Implementing Multiple Spanning Tree Protocol

This module provides conceptual and configuration information for Multiple Spanning Tree Protocol on Cisco ASR 9000 Series Routers. Multiple Spanning Tree Protocol (MSTP) is a spanning-tree protocol used to prevent loops in bridge configurations. Unlike other types of STPs, MSTP can block ports selectively by VLAN.

Feature History for Implementing Multiple Spanning Tree Protocol

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release 3.7.3</td>
<td>This feature was introduced on Cisco ASR 9000 Series Routers.</td>
</tr>
<tr>
<td>Release 3.9.1</td>
<td>Support for MSTP over Bundles feature was added.</td>
</tr>
<tr>
<td>Release 4.0.1</td>
<td>Support for PVST+ and PVSTAG features was added.</td>
</tr>
</tbody>
</table>

Contents

- Prerequisites for Implementing Multiple Spanning Tree Protocol, page 308
- Information About Implementing Multiple Spanning Tree Protocol, page 308
- How to Implement Multiple Spanning Tree Protocol, page 320
- Configuration Examples for Implementing MSTP, page 343
- Additional References, page 352
Prerequisites for Implementing Multiple Spanning Tree Protocol

This prerequisite applies to implementing MSTP:

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command.

If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Information About Implementing Multiple Spanning Tree Protocol

To implement Ethernet services access lists, you must understand these concepts:

- Spanning Tree Protocol Overview
- Multiple Spanning Tree Protocol Overview
- MSTP Supported Features
- Restrictions for configuring MSTP
- Access Gateway
- Multiple VLAN Registration Protocol

Spanning Tree Protocol Overview

Ethernet is no longer just a link-layer technology used to interconnect network vehicles and hosts. Its low cost and wide spectrum of bandwidth capabilities coupled with a simple plug and play provisioning philosophy have transformed Ethernet into a legitimate technique for building networks, particularly in the access and aggregation regions of service provider networks.

Ethernet networks lacking a TTL field in the Layer 2 (L2) header and, encouraging or requiring multicast traffic network-wide, are susceptible to broadcast storms if loops are introduced. However, loops are a desirable property as they provide redundant paths. Spanning tree protocols (STP) are used to provide a loop free topology within Ethernet networks, allowing redundancy within the network to deal with link failures.

There are many variants of STP; however, they work on the same basic principle. Within a network that may contain loops, a sufficient number of interfaces are disabled by STP so as to ensure that there is a loop-free spanning tree, that is, there is exactly one path between any two devices in the network. If there is a fault in the network that affects one of the active links, the protocol recalculates the spanning tree so as to ensure that all devices continue to be reachable. STP is transparent to end stations which cannot detect whether they are connected to a single LAN segment or to a switched LAN containing multiple segments and using STP to ensure there are no loops.
STP Protocol Operation

All variants of STP operate in a similar fashion: STP frames (known as bridge protocol data units (BPDUs)) are exchanged at regular intervals over Layer 2 LAN segments, between network devices participating in STP. Such network devices do not forward these frames, but use the information to construct a loop free spanning tree.

The spanning tree is constructed by first selecting a device which is the root of the spanning tree (known as the root bridge), and then by determining a loop free path from the root bridge to every other device in the network. Redundant paths are disabled by setting the appropriate ports into a blocked state, where STP frames can still be exchanged but data traffic is never forwarded. If a network segment fails and a redundant path exists, the STP protocol recalculates the spanning tree topology and activates the redundant path, by unblocking the appropriate ports.

The selection of the root bridge within an STP network is determined by the configured priority and the embedded bridge ID of each device. The device with the lowest priority, or with equal lowest priority but the lowest bridge ID, is selected as the root bridge.

The selection of the active path among a set of redundant paths is determined primarily by the port path cost. The port path cost represents the cost of transiting between that port and the root bridge - the further the port is from the root bridge, the higher the cost. The cost is incremented for each link in the path, by an amount that is (by default) dependent on the media speed. Where two paths from a given LAN segment have an equal cost, the selection is further determined by the priority and bridge ID of the attached devices, and in the case of two attachments to the same device, by the configured port priority and port ID of the attached ports.

Once the active paths have been selected, any ports that do not form part of the active topology are moved to the blocking state.

Topology Changes

Network devices in a switched LAN perform MAC learning; that is, they use received data traffic to associate unicast MAC addresses with the interface out of which frames destined for that MAC address should be sent. If STP is used, then a recalculation of the spanning tree (for example, following a failure in the network) can invalidate this learned information. The protocol therefore includes a mechanism to notify topology changes around the network, so that the stale information can be removed (flushed) and new information can be learned based on the new topology.

A Topology Change notification is sent whenever STP moves a port from the blocking state to the forwarding state. When it is received, the receiving device flushes the MAC learning entries for all ports that are not blocked other than the one where the notification was received, and also sends its own topology change notification out of those ports. In this way, it is guaranteed that stale information is removed from all the devices in the network.

Variants of STP

There are many variants of the Spanning Tree Protocol:

- Legacy STP (STP)—The original STP protocol was defined in IEEE 802.1D-1998. This creates a single spanning tree which is used for all VLANs and most of the convergence is timer-based.

- Rapid STP (RSTP)—This is an enhancement defined in IEEE 802.1D-2004 to provide more event-based, and hence faster, convergence. However, it still creates a single spanning tree for all VLANs.
Multiple STP (MSTP)—A further enhancement was defined in IEEE 802.1Q-2005. This allows multiple spanning trees to be created over the same physical topology. By assigning different VLANs to the different spanning trees, data traffic can be load-balanced over different physical links. The number of different spanning trees that can be created is restricted to a much smaller number than the number of possible VLANs; however, multiple VLANs can be assigned to the same spanning tree. The BPDUs used to exchange MSTP information are always sent untagged; the VLAN and spanning tree instance data is encoded inside the BPDU.

Per-Vlan STP (PVST)—This is an alternative mechanism for creating multiple spanning trees; it was developed by Cisco before the standardization of MSTP. Using PVST, a separate spanning tree is created for each VLAN. There are two variants: PVST+ (based on legacy STP), and PVRST (based on RSTP). At a packet level, the separation of the spanning trees is achieved by sending standard STP or RSTP BPDUs, tagged with the appropriate VLAN tag.

REP (Cisco-proprietary ring-redundancy protocol)—This is a Cisco-proprietary protocol for providing resiliency in rings. It is included for completeness, as it provides MSTP compatibility mode, using which, it interoperates with an MSTP peer.

Multiple Spanning Tree Protocol Overview

The Multiple Spanning Tree Protocol (MSTP) is an STP variant that allows multiple and independent spanning trees to be created over the same physical network. The parameters for each spanning tree can be configured separately, so as to cause a different network devices to be selected as the root bridge or different paths to be selected to form the loop-free topology. Consequently, a given physical interface can be blocked for some of the spanning trees and unblocked for others.

Having set up multiple spanning trees, the set of VLANs in use can be partitioned among them; for example, VLANs 1 - 100 can be assigned to spanning tree 1, VLANs 101 - 200 can be assigned to spanning tree 2, VLANs 201 - 300 can be assigned to spanning tree 3, and so on. Since each spanning tree has a different active topology with different active links, this has the effect of dividing the data traffic among the available redundant links based on the VLAN - a form of load balancing.

MSTP Regions

Along with supporting multiple spanning trees, MSTP also introduces the concept of regions. A region is a group of devices under the same administrative control and have similar configuration. In particular, the configuration for the region name, revision, and the mapping of VLANs to spanning tree instances must be identical on all the network devices in the region. A digest of this information is included in the BPDUs sent by each device, so as to allow other devices to verify whether they are in the same region.

Figure 31 shows the operation of MST regions when bridges running MSTP are connected to bridges running legacy STP or RSTP. In this example, switches SW1, SW2, SW3, SW4 support MSTP, while switches SW5 and SW6 do not.
To handle this situation, an Internal Spanning Tree (IST) is used. This is always spanning tree instance 0 (zero). When communicating with non-MSTP-aware devices, the entire MSTP region is represented as a single switch. The logical IST topology in this case is shown in Figure 32.

The same mechanism is used when communicating with MSTP devices in a different region. For example, SW5 in Figure 32 could represent a number of MSTP devices, all in a different region compared to SW1, SW2, SW3 and SW4.

**MSTP Port Fast**

MSTP includes a Port Fast feature for handling ports at the edge of the switched Ethernet network. For devices that only have one link to the switched network (typically host devices), there is no need to run MSTP, as there is only one available path. Furthermore, it is undesirable to trigger topology changes (and resultant MAC flushes) when the single link fails or is restored, as there is no alternative path.

By default, MSTP monitors ports where no BPDUs are received, and after a timeout, places them into edge mode whereby they do not participate in MSTP. However, this process can be speeded up (and convergence of the whole network thereby improved) by explicitly configuring edge ports as port fast.
Port Fast is implemented as a Cisco-proprietary extension in Cisco implementations of legacy STP. However, it is encompassed in the standards for RSTP and MSTP, where it is known as Edge Port.

**MSTP Root Guard**

In networks with shared administrative control, it may be desirable for the network administrator to enforce aspects of the network topology and in particular, the location of the root bridge. By default, any device can become the root bridge for a spanning tree, if it has a lower priority or bridge ID. However, a more optimal forwarding topology can be achieved by placing the root bridge at a specific location in the centre of the network.

The administrator can set the root bridge priority to 0 in an effort to secure the root bridge position; however, this is no guarantee against another bridge which also has a priority of 0 and has a lower bridge ID.

The root guard feature provides a mechanism that allows the administrator to enforce the location of the root bridge. When root guard is configured on an interface, it prevents that interface from becoming a root port (that is, a port via which the root can be reached). If superior information is received via BPDUs on the interface that would normally cause it to become a root port, it instead becomes a backup or alternate port. In this case, it is placed in the blocking state and no data traffic is forwarded.

The root bridge itself has no root ports. Thus, by configuring root guard on every interface on a device, the administrator forces the device to become the root, and interfaces receiving conflicting information are blocked.

**MSTP Topology Change Guard**

In certain situations, it may be desirable to prevent topology changes originating at or received at a given port from being propagated to the rest of the network. This may be the case, for example, when the network is not under a single administrative control and it is desirable to prevent devices external to the core of the network from causing MAC address flushing in the core. This behavior can be enabled by configuring Topology Change Guard on the port.

**Note**

Root Guard is implemented as a Cisco-proprietary extension in Cisco implementations of legacy STP and RSTP. However, it is encompassed in the standard for MSTP, where it is known as Restricted Role.

**Note**

Topostry Change Guard is known as Restricted TCN in the MSTP standard.
MSTP Supported Features

Cisco ASR 9000 Series Routers support MSTP, as defined in IEEE 802.1Q-2005, on physical Ethernet interfaces and Ethernet Bundle interfaces. Note that this includes the Port Fast, Backbone Fast, Uplink Fast and Root Guard features found in Cisco implementations of legacy STP, RSTP and PVST, as these are encompassed by the standard MSTP protocol. Cisco ASR 9000 Series Routers can operate in either standard 802.1Q mode, or in Provide Edge (802.1ad) mode. In provider edge mode, a different MAC address is used for BPDUs, and any BPDUs received with the 802.1Q MAC address are forwarded transparently.

In addition, these additional Cisco features are supported:

- **BPDU Guard**—This Cisco feature protects against misconfiguration of edge ports.
- **Flush Containment**—This Cisco feature helps prevent unnecessary MAC flushes that would otherwise occur following a topology change.
- **Bringup Delay**—This Cisco feature prevents an interface from being added to the active topology before it is ready to forward traffic.

Interoperation with RSTP is supported, as described in the 802.1Q standard; however, interoperation with legacy STP is not supported.

BPDU Guard

BPDU Guard is a Cisco feature that protects against misconfiguration of edge ports. It is an enhancement to the MSTP port fast feature. When port fast is configured on an interface, MSTP considers that interface to be an edge port and removes it from consideration when calculating the spanning tree. When BPDU Guard is configured, MSTP additionally shuts down the interface using error-disable if an MSTP BPDU is received.

Flush Containment

Flush containment is a Cisco feature that helps prevent unnecessary MAC flushes due to unrelated topology changes in other areas of a network. This is best illustrated by example. Figure 33 shows a network containing four devices. Two VLANs are in use: VLAN 1 is only used on device D, while VLAN 2 spans devices A, B and C. The two VLANs are in the same spanning tree instance, but do not share any links.

**Figure 33**  Flush Containment

If the link AB goes down, then in normal operation, as C brings up its blocked port, it sends out a topology change notification on all other interfaces, including towards D. This causes a MAC flush to occur for VLAN 1, even though the topology change which has taken place only affects VLAN 2.
Flush containment helps deal with this problem by preventing topology change notifications from being sent on interfaces on which no VLANs are configured for the MSTI in question. In the example network this would mean no topology change notifications would be sent from C to D, and the MAC flushes which take place would be confined to the right hand side of the network.

**Note**
Flush containment is enabled by default, but can be disabled by configuration, thus restoring the behavior described in the IEEE 802.1Q standard.

### Bringup Delay

Bringup delay is a Cisco feature that stops MSTP from considering an interface when calculating the spanning tree, if the interface is not yet ready to forward traffic. This is useful when a line card first boots up, as the system may declare that the interfaces on that card are *Up* before the dataplane is fully ready to forward traffic. According to the standard, MSTP considers the interfaces as soon as they are declared *Up*, and this may cause it to move other interfaces into the blocking state if the new interfaces are selected instead.

Bringup delay solves this problem by adding a configurable delay period which occurs as interfaces that are configured with MSTP first come into existence. Until this delay period ends, the interfaces remain in blocking state, and are not considered when calculating the spanning tree.

Bringup delay only takes place when interfaces which are already configured with MSTP are created, for example, on a card reload. No delay takes place if an interface which already exists is later configured with MSTP.

### Restrictions for configuring MSTP

These restrictions apply when using MSTP:

- MSTP must only be enabled on interfaces where the interface itself (if it is in L2 mode) or all of the subinterfaces have a simple encapsulation configured. These encapsulation matching criteria are considered simple:
  - Single-tagged 802.1Q frames
  - Double-tagged Q-in-Q frames (only the outermost tag is examined)
  - 802.1ad frames (if MSTP is operating in Provider Bridge mode)
  - Ranges or lists of tags (any of the above)

**Note**
Subinterfaces with a *default* or *untagged* encapsulation are not supported.

- If an L2 interface or subinterface is configured with an encapsulation that matches multiple VLANs, then all of those VLANs must be mapped to the same spanning tree instance. There is therefore a single spanning tree instance associated with each L2 interface or subinterface.
- All the interfaces or subinterfaces in a given bridge domain must be associated with the same spanning tree instance.
- Multiple subinterfaces on the same interface must not be associated with the same spanning tree instance, unless those subinterfaces are in the same split horizon group. In other words, hair-pinning is not possible.
Implementing Multiple Spanning Tree Protocol

Information About Implementing Multiple Spanning Tree Protocol

LSC-315

Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Configuration Guide

OL-23107-02

Across the network, L2 interfaces or subinterfaces must be configured on all redundant paths for all the VLANs mapped to each spanning tree instance. This is to avoid inadvertent loss of connectivity due to STP blocking of a port.

Caution

A subinterface with a default or untagged encapsulation will lead to an MSTP state machine failure.

Access Gateway

One common deployment scenario for Cisco ASR 9000 Series Routers is as an nPE gateway device situated between a network of uPE access devices and a core or aggregation network. Each gateway device may provide connectivity for many access networks, as shown in Figure 34. The access networks (typically rings) have redundant links to the core or aggregation network, and therefore must use some variant of STP or a similar protocol to ensure the network remains loopfree.

Figure 34 Core or Aggregation Network

It is possible for the gateway devices to also participate in the STP protocol. However, since each gateway device may be connected to many access networks, this would result in one of two solutions:

- A single topology is maintained covering all of the access networks. This is undesirable as it means topology changes in one access network could impact all the other access networks.

- The gateway devices run multiple instances of the STP protocol, one for each access network. This means a separate protocol database and separate protocol state machines are maintained for each access network, which is undesirable due to the memory and CPU resource that would be required on the gateway device.

It can be seen that both of these options have significant disadvantages.

Another alternative is for the gateway devices to tunnel protocol BPDUs between the legs of each access network, but not to participate in the protocol themselves. While this results in correct loopfree topologies, it also has significant downsides:
• Since there is no direct connection between the legs of the access ring, a failure in one of the leg links is not immediately detected by the access device connected to the other leg. Therefore, recovery from the failure must wait for protocol timeouts, which leads to a traffic loss of at least six seconds.

• As the gateway devices do not participate in the protocol, they are not aware of any topology changes in the access network. The aggregation network may therefore direct traffic destined for the access network over the wrong leg, following a topology change. This can lead to traffic loss on the order of the MAC learning timeout (5 minutes by default).

Access gateway is a Cisco feature intended to address this deployment scenario, without incurring the disadvantages of the solutions described above.

Overview of Access Gateway

Access gateway is based on two assumptions:

• Both gateway devices provide connectivity to the core or aggregation network at all times. Generally, resiliency mechanisms used within the core or aggregation network are sufficient to ensure this is the case. In many deployments, VPLS is used in the core or aggregation network to provide this resiliency.

• The desired root of all of the spanning trees for each access network is one of the gateway devices. This will be the case if (as is typical) the majority of the traffic is between an access device and the core or aggregation network, and there is little if any traffic between the access devices.

With these assumptions, an STP topology can be envisaged where for every spanning tree, there is a virtual root bridge behind (that is, on the core side of) the gateway devices, and both gateway devices have a zero cost path to the virtual root bridge. In this case, the ports that connect the gateway devices to the access network would never be blocked by the spanning tree protocol, but would always be in the forwarding state. This is illustrated in Figure 35.

Figure 35 Access Networks

![Figure 35 Access Networks](image-url)
With this topology, it can be observed that the BPDUs sent by the gateway devices are constant: since the root bridge never changes (as we assume the aggregation or core network always provides connectivity) and the ports are always forwarding, the information sent in the BPDUs never changes.

Access gateway makes use of this by removing the need to run the full STP protocol and associated state machines on the gateway devices, and instead just sends statically configured BPDUs towards the access network. The BPDUs are configured so as to mimic the behavior above, so that they contain the same information that would be sent if the full protocol was running. To the access devices, it appears that the gateway devices are fully participating in the protocol; however, since in fact the gateway devices are just sending static BPDUs, very little memory or CPU resource is needed on the gateway devices, and many access networks can be supported simultaneously.

For the most part, the gateway devices can ignore any BPDUs received from the access network; however, one exception is when the access network signals a topology change. The gateway devices can act on this appropriately, for example by triggering an LDP MAC withdrawal in the case where the core or aggregation network uses VPLS.

In many cases, it is not necessary to have direct connectivity between the gateway devices; since the gateway devices statically send configured BPDUs over the access links, they can each be configured independently (so long as the configuration on each is consistent). This also means that different access networks can use different pairs of gateway devices, as shown in Figure 36.

**Figure 36 Access Networks**

![Access Networks Diagram]

**Note**

Although Figure 36 shows access rings, in general there are no restrictions on the access network topology or the number or location of links to the gateway devices.

Access gateway ensures loop-free connectivity in the event of these failure cases:

- Failure of a link in the access network.
- Failure of a link between the access network and the gateway device.
- Failure of an access device.
- Failure of a gateway device.
Topology Change Propagation

There is one case where the two gateway devices need to exchange BPDUs between each other, and this is to handle topology changes in the access network. If a failure in the access network results in a topology change that causes a previously blocked port to move to forwarding, the access device sends a topology change notification out on that port, so as to notify the rest of the network about the change and trigger the necessary MAC learning flushes. Typically, the topology change notification is sent towards the root bridge, in the case of access gateway, that means it is sent to one of the gateway devices.

As described above, this causes the gateway device itself to take any necessary action; however, if the failure caused the access network to become partitioned, it may also be necessary to propagate the topology change notification to the rest of the access network, that is, the portion connected to the other gateway device. This can be achieved by ensuring there is connectivity between the gateway devices, so that each gateway device can propagate any topology change notifications it receives from the access network to the other device. When a gateway device receives a BPDU from the other gateway device that indicates a topology change, it signals this in the static BPDUs (that it is sending towards the access network).

Topology Change Propagation is only necessary when these two conditions are met:

- The access network contains three or more access devices. If there are fewer than three devices, then any possible failure must be detected by all the devices.
- The access devices send traffic to each other, and not just to or from the core or aggregation network. If all the traffic is to or from the core or aggregation network, then all the access devices must either already be sending traffic in the right direction, or will learn about the topology change from the access device that originates it.

Preempt Delay

One of the assumptions underpinning access gateway is that the gateway devices are always available to provide connectivity to the core or aggregation network. However, there is one situation where this assumption may not hold, which is at bringup time. At bringup, it may be the case that the access facing interface is available before all of the necessary signaling and convergence has completed that means traffic can successfully be forwarded into the core or aggregation network. Since access gateway starts sending BPDUs as soon as the interface comes up, this could result in the access devices sending traffic to the gateway device before it is ready to receive it. To avoid this problem, the preempt delay feature is used.

The preempt delay feature causes access gateway to send out inferior BPDUs for some period of time after the interface comes up, before reverting to the normal values. These inferior BPDUs can be configured such that the access network directs all traffic to the other gateway device, unless the other gateway device is also down. If the other gateway device is unavailable, it is desirable for the traffic to be sent to this device, even if it is only partially available, rather than being dropped completely. For this reason, inferior BPDUs are sent during the preempt delay time, rather than sending no BPDUs at all.
Supported Access Gateway Protocols

Access Gateway is supported on Cisco ASR 9000 Series Routers when the following protocols are used in the access network.

Table 3 Protocols

<table>
<thead>
<tr>
<th>Access Network Protocol</th>
<th>Access Gateway Variant</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSTP</td>
<td>MST Access Gateway (MSTAG)</td>
</tr>
<tr>
<td>REP</td>
<td>REP Access gateway (REPAG)</td>
</tr>
<tr>
<td>PVST+</td>
<td>PVST+ Access Gateway (PVSTAG)</td>
</tr>
<tr>
<td>PVRST</td>
<td>PVRST Access Gateway (PVRSTAG)</td>
</tr>
</tbody>
</table>

1. REP Access Gateway is supported when the access device interfaces that connect to the gateway devices are configured with REP MSTP Compatibility mode.
2. Topology Change Propagation is not supported for PVSTAG.
3. Topology Change Propagation is not supported for PVRSTAG.

Restrictions

Multiple VLAN Registration Protocol

The Multiple VLAN Registration Protocol is defined in IEEE 802.1ak and is used in MSTP based networks to optimize the propagation of multicast and broadcast frames.

By default, multicast and broadcast frames are propagated to every point in the network, according to the spanning tree, and hence to every edge (host) device that is attached to the network. However, for a given VLAN, it may be the case that only certain hosts are interested in receiving the traffic for that VLAN. Furthermore, it may be the case that a given network device, or even an entire segment of the network, has no attached hosts that are interested in receiving traffic for that VLAN. In this case, an optimization is possible by avoiding propagating traffic for that VLAN to those devices that have no stake in it. MVRP provides the necessary protocol signaling that allows each host and device to indicate to its attached peers which VLANs it is interested in.

MVRP-enabled devices can operate in two modes:

- Static mode—In this mode, the device initiates MVRP messages declaring interest in a statically configured set of VLANs. Note that the protocol is still dynamic with respect to the MSTP topology; it is the set of VLANs that is static.

- Dynamic mode—In this mode, the device processes MVRP messages received on different ports, and aggregates them dynamically to determine the set of VLANs it is interested in. It sends MVRP messages declaring interest in this set. In dynamic mode, the device also uses the received MVRP messages to prune the traffic sent out of each port so that traffic is only sent for the VLANs that the attached device has indicated it is interested in.

Cisco ASR 9000 Series Routers support operating in static mode. This is known as MVRP-lite.
How to Implement Multiple Spanning Tree Protocol

This section contains these procedures:

- Configuring MSTP
- Configuring MSTAR or REPAG
- Configuring PVSTAG or PVRSTAG
- Configuring MVRP-lite

Configuring MSTP

This section describes the procedure for configuring MSTP:

- Enabling MSTP
- Configuring MSTP parameters
- Verifying MSTP

Note

This section does not describe how to configure data switching. Refer to the Implementing Multipoint Layer 2 Services module for more information.

Enabling MSTP

By default, STP is disabled on all interfaces. MSTP should be explicitly enabled by configuration on each physical or Ethernet Bundle interface. When MSTP is configured on an interface, all the subinterfaces of that interface are automatically MSTP-enabled.

Configuring MSTP parameters

The MSTP Standard defines a number of configurable parameters. The global parameters are:

- Region Name and Revision
- Bringup Delay
- Forward Delay
- Max Age or Hops
- Transmit Hold Count
- Provider Bridge mode
- Flush Containment
- VLAN IDs (per spanning-tree instance)
- Bridge Priority (per spanning-tree instance)

The per-interface parameters are:

- External port path cost
- Hello Time
- Link Type
Implementing Multiple Spanning Tree Protocol

- Port Fast and BPDU Guard
- Root Guard and Topology Change Guard
- Port priority (per spanning-tree instance)
- Internal port path cost (per spanning-tree instance)

Per-interface configuration takes place in an interface submode within the MST configuration submode.

**Note**
The configuration steps listed in the following sections show all of the configurable parameters. However, in general, most of these can be retained with the default value.
SUMMARY STEPS

1. configure
2. spanning-tree mst protocol instance identifier
3. bringup delay for interval [minutes | seconds]
4. flush containment disable
5. name name
6. revision revision-number
7. forward-delay seconds
8. maximum [age seconds | hops hops]
9. transmit hold-count count
10. provider-bridge
11. instance id
12. priority priority
13. vlan-id vlan-range [,vlan-range][,vlan-range][,vlan-range]
14. interface {Bundle-Ether | GigabitEthernet | TenGigE | FastEthernet} instance
15. instance id port-priority priority
16. instance id cost cost
17. external-cost cost
18. link-type {point-to-point | multipoint}
19. hello-time seconds
20. portfast [bpdu-guard]
21. guard root
22. guard topology-change
23. end
   or
   commit
## DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td><strong>configure</strong></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><strong>RP/0/RSP0/CPU0:router(config)# config Thu Jun 4 07:50:02.660 PST RP/0/RSP0/CPU0:router(config)#</strong></td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td><strong>spanning-tree mst protocol instance identifier</strong></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><strong>RP/0/RSP0/CPU0:router(config)# spanning-tree mst a RP/0/RSP0/CPU0:router(config-mstp)#</strong></td>
</tr>
<tr>
<td><strong>Step 3</strong></td>
<td>**bringup delay for interval (minutes</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><strong>RP/0/RSP0/CPU0:router(config-mstp)# bringup delay for 10 minutes</strong></td>
</tr>
<tr>
<td><strong>Step 4</strong></td>
<td><strong>flush containment disable</strong></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><strong>RP/0/RSP0/CPU0:router(config-mstp)# flush containment disable</strong></td>
</tr>
<tr>
<td><strong>Step 5</strong></td>
<td><strong>name name</strong></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><strong>RP/0/RSP0/CPU0:router(config-mstp)# name m1</strong></td>
</tr>
<tr>
<td><strong>Step 6</strong></td>
<td><strong>revision revision-number</strong></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><strong>RP/0/RSP0/CPU0:router(config-mstp)# revision 10</strong></td>
</tr>
<tr>
<td><strong>Step 7</strong></td>
<td><strong>forward-delay seconds</strong></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><strong>RP/0/RSP0/CPU0:router(config-mstp)# forward-delay 20</strong></td>
</tr>
<tr>
<td><strong>Step 8</strong></td>
<td>**maximum (age seconds</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><strong>RP/0/RSP0/CPU0:router(config-mstp)# max age 40 RP/0/RSP0/CPU0:router(config-mstp)# max hops 30</strong></td>
</tr>
<tr>
<td>Step</td>
<td>Command or Action</td>
</tr>
<tr>
<td>------</td>
<td>------------------</td>
</tr>
<tr>
<td>9</td>
<td>transmit hold-count count</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-mstp)# transmit hold-count 8</td>
</tr>
<tr>
<td>10</td>
<td>provider-bridge</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-mstp)# provider-bridge</td>
</tr>
<tr>
<td>11</td>
<td>instance id</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-mstp)# instance 101</td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-mstp-inst)#</td>
</tr>
<tr>
<td>12</td>
<td>priority priority</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-mstp-inst)# priority 8192</td>
</tr>
<tr>
<td>13</td>
<td>vlan-id vlan-range [,vlan-range] [,vlan-range] [,vlan-range]</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-mstp-inst)# vlan-id 2-1005</td>
</tr>
<tr>
<td>14</td>
<td>interface {Bundle-Ether</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-mstp)# interface FastEthernet 0/0/0/1</td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-mstp-if)#</td>
</tr>
<tr>
<td>15</td>
<td>instance id port-priority priority</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-mstp-if)# instance 101 port-priority 160</td>
</tr>
<tr>
<td>16</td>
<td>instance id cost cost</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-mstp-if)# instance 101 cost 10000</td>
</tr>
<tr>
<td>Command or Action</td>
<td>Purpose</td>
</tr>
<tr>
<td>------------------</td>
<td>---------</td>
</tr>
<tr>
<td><strong>Step 17</strong></td>
<td>Sets the external path cost on the current port. Allowed values for port cost are from 1 through 20000000.</td>
</tr>
<tr>
<td>external-cost</td>
<td></td>
</tr>
<tr>
<td>cost</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-mstp-if)#</td>
<td></td>
</tr>
<tr>
<td>external-cost 10000</td>
<td></td>
</tr>
<tr>
<td><strong>Step 18</strong></td>
<td>Sets the link type of the port to point-to-point or multipoint.</td>
</tr>
<tr>
<td>link-type</td>
<td></td>
</tr>
<tr>
<td>(point-to-point</td>
<td>multipoint)</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-mstp-if)#</td>
<td></td>
</tr>
<tr>
<td>link-type point-to-point</td>
<td></td>
</tr>
<tr>
<td><strong>Step 19</strong></td>
<td>Sets the port hello time in seconds. Allowed values are 1 and 2.</td>
</tr>
<tr>
<td>hello-time</td>
<td></td>
</tr>
<tr>
<td>seconds</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-mstp-if)#</td>
<td></td>
</tr>
<tr>
<td>hello-time 1</td>
<td></td>
</tr>
<tr>
<td><strong>Step 20</strong></td>
<td>Enables PortFast on the port, and optionally enables BPDU guard.</td>
</tr>
<tr>
<td>portfast</td>
<td></td>
</tr>
<tr>
<td>[bpdu-guard]</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-mstp-if)#</td>
<td></td>
</tr>
<tr>
<td>portfast</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-mstp-if)#</td>
<td></td>
</tr>
<tr>
<td>portfast bpduguard</td>
<td></td>
</tr>
<tr>
<td><strong>Step 21</strong></td>
<td>Enables RootGuard on the port.</td>
</tr>
<tr>
<td>guard root</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-mstp-if)#</td>
<td></td>
</tr>
<tr>
<td>guard root</td>
<td></td>
</tr>
<tr>
<td>Command or Action</td>
<td>Purpose</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Step 22</strong> guard topology-change</td>
<td>Enables TopologyChangeGuard on the port.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-mstp-if)#</td>
<td></td>
</tr>
<tr>
<td>guard topology-change</td>
<td></td>
</tr>
</tbody>
</table>

**Note** Repeat steps 14 to 22 for each interface.

| **Step 23** end or commit                  | Saves configuration changes.                                           |
| **Example:**                             |                                                                         |
| RP/0/RSP0/CPU0:router(config-mstp-if)#    |                                                                         |
| end or commit                            |                                                                         |
|                                           |                                                                         |

- When you issue the **end** command, the system prompts you to commit changes:
  Uncommitted changes found, commit them before exiting(yes/no/cancel)?
  [cancel]:
  - Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
  - Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
  - Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
- Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

## Verifying MSTP

These show commands allow you to verify the operation of MSTP:

- `show spanning-tree mst mst-name`
- `show spanning-tree mst mst-name interface interface-name`
- `show spanning-tree mst mst-name errors`
- `show spanning-tree mst mst-name configuration`
- `show spanning-tree mst mst-name bpdu interface interface-name`
- `show spanning-tree mst mst-name topology-change flushes`
Configuring MSTAG or REPAG

This section describes the procedures for configuring MSTAG:

- Configuring an untagged subinterface
- Enabling MSTAG
- Configuring MSTAG parameters
- Configuring MSTAG Topology Change Propagation
- Verifying MSTAG

Note

The procedures for configuring REPAG are identical.

This section does not describe how to configure data switching. Refer to the Implementing Multipoint Layer 2 Services module for more information.

Configuring an untagged subinterface

In order to enable MSTAG on a physical or Bundle Ethernet interface, an L2 subinterface must first be configured which matches untagged packets, using the `encapsulation untagged` command. Refer to The Cisco ASR 9000 Series Routers Carrier Ethernet Model module for more information about configuring L2 subinterfaces.

Enabling MSTAG

MSTAG is enabled on a physical or Bundle Ethernet interface by explicitly configuring it on the corresponding untagged subinterface. When MSTAG is configured on the untagged subinterface, it is automatically enabled on the physical or Bundle Ethernet interface and on all other subinterfaces on that physical or Bundle Ethernet subinterface.

Configuring MSTAG parameters

MSTAG parameters are configured separately on each interface, and MSTAG runs completely independently on each interface. There is no interaction between the MSTAG parameters on different interfaces (unless they are connected to the same access network).

These parameters are configurable for each interface:

- Region Name and Revision
- Bridge ID
- Port ID
- External port path cost
- Max Age
- Provide Bridge mode
- Hello Time
The following MSTAG parameters are configurable for each interface, for each spanning tree instance:

- VLAN IDs
- Root Bridge Priority and ID
- Bridge Priority
- Port Priority
- Internal Port Path Cost

To ensure consistent operation across the access network, these guidelines should be used when configuring:

- Both gateway devices should be configured with a Root Bridge Priority and ID (for each spanning tree instance) that is better (lower) than the Bridge Priority and Bridge ID of any device in the access network. It is recommended to set the Root Bridge Priority and ID to 0 on the gateway devices.

**Note**

To avoid an STP dispute being detected by the access devices, the same root priority and ID should be configured on both gateway devices.

- Both gateway devices should be configured with a Port Path Cost of 0.
- For each spanning tree instance, one gateway device should be configured with the bridge priority and ID that is higher than the root bridge priority and ID, but lower than the bridge priority and ID of any other device in the network (including the other gateway device). It is recommended to set the bridge priority to 0.
- For each spanning tree instance, the second gateway device should be configured with a bridge priority and ID that is higher than the root bridge priority and ID and the first gateway device bridge priority and ID, but lower than the bridge priority and ID of any device in the access network. It is recommended to set the bridge priority to 4096 (this is the lowest allowable value greater than 0).
- All of the access devices should be configured with a higher bridge priority than the gateway devices. It is recommended to use values of 8192 or higher.
- For each spanning tree instance, the port path cost and other parameters may be configured on the access devices so as to ensure the desired port is put into the blocked state when all links are up.

**Caution**

There are no checks on MSTAG configuration—misconfiguration may result in incorrect operation of the MSTP protocol in the access devices (for example, an STP dispute being detected).

The guidelines above are illustrated in Figure 37.

**Note**

These guidelines do not apply to REPAG, as in that case the access devices ignore the information received from the gateway devices apart from when a topology change is signalled.
Figure 37  MSTAG Guidelines

The configuration steps listed in the following sections show all of the configurable parameters. However, in general, most of these can be retained with the default values.

SUMMARY STEPS

1. `configure`
2. `spanning-tree mstag protocol instance identifier`
3. `preempt delay for interval {seconds | minutes | hours}`
4. `interface {Bundle-Ether | GigabitEthernet | TenGigE | FastEthernet} instance.subinterface`
5. `name name`
6. `revision revision-number`
7. `max age seconds`
8. `provider-bridge`
9. `bridge-id id`
10. `port-id id`
11. `external-cost cost`
12. `hello-time seconds`
13. `instance id`
14. `vlan-id vlan-range [,vlan-range][,vlan-range][,vlan-range]`
15. `priority priority`
16. `port-priority priority`
17. `cost cost`
18. `root-bridge id`
19. `root-priority priority`

20. `end`
   or
   `commit`

**DETAILED STEPS**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
</table>
| **Step 1**       | `configure`
|                   | **Purpose**: Enters global configuration mode. |
| **Example:**     | RP/0/RSP0/CPU0:router# `configure`
|                   | Thu Jun 4 07:50:02.660 PST |
|                   | RP/0/RSP0/CPU0:router(config)# |
| **Step 2**       | `spanning-tree mstag protocol instance identifier`
|                   | **Purpose**: Enters the MSTAG configuration submode. |
| **Example:**     | RP/0/RSP0/CPU0:router(config)#
|                   | `spanning-tree mstag a` |
|                   | RP/0/RSP0/CPU0:router(config-mstag)# |
| **Step 3**       | `preempt delay for interval (seconds | minutes | hours)`
|                   | **Purpose**: Specifies the delay period during which startup BPDUs should be sent, before preempting. |
| **Example:**     | RP/0/RSP0/CPU0:router(config-mstag)#
|                   | `preempt delay for 10 seconds` |
| **Step 4**       | `interface (Bundle-Ether | GigabitEthernet | TenGigE | FastEthernet) instance.subinterface`
|                   | **Purpose**: Enters the MSTAG interface configuration submode, and enables MSTAG for the specified port. |
| **Example:**     | RP/0/RSP0/CPU0:router(config-mstag)#
|                   | `interface GigabitEthernet0/2/0/30.1` |
|                   | RP/0/RSP0/CPU0:router(config-mstag-if)# |
| **Step 5**       | `name name`
|                   | **Purpose**: Sets the name of the MSTP region. The default value is the MAC address of the switch, formatted as a text string using the hexadecimal representation specified in IEEE Standard 802. |
| **Example:**     | RP/0/RSP0/CPU0:router(config-mstag-if)#
|                   | `name leo` |
| **Step 6**       | `revision revision-number`
|                   | **Purpose**: Sets the revision level of the MSTP region. Allowed values are from 0 through 65535. |
| **Example:**     | RP/0/RSP0/CPU0:router(config-mstag-if)#
<p>|                   | <code>revision 1</code> |</p>
<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 7</td>
<td>max age seconds</td>
<td>Sets the maximum age performance parameters for the bridge. Allowed values for the maximum age time for the bridge in seconds are from 6 through 40.</td>
</tr>
<tr>
<td></td>
<td>Example: RP/0/RSP0/CPU0:router(config-mstag-if)# max age 20</td>
<td></td>
</tr>
<tr>
<td>Step 8</td>
<td>provider-bridge</td>
<td>Places the current instance of the protocol in 802.1ad mode.</td>
</tr>
<tr>
<td></td>
<td>Example: RP/0/RSP0/CPU0:router(config-mstag-if)# provider-bridge</td>
<td></td>
</tr>
<tr>
<td>Step 9</td>
<td>bridge-id id</td>
<td>Sets the bridge ID for the current switch.</td>
</tr>
<tr>
<td></td>
<td>Example: RP/0/RSP0/CPU0:router(config-mstag-if)# bridge-id 001c.0000.0011</td>
<td></td>
</tr>
<tr>
<td>Step 10</td>
<td>port-id id</td>
<td>Sets the port ID for the current switch.</td>
</tr>
<tr>
<td></td>
<td>Example: RP/0/RSP0/CPU0:router(config-mstag-if)# port-id 111</td>
<td></td>
</tr>
<tr>
<td>Step 11</td>
<td>external-cost cost</td>
<td>Sets the external path cost on the current port. Allowed values for port cost are from 1 through 200000000.</td>
</tr>
<tr>
<td></td>
<td>Example: RP/0/RSP0/CPU0:router(config-mstag-if)# external-cost 10000</td>
<td></td>
</tr>
<tr>
<td>Step 12</td>
<td>hello-time seconds</td>
<td>Sets the port hello time in seconds. Allowed values are from 1 through 2.</td>
</tr>
<tr>
<td></td>
<td>Example: RP/0/RSP0/CPU0:router(config-mstag-if)# hello-time 1</td>
<td></td>
</tr>
<tr>
<td>Step 13</td>
<td>instance id</td>
<td>Enters the MSTI configuration submode. Allowed values for the MSTI ID are from 0 through 4094.</td>
</tr>
<tr>
<td></td>
<td>Example: RP/0/RSP0/CPU0:router(config-mstag-if)# instance 1</td>
<td></td>
</tr>
<tr>
<td>Step 14</td>
<td>vlan-id vlan-range</td>
<td>Associates a set of VLAN IDs with the current MSTI. List of VLAN ranges in the form a-b, c, d, e-f, g, and so on.</td>
</tr>
<tr>
<td></td>
<td>[vlan-range],[vlan-range]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>,vlan-range</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Example: RP/0/RSP0/CPU0:router(config-mstag-if-ins t)# vlan-id 2-1005</td>
<td></td>
</tr>
<tr>
<td>Step 15</td>
<td>priority priority</td>
<td>Sets the bridge priority for the current MSTI. Allowed values are from 0 through 61440 in multiples of 4096.</td>
</tr>
<tr>
<td></td>
<td>Example: RP/0/RSP0/CPU0:router(config-mstag-if-ins t)# priority 4096</td>
<td></td>
</tr>
<tr>
<td>Command or Action</td>
<td>Purpose</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>---------</td>
<td></td>
</tr>
<tr>
<td><strong>Step 16</strong></td>
<td>Sets the port priority performance parameter for the MSTI. Allowed values for port priority are from 0 through 240 in multiples of 16.</td>
<td></td>
</tr>
<tr>
<td><code>port-priority priority</code></td>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-mstag-if-ins t)# <code>port-priority 160</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 17</strong></td>
<td>Sets the internal path cost for a given instance on the current port. Allowed values for port cost are from 1 through 20000000.</td>
<td></td>
</tr>
<tr>
<td><code>cost cost</code></td>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-mstag-if-ins t)# <code>cost 10000</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 18</strong></td>
<td>Sets the root bridge ID for the BPDUs sent from the current port.</td>
<td></td>
</tr>
<tr>
<td><code>root-bridge id</code></td>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-mstag-if-ins t)# <code>root-id 001c.0000.0011</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 19</strong></td>
<td>Sets the root bridge priority for the BPDUs sent from this port. <strong>Note</strong> Repeat steps 4 to 19 to configure each interface, and repeat steps 13 to 19 to configure each MSTI for each interface.</td>
<td></td>
</tr>
<tr>
<td><code>root-priority priority</code></td>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-mstag-if-ins t)# <code>root-priority 4096</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 20</strong></td>
<td>Saves configuration changes. Use the <code>commit</code> command to save the configuration changes to the running configuration file and remain within the configuration session. When you issue the <code>end</code> command, the system prompts you to commit changes: Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:</td>
<td></td>
</tr>
<tr>
<td><code>end</code> or <code>commit</code></td>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-mstag-if-ins t)# <code>end</code> or RP/0/RSP0/CPU0:router(config-mstag-if-ins t)# <code>commit</code></td>
<td></td>
</tr>
</tbody>
</table>
Configuring MSTAG Topology Change Propagation

MSTAG Topology Change Propagation is configured simply by configuring connectivity between the MSTAG-enabled interfaces on the two gateway devices:

1. Configure MSTAG as described above. Take note of the untagged subinterface that is used.
2. Configure connectivity between the gateway devices. This may be via an MPLS Pseudowire, or may be a VLAN subinterface if there is a direct physical link.
3. Configure a point-to-point (P2P) cross-connect on each gateway device that contains the untagged subinterface and the link (PW or subinterface) to the other gateway device.

Once the untagged subinterface that is configured for MSTAG is added to the P2P cross-connect, MSTAG Topology Change Propagation is automatically enabled. MSTAG forwards BDPUs via the cross-connect to the other gateway device, so as to signal when a topology change has been detected.

For more information on configuring MPLS pseudowire or P2P cross-connects, refer to the Implementing Point to Point Layer 2 Services module.

Verifying MSTAG

These show commands allow you to verify the operation of MSTAG:

- `show spanning-tree mstag mst-name`
- `show spanning-tree mstag mst-name bpdu interface interface-name`
- `show spanning-tree mstag mst-name topology-change flushes`

Analogous commands are available for REPAG.

Configuring PVSTAG or PVRSTAG

This section describes the procedures for configuring PVSTAG:

- **Enabling PVSTAG**
- **Configuring PVSTAG parameters**
- **Configuring Subinterfaces**
- **Verifying PVSTAG**

The procedures for configuring PVRSTAG are identical.

Note

This section does not describe how to configure data switching. Refer to the Implementing Multipoint Layer 2 Services module for more information.

Enabling PVSTAG

PVSTAG is enabled for a particular VLAN, on a physical interface, by explicit configuration of that physical interface and VLAN for PVSTAG.
Configuring PVSTAG parameters

The configurable PVSTAG parameters for each interface on each VLAN are:

- Root Priority and ID
- Root cost
- Bridge Priority and ID
- Port priority and ID
- Max Age
- Hello Time

For correct operation, these guidelines must be followed when configuring PVSTAG.

- Both gateway devices should be configured with a root bridge priority and ID that is better (lower) than the bridge priority and Bridge ID of any device in the access network. It is recommended that you set the root bridge priority and ID to 0 on the gateway devices.
- Both gateway devices should be configured with a root cost of 0.
- One gateway device should be configured with a bridge priority and ID that is higher than the root bridge priority and ID, but lower than the bridge priority and ID of any other device in the network (including the other gateway device). It is recommended that you set the bridge priority to 0.
- The second gateway device should be configured with a bridge priority and ID that is higher than the root bridge priority and ID and the first gateway device bridge priority and ID, but lower than the bridge priority and ID of any device in the access network. It is recommended that you set the bridge priority to 1 for PVSTAG or 4096 for PVRSTAG. (For PVRSTAG, this is the lowest allowable value greater than 0.)
- All access devices must be configured with a higher bridge priority than the gateway devices. It is recommended that you use values of 2 or higher for PVSTAG, or 8192 or higher for PVRSTAG.
- For each spanning tree instance, the port path cost and other parameters may be configured on the access devices, so as to ensure the desired port is placed into the blocked state when all links are up.

Caution

There are no checks on PVSTAG configuration—misconfiguration may result in incorrect operation of the PVST protocol in the access devices (for example, an STP dispute being detected).

These guidelines are illustrated in Figure 38.
PVSTAG Topology Restrictions

These restrictions are applicable to PVSTAG topology:

- Only a single access device can be attached to the gateway devices.
- Topology change notifications on a single VLAN affect all VLANs and bridge domains on that physical interface.

Note

The configuration steps listed in the following sections show all of the configurable parameters. However, in general, most of these can be retained with the default values.

SUMMARY STEPS

1. configure
2. spanning-tree pvstag protocol instance identifier
3. preempt delay for interval {seconds | minutes | hours}
4. interface interface-instance.subinterface
5. vlan vlan-id
6. root-priority priority
7. root-id id
8. root-cost cost
9. priority priority
10. bridge-id id
11. port-priority priority
12. port-id \textit{id}
13. hello-time \textit{seconds}
14. max age \textit{seconds}
15. end
   or
   commit
## DETAILED STEPS

<table>
<thead>
<tr>
<th>Step 1</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>configure</strong></td>
<td>Enters global configuration mode.</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router# configure</td>
<td>Thu Jun 4 07:50:02.660 PST</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config)#</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 2</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>spanning-tree pvstag</strong> protocol instance identifier</td>
<td>Enters the PVSTAG configuration submode.</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config)# spanning-tree pvstag a</td>
<td>RP/0/RSP0/CPU0:router(config-pvstag)#</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 3</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>**preempt delay for interval (seconds</td>
<td>minutes</td>
<td>hours)**</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-pvstag)# preempt delay for 10 seconds</td>
<td>RP/0/RSP0/CPU0:router(config-pvstag-if)#</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 4</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>interface</strong> interface-instance.subinterface</td>
<td>Enters the PVSTAG interface configuration submode, and enables PVSTAG for the specified port.</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-pvstag)# interface GigabitEthernet0/2/0/30.1</td>
<td>RP/0/RSP0/CPU0:router(config-pvstag-if)#</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 5</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>vlan vlan-id</strong></td>
<td>Enables and configures a VLAN on this interface.</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-pvstag-if)# vlan 200</td>
<td>RP/0/RSP0/CPU0:router(config-pvstag-if-vlan)#</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 6</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>root-priority priority</strong></td>
<td>Sets the root bridge priority for the BPDUs sent from this port.</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-pvstag-if-vlan)# root-priority 4096</td>
<td>RP/0/RSP0/CPU0:router(config-pvstag-if-vlan)#</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 7</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>root-id id</strong></td>
<td>Sets the identifier of the root bridge for BPDUs sent from a port.</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-pvstag-if-vlan)# root-id 0000.0000.0000</td>
<td>RP/0/RSP0/CPU0:router(config-pvstag-if-vlan)#</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 8</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>root-cost cost</strong></td>
<td>Set the root path cost to sent in BPDUs from this interface.</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-pvstag-if-vlan)# root-cost 10000</td>
<td>RP/0/RSP0/CPU0:router(config-pvstag-if-vlan)#</td>
<td></td>
</tr>
</tbody>
</table>
### Command or Action

<table>
<thead>
<tr>
<th>Step 9</th>
<th>priority priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-pvstag-if-vlan)# priority 4096</td>
</tr>
<tr>
<td>Purpose</td>
<td>Sets the bridge priority for the current MSTI. For PVSTAG, allowed values are from 0 through 65535; for PVRSTAG, the allowed values are from 0 through 61440 in multiples of 4096.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 10</th>
<th>bridge-id id</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-pvstag-if-vlan)# bridge-id 001c.0000.0011</td>
</tr>
<tr>
<td>Purpose</td>
<td>Sets the bridge ID for the current switch.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 11</th>
<th>port-priority priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-pvstag-if-vlan)# port-priority 160</td>
</tr>
<tr>
<td>Purpose</td>
<td>Sets the port priority performance parameter for the MSTI. For PVSTAG, allowed values for port priority are from 0 through 255; for PVRSTAG, the allowed values are from 0 through 240 in multiples of 16.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 12</th>
<th>port-id id</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-pvstag-if-vlan)# port-id 111</td>
</tr>
<tr>
<td>Purpose</td>
<td>Sets the port ID for the current switch.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 13</th>
<th>hello-time seconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-pvstag-if-vlan)# hello-time 1</td>
</tr>
<tr>
<td>Purpose</td>
<td>Sets the port hello time in seconds. Allowed values are from 1 through 2.</td>
</tr>
</tbody>
</table>
Implementing Multiple Spanning Tree Protocol

How to Implement Multiple Spanning Tree Protocol

LSC-339

Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Configuration Guide

OL-23107-02

Configuring Subinterfaces

For each VLAN that is enabled for PVSTAG on an interface, a corresponding subinterface that matches traffic for that VLAN must be configured. This is used both for data switching and for PVST BPDUs. Follow these guidelines when configuring subinterfaces:

- VLAN 1 is treated as the native VLAN in PVST. Therefore, for VLAN 1, a subinterface that matches untagged packets (encapsulation untagged) must be configured. It may also be necessary to configure a subinterface that matches packets tagged explicitly with VLAN 1 (encapsulation dot1q 1).

- Only dot1q packets are allowed in PVST; Q-in-Q and dot1ad packets are not supported by the protocol, and therefore subinterfaces configured with these encapsulation will not work correctly with PVSTAG.

- Subinterfaces that match a range of VLANs are supported by PVSTAG; it is not necessary to configure a separate subinterface for each VLAN, unless it is desirable for provisioning the data switching.

- PVSTAG does not support:
  - Physical interfaces configured in L2 mode
  - Subinterface configured with a default encapsulation (encapsulation default)
  - Subinterfaces configured to match any VLAN (encapsulation dot1q any)
For more information about configuring L2 subinterfaces, refer to the Implementing Point to Point Layer 2 Services module.

Verifying PVSTAG

These `show` commands allow you to verify the operation of PVSTAG or PVRSTAG:

- `show spanning-tree pvstag mst-name`
- `show spanning-tree pvstag mst-name`

In particular, these commands display the subinterface that is being used for each VLAN.

Configuring MVRP-lite

This section describes the procedure for configuring MVRP-lite:

- **Enabling MVRP-lite**
- **Configuring MVRP-lite parameters**
- **Verifying MVRP-lite**

Enabling MVRP-lite

When MVRP-lite is configured, it is automatically enabled on all interfaces where MSTP is enabled. MSTP must be configured before MVRP can be enabled. For more information on configuring MSTP, see Configuring MSTP, page 320.

Configuring MVRP-lite parameters

The configurable MVRP-lite parameters are:

- Periodic Transmission
- Join Time
- Leave Time
- Leave-all Time

Summary Steps

1. `configure`
2. `spanning-tree mst protocol instance name`
3. `mvrp static`
4. `periodic transmit [interval seconds]`
5. `join-time milliseconds`
6. `leave-time seconds`
7. `leaveall-time seconds`
8. `end`
   or
   `commit`
### Detailed Steps

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td>Thu Jun 4 07:50:02.660 PST</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config)#</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> spanning-tree mst protocol</td>
<td>Enters the MSTP configuration submode.</td>
</tr>
<tr>
<td>instance identifier</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config)#</td>
<td></td>
</tr>
<tr>
<td>spanning-tree mst a</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-mstp)#</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> mvrp static</td>
<td>Configures MVRP to run over this MSTP protocol instance.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-mstp)#</td>
<td>mvrp</td>
</tr>
<tr>
<td>static</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> periodic transmit [interval</td>
<td>Sends periodic Multiple VLAN Registration Protocol Data Unit (MVRPDU) on all active ports.</td>
</tr>
<tr>
<td>seconds]</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-mvrp)#</td>
<td>periodic transmit</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Step 5</strong> join-time milliseconds</td>
<td>Sets the join time for all active ports.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-mvrp)#</td>
<td>join-time</td>
</tr>
<tr>
<td>hello-time 1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Step 6</strong> leave-time seconds</td>
<td>Sets the leave time for all active ports.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-mvrp)#</td>
<td>leave-time 20</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Implementing Multiple Spanning Tree Protocol

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 7</strong> leaveall-time <em>seconds</em></td>
<td>Sets the leave all time for all active ports.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-mvrp)# leaveall-time 20</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 8</th>
<th>end or commit</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-mvrp)# end or commit</td>
<td>Saves configuration changes.</td>
</tr>
<tr>
<td></td>
<td>When you issue the <strong>end</strong> command, the system prompts you to commit changes:</td>
</tr>
<tr>
<td></td>
<td>Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:</td>
</tr>
<tr>
<td></td>
<td>– Entering <strong>yes</strong> saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.</td>
</tr>
<tr>
<td></td>
<td>– Entering <strong>no</strong> exits the configuration session and returns the router to EXEC mode without committing the configuration changes.</td>
</tr>
<tr>
<td></td>
<td>– Entering <strong>cancel</strong> leaves the router in the current configuration session without exiting or committing the configuration changes.</td>
</tr>
<tr>
<td></td>
<td>– Use the <strong>commit</strong> command to save the configuration changes to the running configuration file and remain within the configuration session.</td>
</tr>
</tbody>
</table>

### Verifying MVRP-lite

These show commands allow you to verify the operation of MVRP-lite:

- `show ethernet mvrp mad`
- `show ethernet mvrp status`
- `show ethernet mvrp statistics`
Configuration Examples for Implementing MSTP

This section provides configuration examples for the following:

- Configuring MSTP: Examples
- Configuring MSTAG: Examples
- Configuring PVSTAG: Examples
- Configuring MVRP-Lite: Examples

Configuring MSTP: Examples

This example shows MSTP configuration for a single spanning-tree instance with MSTP enabled on a single interface:

```
config
spanning-tree mst example
  name m1
  revision 10
  forward-delay 20
  maximum hops 40
  maximum age 40
  transmit hold-count 8
  provider-bridge
  bringup delay for 60 seconds
  flush containment disable
  instance 101
    vlans-id 101-110
    priority 8192
    !
  interface GigabitEthernet0/0/0/0
    hello-time 1
    external-cost 10000
    link-type point-to-point
    portfast
    guard root
    guard topology-change
    instance 101 cost 10000
    instance 101 port-priority 160
    !
```

This example shows the output from the `show spanning-tree mst` command, which produces an overview of the spanning tree protocol state:

```
# show spanning-tree mst example
Role:  ROOT=Root, DSGN=Designated, ALT=Alternate, BKP=Backup, MSTR=Master
State: FWD=Forwarding, LRN=Learning, BLK=Blocked, DLY=Bringup Delayed

Operating in dot1q mode

MSTI 0 (CIST):

VLANS Mapped: 1-9,11-4094

CIST Root  Priority  4096
          Address   6262.6262.6262
This bridge is the CIST root
```

This example shows MSTP configuration for a single spanning-tree instance with MSTP enabled on a single interface.
In the `show spanning-tree mst` output, the first line indicates whether MSTP is operating in dot1q or the Provider Bridge mode, and this information is followed by details for each MSTI.

For each MSTI, the following information is displayed:

- The list of VLANs for the MSTI.
- For the CIST, the priority and bridge ID of the CIST root, and the external path cost to reach the CIST root. The output also indicates if this bridge is the CIST root.
- The priority and bridge ID of the root bridge for this MSTI, and the internal path cost to reach the root. The output also indicates if this bridge is the root for the MSTI.
- The max age and forward delay times received from the root bridge for the MSTI.
- The priority and bridge ID of this bridge, for this MSTI.
- The maximum age, forward delay, max hops and transmit hold-count for this bridge (which is the same for every MSTI).

- A list of MSTP-enabled interfaces. For each interface, the following information is displayed:
  - The interface name
  - The port priority and port ID for this interface for this MSTI.
  - The port cost for this interface for this MSTI.
  - The current port role:
    - DSGN—Designated: This is the designated port on this LAN, for this MSTI
    - ROOT—Root: This is the root port for the bridge for this MSTI.
    - ALT—Alternate: This is an alternate port for this MSTI.
    - BKP—Backup: This is a backup port for this MSTI
    - MSTR—Master: This is a boundary port that is a root or alternate port for the CIST.
    The interface is down, or the bringup delay timer is running and no role has been assigned yet.
  - The current port state:
    - BLK—The port is blocked.
    - LRN—The port is learning.
    - FWD—The port is forwarding.
    - DLY—The bringup-delay timer is running.
  - If the port is a boundary port, and not CIST and the port is not designated, then only the BOUNDARY PORT is displayed and the remaining information is not displayed.
  - If the port is not up, or the bringup delay timer is running, no information is displayed for the remaining fields. Otherwise, the bridge priority and bridge ID of the designated bridge on the LAN that the interface connects to is displayed, followed by the port priority and port ID of the designated port on the LAN. If the port role is Designated, then the information for this bridge or port is displayed.

The following example shows the output from the `show spanning-tree mst` command, which produces more detailed information regarding interface state than the standard command as described above:

```
# show spanning-tree mst a interface GigabitEthernet0/1/2/1
GigabitEthernet0/1/2/1
Cost: 20000
link-type: point-to-point
hello-time 1
Portfast: no
BPDU Guard: no
Guard root: no
Guard topology change: no
BPDU:s sent 492, received 3
```
MST 3:
   Edge port:
      Boundary : internal
      Designated forwarding
   VLANs mapped to MST 3: 1-2, 4-2999, 4000-4094
   Port info port id 128.193 cost 200000
   Designated root address 0050.3e66.d000 priority 8193 cost 20004
   Designated bridge address 0002.172c.f400 priority 49152 port id 128.193
   Timers: message expires in 0 sec, forward delay 0, forward transitions 1
   Transitions to reach this state: 12

The output includes interface information about the interface which applies to all MSTIs:
- Cost
- link-type
- hello-time
- portfast (including whether BPDU guard is enabled)
- guard root
- guard topology change
- BPDU sent, received.

It also includes information specific to each MSTI:
- Port ID, priority, cost
- BPDU information from root (bridge ID, cost, and priority)
- BPDU information being sent on this port (Bridge ID, cost, priority)
- State transitions to reach this state.
- Topology changes to reach this state.
- Flush containment status for this MSTI.

This example shows the output of `show spanning-tree mst errors`, which produces information about interfaces that are configured for MSTP but where MSTP is not operational. Primarily this shows information about interfaces which do not exist:

```
# show spanning-tree mst a errors
Interface                Error
-------------------------------
GigabitEthernet1/2/3/4   Interface does not exist.
```

This example shows the output of `show spanning-tree mst configuration`, which displays the VLAN ID to MSTI mapping table. It also displays the configuration digest which is included in the transmitted BPDU—this must match the digest received from other bridges in the same MSTP region:

```
# show spanning-tree mst a configuration
Name           leo
Revision       2702
Config Digest  9D-14-5C-26-7D-BE-9F-B5-D8-93-44-18-E3-BA-08-CE
Instance       Vlans mapped
               -------
   0         1-9, 11-19, 21-29, 31-39, 41-4094
   1         10, 20, 30, 40
               -------
```
This example shows the output of `show spanning-tree mst bpdu interface`, which produces details on the BPDUs being output and received on a given local interface:

```
Note
Several received packets can be stored in case of MSTP operating on a shared LAN.
```

```
# show spanning-tree mst a bpdu interface GigabitEthernet0/1/2/2 direction transmit
MSTI 0 (CIST):
   Root ID: 0004.9b78.0800
   Path Cost: 83
   Bridge ID: 0004.9b78.0800
   Port ID: 12
   Hello Time: 2
   ...
```

This example shows the output of `show spanning-tree mst topology-change flushes`, which displays details about the topology changes that have occurred for each MSTI on each interface:

```
# show spanning-tree mst M topology-change flushes instance$
MSTI 1:
Interface     Last TC               Reason                           Count
------------  --------------------  -------------------------------- -----
Te0/0/0/1     04:16:05 Mar 16 2010  Role change: DSGN to ----           10
#
# show spanning-tree mst M topology-change flushes instance$
MSTI 0 (CIST):
Interface     Last TC               Reason                           Count
------------  --------------------  -------------------------------- -----
Te0/0/0/1     04:16:05 Mar 16 2010  Role change: DSGN to ----           10
#
```

**Configuring MSTAG: Examples**

This example shows MSTAG configuration for a single spanning-tree instance on a single interface:

```
config
interface GigabitEthernet0/0/0/0.1 l2transport
   encapsulation untagged
!
spanning-tree mstag example
   preempt delay for 60 seconds
   interface GigabitEthernet0/0/0/0.1
      name m1
      revision 10
      external-cost 0
      bridge-id 0.0.1
      port-id 1
      maximum age 40
      provider-bridge
      hello-time 1
      instance 101
      vlans-id 101-110
      root-priority 0
      root-id 0.0.0
      cost 0
```
Implementing Multiple Spanning Tree Protocol

Configuration Examples for Implementing MSTP

This example shows additional configuration for MSTAG Topology Change Propagation:

```
priority 0
port-priority 0
!
!
```

This example shows the output of `show spanning-tree mstag`:

```
# show spanning-tree mstag A
GigabitEthernet0/0/0/1
  Preempt delay is disabled.
  Name: 6161:6161:6161
  Revision: 0
  Max Age: 20
  Provider Bridge: no
  Bridge ID: 6161.6161.6161
  Port ID: 1
  External Cost: 0
  Hello Time: 2
  Active: no
  BPDUs sent: 0
  MSTI 0 (CIST):
    VLAN IDs: 1-9,32-39,41-4094
    Bridge Priority: 32768
    Port Priority: 128
    Cost: 0
    Root Bridge: 6161.6161.6161
    Root Priority: 32768
    Topology Changes: 123
  MSTI 2
    VLAN IDs: 10-31
    Bridge Priority: 32768
    Port Priority: 128
    Cost: 0
    Root Bridge: 6161.6161.6161
    Root Priority: 32768
    Topology Changes: 123
```

This example shows the output of `show spanning-tree mstag bpdu interface`, which produces details on the BPDUs being output and received on a given local interface:

```
RP/0/RSP0/CPU0:router#show spanning-tree mstag foo bpdu interface GigabitEthernet 0/0/0/0
Transmitted:
  MSTI 0 (CIST):
    ProtocolIdentifier: 0
    ProtocolVersionIdentifier: 3
    BPDUType: 2
    CISTFlags: Top Change Ack 0
    Agreement 1
    Forwarding 1
    Learning 1
    Role 3
```
This example shows the output of `show spanning-tree mstag topology-change flushes`, which displays details about the topology changes that have occurred for each interface:

```
#show spanning-tree mstag b topology-change flushes

MSTAG Protocol Instance b
Interface Last TC Reason Count
-------- ------- ----------- ------
Gi0/0/0/1 18:03:24 2009-07-14 Gi0/0/0/0/10 egress TCN 65535
Gi0/0/0/2 21:05:04 2009-07-15 Gi0/0/0/2/1234567890 ingress TCN 2
```
Configuring PVSTAG: Examples

This example shows PVSTAG configuration for a single VLAN on a single interface:

```bash
config
spanning-tree pvstag example
    preempt delay for 60 seconds
    interface GigabitEthernet0/0/0/0
    vlan 10
    root-priority 0
    root-id 0.0.0
    root-cost 0
    priority 0
    bridge-id 0.0.1
    port-priority 0
    port-id 1
    max age 40
    hello-time 1
    !
    !
```

This example shows the output of `show spanning-tree pvstag`:

```
# show spanning-tree pvstag interface GigabitEthernet0/0/0/1
GigabitEthernet0/0/0/1
VLAN 10
    Preempt delay is disabled.
    Sub-interface:  GigabitEthernet0/0/0/1.20 (Up)
    Max Age: 20
    Root Priority: 0
    Root Bridge: 0000.0000.0000
    Cost: 0
    Bridge Priority: 32768
    Bridge ID: 6161.6161.6161
    Port Priority: 128
    Port ID: 1
    Hello Time: 2
    Active: no
    BPDUs sent: 0
    Topology Changes: 123
    VLAN 20
```

Configuring MVRP-Lite: Examples

This example shows MVRP-lite configuration:

```bash
config
spanning-tree mst example
    mvrp static
        periodic transmit
        join-time 200
        leave-time 30
        leaveall-time 10
        !
    !
```
This example shows the output of `show ethernet mvrp mad`:

```
RP/0/RSP0/CPU0:router# show ethernet mvrp mad interface GigabitEthernet 0/1/0/1
GigabitEthernet0/1/0/1
Participant Type: Full; Point-to-Point: Yes
Admin Control: Applicant Normal; Registrar Normal
LeaveAll Passive (next in 5.92s); periodic disabled
Leave in 25.70s; Join not running
Last peer 0293.6926.9585; failed registrations: 0

<table>
<thead>
<tr>
<th>VID</th>
<th>Applicant</th>
<th>Registrar</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Very Anxious Observer</td>
<td>Leaving</td>
</tr>
<tr>
<td>283</td>
<td>Quiet Passive</td>
<td>Empty</td>
</tr>
</tbody>
</table>
```

This example shows the output of `show ethernet mvrp status`:

```
RP/0/RSP0/CPU0:router# show ethernet mvrp status interface GigabitEthernet 0/1/0/1
GigabitEthernet0/1/0/1
Statically declared: 1-512,768,980-1034
Dynamically declared: 2048-3084
Registered: 1-512
```

This example shows the output of `show ethernet mvrp statistics`:

```
RP/0/RSP0/CPU0:router# show ethernet mvrp statistics interface GigabitEthernet 0/1/0/1
GigabitEthernet0/1/0/1
MVRPDU TX: 1245
MVRPDU RX: 7
Dropped TX: 0
Dropped RX: 42
Invalid RX: 12
```
## Additional References

These sections provide references related to implementing Multiple Spanning Tree Protocol (MSTP) on Cisco ASR 9000 Series Routers.

## Related Documents

<table>
<thead>
<tr>
<th>Related Topic</th>
<th>Document Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multiple Spanning Tree Protocol Commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples</td>
<td>Multiple Spanning Tree Protocol Commands module in Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Command Reference</td>
</tr>
</tbody>
</table>

## Standards

<table>
<thead>
<tr>
<th>Standards</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEEE 802.1Q-2005</td>
<td>IEEE Standard for Local and Metropolitan Area Networks - Virtual Bridged Local Area Networks</td>
</tr>
</tbody>
</table>

## MIBs

<table>
<thead>
<tr>
<th>MIBs</th>
<th>MIBs Link</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at this URL and choose a platform under the Cisco Access Products menu: <a href="http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml">http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml</a></td>
</tr>
</tbody>
</table>

## RFCs

<table>
<thead>
<tr>
<th>RFCs</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.</td>
<td>—</td>
</tr>
</tbody>
</table>
# Technical Assistance

<table>
<thead>
<tr>
<th>Description</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>able technical content, including links to products, technologies, solutions,</td>
<td></td>
</tr>
<tr>
<td>technical tips, and tools. Registered Cisco.com users can log in from this</td>
<td></td>
</tr>
<tr>
<td>page to access even more content.</td>
<td></td>
</tr>
</tbody>
</table>