the VRF with the same parameters.
Step 7	**route-target import***route-target-id* **Example:** `Device(config-vrf-af)# route-target import 1:1`	Creates a list of import RTs for the VRF with the same parameters.
Step 8	**route-target import***route-target-id***stitching** **Example:** `Device(config-vrf-af)# route-target import 1:1 stitching`	Configures importing of routes from the EVPN BGP that have the matching route-target value.

	Command or Action	Purpose
Step 9	**route-target export***route-target-id***stitching** **Example:** `Device(config-vrf-af)# route-target export 1:1 stitching`	Configures exporting of routes from the VRF to the EVPN BGP and assigns the specified route-target identifiers to the BGP EVPN.
Step 10	**exit-address-family** **Example:** `Device(config-vrf-af)# exit-address-family`	Exits address-family configuration mode.

Verifying the VXLAN BGP EVPN Configuration

Command	Purpose
show nve vni	Displays VNIs associated in the NVE.
show ip mroute	Displays multicast routing table information.
show ip mfib	Displays forwarding entries and interfaces in the IPv4 Multicast Forwarding Information Base (MFIB).
show ip pim neighbors	Displays PIM neighbour table.
show ip pim tunnel	Displays information about the PIM register encapsulation and decapsulation tunnels on an interface.
show ip pim rp	Displays mapping information for the RP.
show l2vpn evpn evi [*evpn-id* | *all*]	Displays detailed information for a particular EVI or all EVIs.
show mac address-table vlan vlan id	Displays information for a specific VLAN.
show l2route evpn mac [**all** | **evi***vlan-id*]	Displays MAC and IP address information learnt by the switch in the EVPN control plane.
show bgp l2vpn evpn	Displays BGP information for L2VPN-EVPN address family.
show ip vrf*vrf-name*	Displays a summary of all VRFs present on the current router and their associated route-distinguishers and interface(s).
show bgp vpnv4 unicast vrf*vrf-name*	Displays VPNv4 routes from BGP table for a specific vrf.
show ip route vrf*vrf-name*	Displays the IP routing table associated with a specific VRF.

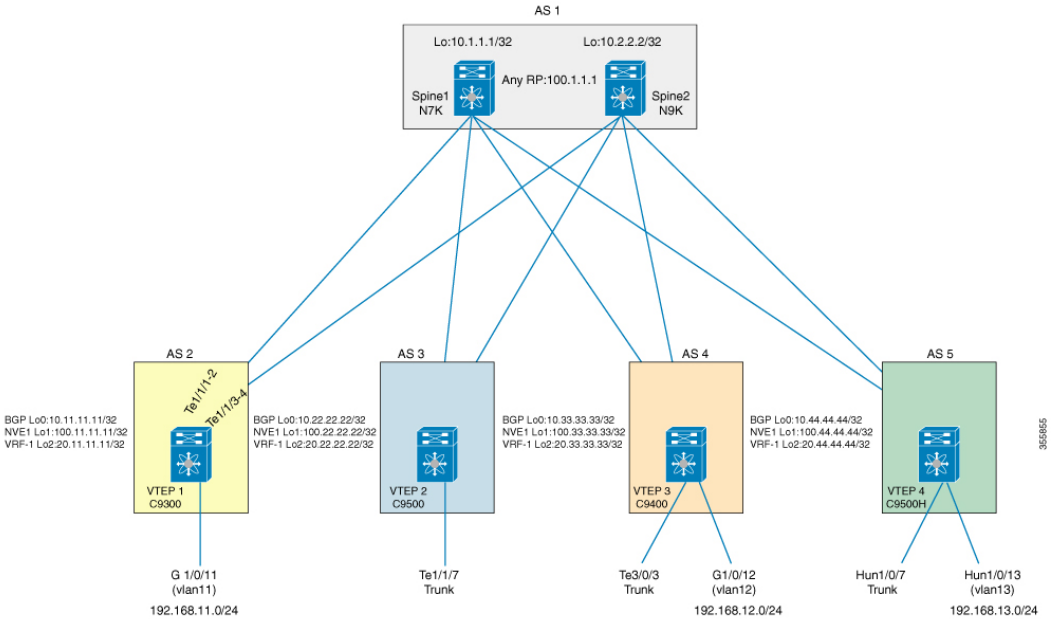
Command	Purpose
show l2vpn evpn mac	Displays the MAC address database for Layer 2 EVPN.
show l2vpn evpn mac ip	Displays the IP address database for Layer 2 EVPN.
show l2route evpn mac ip	Displays MAC IP routes.

Note Although the **show ip bgp** command is available for verifying a BGP configuration, as a best practice, it is preferable to use the**show bgp** command instead.

Examples of VXLAN BGP EVPN (EBGP)

Example: Configuring eBGP Multi-AS EVPN VxLAN design model

Figure 29: shows the topology used in the eBGP Multi –AS design model



Example: Configuring Underlay Transport (Unicast and Multicast) between all the VTEPs and the Spine(s):

eBGP peering between the spine and the VTEPs requires IP connectivity. This can be achieved by using static routes to reach loopback addresses between VTEPs and spines.

Configuring the spine

Note The following Spine configuration is applicable to Cisco Nexus Series Switches and is not applicable to Cisco Catalyst 9000 Family Switches.

```
ip pim rp-address 100.1.1.1 group-list 239.0.0.0/8
ip pim rp-candidate loopback1 group-list 239.0.0.0/8
ip pim anycast-rp 100.1.1.1 10.1.1.1
ip pim anycast-rp 100.1.1.1 10.2.2.2
!
interface loopback0
ip address 10.1.1.1/32
ip pim sparse-mode
!
interface loopback1
ip address 100.1.1.1/32
ip pim sparse-mode
!
interface port-channel1
mtu 9198
medium p2p
ip address 10.10.1.1/30
ip pim sparse-mode
!
interface port-channel2
mtu 9198
medium p2p
ip address 10.10.2.1/30
ip pim sparse-mode
!
interface port-channel3
mtu 9198
medium p2p
ip address 10.10.3.1/30
ip pim sparse-mode
```

Configuring the VTEP

```
ip multicast-routing
ip pim rp-address 100.1.1.1
!
ip routing
!
interface Loopback0
ip address 10.11.11.11 255.255.255.255
ip pim sparse-mode
!
interface Loopback1
ip address 100.11.11.11 255.255.255.255
ip pim sparse-mode
!
interface Loopback2
vrf forwarding tenant_1
ip address 11.11.11.11 255.255.255.255
!
interface Port-channel1
no switchport
ip address 10.10.1.2 255.255.255.252
ip pim sparse-mode
!
interface Port-channel11
```

```
no switchport
ip address 20.20.1.2 255.255.255.252
ip pim sparse-mode
!
interface TenGigabitEthernet1/1/2
no switchport
no ip address
channel-group 1 mode active
!
interface TenGigabitEthernet1/1/3
no switchport
no ip address
channel-group 11 mode active
```

Example: Configuring eBGP with EVPN address family between the Spine(s) and VTEPs:

Configuring the spine

Note The following Spine configuration is applicable to Cisco Nexus Series Switches and is not applicable to Cisco Catalyst 9000 Family Switches.

```
ip prefix-list lo_prefix seq 5 permit 0.0.0.0/0 le 32
route-map NH-UNCHANGED permit 10
set ip next-hop unchanged
route-map any_prefix permit 10
match ip address prefix-list lo_prefix
!
router bgp 1
router-id 10.1.1.1
log-neighbor-changes
address-family ipv4 unicast
redistribute direct route-map any_prefix
address-family l2vpn evpn
next-hop route-map NH-UNCHANGED
retain route-target all
!
neighbor 10.11.11.11
remote-as 2
update-source loopback0
ebgp-multihop 10
address-family ipv4 unicast
send-community both
soft-reconfiguration inbound
address-family l2vpn evpn
send-community both
route-map NH-UNCHANGED out
```

Configuring the VTEP

```
router bgp 2
bgp router-id interface Loopback0
bgp log-neighbor-changes
bgp graceful-restart
neighbor 10.1.1.1 remote-as 1
neighbor 10.1.1.1 ebgp-multihop 10
neighbor 10.1.1.1 update-source Loopback0
!
```

```
address-family ipv4
redistribute connected
neighbor 10.1.1.1 activate
exit-address-family
!
address-family l2vpn evpn
neighbor 10.1.1.1 activate
neighbor 10.1.1.1 send-community both
maximum-paths 2
exit-address-family
!
address-family ipv4 vrf tenant_1
advertise l2vpn evpn
redistribute connected
exit-address-family
```

Example: Configuring NVE on all VTEPs

Configuring the VTEP

```
interface nve1
no ip address
source-interface Loopback1
host-reachability protocol bgp
member vni 11001 mcast-group 239.0.1.1
member vni 11002 mcast-group 239.0.1.1
member vni 900001 vrf tenant_1
```

Example: Configuring L2VPN EVPN on VTEPs

Configuring the VTEP

```
l2vpn evpn
replication-type static
router-id Loopback1
!
l2vpn evpn instance 1 vlan-based
encapsulation vxlan
route-target export 2:1
route-target import 2:1
no auto-route-target
!
l2vpn evpn instance 2 vlan-based
encapsulation vxlan
route-target export 2:2
route-target import 2:2
no auto-route-target
```

Example: Configuring Access customer facing VLAN VTEPs

Configuring the VTEP

```
interface GigabitEthernet1/0/11
switchport access vlan 11
switchport mode access
!
interface TenGigabitEthernet1/1/7
switchport trunk allowed vlan 11-210,901-905
switchport mode trunk
```

Example: Configuring additional VNI, EVI and VLAN on VTEPs

```
VTEP1(config)#vlan 4000
VTEP1 (config-vlan)#state active
VTEP1 (config)#vlan configuration 4000
VTEP1 (config-vlan-config)#member evpn-instance 20000
```

Configuring the VTEP

```
vlan 11
state active
vlan 12
state active
vlan 901
state active
!
vlan configuration 11
member evpn-instance 1 vni 11001
!
vlan configuration 12
member evpn-instance 2 vni 11002
!
vlan configuration 901
member vni 900001
!
interface Vlan901
description connected to vni_900001
vrf forwarding tenant_1
ip unnumbered Loopback2
!
interface nve1
no ip address
source-interface Loopback1
host-reachability protocol bgp
member vni 11001 mcast-group 239.0.1.1
member vni 11002 mcast-group 239.0.1.1
member vni 900001 vrf tenant_1
```

Example: Configuring IP VRF on VTEPs for Inter-VxLAN routing

Configuring the VTEP

```
vrf definition tenant_1
rd 1:1
!
address-family ipv4
route-target export 1:1
route-target import 1:1
route-target export 1:1 stitching
route-target import 1:1 stitching
exit-address-family
```

Example: Configuring Access VLAN Interfaces (SVIs) on VTEPs

Configuring the VTEP

```
interface Vlan11
description vni_11001
mac-address 0001.0001.0001
vrf forwarding tenant_1
ip address 192.168.1.254 255.255.255.0
```

```
!
interface Vlan12
description vni_11002
mac-address 0001.0001.0001
vrf forwarding tenant_1
ip address 192.168.2.254 255.255.255.0
```

Example: Configuring additional L3-VNI in NVE interfaces

Configuring the VTEP

```
interface nve1
no ip address
source-interface Loopback1
host-reachability protocol bgp
member vni 11001 mcast-group 239.0.1.1
member vni 11002 mcast-group 239.0.1.1
member vni 900001 vrf tenant_1
```

Example: Configuring Core-facing VLANs and VLAN Interfaces

Configuring the VTEP

```
vlan configuration 901
member vni 900001
!
interface Vlan901
description connected to vni_900001
vrf forwarding tenant_1
ip unnumbered Loopback2
```

Example: Configuring iBGP/IGP EVPN VxLAN design model

Configuring the spine:

Note The following Spine configuration is applicable to Cisco Nexus Series Switches and is not applicable to Cisco Catalyst 9000 Family Switches.

```
feature-set fabric
hostname spine-1
!
feature telnet
feature scp-server
feature fabric forwarding
nv overlay evpn
feature ospf
feature bgp
feature pim
feature ipp
feature isis
feature fabric multicast
feature interface-vlan
feature lldp
feature fabric access
feature nv overlay
feature nxapi
```

```
!
ip pim rp-address 4.5.4.5 group-list 224.0.0.0/4
!
vlan 1
!
interface Vlan1
!
interface Ethernet1/1 ip address 10.14.1.4/24
ip router ospf 1 area 0.0.0.0
ip pim sparse-mode
no shutdown
!
interface loopback0
ip address 4.4.4.4/32
ip router ospf 1 area 0.0.0.0
ip pim sparse-mode
interface loopback1
ip address 4.5.4.5/32
ip router ospf 1 area 0.0.0.0
ip pim sparse-mode
!
router ospf 1
router-id 4.4.4.4
!
router bgp 100
router-id 4.4.4.4
address-family l2vpn evpn
neighbor 1.1.1.1 remote-as 100
update-source loopback0
address-family ipv4 unicast
send-community both
route-reflector-client
address-family l2vpn evpn
send-community both
route-reflector-client
```

Configuring the VTEP

```
vrf definition l3vni50000
rd 101:1
!
address-family ipv4
route-target export 100:1 stitching
route-target import 100:1 stitching
exit-address-family
!
ip multicast-routing
ip pim rp-address 4.5.4.5
!
l2vpn evpn
replication-type static
!
vlan 10
State active
vlan 11
State active
vlan 501
state active
!
vlan configuration 10
member evpn-instance 10 vni 100010
!
vlan configuration 11
member evpn-instance 11 vni 100011
!
```

```
vlan configuration 501
member vni 50000
!
interface Loopback0
ip address 1.1.1.1 255.255.255.255
ip pim sparse-mode
ip ospf 1 area 0
!
interface GigabitEthernet1/0/1
switchport mode trunk
!
interface GigabitEthernet1/0/2
switchport access vlan 10
switchport mode access
!
interface TenGigabitEthernet3/0/1
description To Spine1
no switchport
ip address 10.14.1.1 255.255.255.0
ip pim sparse-mode
ip ospf 1 area 0
!
interface TenGigabitEthernet3/0/2
description To Spine1
no switchport
ip address 10.15.1.1 255.255.255.0
ip pim sparse-mode
ip ospf 1 area 0
!
interface Vlan10
description connected to 100010
mac-address 0001.0001.0001
vrf forwarding l3vni50000
ip address 192.168.10.1 255.255.255.0
!
interface Vlan11
description connected to 100011
mac-address 0001.0001.0001
vrf forwarding l3vni50000
ip address 192.168.11.1 255.255.255.0
!
interface Vlan501
description connected to 50000
vrf forwarding l3vni50000
ip unnumbered Loopback0
!
router ospf 1
router-id 1.1.1.1
nsr
!
router bgp 100
bgp router-id 1.1.1.1
bgp log-neighbor-changes
bgp graceful-restart
neighbor 4.4.4.4 remote-as 100
neighbor 4.4.4.4 update-source Loopback0
!
address-family ipv4
redistribute connected
neighbor 4.4.4.4 activate
exit-address-family
!
address-family l2vpn evpn
neighbor 4.4.4.4 activate
```

```
neighbor 4.4.4.4 send-community both
exit-address-family
!
address-family ipv4 vrf l3vni50000
advertise l2vpn evpn
redistribute connected
exit-address-family
!
interface nve1
no ip address
source-interface Loopback0
host-reachability protocol bgp
member vni 100010 mcast-group 227.0.0.1
member vni 100011 mcast-group 227.0.0.1
member vni 50000 vrf l3vni50000
```

Example: Verifying L2/L3 VNI in NVE

```
# show nve vni

Interface VNI Multicast-group VNI state Mode VLAN cfg vrf
nve1 60519 233.1.1.19 Up L2CP 519 CLI N/A
nve1 60518 233.1.1.18 Up L2CP 518 CLI N/A
```

Example: Verifying Multicast in multicast routing table

```
# show ip mroute
Leaf_1#sh ip mroute
IP Multicast Routing Table
Outgoing interface flags: H - Hardware switched, A - Assert winner, p - PIM Join
Timers: Uptime/Expires
Interface state: Interface, Next-Hop or VCD, State/Mode
(*, 239.0.1.1), 5d16h/stopped, RP 100.1.1.1, flags: SJCFx
Incoming interface: Port-channel11, RPF nbr 20.20.1.1
Outgoing interface list:
Tunnel0, Forward/Sparse-Dense, 5d16h/00:01:17
!
(100.11.11.11, 239.0.1.1), 00:02:18/00:00:41, flags: FTx
Incoming interface: Loopback1, RPF nbr 0.0.0.0, Registering
Outgoing interface list:
Port-channel11, Forward/Sparse, 00:02:18/00:03:14


# show ip mfib
Forwarding Counts: Pkt Count/Pkts per second/Avg Pkt Size/Kbits per second
Other counts: Total/RPF failed/Other drops
I/O Item Counts: FS Pkt Count/PS Pkt Count
Default
(*,224.0.0.0/4) Flags: C HW
SW Forwarding: 0/0/0/0, Other: 0/0/0
HW Forwarding: 0/0/0/0, Other: 0/0/0
(*,224.0.1.40) Flags: C HW
SW Forwarding: 0/0/0/0, Other: 0/0/0
HW Forwarding: 0/0/0/0, Other: 0/0/0
Port-channel11 Flags: A NS
Loopback0 Flags: F IC NS
Pkts: 0/0
(*,239.0.1.1) Flags: C HW
SW Forwarding: 0/0/0/0, Other: 0/0/0
HW Forwarding: 187/0/190/0, Other: 0/0/0
Port-channel11 Flags: A NS
```

```
Tunnel0, VXLAN Decap Flags: F NS
Pkts: 0/0
(100.11.11.11,239.0.1.1) Flags: HW
SW Forwarding: 0/0/0/0, Other: 0/0/0
HW Forwarding: 0/0/0/0, Other: 0/0/0
Null0 Flags: A NS
Port-channel11 Flags: F NS
Pkts: 0/0
Tunnel1 Flags: F
Pkts: 0/0


# show ip pim neighbors
PIM Neighbor Table
Mode: B - Bidir Capable, DR - Designated Router, N - Default DR Priority,
P - Proxy Capable, S - State Refresh Capable, G - GenID Capable,
L - DR Load-balancing Capable
Neighbor Interface Uptime/Expires Ver DR
Address Prio/Mode
10.10.1.1 Port-channel1 5d16h/00:01:40 v2 1 / G
20.20.1.1 Port-channel11 5d16h/00:01:20 v2 1 / G


# show ip pim tunnel
Tunnel1*
Type : PIM Encap
RP : 100.1.1.1
Source : 20.20.1.2
State : UP
Last event : Created (5d16h)
# sh ip pim rp
Group: 239.0.1.1, RP: 100.1.1.1, uptime 5d16h, expires never
```

Example: Verifying EVPN Instance in EVPN Manager

```
# show l2vpn evpn evi 1 detail
EVPN instance: 1 (VLAN Based)
RD: 100.11.11.11:1 (auto)
Import-RTs: 2:1
Export-RTs: 2:1
Per-EVI Label: none
State: Established
Encapsulation: vxlan
Vlan: 11
Ethernet-Tag: 0
State: Established
Core If: Vlan901
Access If: Vlan11
RMAC: ec1d.8b75.eac8
Core Vlan: 901
L2 VNI: 11001
L3 VNI: 900001
VTEP IP: 100.11.11.11
MCAST IP: 239.0.1.1
VRF: tenant_1
Pseudoports:
TenGigabitEthernet1/1/7 service instance 11
```

Example: Verifying MAC Table

```
# show mac address-table vlan 11
Mac Address Table
-------------------------------------------
Vlan Mac Address Type Ports
---- ----------- -------- -----
11 0001.0001.0001 STATIC Vl11 ---------□ SVI mac for Anycast Gateway
11 0011.0011.0005 DYNAMIC Te1/1/7--------□ dynamically learned
Total Mac Addresses for this criterion: 2
```

Example: Verifying MAC entries in EVPN Manager

```
# show l2vpn evpn mac
MAC Address EVI VLAN ESI Ether Tag Next Hop
-------------- ----- ----- ---------------------- ---------- ---------------
0011.0011.00c9 1 11 0000.0000.0000.0000.0000 0 Te1/1/7:11
0012.0012.0001 1 11 0000.0000.0000.0000.0000 0 100.22.22.22
0013.0013.0001 1 11 0000.0000.0000.0000.0000 0 100.33.33.33
0014.0014.0001 1 11 0000.0000.0000.0000.0000 0 100.44.44.44
```

Example: Verifying MAC routes in BGP

```
# show bgp l2vpn evpn evi 1---------□ only evi 1 will be shown
BGP table version is 654847, local router ID is 10.11.11.11
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
x best-external, a additional-path, c RIB-compressed,
t secondary path, L long-lived-stale,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found
Network Next Hop Metric LocPrf Weight Path
Route Distinguisher: 100.11.11.11:1
*> [2][100.11.11.11:1][0][48][0011001100C9][0][*]/20
:: 32768 ?
*> [2][100.11.11.11:1][0][48][001200120001][0][*]/20
100.22.22.22 0 1 3 ?
*> [2][100.11.11.11:1][0][48][001200120001][32][192.168.1.2]/24
100.22.22.22 0 1 3 ?
*> [2][100.11.11.11:1][0][48][001300130001][0][*]/20
100.33.33.33 0 1 4 ?
*> [2][100.11.11.11:1][0][48][001300130001][32][192.168.1.3]/24
100.33.33.33 0 1 4 ?
*> [2][100.11.11.11:1][0][48][001400140001][0][*]/20
100.44.44.44 0 1 4 ?
*> [2][100.11.11.11:1][0][48][001400140001][32][192.168.1.4]/24
100.44.44.44 0 1 4 ?
```

Example: Verifying MAC routes in Layer 2 Routing Information Base

```
#show l2route evpn mac
EVI ETag Prod Mac Address Next Hop(s) Seq Number
----- ---------- ----- -------------- ----------------------------- ----------
1 0 BGP 0012.0012.0001 V:11001 100.22.22.22 0
1 0 BGP 0013.0013.0001 V:11001 100.33.33.33 0
1 0 BGP 0014.0014.0001 V:11001 100.44.44.44 0
1 0 L2VPN 0011.0011.00c9 Te1/1/7:11 0
```

Example: Verifying IP VRF with all SVIs

```
# show ip vrf
Name                              Default RD            Interfaces
Mgmt-vrf                          <not set>             Gi0/0
tenant_1                          1:1                   Lo2
                                                        Vl11
                                                        Vl12
```

Example: Verifying MAC/IP entries in MAC VRFs (EVIs)

```
# show bgp l2vpn evpn evi 1 route-type 2
BGP routing table entry for [2][100.11.11.11:1][0][48][0011001100C9][32][10.0.0.2]/24,
version 7
Paths: (1 available, best #1, table evi_1)
  Advertised to update-groups:
     1
  Refresh Epoch 1
  Local
    :: (via default) from 0.0.0.0 (10.11.11.11)
      Origin incomplete, localpref 100, weight 32768, valid, sourced, local, best
      EVPN ESI: 00000000000000000000, Label1 11001-□ L2 VNI
      Extended Community: RT:2:1 ENCAP:8
      Local irb vxlan vtep:
        vrf:tenant_1, l3-vni:900001--------□ IP VRF and L3 VNI
        local router mac:EC1D.8B75.EAC8
        core-irb interface:Vlan901----□ core SVI
        vtep-ip:100.11.11.11
      rx pathid: 0, tx pathid: 0x0
```

Example: Verifying Remote MAC/IP and IP Prefix routes in L3VNI (IP VRF)

```
# show bgp vpnv4 unicast vrf tenant_1-------□ not all routes will be shown
BGP table version is 8583, local router ID is 10.11.11.11
     Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 1:1 (default for vrf tenant_1)
AF-Private Import to Address-Family: L2VPN E-VPN, Pfx Count/Limit: 11/1000
 *>   11.11.11.11/32   0.0.0.0                  0         32768 ?
 *>   11.22.22.22/32   100.22.22.22                         0 1 3 ?
 *>   11.33.33.33/32   100.33.33.33                         0 1 4 ?
 *>   11.44.44.44/32   100.44.44.44                         0 1 4 ?
 *    192.168.1.0         100.44.44.44                         0 1 4 ?
*                                 100.33.33.33                           0 1 4 ?
 *                                 100.22.22.22                           0 1 3 ?
 *>                                 0.0.0.0                  0         32768 ?
 *>   192.168.1.2/32   100.22.22.22                         0 1 3 ?
 *>   192.168.1.3/32   100.33.33.33                         0 1 4 ?
 *>   192.168.1.4/32   100.44.44.44                         0 1 4 ?
 *    192.168.2.0          100.44.44.44                         0 1 4 ?
 *                                   100.33.33.33                           0 1 4 ?
 *                                   100.22.22.22                           0 1 3 ?
 *>                                   0.0.0.0                  0         32768 ?
```

Example: Verifying IP routes are installed in L3 VNI (IP VRF)

```
# show ip route vrf tenant_1
Routing Table: tenant_1
Gateway of last resort is not set
```

```
      11.0.0.0/32 is subnetted, 3 subnets
C        11.11.11.11 is directly connected, Loopback2
B        11.22.22.22 [20/0] via 100.22.22.22, 00:13:21, Vlan901
B        11.33.33.33 [20/0] via 100.33.33.33, 00:13:21, Vlan901
B        11.44.44.44 [20/0] via 100.44.44.44, 00:12:51, Vlan901
      192.168.1.0/24 is variably subnetted, 4 subnets, 2 masks
C        192.168.1.0/24 is directly connected, Vlan11
B        192.168.1.3/32 [20/0] via 100.33.33.33, 16:26:48, Vlan901
B        192.168.1.4/32 [20/0] via 100.44.44.44, 2d19h, Vlan901
L        192.168.1.254/32 is directly connected, Vlan11
      192.168.2.0/24 is variably subnetted, 4 subnets, 2 masks
C        192.168.2.0/24 is directly connected, Vlan12
B        192.168.2.3/32 [20/0] via 100.33.33.33, 02:52:20, Vlan901
B        192.168.2.4/32 [20/0] via 100.44.44.44, 2d19h, Vlan901
L        192.168.2.254/32 is directly connected, Vlan12
      192.168.3.0/24 is variably subnetted, 4 subnets, 2 masks
C        192.168.3.0/24 is directly connected, Vlan13
B        192.168.3.3/32 [20/0] via 100.33.33.33, 2d19h, Vlan901
```

Example: Verifying MAC/IP entries in EVPN Manager

```
#show l2vpn evpn mac ip

IP Address EVI VLAN MAC Address Next Hop(s)
------------------------ ----- ----- -------------- -------------------------
10.0.0.1 1 11 0011.0011.00c9 Te1/1/7:11
10.0.0.2 1 11 0012.0012.0001 100.22.22.22
```

Example: Verifying MAC/IP routes in Layer 2 Routing Informatio Base

```
#show l2route evpn mac ip

EVI ETag Prod Mac Address Host IP Next Hop(s)
----- ---------- ----- -------------- --------------- --------------------------
1 0 BGP 0012.0012.0001 10.0.0.2 V:11001 100.22.22.22
1 0 L2VPN 0011.0011.00c9 10.0.0.1 Te1/1/7:11
```

Feature History and Information for VXLAN BGP EVPN

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Table 17: Feature History for VXLAN BGP EVPN

Release	Feature Information
Cisco IOS XE Fuji 16.9.1	The feature was introduced.