Configure Virtual LANs in Layer 2 VPNs

The Layer 2 Virtual Private Network (L2VPN) feature enables Service Providers (SPs) to provide L2 services to geographically disparate customer sites.

A virtual local area network (VLAN) is a group of devices on one or more LANs that are configured so that they can communicate as if they were attached to the same wire, when in fact they are located on a number of different LAN segments. The IEEE's 802.1Q specification establishes a standard method for inserting VLAN membership information into Ethernet frames.

VLANs are very useful for user and host management, bandwidth allocation, and resource optimization. Using VLANs addresses the problem of breaking large networks into smaller parts so that broadcast and multicast traffic does not consume more bandwidth than necessary. VLANs also provide a higher level of security between segments of internal networks.

The 802.1Q specification establishes a standard method for inserting VLAN membership information into Ethernet frames. Cisco IOS XR software supports VLAN sub-interface configuration on Gigabit Ethernet and 10-Gigabit Ethernet interfaces.

The configuration model for configuring VLAN Attachment Circuits (ACs) is similar to the model used for configuring basic VLANs, where the user first creates a VLAN sub-interface, and then configures that VLAN in sub-interface configuration mode. To create an Attachment Circuit, you need to include the **l2transport** keyword in the **interface** command string to specify that the interface is a L2 interface.

VLAN ACs support the following modes of L2VPN operation:

- Basic Dot1Q Attachment Circuit—The Attachment Circuit covers all frames that are received and sent with a specific VLAN tag.
- QinQ Attachment Circuit—The Attachment Circuit covers all frames received and sent with a specific outer VLAN tag and a specific inner VLAN tag. QinQ is an extension to Dot1Q that uses a stack of two tags.

Encapsulation

Encapsulation defines the matching criteria that maps a VLAN, a range of VLANs. Different types of encapsulations are default, dot1q, dot1ad. The following are the supported encapsulation types:

- **encapsulation default**: Configures the default service instance on a port.
- **encapsulation dot1q vlan-id** : Defines the matching criteria to map 802.1Q frames ingress on an interface to the appropriate service instance.

- **encapsulation dot1ad vlan-id** : Defines the matching criteria to map 802.1ad frames ingress on an interface to the appropriate service instance.
- **encapsulation dot1q second-dot1q**: Defines the matching criteria to map Q-in-Q ingress frames on an interface to the appropriate service instance.
- **encapsulation dot1ad dot1q**: Defines the matching criteria to be used in order to map single-tagged 802.1ad frames ingress on an interface to the appropriate service instance.

Restrictions and Limitations

To configure VLANs for Layer 2 VPNs, the following restrictions are applicable.

- In a point-to-point connection, the two Attachment Circuits do not have to be of the same type. For example, a port mode Ethernet Attachment Circuit can be connected to a Dot1Q Ethernet Attachment Circuit.
- Pseudowires can run in VLAN mode or in port mode. A pseudowire running in VLAN mode always carries Dot1Q or Dot1ad tag(s), while a pseudowire running in port mode may or may NOT carry tags. To connect these different types of circuits, popping, pushing, and rewriting tags is required.
- The Attachment Circuits on either side of an MPLS pseudowire can be of different types. In this case, the appropriate conversion is carried out at one or both ends of the Attachment Circuit to pseudowire connection.
- When receiving single or double Dot1Q tagged traffic on an L2VPN pseudowire, the egress rewrite action Push 1 configured in an attachment circuit is not supported. The egress rewrite action Push 1 configured in an attachment circuit is supported only for untagged traffic received on an L2VPN pseudowire.

- Configure VLAN Subinterfaces, on page 2
- Introduction to Ethernet Flow Point, on page 5
- Configure VLAN Header Rewrite, on page 10
- Rewrite of Priority Tag, on page 22

Configure VLAN Subinterfaces

Subinterfaces are logical interfaces created on a hardware interface. These software-defined interfaces allow for segregation of traffic into separate logical channels on a single hardware interface as well as allowing for better utilization of the available bandwidth on the physical interface.

Subinterfaces are distinguished from one another by adding an extension on the end of the interface name and designation. For instance, the Ethernet subinterface 23 on the physical interface designated TenGigE 0/1/0/0 would be indicated by TenGigE 0/1/0/0.23.

Before a subinterface is allowed to pass traffic, it must have a valid tagging protocol encapsulation and VLAN identifier assigned. All Ethernet subinterfaces always default to the 802.1Q VLAN encapsulation. However, the VLAN identifier must be explicitly defined.

The subinterface Maximum Transmission Unit (MTU) is inherited from the physical interface with 4 bytes allowed for the 802.1Q VLAN tag.

The following modes of VLAN subinterface configuration are supported:

- Basic dot1q Attachment Circuit
- Basic dot1ad Attachment Circuit
- Q-in-Q Attachment Circuit

To configure a basic dot1q Attachment Circuit, use this encapsulation mode:

encapsulation dot1q *vlan extra-id*

To configure a basic dot1ad Attachment Circuit, use this encapsulation mode:

encapsulation dot1ad *vlan-id*

To configure a Q-in-Q Attachment Circuit, use the following encapsulation modes:

- **encapsulation dot1q** *vlan-id* **second-dot1q** *vlan-id*
- **encapsulation dot1ad** *vlan-id* **dot1q** *vlan-id*

- BVI with Double-Tagged AC—You can configure the attachment circuit (AC) with double-VLAN tag encapsulation on the bridge-group virtual interface (BVI). You must specify the rewrite ingress pop 2 symmetric option when you configure the AC on the BVI with double-VLAN tag encapsulation.

Note On Cisco N540X-6Z18G-SYS-A/D and N540X-8Z16G-SYS- A/D variants a maximum of up to 1K subinterfaces (includes both bundle and physical) are supported overall.

Restrictions and Limitations

To configure VLAN subinterface, the following restrictions are applicable.

- At least 64 VLAN-IDs in a VLAN list is required to overcome the limitation of only 9 VLAN ranges per NPU.
- For double-tagged packet, the VLAN range is supported only on the inner tag.
- VLANs separated by comma are called a VLAN list. VLAN list isn’t supported on the router.
- If 0x9100/0x9200 is configured as tunneling ether-type, then dot1ad (0x88a8) encapsulation isn’t supported.
- If any subinterface is already configured under a main interface, modifying the tunneling ether-type isn’t supported.
- Following limitations are applicable to both outer and inner VLAN ranges:
  - 32 unique VLAN ranges are supported per NPU.
  - The overlap between outer VLAN ranges on subinterfaces of the same Network Processor Unit (NPU) isn’t supported. A subinterface with a single VLAN tag that falls into a range configured on another subinterface of the same NPU is also considered an overlap.
  - The overlap between inner VLAN ranges on subinterfaces of the same NPU isn’t supported.
  - Range 'any' doesn’t result in explicit programming of a VLAN range in hardware and therefore doesn’t count against the configured ranges.

Configuration Example

Configuring a VLAN subinterface involves:

- Creating a Ten Gigabit Ethernet subinterface
- Enabling L2 transport mode on the interface
- Defining the matching criteria (encapsulation mode) to be used in order to map ingress frames on an interface to the appropriate service instance.

Configuration of Basic dot1q Attachment Circuit

```
Router# configure
Router(config)# interface TenGigE 0/0/0/10.1 l2transport
Router(config-if)# encapsulation dot1q 10 exact
Router(config-if)# no shutdown

Router# configure
Router(config)#interface TenGigE 0/0/0/1.101 l2transport
Router(config-subif)#encapsulation list-extended dot1q
66-67,68-69,70-71,118-119,120-121,122-123,229,230,231
```

Running Configuration

```
configure
 interface TenGigE 0/0/0/10.1
  l2transport
  encapsulation dot1q 10 exact
 !
!

Configure
 interface TenGigE 0/0/0/1.101
  l2transport
  encapsulation list-extended dot1q 66-67,68-69,70-71,118-119,120-121,122-123,229,230,231
```

Verification

Verify that the VLAN subinterface is active:

```
Router# show interfaces TenGigE 0/0/0/10.1

...
TenGigE0/0/0/10.1 is up, line protocol is up
  Interface state transitions: 1
  Hardware is VLAN sub-interface(s), address is 0011.1aac.a05a
  Layer 2 Transport Mode
  MTU 1518 bytes, BW 10000000 Kbit (Max: 10000000 Kbit)
     reliability Unknown, txload Unknown, rxload Unknown
  Encapsulation 802.1Q Virtual LAN,
    Outer Match: Dot1Q VLAN 10
    Ethertype Any, MAC Match src any, dest any
  loopback not set,
...
```

```
Router#show interfaces TenGigE 0/0/0/1.101
TenGigabitEthernet0/0/0/1.101 is down, line protocol is down
   Interface state transitions: 0
   Hardware is VLAN sub-interface(s), address is 008a.9678.0c04
   Layer 2 Transport Mode
   MTU 1518 bytes, BW 10000000 Kbit (Max: 10000000 Kbit)
      reliability Unknown, txload Unknown, rxload Unknown
   Encapsulation 802.1Q Virtual LAN,
      Outer Match: Dot1Q VLAN 66-67,68-69,70-71,118-119,120-121,122-123,229,230,231
      Ethertype Any, MAC Match src any, dest any
   loopback not set,
   Last input never, output never
   Last clearing of "show interface" counters never
      0 packets input, 0 bytes
      0 input drops, 0 queue drops, 0 input errors
      0 packets output, 0 bytes
      0 output drops, 0 queue drops, 0 output errors
```

Associated Commands

- encapsulation dot1ad dot1q
- encapsulation dot1q
- encapsulation dot1q second-dot1q
- l2transport (Ethernet)
- encapsulation dot1ad

Introduction to Ethernet Flow Point

An Ethernet Flow Point (EFP) is a Layer 2 logical sub-interface used to classify traffic under a physical or a bundle interface. An EFP is defined by a set of filters (a set of entries) that are applied to all the ingress traffic to classify the frames that belong to a particular EFP. Each entry usually contains 0, 1 or 2 VLAN tags. You can specify a VLAN or QinQ tagging to match against on ingress. A packet that starts with the same tags as an entry in the filter is said to match the filter; if the start of the packet does not correspond to any entry in the filter, then the packet does not match the filter.

All traffic on ingress are processed by that EFP if a match occurs, and this can in turn change VLAN IDs, add or remove VLAN tags, and change ethertypes. After the frames are matched to a particular EFP, any appropriate feature (such as, any frame manipulations specified by the configuration as well as things such as QoS and ACLs) can be applied.

The benefits of EFP include:

- Identifying all frames that belong to a particular flow on a given interface
- Performing VLAN header rewrites

  (See, Configure VLAN Header Rewrite, on page 10)
- Adding features to the identified frames
- Optionally defining how to forward the identified frames in the data path

Limitations of EFP

Egress EFP filtering is not supported on Cisco IOS XR.

Identify Frames of an EFP

The EFP identifies frames belonging to a particular flow on a given port, independent of their Ethernet encapsulation. An EFP can flexibly map frames into a flow or EFP based on the fields in the frame header. The frames can be matched to an EFP using VLAN tags.

The frames can't be matched to an EFP through this:

- Any information outside the outermost Ethernet frame header and its associated tags such as
  - IPv4, IPv6, or MPLS tag header data
  - C-DMAC, C-SMAC, or C-VLAN

VLAN Tag Identification

Below table describes the different encapsulation types and the EFP identifier corresponding to each.

Encapsulation Type	EFP Identifier
Single tagged frames	802.1Q customer-tagged Ethernet frames
Double tagged frames Double tagged frames can be of the following types: • Single range • Range-in-Q • Q-in-Range	802.1Q (ethertype 0x9100) double tagged frames 802.1ad (ethertype 0x9200) double tagged frames • In single range, a range of VLAN IDs can be added for an EFP. • In Range-in-Q, a range of outer VLAN IDs can have a single inner VLAN ID. • In Q-in-Range, a single outer VLAN ID can have a range of inner VLAN IDs.

You can use wildcards while defining frames that map to a given EFP. EFPs can distinguish flows based on a single VLAN tag, a stack of VLAN tags or a combination of both (VLAN stack with wildcards). It provides the EFP model, a flexibility of being encapsulation agnostic, and allows it to be extensible as new tagging or tunneling schemes are added.

Apply Features

After the frames are matched to a particular EFP, any appropriate features can be applied. In this context, "features" means any frame manipulations specified by the configuration as well as things such as QoS and ACLs. The Ethernet infrastructure provides an appropriate interface to allow the feature owners to apply their features to an EFP. Hence, IM interface handles are used to represent EFPs, allowing feature owners to manage their features on EFPs in the same way the features are managed on regular interfaces or sub-interfaces.

The only L2 features that can be applied on an EFP that is part of the Ethernet infrastructure are the L2 header encapsulation modifications. The L2 features are described in this section.

Encapsulation Modifications

EFP supports these L2 header encapsulation modifications on both ingress and egress:

- Push 1 or 2 VLAN tags
- Pop 1 or 2 VLAN tags

Note This modification can only pop tags that are matched as part of the EFP.

- Rewrite 1 or 2 VLAN tags:
  - Rewrite outer tag
  - Rewrite outer 2 tags
  - Rewrite outer tag and push an additional tag

  For each of the VLAN ID manipulations, these can be specified:
- The VLAN tag type, that is, C-VLAN, S-VLAN, or I-TAG. The ethertype of the 802.1Q C-VLAN tag is defined by the dot1q tunneling type command.
- The VLAN ID. 0 can be specified for an outer VLAN tag to generate a priority-tagged frame.

Note For tag rewrites, the CoS bits from the previous tag should be preserved in the same way as the DEI bit for 802.1ad encapsulated frames.

Define Data-Forwarding Behavior

The EFP can be used to designate the frames belonging to a particular Ethernet flow forwarded in the data path. These forwarding cases are supported for EFPs in Cisco IOS XR software:

- L2 Switched Service (Bridging)—The EFP is mapped to a bridge domain, where frames are switched based on their destination MAC address. This includes multipoint services:
  - Ethernet to Ethernet Bridging
  - Multipoint Layer 2 Services
- L2 Stitched Service (AC to AC xconnect)—This covers point-to-point L2 associations that are statically established and do not require a MAC address lookup.
  - Ethernet to Ethernet Local Switching—The EFP is mapped to an S-VLAN either on the same port or on another port. The S-VLANs can be identical or different.
- Tunneled Service (xconnect)—The EFP is mapped to a Layer 3 tunnel. This covers point-to-point services, such as EoMPLS.

Ethernet Flow Points Visibility

EFP Visibility feature enables you to configure multiple VLANs only when IGMP snooping is enabled and multiple VLANs and sub-interfaces of same port is configured under the same bridge domain.

An Ethernet flow point (EFP) service instance is a logical interface that connects a bridge domain to a physical port or to an EtherChannel group. A VLAN tag is used to identify the EFP.

Earlier only one EFP was allowed per bridge-domain. With EFP visibility feature, you can configure a maximum of:

- 600 EFPs per bridge-domain.
- 100 EFPs per port.
- The maximum attachment circuits (VLAN interfaces) per bridge-domain is 128 for Cisco N540X-6Z18G-SYS-A/D and N540X-8Z16G-SYS - A/D variants.

Irrespective of number of ports available, you have flexibility to add more EFPs in one bridge group.

Configuring EFP Visibility

This example shows how to configure IGMP snooping on VLAN interfaces under a bridge domain with multiple EFPs.

```
/* Configure two IGMP Snooping profiles */
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# igmp snooping profile 1
RP/0/RP0/CPU0:router(config-igmp-snooping-profile)# exit
RP/0/RP0/CPU0:router(config)# igmp snooping profile 2
RP/0/RP0/CPU0:router(config-igmp-snooping-profile)#commit

 !

/* Configure VLAN interfaces for L2 transport */
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface gigabitEthernet 0/8/0/8
RP/0/RP0/CPU0:router(config-if)# bundle id 2 mode on
RP/0/RP0/CPU0:router(config-if)# no shut
RP/0/RP0/CPU0:router(config-if)# exit
RP/0/RP0/CPU0:router(config)# interface gigabitEthernet 0/8/0/9
RP/0/RP0/CPU0:router(config-if)# bundle id 3 mode on
RP/0/RP0/CPU0:router(config-if)# no shut
RP/0/RP0/CPU0:router(config-if)# exit

RP/0/RP0/CPU0:router(config)# interface Bundle-Ether2
RP/0/RP0/CPU0:router(config-if)# exit
RP/0/RP0/CPU0:router(config)# interface Bundle-Ether3
RP/0/RP0/CPU0:router(config-if)# exit

RP/0/RP0/CPU0:router(config)# interface Bundle-Ether2.2 l2transport
RP/0/RP0/CPU0:router(config-subif)# encapsulation dot1q 2
RP/0/RP0/CPU0:router(config-subif)# rewrite ingress tag pop 1 symmetric
RP/0/RP0/CPU0:router(config-subif)# exit
RP/0/RP0/CPU0:router(config)# interface Bundle-Ether2.3 l2transport
RP/0/RP0/CPU0:router(config-subif)# encapsulation dot1q 3
RP/0/RP0/CPU0:router(config-subif)# rewrite ingress tag pop 1 symmetric
RP/0/RP0/CPU0:router(config-subif)# exit
RP/0/RP0/CPU0:router(config)# interface Bundle-Ether2.4 l2transport
RP/0/RP0/CPU0:router(config-subif)# encapsulation dot1q 4
RP/0/RP0/CPU0:router(config-subif)# rewrite ingress tag pop 1 symmetric
RP/0/RP0/CPU0:router(config-subif)# exit
RP/0/RP0/CPU0:router(config)# interface Bundle-Ether2.5 l2transport
RP/0/RP0/CPU0:router(config-subif)# encapsulation dot1q 5
RP/0/RP0/CPU0:router(config-subif)# rewrite ingress tag pop 1 symmetric
RP/0/RP0/CPU0:router(config-subif)# exit
```

```
RP/0/RP0/CPU0:router(config)# interface Bundle-Ether3.2 l2transport
RP/0/RP0/CPU0:router(config-subif)# encapsulation dot1q 2
RP/0/RP0/CPU0:router(config-subif)# rewrite ingress tag pop 1 symmetric
RP/0/RP0/CPU0:router(config-subif)# exit
RP/0/RP0/CPU0:router(config)# interface Bundle-Ether3.3 l2transport
RP/0/RP0/CPU0:router(config-subif)# encapsulation dot1q 3
RP/0/RP0/CPU0:router(config-subif)# rewrite ingress tag pop 1 symmetric
RP/0/RP0/CPU0:router(config-subif)# exit
RP/0/RP0/CPU0:router(config)# commit

/* Attach a profile and add interfaces to the bridge domain.
Attach a profile to one of the interfaces. The other interface
inherits IGMP snooping configuration attributes from the bridge domain profile */

RP/0/RP0/CPU0:router(config)#l2vpn
RP/0/RP0/CPU0:router(config-l2vpn)#bridge group VLAN2
RP/0/RP0/CPU0:router(config-l2vpn-bg)#bridge-domain VLAN2
RP/0/RP0/CPU0:router(config-l2vpn-bg-bd)#efp-visibility
RP/0/RP0/CPU0:router(config-l2vpn-bg-bd)#igmp snooping profile 1
RP/0/RP0/CPU0:router(config-l2vpn-bg-bd)#interface bundle-Ether2.2
RP/0/RP0/CPU0:router(config-l2vpn-bg-bd-ac)#exit
RP/0/RP0/CPU0:router(config-l2vpn-bg-bd)#interface bundle-Ether 2.3
RP/0/RP0/CPU0:router(config-l2vpn-bg-bd-ac)#exit
RP/0/RP0/CPU0:router(config-l2vpn-bg-bd)#interface bundle-Ether 2.4
RP/0/RP0/CPU0:router(config-l2vpn-bg-bd-ac)#exit
RP/0/RP0/CPU0:router(config-l2vpn-bg-bd)#interface bundle-Ether 2.5
RP/0/RP0/CPU0:router(config-l2vpn-bg-bd-ac)#exit
RP/0/RP0/CPU0:router(config-l2vpn-bg-bd)#exit
RP/0/RP0/CPU0:router(config-l2vpn-bg)#bridge-domain vlan3
RP/0/RP0/CPU0:router(config-l2vpn-bg-bd)#efp-visibility
RP/0/RP0/CPU0:router(config-l2vpn-bg-bd)#igmp snooping profile 2
RP/0/RP0/CPU0:router(config-l2vpn-bg-bd)#interface bundle-Ether3.2
RP/0/RP0/CPU0:router(config-l2vpn-bg-bd-ac)#exit
RP/0/RP0/CPU0:router(config-l2vpn-bg-bd)#interface bundle-Ether 3.3
RP/0/RP0/CPU0:router(config-l2vpn-bg-bd-ac)#exit
RP/0/RP0/CPU0:router(config-l2vpn-bg-bd)#routed interface bvi2
RP/0/RP0/CPU0:router(config-l2vpn-bg-bd-bvi)#exit
RP/0/RP0/CPU0:router(config-l2vpn-bg-bd)#evi 2
RP/0/RP0/CPU0:router(config-l2vpn-bg-bd-evi)#exit
RP/0/RP0/CPU0:router(config-l2vpn-bg-bd)#commit
```

Verification

Verify the configured bridge ports:

```
RP/0/RP0/CPU0:router# show igmp snooping port
                      Bridge Domain VLAN2:VLAN2

                                                        State
Port                                                Oper  STP  Red   #Grps   #SGs
----                                                ----  ---  ---   -----   ----
BVI2                                                Up    -    -       0      0
Bundle-Ether2.2                                     Up    -    -     100      0
Bundle-Ether2.3                                     Up    -    -     100      0
Bundle-Ether2.4                                     Up    -    -     100      0
Bundle-Ether2.5                                     Up    -    -     100      0

 Bridge Domain VLAN3:VLAN3

                                                        State
Port                                                Oper  STP  Red   #Grps   #SGs
----                                                ----  ---  ---   -----   ----
BVI3                                                Up    -    -       0      0
```

```
Bundle-Ether3.2                                  Up    -    -     100      0
Bundle-Ether3.3                                  Up    -    -     100      0
```

In the above output verify the status of BVI and EFPs are **Up**, and the **#Grps** and **#SG** show the correct number of IGMP join received.

Configure VLAN Header Rewrite

EFP supports the following VLAN header rewrites on both ingress and egress ports:

- Push 1 VLAN tag
- Pop 1 VLAN tag

Note This rewrite can only pop tags that are matched as part of the EFP.

- Translate 1 or 2 VLAN tags:
  - Translate 1-to-1 tag: Translates the outermost tag to another tag
  - Translate 1-to-2 tags: Translates the outermost tag to two tags
  - Translate 2-to-2 tags: Translates the outermost two tags to two other tags

Various combinations of ingress, egress VLAN rewrites with corresponding tag actions during ingress and egress VLAN translation, are listed in the following sections:

Limitations

The limitations for VLAN header rewrites are as follows:

- Push 1 is not supported for dot1ad configuration.
- Push 2 is supported only on:
  - Untagged EFP
  - Dot1q EFP with **exact** configuration statement
- Translate 1 to 1 is not supported for dot1ad configuration.
- Translate 1 to 2 is not supported with **dot1q tunneling ethertype** configuration statement.
- From Cisco IOS XE 7.3.1 release, pop 2 is supported only on Cisco N540X-6Z18G-SYS-A/D and N540X-8Z16G-SYS - A/D variants.
- Translate 2 to 1 is not supported.
- When a single-tag range is used, double tagged traffic does not match.

  For example, in the following configuration, dot1q 2-6 is the outer tag.

```
Router#configure
```

```
Router(config)# interface GigabitEthernet0/0/0/0.0 l2transport
Router(config-if)# encapsulation dot1q 2-6
```

- An incoming packet with an outer tag of 2 and ANY inner tag does not match. For example, the double tag packet of outer tag 2 and inner tag 1 is not be accepted on the interface 0/0/0/0.0.
- But, an incoming packet with a single tag of 2 is accepted. For example, the single tag packet of outer tag between 2 to 6 is accepted on the interface 0/0/0/0.0.

Configuration Example

This topic covers VLAN header rewrites on various attachment circuits, such as:

- L2 single-tagged sub-interface
- L2 double-tagged sub-interface

Configuring VLAN header rewrite involves:

- Creating a TenGigabit Ethernet sub-interface
- Enabling L2 transport mode on the interface
- Defining the matching criteria (encapsulation mode) to be used in order to map single-tagged frames ingress on an interface to the appropriate service instance
- Specifying the encapsulation adjustment that is to be performed on the ingress frame

Configuration of VLAN Header Rewrite (single-tagged sub-interface)

```
Router# configure
Router(config)# interface TenGigE 0/0/0/10.1 l2transport
Router(config-if)# encapsulation dot1q 10 exact
Router(config-if)# rewrite ingress tag push dot1q 20 symmteric
```

Running Configuration

```
/* Configuration without rewrite */

configure
 interface TenGigE0/0/0/0.1 l2transport
  encapsulation dot1q 10 exact
 !
!

/* Configuration with rewrite */

/* PUSH 1 */
interface TenGigE0/0/0/0.1 l2transport
 encapsulation dot1q 10
 rewrite ingress tag push dot1q 20 symmteric
 !
!

/* POP 1 */
interface TenGigE0/0/0/0.1 l2transport
 encapsulation dot1q 10
```

```
  rewrite ingress tag pop 1
 !
!

/* TRANSLATE 1-1 */

interface TenGigE0/0/0/0.1 l2transport
 encapsulation dot1q 10
  rewrite ingress tag translate 1-to-1 dot1q 20
 !
!

/* TRANSLATE 1-2 */

interface TenGigE0/0/0/0.1 l2transport
 encapsulation dot1q 10
  rewrite ingress tag translate 1-to-2 dot1q 20 second-dot1q 30
 !
!
```

Running Configuration (VLAN header rewrite on double-tagged sub-interface)

```
/* Configuration without rewrite */

interface TenGigE0/0/0/0.1 l2transport
 encapsulation dot1q 10 second-dot1q 11
 !
!

/* Configuration with rewrite */

/* PUSH 1 */
interface TenGigE0/0/0/0.1 l2transport
 encapsulation dot1q 10 second-dot1q 11
  rewrite ingress tag push dot1q 20 symmteric
 !
!

/* TRANSLATE 1-1 */

interface TenGigE0/0/0/0.1 l2transport
 encapsulation dot1q 10 second-dot1q 11
  rewrite ingress tag translate 1-to-1 dot1q 20
 !
!

/* TRANSLATE 1-2 */

interface TenGigE0/0/0/0.1 l2transport
 encapsulation dot1q 10 second-dot1q 11
  rewrite ingress tag translate 1-to-2 dot1q 20 second-dot1q 30
 !
!

/* TRANSLATE 2-2 */

interface TenGigE0/0/0/0.1 l2transport
 encapsulation dot1q 10 second-dot1q 11
  rewrite ingress tag translate 2-to-2 dot1q 20 second-dot1q 30
 !
!
```

Associated Commands

- encapsulation dot1ad dot1q
- encapsulation dot1q
- encapsulation dot1q second-dot1q
- l2transport (Ethernet)
- rewrite ingress tag

Valid Ingress Rewrite Actions

Table 1: Valid Ingress Rewrite Actions

Interface Configuration	Ingress Rewrite Action
dot1q	No rewrite
dot1q	Pop 1
dot1q	Push 1
dot1q	Push 2
dot1q	Translate 1 to 1
dot1q	Translate 1 to 2
QinQ	No rewrite
QinQ	Pop 1
QinQ	Push 1
QinQ	Translate 1 to 1
QinQ	Translate 1 to 2
QinQ	Translate 2 to 2
Untagged	No rewrite
Untagged	Push 1
Untagged	Push 2

The following notations are used for the rewrite actions mentioned in the table:

- Translate 1-to-1 tag: Translates the outermost tag to another tag.
- Translate 1-to-2 tags: Translates the outermost tag to two tags.
- Translate 2-to-2 tags: Translates the outermost two tags to two other tags.

Valid Ingress-Egress Rewrite Combinations

Table 2: Valid Ingress-Egress Rewrite Combinations

Ingress Interface Configuration	Ingress Interface Rewrite Action	Egress Interface Configuration	Egress Interface Rewrite Action
dot1q	No rewrite	dot1q	No rewrite
dot1q	No rewrite	dot1q	Pop 1
dot1q	No rewrite	dot1q	Push 1
dot1q	No rewrite	dot1q	Translate 1-to-1
dot1q	Pop 1	dot1q	No rewrite
dot1q	Pop 1	dot1q	Pop 1
dot1q	Push 1	dot1q	No rewrite
dot1q	Push 1	dot1q	Push 1
dot1q	Push 1	dot1q	Push 2
dot1q	Push 1	dot1q	Translate 1-to-1
dot1q	Push 1	dot1q	Translate 1-to-2
dot1q	Push 2 / Translate 1-to-2	dot1q	Push 1
dot1q	Push 2 / Translate 1-to-2	dot1q	Push 2
dot1q	Push 2 / Translate 1-to-2	dot1q	Translate 1-to-2
dot1q	Translate 1-to-1	dot1q	No rewrite
dot1q	Translate 1-to-1	dot1q	Push 1
dot1q	Translate 1-to-1	dot1q	Translate 1-to-1
dot1q	No rewrite	dot1q range	No rewrite
dot1q	No rewrite	dot1q range	Push 1
dot1q	Pop 1	dot1q range	No rewrite
dot1q	Push 1	dot1q range	No rewrite
dot1q	Push 1	dot1q range	Push 1
dot1q	Push 1	dot1q range	Push 2
dot1q	Translate 1-to-1	dot1q range	No rewrite

Ingress Interface Configuration	Ingress Interface Rewrite Action	Egress Interface Configuration	Egress Interface Rewrite Action
dot1q	Translate 1-to-1	dot1q range	Push 1
dot1q	Translate 1-to-2	dot1q range	Push 1
dot1q	Translate 1-to-2	dot1q range	Push 2
dot1q	No rewrite / Translate 1-to-1	QinQ	No rewrite
dot1q	No rewrite / Translate 1-to-1	QinQ	Pop 1
dot1q	No rewrite / Translate 1-to-1	QinQ	Push 1
dot1q	No rewrite / Translate 1-to-1	QinQ	Translate 1-to-1
dot1q	Pop 1	QinQ	No rewrite
dot1q	Pop 1	QinQ	Pop 1
dot1q	Push 1	QinQ	No rewrite
dot1q	Push 1	QinQ	Pop 1
dot1q	Push 1	QinQ	Push 1
dot1q	Push 1	QinQ	Translate 1-to-1
dot1q	Push 1	QinQ	Translate 1-to-2
dot1q	Push 1	QinQ	Translate 2-to-2
dot1q	Push 2 / Translate 1-to-2	QinQ	No rewrite
dot1q	Push 2 / Translate 1-to-2	QinQ	Push 1
dot1q	Push 2 / Translate 1-to-2	QinQ	Translate 1-to-1
dot1q	Push 2 / Translate 1-to-2	QinQ	Translate 1-to-2
dot1q	Push 2 / Translate 1-to-2	QinQ	Translate 2-to-2
dot1q	No rewrite / Translate 1-to-1	QinQ range	No rewrite
dot1q	No rewrite / Translate 1-to-1	QinQ range	Pop 1
dot1q	No rewrite / Translate 1-to-1	QinQ range	Push 1
dot1q	No rewrite / Translate 1-to-1	QinQ range	Translate 1-to-1
dot1q	Pop 1	QinQ range	No rewrite
dot1q	Pop 1	QinQ range	Pop 1
dot1q	Push 1	QinQ range	No rewrite

Ingress Interface Configuration	Ingress Interface Rewrite Action	Egress Interface Configuration	Egress Interface Rewrite Action
dot1q	Push 1	QinQ range	Pop 1
dot1q	Push 1	QinQ range	Push 1
dot1q	Push 1	QinQ range	Translate 1-to-1
dot1q	Push 1	QinQ range	Translate 1-to-2
dot1q	Push 2 / Translate 1-to-2	QinQ range	No rewrite
dot1q	Push 2 / Translate 1-to-2	QinQ range	Push 1
dot1q	Push 2 / Translate 1-to-2	QinQ range	Translate 1-to-1
dot1q	Push 2 / Translate 1-to-2	QinQ range	Translate 1-to-2
dot1q	No rewrite	QinQ range	No rewrite
dot1q	No rewrite	Untagged	Push 1
dot1q	Pop 1	Untagged	No rewrite
dot1q	Push 1	Untagged	Push 1
dot1q	Push 1	Untagged	Push 2
dot1q	Push 2	Untagged	Push 2
dot1q	Translate 1-to-1	Untagged	Push 1
dot1q	Translate 1-to-2	Untagged	Push 2
dot1q range	No rewrite	dot1q range	No rewrite
dot1q range	No rewrite	dot1q range	Push 1
dot1q range	Push 1	dot1q range	No rewrite
dot1q range	Push 1	dot1q range	Push 1
dot1q range	Push 1	dot1q range	Push 2
dot1q range	Push 2	dot1q range	Push 1
dot1q range	Push 2	dot1q range	Push 2
dot1q range	No rewrite	QinQ	No rewrite
dot1q range	No rewrite	QinQ	Pop 1
dot1q range	No rewrite	QinQ	Push 1
dot1q range	No rewrite	QinQ	Translate 1-to-1

Ingress Interface Configuration	Ingress Interface Rewrite Action	Egress Interface Configuration	Egress Interface Rewrite Action
dot1q range	Push 1	QinQ	No rewrite
dot1q range	Push 1	QinQ	Pop 1
dot1q range	Push 1	QinQ	Push 1
dot1q range	Push 1	QinQ	Translate 1-to-1
dot1q range	Push 1	QinQ	Translate 1-to-2
dot1q range	Push 1	QinQ	Translate 2-to-2
dot1q range	Push 2	QinQ	No rewrite
dot1q range	Push 2	QinQ	Push 1
dot1q range	Push 2	QinQ	Translate 1-to-1
dot1q range	Push 2	QinQ	Translate 1-to-2
dot1q range	Push 2	QinQ	Translate 2-to-2
dot1q range	No rewrite	QinQ range / QinAny	No rewrite
dot1q range	No rewrite	QinQ range	Pop 1
dot1q range	No rewrite	QinQ range / QinAny	Push 1
dot1q range	No rewrite	QinQ range / QinAny	Translate 1-to-1
dot1q range	Push 1	QinQ range / QinAny	No rewrite
dot1q range	Push 1	QinQ range / QinAny	Pop 1
dot1q range	Push 1	QinQ range	Push 1
dot1q range	Push 1	QinQ range / QinAny	Translate 1-to-1
dot1q range	Push 1	QinQ range / QinAny	Translate 1-to-2

Ingress Interface Configuration	Ingress Interface Rewrite Action	Egress Interface Configuration	Egress Interface Rewrite Action
dot1q range	Push 2	QinQ range / QinAny	No rewrite
dot1q range	Push 2	QinQ range / QinAny	Push 1
dot1q range	Push 2	QinQ range / QinAny	Translate 1-to-1
dot1q range	Push 2	QinQ range / QinAny	Translate 1-to-2
dot1q range	No rewrite	Untagged	No rewrite
dot1q range	No rewrite	Untagged	Push 1
dot1q range	Push 1	Untagged	Push 1
dot1q range	Push 1	Untagged	Push 2
dot1q range	Push 2	Untagged	Push 2
QinQ	No rewrite / push 1 / Translate 1-to-1	QinQ	No rewrite
QinQ	No rewrite / push 1 / Translate 1-to-1	QinQ	Pop 1
QinQ	No rewrite / push 1 / Translate 1-to-1	QinQ	Push 1
QinQ	No rewrite / push 1 / Translate 1-to-1	QinQ	Translate 1-to-1
QinQ	No rewrite / push 1 / Translate 1-to-1	QinQ	Translate 1-to-2
QinQ	No rewrite / push 1 / Translate 1-to-1	QinQ	Translate 2-to-2
QinQ	Pop 1	QinQ	No rewrite
QinQ	Pop 1	QinQ	Pop 1
QinQ	Pop 1	QinQ	Push 1
QinQ	Pop 1	QinQ	Translate 1-to-1
QinQ	Translate 1-to-2 / Translate 2-to-2	QinQ	No rewrite

Ingress Interface Configuration	Ingress Interface Rewrite Action	Egress Interface Configuration	Egress Interface Rewrite Action
QinQ	Translate 1-to-2 / Translate 2-to-2	QinQ	Push 1
QinQ	Translate 1-to-2 / Translate 2-to-2	QinQ	Translate 1-to-1
QinQ	Translate 1-to-2 / Translate 2-to-2	QinQ	Translate 1-to-2
QinQ	Translate 1-to-2 / Translate 2-to-2	QinQ	Translate 2-to-2
QinQ	No rewrite / push 1 / Translate 1-to-1	QinQ range / QinAny	No rewrite
QinQ	No rewrite / push 1 / Translate 1-to-1	QinQ range / QinAny	Pop 1
QinQ	No rewrite / push 1 / Translate 1-to-1	QinQ range / QinAny	Push 1
QinQ	No rewrite / push 1 / Translate 1-to-1	QinQ range / QinAny	Translate 1-to-1
QinQ	No rewrite / push 1 / Translate 1-to-1	QinQ range / QinAny	Translate 1-to-2
QinQ	Pop 1	QinQ range / QinAny	No rewrite
QinQ	Pop 1	QinQ range / QinAny	Pop 1
QinQ	Pop 1	QinQ range / QinAny	Push 1
QinQ	Pop 1	QinQ range / QinAny	Translate 1-to-1
QinQ	Translate 1-to-2 / Translate 2-to-2	QinQ range / QinAny	No rewrite
QinQ	Translate 1-to-2 / Translate 2-to-2	QinQ range / QinAny	Push 1
QinQ	Translate 1-to-2 / Translate 2-to-2	QinQ range / QinAny	Translate 1-to-1

Ingress Interface Configuration	Ingress Interface Rewrite Action	Egress Interface Configuration	Egress Interface Rewrite Action
QinQ	Translate 1-to-2 / Translate 2-to-2	QinQ range / QinAny	Translate 1-to-2
QinQ	No rewrite	Untagged	No rewrite
QinQ	No rewrite	Untagged	Push 1
QinQ	No rewrite	Untagged	Push 2
QinQ	Pop 1	Untagged	No rewrite
QinQ	Pop 1	Untagged	Push 1
QinQ	Push 1 / Translate 1-to-1	Untagged	Push 1
QinQ	Push 1 / Translate 1-to-1	Untagged	Push 2
QinQ	Translate 1-to-2 / Translate 2-to-2	Untagged	Push 2
QinQ range / QinAny	No rewrite / push 1 / Translate 1-to-1	QinQ range / QinAny	No rewrite
QinQ range / QinAny	No rewrite / push 1 / Translate 1-to-1	QinQ range / QinAny	Pop 1
QinQ range / QinAny	No rewrite / push 1 / Translate 1-to-1	QinQ range / QinAny	Push 1
QinQ range / QinAny	No rewrite / push 1 / Translate 1-to-1	QinQ range / QinAny	Translate 1-to-1
QinQ range / QinAny	No rewrite / push 1 / Translate 1-to-1	QinQ range / QinAny	Translate 1-to-2
QinQ range / QinAny	Pop 1	QinQ range / QinAny	No rewrite
QinQ range / QinAny	Pop 1	QinQ range / QinAny	Pop 1
QinQ range / QinAny	Pop 1	QinQ range / QinAny	Push 1
QinQ range	Pop 1	QinQ range	Translate 1-to-1

Ingress Interface Configuration	Ingress Interface Rewrite Action	Egress Interface Configuration	Egress Interface Rewrite Action
QinQ range / QinAny	Translate 1-to-2	QinQ range / QinAny	No rewrite
QinQ range / QinAny	Translate 1-to-2	QinQ range / QinAny	Push 1
QinQ range / QinAny	Translate 1-to-2	QinQ range / QinAny	Translate 1-to-1
QinQ range / QinAny	Translate 1-to-2	QinQ range / QinAny	Translate 1-to-2
QinQ range / QinAny	No rewrite	Untagged	No rewrite
QinQ range / QinAny	No rewrite	Untagged	Push 1
QinQ range / QinAny	No rewrite	Untagged	Push 2
QinQ range / QinAny	Pop 1	Untagged	No rewrite
QinQ range / QinAny	Pop 1	Untagged	Push 1
QinQ range / QinAny	Push 1 / Translate 1-to-1	Untagged	Push 1
QinQ range / QinAny	Push 1 / Translate 1-to-1	Untagged	Push 2
QinQ range / QinAny	Translate 1-to-2	Untagged	Push 2
Untagged	No rewrite	Untagged	No rewrite
Untagged	Push 1	Untagged	Push 1
Untagged	Push 2	Untagged	Push 2

The following notations are used for the rewrite actions mentioned in the table:

- Translate 1-to-1 tag: Translates the outermost tag to another tag

- Translate 1-to-2 tags: Translates the outermost tag to two tags
- Translate 2-to-2 tags: Translates the outermost two tags to two other tags

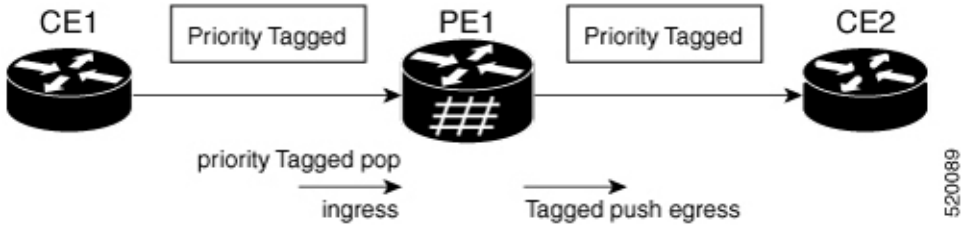
Rewrite of Priority Tag

The Rewrite of Priority Tag feature allows you to configure rewrite tag for a priority-tagged VLAN. This feature removes the priority-tagged VLAN in the ingress direction and adds the priority-tagged VLAN in the egress direction.

You can configure the **rewrite ingress tag symmetric** command for priority-tagged Ethernet Virtual Connections (EVC) on PE1.

This feature supports only rewrite tag pop1 for priority-tag.

Figure 1: Rewrite of Priority Tag



Configure Rewrite of Priority Tag

Perform this task to configure Rewrite of Priority Tag feature.

```
Router#configure
Router(config)#interface FortyGigE0/5/0/0.1 l2transport
Router(config-subif)#encapsulation dot1q priority-tagged
Router(config-subif)#rewrite ingress tag pop 1 symmetric
Router(config-subif)#commit
```

Running Configuration

This section shows Rewrite of Priority Tag running configuration.

```
configure
 interface FortyGigE0/5/0/0.1 l2transport
  encapsulation dot1q priority-tagged
  rewrite ingress tag pop 1 symmetric
!
```

Related Topics

Rewrite of Priority Tag, on page 22

Associated Commands

- rewrite ingress tag pop 1 symmetric