Configuring 802.1Q VLAN Interfaces

This module describes the configuration and management of 802.1Q VLAN interfaces.

The IEEE 802.1Q specification establishes a standard method for tagging Ethernet frames with VLAN membership information. It defines the operation of VLAN bridges that permit the definition, operation, and administration of VLAN topologies within a bridged LAN infrastructure.

The 802.1Q standard is intended to address the problem of how to divide large networks into smaller parts so broadcast and multicast traffic does not use more bandwidth than necessary. The standard also helps provide a higher level of security between segments of internal networks.

Feature History for Configuring 802.1Q VLAN Interfaces

Release	Modification
Release 7.0.11	This feature was introduced.
Release 7.2.12	Support for Layer 2 interfaces was introduced.

Prerequisites for Configuring 802.1Q VLAN Interfaces

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.

Before configuring 802.1Q VLAN interfaces, ensure that you meet the following conditions:

- You must have configured a HundredGigE interface, a FourHundredGigE interface, or an Ethernet bundle interface.

Information About Configuring 802.1Q VLAN Interfaces

To configure 802.1Q VLAN interfaces, you must understand the following concepts:

802.1Q VLAN Overview

Table 1: Feature History Table

Feature Name	Release Information	Feature Description
802.1Q VLAN	Release 24.4.1	Introduced in this release on: Fixed Systems (8700) (select variants only*) * The 802.1Q VLAN functionality is now extended to the Cisco 8712-MOD-M routers.
802.1Q VLAN	Release 24.3.1	Introduced in this release on: Fixed Systems (8200 [ASIC: Q200, P100], 8700 [ASIC: P100])(select variants only*); Modular Systems (8800 [LC ASIC: Q100, Q200, P100])(select variants only*) * The 802.1Q VLAN functionality is now extended to: • 8212-48FH-M • 8711-32FH-M • 88-LC1-52Y8H-EM • 88-LC1-12TH24FH-E
802.1Q VLAN	Release 24.2.11	Introduced in this release on: Modular Systems (8800 [LC ASIC: P100]) (select variants only*) A VLAN is a logical grouping of devices across different LAN segments that communicate as if they are on the same physical network, offering flexibility in user management, bandwidth allocation, and resource optimization. The IEEE 802.1Q protocol standard helps divide large networks into smaller segments to efficiently manage broadcast and multicast traffic, enhancing bandwidth usage and security. * This feature is now supported on routers with 88-LC1-36EH line cards.
Double-Tagged 802.1ad Encapsulation Options for Layer 3 Physical and Bundle Subinterfaces	Release 24.4.1	Introduced in this release on: Fixed Systems (8200 [ASIC: Q200, P100], 8700 [ASIC: P100, K100]); Centralized Systems (8600 [ASIC:Q200]); Modular Systems (8800 [LC ASIC: Q100, Q200, P100]) The support for Double-Tagged 802.1ad Encapsulation Options for Layer 3 Physical and Bundle Subinterfaces is now extended to all Systems in the Cisco 8000 Series Routers.

Double-Tagged 802.1ad Encapsulation Options for Layer 3 Physical and Bundle Subinterfaces	Release 7.3.2	This feature enables you to increase the number of VLAN tags in an interface and an subinterface. You can enable this feature either on a physical interface or a bundle interface. When you configure this feature with the dual tag, interfaces check for IP addresses along with MAC addresses. **Verified Scalability Limits:** • Total number of VLAN tags in an interface and a subinterface with single tag: 4094 • Total number of VLAN tags in an interface and a subinterface with double tag: 4094 * 4094

A VLAN is a group of devices on one or more LANs that you can configure so that the devices can communicate as if they were attached to the same wire. When in fact, they are located on several different LAN segments. Because VLANs are based on logical instead of physical connections, they are flexible for user and host management, bandwidth allocation, and resource optimization.

The IEEE 802.1Q protocol standard addresses the problem of dividing large networks into smaller parts so broadcast and multicast traffic does not consume more bandwidth than necessary. The standard also helps 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 subinterface configuration on 40Gigabit, HundredGig, FourHundredGig, and bundle interfaces.

802.1Q Tagged Frames

The IEEE 802.1Q tag-based VLAN uses an extra tag in the MAC header to identify the VLAN membership of a frame across bridges. This tag is used for VLAN and quality of service (QoS) priority identification. The VLANs can be created statically by manual entry or dynamically through Generic Attribute Registration Protocol (GARP) VLAN Registration Protocol (GVRP). The VLAN ID associates a frame with a specific VLAN and provides the information that switches must process the frame across the network. A tagged frame is four bytes longer than an untagged frame and contains two bytes of Tag Protocol Identifier (TPID) residing within the type and length field of the Ethernet frame and two bytes of Tag Control Information (TCI) which starts after the source address field of the Ethernet frame.

Subinterfaces

Subinterfaces are logical interfaces that you can create on a hardware interface. These software-defined interfaces allow the segregation of traffic into separate logical channels on a single hardware interface. It also allows for the better utilization of the available bandwidth on the physical interface.

You can distinguish subinterfaces from each other by adding an extension at the end of the interface name and designation. For instance, the system indicates Ethernet subinterface 23 on the physical interface designated TenGigE 0/1/0/0, by TenGigE 0/1/0/0.23.

Before the system allows a subinterface 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, you must explicitly define the VLAN identifier.

Supported Encapsulation

Table 2: 802.1ad Encapsulation Support for Layer 3 Interfaces and subinterfaces

Interface Type	Encapsulation	Standard	Support Status
Layer 3 interface Layer 3 subinterface Layer 3 bundle subinterface	Single-Tag Encapsulation	dot1ad	Supported (From Cisco IOS XR Software Release 24.4.1 onwards)
		dot1q	Supported.
	Double-Tag Encapsulation	dot1ad <> dot1q<>	Supported.
		dot1q <> dot1q<>	[1]Supported.

1 The **encapsulation dot1q <x> second-dot1q <y>** encapsulation type is supported on Q200-based line cards from Cisco IOS XR Software Release 24.1.1 onwards and supported for all hardware platforms in the Cisco 8000 Series Routers from Cisco IOS XR Software Release 24.4.1 onwards.

For information about supported encapsulation for Layer 2 Interfaces and subinterfaces, see Virtual LANs in Layer 2 VPNs.

Subinterface MTU

The system inherits the subinterface maximum transmission unit (MTU)from the physical interface with an additional four bytes allowed for the 802.1Q VLAN tag.

Native VLAN

The router does not support a native VLAN. However, the equivalent functionality is accomplished using an **encapsulation** command as follows:

```
encapsulation dot1q TAG-ID
```

Layer 2 VPN on VLANs

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

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 subinterface, and then configures that VLAN in subinterface configuration mode. To create an AC, you need to include the **l2transport** keyword in the **interface** command string to specify that the interface is a Layer 2 interface.

VLAN ACs support three modes of L2VPN operation:

- Basic Dot1Q AC—The AC covers all frames that are received and sent with a specific VLAN tag.
- QinQ AC— Only outer tag (s-tag) of 0x88a8 and inner tag (c-tag) of 0x8100 is supported.

Keep the following in mind when configuring L2VPN on a VLAN:

- Cisco IOS XR software supports 255 ACs per LC.

Use the **show interfaces** command to display AC information.

How to Configure 802.1Q VLAN Interfaces

This section contains the following procedures:

Configuring 802.1Q VLAN Subinterfaces

This task explains how to configure 802.1Q VLAN subinterfaces. To remove these subinterfaces, see the "Removing an 802.1Q VLAN Subinterface" section.

Tip You can programmatically configure and retrieve the VLAN interfaces and subinterfaces parameters using `openconfig-vlan.yang` OpenConfig data model. To get started with using data models, see the *Programmability Configuration Guide for Cisco 8000 Series Routers*.

```
RP/0/RP0/CPU0:router# configure
```

/* Enter subinterface configuration mode and specifies the interface type, location, and subinterface number. */

```
RP/0/RP0/CPU0:router(config)# interface TenGigE 0/2/0/4.10
```

- Replace the *interface-path-id* argument with one of the following instances:
- Physical Ethernet interface instance, or with an Ethernet bundle instance. Naming notation is *rack*/*slot*/*module*/*port*, and a slash between values is required as part of the notation.
- Ethernet bundle instance. Range is from 1 through 65535.
- Replace the *subinterface* argument with the subinterface value. Range is from 0 through 4095.
- Naming notation is *interface-path-id.subinterface*, and a period between arguments is required as part of the notation.

/* Set the Layer 2 encapsulation of an interface. */

```
RP/0/RP0/CPU0:router(config-subif)# encapsulation dot1q 100
```

Note
- The **dot1q vlan** command is replaced by the **encapsulation dot1q** command on the Cisco 8000 Series Router. It is still available for backward-compatibility, but only for Layer 3 interfaces.

/* Assign an IP address and subnet mask to the subinterface. */

```
RP/0/RP0/CPU0:router(config-subif)# ipv4 address 178.18.169.23/24
```

- Replace *ip-address* with the primary IPv4 address for an interface.
- Replace *mask* with the mask for the associated IP subnet. The network mask can be specified in either of two ways:
- The network mask can be a four-part dotted decimal address. For example, 255.0.0.0 indicates that each bit equal to 1 means that the corresponding address bit belongs to the network address.
- The network mask can be indicated as a slash (/) and number. For example, /8 indicates that the first 8 bits of the mask are ones, and the corresponding bits of the address are network address.

/* The **exit** command is not explicitly required. */

```
RP/0/RP0/CPU0:router(config-subif)# exit
RP/0/RP0/CPU0:router(config)# end
or
RP/0/RP0/CPU0:router(config)# 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.

Example

```
"RP/0/RP0/CPU0:S3(config)#interface fourHundredGigE 0/5/0/1.100
RP/0/RP0/CPU0:S3(config-subif)#ipv4 address 100.100.100.100/31
RP/0/RP0/CPU0:S3(config-subif)#encapsulation dot1q 100
RP/0/RP0/CPU0:S3(config-subif)#no shutdown
RP/0/RP0/CPU0:S3(config-subif)#commit
Mon Jul  8 23:05:01.979 PDT
RP/0/RP0/CPU0:S3(config-subif)#end
RP/0/RP0/CPU0:S3#show interfaces fourHundredGigE 0/5/0/1.100 brief
Mon Jul  8 23:05:08.784 PDT

               Intf       Intf        LineP              Encap  MTU        BW
               Name       State       State               Type (byte)    (Kbps)
--------------------------------------------------------------------------------
      FH0/5/0/1.100        up        up             802.1Q  1518  400000000

RP/0/RP0/CPU0:S3#show interfaces brief location 0/5/CPU0 | include 802.1Q
Mon Jul  8 23:07:43.929 PDT
      FH0/5/0/1.100        up        up             802.1Q  1518  400000000
RP/0/RP0/CPU0:S3#

RP/0/RP0/CPU0:S3#"
```

Configuring an Attachment Circuit on a VLAN

Use the following procedure to configure an attachment circuit on a VLAN.

SUMMARY STEPS

1. **configure**
2. **interface [HundredGigE | TenGigE | Bundle-Ether | TenGigE]** *interface-path*] *id.subinterface* **l2transport**
3. **encapsulation dot1q** *vlan-id*
4. **end** or **commit**
5. **show interfaces** [**HundredGigE | TenGigE**] *interface-path-id.subinterface*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure terminal`	Enters global configuration mode.
Step 2	**interface [HundredGigE | TenGigE | Bundle-Ether | TenGigE]** *interface-path*] *id.subinterface* **l2transport** **Example:** `RP/0/RP0/CPU0:router(config)# interface TenGigE 0/1/0/0.1 l2transport`	Enters subinterface configuration and specifies the interface type, location, and subinterface number. • Replace the argument with one of the following instances: • Physical Ethernet interface instance, or with an Ethernet bundle instance. Naming notation is *rack*/*slot*/*module*/*port*, and a slash between values is required as part of the notation. • Ethernet bundle instance. Range is from 1 through 65535. • Replace the *subinterface* argument with the subinterface value. Range is from 0 through 4095. • Naming notation is *instance.subinterface*, and a period between arguments is required as part of the notation. • You must include the **l2transport** keyword in the command string; otherwise, the configuration creates a Layer 3 subinterface rather that an AC.
Step 3	**encapsulation dot1q** *vlan-id* **Example:** `RP/0/RP0/CPU0:router(config-subif)# encapsulation dot1q 100`	Sets the Layer 2 encapsulation of an interface.

	Command or Action	Purpose

Step 4	**end** or **commit**
	Example:
	`RP/0/RP0/CPU0:router(config-if-l2)# end`
	or
	`RP/0/RP0/CPU0:router(config-if-l2)# commit`
		Saves configuration changes.
		• 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.

Step 5	**show interfaces** [**HundredGigE** | **TenGigE**] *interface-path-id.subinterface* **Example:** `RP/0/RP0/CPU0:router# show interfaces TenGigE 0/3/0/0.1`	(Optional) Displays statistics for interfaces on the router.

Removing an 802.1Q VLAN Subinterface

This task explains how to remove 802.1Q VLAN subinterfaces that have been previously configured using the Configuring 802.1Q VLAN subinterfaces section in this module.

```
RP/0/RP0/CPU0:router# configure
```

/* Remove the subinterface, which also automatically deletes all the configuration applied to the subinterface. */

```
RP/0/RP0/CPU0:router(config)# no interface TenGigE 0/2/0/4.10
```

- Replace the *instance* argument with one of the following instances:
- Physical Ethernet interface instance, or with an Ethernet bundle instance. Naming notation is *rack*/*slot*/*module*/*port*, and a slash between values is required as part of the notation.
- Ethernet bundle instance. Range is from 1 through 65535.
- Replace the *subinterface* argument with the subinterface value. Range is from 0 through 4095.

Naming notation is *instance.subinterface*, and a period between arguments is required as part of the notation.

Note Repeat to remove other VLAN subinterfaces.

```
RP/0/RP0/CPU0:router(config)# end
or
RP/0/RP0/CPU0:router(config)# 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.

Configuration Examples for VLAN Interfaces

This section contains the following example:

VLAN Subinterfaces: Example

The following example shows how to create three VLAN subinterfaces at one time:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface TenGigE 0/2/0/4.1

RP/0/RP0/CPU0:router(config-subif)# encapsulation dot1q 100
RP/0/RP0/CPU0:router(config-subif)# ipv4 address 10.0.10.1/24
RP/0/RP0/CPU0:router(config-subif)# interface TenGigE0/2/0/4.2

RP/0/RP0/CPU0:router(config-subif)# encapsulation dot1q 101
RP/0/RP0/CPU0:router(config-subif)# ipv4 address 10.0.20.1/24
RP/0/RP0/CPU0:router(config-subif)# interface TenGigE0/2/0/4.3

RP/0/RP0/CPU0:router(config-subif)# encapsulation dot1q 102
RP/0/RP0/CPU0:router(config-subif)# ipv4 address 10.0.30.1/24
RP/0/RP0/CPU0:router(config-subif)# commit
RP/0/RP0/CPU0:router(config-subif)# exit
RP/0/RP0/CPU0:router(config)# exit
```

```
RP/0/RP0/CPU0:router# show ethernet trunk bundle-Ether 1
Trunk                               Sub types        Sub states
VLAN trunks: 1,
  1 are 802.1Q (Ether)
Sub-interfaces: 3,
  3 are up.
802.1Q VLANs: 3,
  3 have VLAN Ids,

RP/0/RP0/CPU0:router# show vlan interfaceInterface     St Ly    MTU    Subs    L3
   Up    Down  Ad-Down
Te0/2/0/4.1         802.1Q             10  up
Te0/2/0/4.2         802.1Q             20  up
Te0/2/0/4.3         802.1Q             30  up
RP/0/RP0/CPU0:router# show vlan trunks briefBE1           Up L3   1514    1000
  0    1000     1000       0       0

Summary                      1000        0    1000     1000       0       0

Te0/2/0/4           802.1Q (Ether)        up
```

The following example shows how to create two VLAN subinterfaces on an Ethernet bundle:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface bundle-ether 2
RP/0/RP0/CPU0:router(config-if)# ipv4 address 192.168.2.1/24
RP/0/RP0/CPU0:router(config-if)# exit
RP/0/RP0/CPU0:router(config)# interface bundle-ether 2.1

RP/0/RP0/CPU0:router(config-subif)# encapsulation dot1q 100
RP/0/RP0/CPU0:router(config-subif)# ipv4 address 192.168.100.1/24
RP/0/RP0/CPU0:router(config-subif)# exit
RP/0/RP0/CPU0:router(config)# interface bundle-ether 2.2

RP/0/RP0/CPU0:router(config-subif)# encapsulation dot1q 200
RP/0/RP0/CPU0:router(config-subif)# ipv4 address 192.168.200.1/24
RP/0/RP0/CPU0:router(config-subif)# exit
RP/0/RP0/CPU0:router(config)# commit
```

Layer 2 interface VLAN encapsulation using VLAN ranges and lists

Table 3: Feature History Table

Feature Name	Release Information	Feature Description

Layer 2 interface VLAN encapsulation using VLAN ranges and lists	Release 24.4.1	Introduced in this release on: Fixed Systems (8200, 8700)(select variants only*); Centralized Systems (8600); Modular Systems (8800 [LC ASIC: Q200, P100])(select variants only*) You can now leverage the VLAN ranges and lists to effectively separate networks operating over shared links and devices. VLAN encapsulation is typically determined by the access network and customer edge (CE) device, limiting the network provider's control over the VLAN tag or Ethernet type of customer traffic. The VLAN ranges and lists support various customer traffic types, enhancing network flexibility and management. *This feature is supported on: • 8201-32FH • 8201-24H8FH • 8202-32FH-M • 8212-48FH-M • 8608 • 8711-32FH-M • 8712-MOD-M • 88-LC0-34H14FH • 88-LC0-36FH • 88-LC0-36FH-M • 88-LC1-36EH • 88-LC1-52Y8H-EM • 88-LC1-12TH24FH-E This feature modifies these changes: **CLI:** • **encapsulation dot1ad** • **encapsulation dot1ad dot1q** • **encapsulation dot1q** • **encapsulation dot1q second-dot1q**

VLAN encapsulation is a networking technique that:

- embeds VLAN tags within Ethernet frames to distinguish and identify network traffic

- relies on unique VLAN IDs, organized through ranges and lists, to manage and distinguish different VLANs, and
- ensures logical separation of traffic types to enhance network performance, security, and manageability.

VLAN ranges and lists

VLAN ranges and lists differ in their approach to VLAN encapsulation:

- A VLAN range is a sequence of contiguous VLAN IDs used for encapsulation. For instance, VLAN IDs from 200 to 250 can be grouped in a range.
- A VLAN list is a specific enumeration of VLAN IDs used for encapsulation. For instance, VLAN IDs 100, 123, and 150 can be grouped in a list.

VLAN list and range can be used together in configuration.

Benefits of Layer 2 interface VLAN encapsulation using VLAN ranges and lists

The VLAN encapsulation using VLAN ranges and lists offers several key benefits and is crucial for efficient network management:

- Simplified configuration: Configuring multiple VLANs with a single command, rather than individually, reduces the time and effort required for network setup and modification.
- Enhanced manageability: Managing VLANs becomes easier when grouped into lists or ranges. Changes can be applied to multiple VLANs simultaneously, simplifying network management tasks.
- Improved scalability: Grouping VLANs supports more efficient scaling in large-scale network environments, enabling the network to handle more VLANs without significantly increasing configuration complexity. A single subinterface can be used instead of multiple subinterfaces.
- Reduced configuration errors: Using fewer commands to configure multiple VLANs decreases the likelihood of configuration errors, resulting in a more stable and reliable network setup.

Restrictions of Layer 2 interface VLAN encapsulation using VLAN ranges and lists

These are the restrictions of Layer 2 interface VLAN encapsulation using VLAN ranges and lists:

- Supported only on Q200, P100, and K100 line cards and systems.
- Double-tagged encapsulations support the inner or outer tag as a list or range of VLAN IDs, but not both. For example:
    - encapsulation dot1ad 100-200 dot1q 300 (supported)
    - encapsulation dot1ad 100 dot1q 200,300 (supported)
    - encapsulation dot1ad 10-20 dot1q 30-40 (not supported)
- Supported only on Layer 2 subinterfaces.
- The maximum number of list entries, individual or range per encapsulation, is 64.
- Ranges of up to 4094 VLAN IDs are supported.
- For lists with more than ten list items, use **encapsulation list-extended** command.

For example, **encapsulation list-extended dot1q <x1>, … , <x10>**

Supported encapsulations

- encapsulation dot1q <x> , <y> (For example, encapsulation dot1q 200, 210, 300, 310, 400)
- encapsulation dot1q <x> - <y> (For example, encapsulation dot1q 200 – 400, 600-801, 901-1000)
- encapsulation dot1q <x> second-dot1q <y>, <z>
- encapsulation dot1q <x> second-dot1q <y> - <z>
- encapsulation dot1q <x> - <y> second-dot1q <z>
- encapsulation dot1ad <x> dot1q <y>, <z>
- encapsulation dot1ad <x> dot1q <y> - <z>
- encapsulation dot1ad <x> - <y> dot1q <z>
- encapsulation dot1ad <x>, <y> dot1q <z>
- encapsulation dot1ad <x>, <y>
- encapsulation dot1ad <x> - <y>

Configure a VLAN range and a VLAN list on Layer 2 subinterfaces

Define the matching criteria to map VLAN-tagged frames to the appropriate service instance.

Procedure

Step 1 Configure a VLAN range and a VLAN list on Layer 2 subinterfaces using the appropriate encapsulation method based on your requirements.

- If you want to configure a service with a list of noncontiguous customer VLAN (C-VLAN) ranges:

```
Router# configure
Router(config)# interface HundredGigE0/3/0/0.1 l2transport
Router(config-subif)# encapsulation dot1ad 200-400,600-801
Router(config-subif)# commit
```

- If you want to configure a service with a list of C-VLANs:

```
Router# configure
Router(config)# interface HundredGigE0/2/0/0.1 l2transport
Router(config-subif)# encapsulation dot1q 200,210,300,310,400
Router(config-subif)# commit
```

- If you want to configure a service with a range of service provider VLANs (S-VLANs) and a single inner C-VLAN:

```
Router# configure
Router(config)# interface HundredGigE0/4/0/0.1 l2transport
Router(config-subif)# encapsulation dot1ad 100-200 dot1q 300
Router(config-subif)# commit
```

- If you want to configure a QinQ service with a single S-VLAN and a list of C-VLANs:

```
Router# configure
Router(config)# interface HundredGigE0/5/0/0.1 l2transport
Router(config-subif)# encapsulation dot1q 100 second-dot1q 200,300
Router(config-subif)# commit
```

Step 2 Execute the **show interfaces** command to verify the VLAN range and VLAN list on the Layer 2 subinterface.

The corresponding value of the outer match appears depending on the VLAN configuration. The *Outer Match* field corresponds to the Dot1ad VLAN range and list in this example.

```
Router# show interfaces HundredGigE0/3/0/0.1
HundredGigE0/3/0/0.1 is UP, line protocol is UP
Interface state transitions: 0
Hardware is VLAN sub-interface(s), address is dc05.39c7.9440
Layer 2 Transport Mode
MTU 1518 bytes, BW 100000000 Kbit (Max: 100000000 Kbit)
reliability Unknown, txload Unknown, rxload Unknown
Encapsulation 802.1ad Virtual LAN,
Outer Match: Dot1ad VLAN 200-400,600-801
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
```

In this example, the *Outer Match* and *Inner Match* fields correspond to the Dot1ad VLAN range 100-200 and VLAN list Dot1Q VLAN 300.

```
Router# show interfaces HundredGigE0/4/0/0.1

HundredGigE0/4/0/0.1 is UP, line protocol is UP
Interface state transitions: 0
Hardware is VLAN sub-interface(s), address is dc05.39c7.9440
Layer 2 Transport Mode
MTU 1522 bytes, BW 100000000 Kbit (Max: 100000000 Kbit)
reliability Unknown, txload Unknown, rxload Unknown
Encapsulation 802.1ad-802.1Q Virtual LAN,
Outer Match: Dot1ad VLAN 100-200
Inner Match: Dot1Q VLAN 300
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
```