Configuring Subscriber Features

Subscriber features that are configured on BNG enable service providers to deploy certain specific functionalities like restricting the use of certain network resources, allowing Law Enforcement Agencies (LEAs) to conduct electronic surveillance, providing multicast services to the subscriber, and so on.

Table 1: Feature History for Configuring Subscriber Features

Release	Modification
Release 6.0.1	Added activating IPv6 router advertisement on an IPv4 subscriber interface enhancements
Release 6.0.1	Added Linking to Subscriber Traffic in a Shared Policy Instance Group feature
Release 6.2.1	These new features were introduced: • IGMP QoS Correlation for IPoE Subscribers • SNMP Lawful Intercept Using Circuit-Id • Controlling Subscriber Plans Using Protocol Options

The subscriber features covered in this chapter are:

• Additional References, on page 60

Excessive Punt Flow Trap

The Excessive Punt Flow Trap feature attempts to identify and mitigate control packet traffic from remote devices that send more than their allocated share of control packet traffic. A remote device can be a subscriber device, a device on a VLAN interface, or a device identified by its source MAC address.

When remote devices send control packet traffic to the router, the control packets are punted and policed by a local packet transport service (LPTS) queue to protect the router's CPU. If one device sends an excessive rate of control packet traffic, the policer queue fills up, causing many packets to be dropped. If the rate from one "bad actor" device greatly exceeds that of other devices, most of the other devices do not get any of their control packets through to the router. The Excessive Punt Flow Trap feature addresses this situation.

Note Even when the Excessive Punt Flow Trap feature is not enabled, the "bad actors" can affect services for only other devices; they cannot bring down the router.

The Excessive Punt Flow Trap feature is supported on both subscriber interfaces, and non-subscriber interfaces such as L2 and L3 VLAN sub-interfaces and bundle virtual interfaces (BVIs). If the source that floods the punt queue with packets is a device with an interface handle, then all punts from that bad actor interface are penalty policed. The default penalty rate, for each protocol, is 10 protocols per second (pps). Otherwise, if the source is a device that does not have an interface handle, then all packets from this bad actor are dropped.

Note In the 4.2.x releases, the Excessive Punt Flow Trap feature was called as "Subscriber Control Plane Policing (CoPP)" that only operated on subscriber interfaces.

Functioning of Excessive Punt Flow Trap Feature

The Excessive Punt Flow Trap feature monitors control packet traffic arriving from physical interfaces, sub-interfaces, BVI, and subscriber interfaces. It divides interfaces into two categories:

- "Parent" interfaces, which can have other interfaces under them.
- "Non-parent" interfaces, which have no interfaces under them.

A physical interface is always a parent interface because it has VLAN sub-interfaces. An L3 VLAN sub-interface can either be a parent or a non-parent interface. If the VLAN sub-interface is enabled for subscribers, then it is a parent interface, otherwise it is a non-parent interface. A subscriber interface (IPoE or PPPoE) is always a non-parent interface.

When a flow is trapped, the Excessive Punt Flow Trap feature tries to identify the source of the flow. The first thing it determines is from which interface the flow came. If this interface is not a "parent" interface, then the feature assumes that it is the end-point source of the flow and penalty policing is applied. The software applies a penalty-policer in the case of a BVI interface also. If the trapped interface is a "parent" interface, then instead of penalizing the entire interface (which would penalize all the interfaces under it), this feature takes the source MAC address of the bad flow and drops all packets from the MAC address under the parent. Due to platform limitation, the penalty policer cannot be applied on a MAC address; therefore all packets are dropped.

For more information about enabling the Excessive Punt Flow Trap feature, see Enabling Excessive Punt Flow Trap Processing, on page 6.

Note The Excessive Punt Flow Trap feature monitors all punt traffic. There is no way to remove a particular interface from the initial monitoring, nor can an interface be prevented from being flagged as bad if it is the source of excessive flows.

Bad actors are policed for each protocol. The protocols that are supported by the Excessive Punt Flow Trap feature are Broadcast, Multicast, ARP, DHCP, PPP, PPPoE, ICMP, IGMP, L2TP and IP (covers many types of L3 based punts, both IPv4 and IPv6). Each protocol has a static punt rate and a penalty rate. For example, the sum total of all ICMP punts from remote devices is policed at 1500 packets per second (pps) to the router's CPU. If one remote device sends an excessive rate of ICMP traffic and is trapped, then ICMP traffic from that bad actor is policed at 10 pps. The remaining (non-bad) remote devices continue to use the static 1500 pps queue for ICMP.

Note The excessive rate required to cause an interface to get trapped has nothing to do with the static punt rate (e.g. 1500 pps for ICMP). The excessive rate is a rate that is significantly higher than the current average rate of other control packets being punted. The excessive rate is not a fixed rate, and is dependent on the current overall punt packet activity.

Once a bad actor is trapped, it is penalty policed on all its punted protocols (ARP, DHCP, PPP, etc.), irrespective of the protocol that caused it to be identified as a bad actor. A penalty rate of 10 pps is sufficient to allow the other protocols to function normally. However, if the bad actor is trapped by source MAC address, then all its packets are dropped.

When an interface is trapped, it is placed in a "penalty box" for a period of time (a default of 15 minutes). At the end of the penalty timeout, it is removed from penalty policing (or dropping). If there is still an excessive rate of control packet traffic coming from the remote device, then the interface is trapped again.

Restrictions

These restrictions apply to implementing Excessive Punt Flow Trap feature:

- The A9K-8x100G-LB-SE and A9K-8x100G-LB-TR line cards do not support BNG subscriber interfaces.
- This feature does not support interfaces on SIP-700 line cards and ASR 9000 Ethernet Line Card.
- This feature is non-deterministic. In some cases, the Excessive Punt Flow Trap feature can give a false positive, i.e. it could trap an interface that is sending legitimate punt traffic.
- The Excessive Punt Flow Trap feature traps flows based on the relative rate of different flows; thus, the behavior depends on the ambient punt rates. A flow that is significantly higher than other flows could be trapped as a bad actor. Thus the feature is less sensitive when there are many flows, and more sensitive when there are fewer flows present.
- Sometimes control packet traffic can occur in bursts. The Excessive Punt Flow Trap has safeguards against triggering on short bursts, but longer bursts could trigger a false positive trap.

MAC-based EPFT on Non-subscriber Interface

This feature supports dropping of the excessive punt packets from a bad actor flow, based on the source MAC address. Before this release, EPFT on non-subscriber interfaces was only performed based on the *ifhandle* (interface handle) of the VLAN sub-interface, wherein all the ingress punt packets on the VLAN sub-interface are penalty policed, irrespective of their source MAC addresses.

In an aggregation scenario, packets may come from multiple source MAC addresses to a VLAN sub-interface. If one particular source MAC sends excessive punt packets, it drains the punt queue; punt packets of other source MAC addresses on that non-subscriber interface may get dropped. MAC-based EPFT on the non-subscriber interface feature performs EPFT (that is, it drops the packets) based on a source MAC address, if the flow is a bad actor flow sending excessive punt packets.

To enable MAC-based EPFT on non-subscriber interface, you must use this command in global configuration mode:

lpts punt excessive-flow-trap non-subscriber-interfaces [**mac**]

Note If the **mac** option is not configured, the default behavior is to perform EPFT, based on the *ifhandle* of the non-subscriber interface.

Tunable Sampler Parameters for Control Plane Policing

This feature allows configuring various EPFT sampler parameters to fine-tune the Elephant Trap algorithm, to achieve the best behavior for realistic traffic streams, and to reduce situations like false positives to a great extent. Before this release, these parameter values were fixed and read from a configuration file.

The commands available for this feature are privileged (Cisco-support) commands.

This table lists configurable EPFT sampler parameters:

EPFT Sampler Parameter	Description
Elephant Trap size	The maximum number of flows that is concurrently stored in Elephant Trap. The range is from 1 to 128; default is 64. The value must be a power of 2, that is 1, 2, 4,8, 16, 32, 64 and 128 are the valid values.
Sampling probability	Sampling probability of Elephant Trap; that is, the probability value to sample any particular packet and feed it into the trap. This is a floating point number ranging from 0 to 1 enclosed in double quotes (""). By default, the value is "0.01", which means that 1 out of 100 packets is randomly picked for sampling.
Report threshold	Threshold at which a flow is reported as a bad actor. The range is from 1 to 65535; default is 5.
Eviction threshold	Threshold below which a flow can be evicted from the Elephant Trap. The range is from 1 to 65535; default is 2.

EPFT Sampler Parameter	Description
Eviction search limit	Maximum number of entries to check before cancelling an eviction search. The range is from 1 to 128; default is 64. Eviction search limit must not be more than the Elephant Trap size.
Maximum flow gap	The maximum time, in milliseconds, that the Elephant Trap allows between successive samples while incrementing the hit counter. The range is from 1 to 60000; default is 800.

False Positive Suppression

Due to the probabilistic nature of the Elephant Trap algorithm, there is possibility of good flows being trapped as bad flows. This probability is more in scenarios where the number of flows is less. Such false positives can be suppressed using these features:

- **Support of tunable sampler parameters for control plane policing**

  For details, see Tunable Sampler Parameters for Control Plane Policing, on page 4.

- **False positive suppression through dampening**

  This feature allows trapping only repeated bad actor flows. The Flowtrap process maintains a trap similar to the Elephant Trap that stores information about each flow for which the bad actor notification is received by the sampler process. The bad actor notifications for penalty policing the flow, or dropping the packets from the flow, is carried out only if the notification is received twice within a specified time (a configurable time in seconds). Although it extends the duration before which a true bad actor is throttled, it also reduces false positives.

  By default, the dampening feature is disabled. To enable this feature, you must use this command in global configuration mode:

  lpts punt excessive-flow-trap dampening [*time*]

  The range of *time* (in milliseconds) is from 1 to 60000. If the *time* option is not used after the **dampening** keyword, a default time value of 30 is used.

EPFT Support for Packet-Triggered Sessions

Before Cisco IOS XR Software Release 5.3.0, punt packets on a packet-triggered subscriber-interface and on a packet-triggered access-interface were policed as per the LPTS rates. The policing rate earlier was high (2000 packets per second) and system wide. With EPFT support for packet triggered sessions, punt packets on packet-triggered interfaces (subscriber and access) go through EPFT node. If identified as bad actor flows, they are penalty-policed according to the EPFT penalty rates (only 20 to 200 packets per second). This is the default behavior from Cisco IOS XR Software Release 5.3.0 and later.

This feature is enabled by default (users need not explicitly configure any command to enable this feature). However, you can use these commands to set the **penalty-rate** and **penalty-timeout** for punt packets of **unclassified-source** type:

lpts punt excessive-flow-trap penalty-rate unclassified *rate*

The range of rate (in pps - packets per second) is from 2 to 100, the default is 10.

lpts punt excessive-flow-trap penalty-timeout unclassified *timeout*

The range of timeout (in minutes) is from 1 to 1000, the default is 15.

Interface-based Flow

For the Elephant Trap sampler, the MAC address is one of the key fields used to uniquely identify a flow. Certain cases of DoS attacks have dynamically changing source MAC addresses. An individual flow does not cross the threshold in such cases, and hence the EPFT does not trap the flow. With the interface-based flow feature, Elephant Trap does not consider MAC addresses as a key for uniquely identifying a flow. Hence, all packets received on a non-subscriber interface (irrespective of the source MAC address) are considered to be a part of a single flow. When excessive punts are received on the interface, EPFT does *ifhandle*-based trap, thereby penalty policing the punt traffic on that particular interface.

To enable interface-based flow, you must use this command in global configuration mode:

lpts punt excessive-flow-trap interface-based-flow

Note You cannot enable this command if EPFT is turned on for the subscriber-interfaces and non-subscriber-interfaces MAC, or vice versa. This is because interface-based flow feature is mutually exclusive with MAC-based EPFT on non-subscriber interface feature.

Enabling Excessive Punt Flow Trap Processing

Perform this task to enable the Excessive Punt Flow Trap feature for both subscriber and non-subscriber interfaces. The task also enables you to set the penalty policing rate and penalty timeout for a protocol.

SUMMARY STEPS

1. **configure**
2. **lpts punt excessive-flow-trap subscriber-interfaces**
3. **lpts punt excessive-flow-trap non-subscriber-interfaces**
4. **lpts punt excessive-flow-trap penalty-rate** *protocol penalty_policer_rate*
5. **lpts punt excessive-flow-trap penalty-timeout** *protocol time*
6. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**lpts punt excessive-flow-trap subscriber-interfaces** **Example:** `RP/0/RSP0/CPU0:router(config)# lpts punt excessive-flow-trap subscriber-interfaces`	Enables the Excessive Punt Flow Trap feature on subscriber interfaces.
Step 3	**lpts punt excessive-flow-trap non-subscriber-interfaces**	Enables the Excessive Punt Flow Trap feature on non-subscriber interfaces.

	Command or Action	Purpose
	Example: `RP/0/RSP0/CPU0:router(config)# lpts punt excessive-flow-trap non-subscriber-interfaces`	**Note** If both Step 2 and Step 3 configurations are applied, the Excessive Punt Flow Trap feature is enabled for all interfaces.
Step 4	**lpts punt excessive-flow-trap penalty-rate** *protocol penalty_policer_rate* **Example:** `RP/0/RSP0/CPU0:router(config)# lpts punt excessive-flow-trap penalty-rate icmp 10`	Sets the penalty policing rate for a protocol. The penalty policer rate is in packets-per-second (pps) and ranges from 2 to 100. **Note** The penalty policing rate for a protocol consumes a policer rate profile.
Step 5	**lpts punt excessive-flow-trap penalty-timeout** *protocol time* **Example:** `RP/0/RSP0/CPU0:router(config)# lpts punt excessive-flow-trap penalty-timeout igmp 10`	Sets the penalty timeout value, which is a period of time that the interface trapped is placed in the penalty box, for a protocol. The penalty timeout value is in minutes and ranges from 1 to 1000. The default penalty timeout value is 15 minutes.
Step 6	**commit**	

Enabling Excessive Punt Flow Trap Processing: Examples

This is an example for enabling the Excessive Punt Flow Trap for subscriber interfaces, using the default penalty timeout (15 minutes) and setting a penalty rate of 20 pps for PPP and PPPoE protocols.

```
configure
lpts punt excessive-flow-trap subscriber-interfaces
lpts punt excessive-flow-trap penalty-rate ppp 20
lpts punt excessive-flow-trap penalty-rate pppoe 20
end
!!
```

This is an example for enabling the Excessive Punt Flow Trap for non-subscriber interfaces, using the default penalty rate (10 pps) and setting the ARP penalty timeout to 2 minutes.

```
configure
lpts punt excessive-flow-trap non-subscriber-interfaces
lpts punt excessive-flow-trap penalty-timeout arp 2
end
!!
```

Access Control List and Access Control List-based Forwarding

An Access Control List (ACL) is used to define access rights for a subscriber. It is also used for filtering content, blocking access to various network resources, and so on.

Certain service providers need to route certain traffic be routed through specific paths, instead of using the path computed by routing protocols. For example, a service provider may require that voice traffic traverse through certain expensive routes, but data traffic to use the regular routing path. This is achieved by specifying

the next-hop address in the ACL configuration, which is then used for forwarding packet towards its destination. This feature of using ACL for packet forwarding is called ACL-based Forwarding (ABF).

The ACL is defined through CLI or XML; however, it can be applied to a subscriber session either through a dynamic-template, or through VSAs from RADIUS. Deploying ABF (using ACL) involves these stages:

- Defining an ACL, see Configuring Access-Control Lists, on page 8.
- Applying the ACL to an access-interface, see Activating ACL, on page 9.

Configuring Access-Control Lists

Perform this task to create an access control list. As an example, this access list is created to deploy ABF; therefore, it defines the next hop address.

SUMMARY STEPS

1. **configure**
2. {**ipv4** | **ipv6**} **access-list** *access-list-name*
3. *sequence-number* **permit tcp any any**
4. *sequence-number* **permit** {**ipv4** | **ipv6**} **host** *source_address* **nexthop** *source_address destination_address*
5. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

Step 2	{**ipv4** | **ipv6**} **access-list** *access-list-name*
	Example:
	`RP/0/RSP0/CPU0:router(config)# ipv4 access-list foo_in`
	or
	`RP/0/RSP0/CPU0:router(config)# ipv6 access-list foo_in`
		Configures the access-list.

Step 3	*sequence-number* **permit tcp any any** **Example:** `RP/0/RSP0/CPU0:router(config)# 10 permit tcp any any`	Enters an access control list rule to tcp traffic.
Step 4	*sequence-number* **permit** {**ipv4** | **ipv6**} **host** *source_address* **nexthop** *source_address destination_address* **Example:**	Specifies packets to forward on ipv4 protocol from source IP address to destination IP address. **Note** Repeat steps 1 to 4 to configure the foo_out access-list.

	Command or Action	Purpose
	RP/0/RSP0/CPU0:router(config)# 10 permit ipv4 host 9.8.8.9 nexthop 6.6.6.6 7.7.7.7 or RP/0/RSP0/CPU0:router(config)# 10 permit ipv6 host 192:2:1:9 nexthop 192:2:6:8	
Step 5	**commit**	

Configuring Access-Control Lists: Examples

```
//For IPv4
configure
ipv4 access-list foo_in
10 permit tcp any any
10 permit ipv4 host 9.8.8.9 nexthop 6.6.6.6 7.7.7.7
!
!
end

//For IPv6
configure
ipv6 access-list foo_in
10 permit tcp any any
10 permit ipv4 host 192:2:1:9 nexthop 192:2:6:8
!
!
end
```

Activating ACL

Perform this task to define a dynamic-template that is used to activate an access-control list.

SUMMARY STEPS

1. **configure**
2. **dynamic-template**
3. **type**{**ipsubscriber** | **ppp** | **service**} *dynamic-template-name*
4. {**ipv4** | **ipv6**} **access-group** *access-list-name* **ingress**
5. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

	Command or Action	Purpose
Step 2	**dynamic-template** **Example:** `RP/0/RSP0/CPU0:router(config)# dynamic-template`	Enters the dynamic-template configuration mode.
Step 3	**type** { **ipsubscriber** | **ppp** | **service** } *dynamic-template-name* **Example:** `RP/0/RSP0/CPU0:router(config-dynamic-template)# type service foo`	Creates a service dynamic-template type.

Step 4	{ **ipv4** | **ipv6** } **access-group** *access-list-name* **ingress**
	Example:
	`RP/0/RSP0/CPU0:router(config-dynamic-template-type)# ipv4 access-group foo_in ingress`
	or
	`RP/0/RSP0/CPU0:router(config-dynamic-template-type)# ipv6 access-group foo_in ingress`
		Specifies access-control for the incoming packets.
		Note Similarly, create another access-group for the outgoing packets called foo_out.

Step 5	**commit**	

Activating ACL: Examples

```
//For IPv4
configure
dynamic-template
type service foo
ipv4 access-group foo_in ingress
!
!
end

//For IPv6
configure
dynamic-template
type service foo
ipv6 access-group foo_in ingress
!
!
end
```

Support for Lawful Intercept

Lawful Intercept allows Law Enforcement Agencies (LEAs) to conduct electronic surveillance as authorized by judicial or administrative order. Increasingly, legislation is being adopted and regulations are being enforced

that require service providers (SPs) and Internet service providers (ISPs) to implement their networks to explicitly support authorized electronic surveillance. The types of SPs or ISPs that are subject to Lawful Intercept mandates vary greatly from country to country. Lawful Intercept compliance in the United States is specified by the Communications Assistance for Law Enforcement Act (CALEA).

Cisco ASR 9000 Series Router supports the Cisco Service Independent Intercept (SII) architecture and PacketCable™[1] Lawful Intercept architecture. The Lawful Intercept components by themselves do not ensure customer compliance with applicable regulations but rather provide tools that can be used by SPs and ISPs to construct an Lawful Intercept compliant network.

BNG supports the Per-session Lawful Intercept and Radius-based Lawful Intercept for subscribers. Both, per-session and radius-based lawful intercepts are executed on IPoE, PPPoE, and PPPoE LAC subscriber sessions in BNG.

Caution This guide does not address legal obligations for the implementation of lawful intercept. Service providers are responsible for ensuring that network complies with applicable lawful intercept statutes and regulations. It is recommended that legal advice be sought to determine obligations.

Note By default, Lawful Intercept is not a part of the Cisco IOS XR software. To enable Lawful Intercept, you must install and activate the **asr9k-li-px.pie**.

For more information about Lawful Intercept-related router configuration, see *Implementing Lawful Intercept* chapter in *Cisco ASR 9000 Series Aggregation Services Router System Security Configuration Guide*.

Per-session Lawful Intercept

Lawful interception of all Layer 2 or Layer 3 traffic on a specified subscriber interface, on both ingress as well egress directions, and sending the replicated stream to mediation device, is called the per-session Lawful Intercept. This Lawful Intercept implements IPv4, IPv6, and multicast traffic interception using the Cisco-defined MIBs. By default, the SNMP-based Lawful Intercept feature is enabled on the Cisco ASR 9000 Series Router, which allows you to configure the taps. For more information about disabling SNMP-based Lawful Intercept, see Disabling SNMP-based Lawful Intercept, on page 12.

The subscriber session is identified by Account-session-ID, which acts as a key in identifying the specified subscriber interface for the subscriber user, whose traffic is getting intercepted.

Lawful Intercept, in general, can be implemented using either SII architecture or PacketCable™ specifications. The Cisco IOS-XR implementation of SNMP-based Lawful Intercept is based on service-independent intercept (SII) architecture. SNMPv3 authenticates data origin and ensures that the connection from Cisco ASR 9000 Series Router to the mediation device is secure. This ensures that unauthorized parties cannot forge an intercept target.

[1] PacketCable™ architecture addresses device interoperability and product compliance issues using the PacketCable™ Specifications.

Note To implement lawful intercept, you must understand how the SNMP server functions. For this reason, carefully review the information described in the module *Implementing SNMP* in *System Management Configuration Guide for Cisco ASR 9000 Series Routers*.

Lawful intercept must be explicitly disabled. It is automatically enabled on a provisioned router after installing and activating the **asr9k-li-px.pie**. However, you should not disable LI if there is an active tap in progress, because this deletes the tap.

Management plane must be configured to enable SNMPv3. Allows the management plane to accept SNMP commands, so that the commands go to the interface (preferably, a loopback) on the router. This allows the mediation device (MD) to communicate with a physical interface. For more information about Management Plane Protection feature, see Configuring the Inband Management Plane Protection Feature, on page 13 and for more information about enabling the mediation device, see Enabling the Mediation Device to Intercept VoIP and Data Sessions, on page 13.

Lawful Intercept MIBs

An external mediation device also known as collectors can create IPv4 or IPv6 address based TAPs using IP-TAP-MIB. The SNMPv3 protocol is used to provision the mediation device (defined by CISCO-TAP2-MIB) and the Taps(defined by CISCO-USER-CONNECTION-TAP-MIB). The Cisco ASR 9000 Series Router supports a total of 511 concurrent taps that includes both SNMP and Radius.

Lawful intercept uses these MIBs for interception:

- CISCO-TAP2-MIB—Used for lawful intercept processing. It contains SNMP management objects that control lawful intercepts on a Cisco ASR 9000 Series Router. The mediation device uses the MIB to configure and run lawful intercepts on targets sending traffic through the Cisco ASR 9000 Series Router. The CISCO-TAP2-MIB supports the SII feature and defines the provisioning of the mediation devices and generic Taps. It primarily consists of the mediation device table and a stream table. The mediation device table contains information about mediation devices with which the Cisco ASR 9000 Series Router communicates; for example, the device's address, the interfaces to send intercepted traffic over, and the protocol to use to transmit the intercepted traffic. The stream table contains a list of generic Taps that are provisioned by the MD table entries.
- CISCO-USER-CONNECTION-TAP-MIB—Used for intercepting traffic for individual subscribers. The MIB contains SNMP management objects to configure and execute wiretaps on individual user connections on the Cisco ASR 9000 Series Router. This MIB contains information about the user connections, each identified by a unique session ID. The CISCO-USER-CONNECTION-TAP-MIB cannot be configured without configuring the CISCO-TAP2-MIB.

Note It is not possible to configure an SNMP tap and a Radius tap at the same time. Also, the same session cannot be tapped more than once at a time.

Disabling SNMP-based Lawful Intercept

Lawful Intercept is enabled by default on the Cisco ASR 9000 Series Router after installing and activating the **asr9k-li-px.pie**.

- To disable Lawful Intercept, enter the **lawful-intercept disable** command in global configuration mode.

- To re-enable it, use the **no** form of this command.

Disabling SNMP-based Lawful Intercept: An example

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# lawful-intercept disable
```

Note The **lawful-intercept disable** command is available only after installing and activating the **asr9k-li-px.pie**.

All SNMP-based taps are dropped when lawful intercept is disabled.

Configuring the Inband Management Plane Protection Feature

If MPP was not earlier configured to work with another protocol, then ensure that the MPP feature is also not configured to enable the SNMP server to communicate with the mediation device for lawful interception. In such cases, MPP must be configured specifically as an inband interface to allow SNMP commands to be accepted by the router, using a specified interface or all interfaces.

Note Ensure this task is performed, even if you have recently migrated to Cisco IOS XR Software from Cisco IOS, and you had MPP configured for a given protocol.

For lawful intercept, a loopback interface is often the choice for SNMP messages. If you choose this interface type, you must include it in your inband management configuration.

Enabling the Mediation Device to Intercept VoIP and Data Sessions

These SNMP server configuration tasks enable the Cisco SII feature on a router running Cisco IOS XR Software by allowing the MD to intercept VoIP or data sessions.

SUMMARY STEPS

1. **configure**
2. **snmp-server view** *view-name* **ciscoTap2MIB included**
3. **snmp-server view** *view-name* **ciscoUserConnectionTapMIB included**
4. **snmp-server group** *group-name* **v3auth read** *view-name* **write** *view-name* **notify** *view-name*
5. **snmp-server host** *ip-address* **traps version 3 auth** *username* **udp-port** *port-number*
6. **snmp-server user** *mduser-id groupname* **v3 auth md5** md-*password*
7. **commit**
8. show snmp users
9. show snmp group
10. show snmp view

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**snmp-server view** *view-name* **ciscoTap2MIB included** **Example:** `RP/0//CPU0:router(config)# snmp-server view TapName ciscoTap2MIB included`	Creates or modifies a view record and includes the CISCO-TAP2-MIB family in the view. The SNMP management objects in the CISCO-TAP2-MIB that controls lawful intercepts are included. This MIB is used by the mediation device to configure and run lawful intercepts on targets sending traffic through the router.
Step 3	**snmp-server view** *view-name* **ciscoUserConnectionTapMIB included** **Example:** `RP/0//CPU0:router(config)# snmp-server view TapName ciscoUserConnectionTapMIB included`	Creates or modifies a view record and includes the CISCO-USER-CONNECTION-TAP-MIB family, to manage the Cisco intercept feature for user connections. This MIB is used along with the CISCO-TAP2-MIB to intercept and filter user traffic.
Step 4	**snmp-server group** *group-name* **v3auth read** *view-name* **write** *view-name* **notify** *view-name* **Example:** `RP/0//CPU0:router(config)# snmp-server group TapGroup v3 auth read TapView write TapView notify TapView`	Configures a new SNMP group that maps SNMP users to SNMP views. This group must have read, write, and notify privileges for the SNMP view.
Step 5	**snmp-server host** *ip-address* **traps version 3 auth** *username* **udp-port** *port-number* **Example:** `RP/0//CPU0:router(config)# snmp-server host 223.255.254.224 traps version 3 auth bgreen udp-port 2555`	Specifies SNMP trap notifications, the version of SNMP to use, the security level of the notifications, and the recipient (host) of the notifications.
Step 6	**snmp-server user** *mduser-id groupname* **v3 auth md5** md-*password* **Example:** `RP/0//CPU0:router(config)# snmp-server mduser-id TapGroup v3 auth md5 mdpassword`	Configures the MD user as part of an SNMP group, using the v3 security model and the HMAC MD5 algorithm, which you associate with the MD password. • The *mduser-id* and *mdpassword* must match that configured on MD. Alternatively, these values must match those in use on the router. • Passwords must be eight characters or longer to comply with SNMPv3 security minimums. • Minimum Lawful Intercept security level is auth; The noauth option will not work, as it indicates noAuthnoPriv security level. The Lawful Intercept security level must also match that of the MD. • Choices other than MD5 are available on the router, but the MD values must match.

	Command or Action	Purpose
		Most MDs default to or support only MD5.
Step 7	**commit**	
Step 8	show snmp users **Example:** `RP/0//CPU0:router# show snmp users`	Displays information about each SNMP username in the SNMP user table.
Step 9	show snmp group **Example:** `RP/0//CPU0:router# show snmp group`	Displays information about each SNMP group on the network.
Step 10	show snmp view **Example:** `RP/0//CPU0:router# show snmp view`	Displays information about the configured views, including the associated MIB view family name, storage type, and status.

Enabling the Mediation Device to Intercept VoIP and Data Sessions: An example

```
configure
snmp-server view TapName ciscoTap2MIB included
snmp-server view TapName ciscoUserConnectionTapMIB included
snmp-server group TapGroup v3 auth read TapView write TapView notify TapView
snmp-server host 223.255.254.224 traps version 3 auth bgreen udp-port 2555
snmp-server mduser-id TapGroup v3 auth md5 mdpassword
end
!
!
```

Radius-based Lawful Intercept

Radius-based Lawful Intercept feature provides mechanisms for interception of the BNG subscriber traffic by using the RADIUS attributes. This is a preferred method over SNMP user-connection MIB, as SNMP-based method prevents a session to be tapped until an IP address has been assigned to the session. In the Radius-based LI mechanism, tapping is possible as soon as a session is established.

A RADIUS-based Lawful Intercept solution enables intercept requests to be sent (through Access-Accept packets or Change of Authorization (CoA)-Request packets) to the network access server (NAS) or to the Layer 2 Tunnel Protocol access concentrator (LAC) from the RADIUS server. All traffic data going to or from a PPP or L2TP session is passed to a mediation device. Another advantage of RADIUS-based Lawful Intercept solution is to set the tap with Access-Accept packets that allows all target traffic to be intercepted simultaneously.

The RADIUS-based Lawful Intercept feature provides tap initiation support for these modes:

- Access-Accept based Lawful Intercept for the new session
- CoA based Lawful Intercept for existing session

Note By default, the Radius-based Lawful Intercept functionality is not enabled. For more information about enabling Radius-based Lawful Intercept, see Enabling RADIUS-based Lawful Intercept, on page 16.

Enabling RADIUS-based Lawful Intercept

Perform this task to enable the Radius-based Lawful Intercept feature.

SUMMARY STEPS

1. **configure**
2. **aaa intercept**
3. **aaa server radius dynamic-author**
4. **port** *port_number*
5. **server-key** [**0**|**7**] *word*
6. **client** *hostname*{ **vrf** *vrf_name* | **server-key** [**0**|**7**] *word* }
7. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**aaa intercept** **Example:** `RP/0/RSP0/CPU0:router(config)# aaa intercept`	Enables the radius-based lawful intercept feature. **Note** This command is available only after installing and activating **asr9k-li-px.pie**. When you disable aaa intercept, all radius-based taps are removed from the Cisco ASR 9000 Series Router.
Step 3	**aaa server radius dynamic-author** **Example:** `RP/0/RSP0/CPU0:router(config)# aaa server radius dynamic-author`	Configures the lawful intercept as a AAA server and enters the dynamic authorization local server configuration mode.
Step 4	**port** *port_number* **Example:** `RP/0/RSP0/CPU0:router(config-Dynamic Author)# port 1600`	Specifies the RADIUS server port. The default port number is 1700.
Step 5	**server-key** [**0**|**7**] *word* **Example:** `RP/0/RSP0/CPU0:router(config-Dynamic Author)# server-key cisco`	Specifies the encryption key shared with the RADIUS client.
Step 6	**client** *hostname*{ **vrf** *vrf_name*	**server-key** [**0**|**7**] *word* }	Specifies the client with which the AAA server will be communicating.

	Command or Action	Purpose
	Example: `RP/0/RSP0/CPU0:router(config-Dynamic Author)# client 3.0.0.28 vrf default server-key cisco`	**Note** You can configure the server key in a global mode and also as a per client type key.
Step 7	**commit**	

Enabling RADIUS-based Lawful Intercept: An example

```
configure
aaa intercept
aaa server radius dynamic-author
port 1600
server-key cisco
client 3.0.0.28 vrf default server-key cisco
end
!
!
```

What to do next

These attributes need to be present in the user profile to configure the Radius-based Lawful Intercept.

```
xyz_user1@domain.com Password == "cisco"
    Cisco-avpair = "md-ip-addr=192.1.1.4",
    Cisco-avpair += "md-port=203",
    Cisco-avpair += "md-dscp=3",
    Cisco-avpair += "intercept-id=abcd0003",
    Cisco-avpair += "li-action=1"
```

TCP MSS Adjustment

The TCP MSS Adjustment feature allows the configuration of the maximum segment size (MSS) on transient packets that traverse a Cisco ASR 9000 Series Router.

When dealing with PPPoE or L2TP cases, an additional header that the client initiating a TCP session may not be aware of is added to the packet. This can result in lost packets, broken transmissions, or fragmentation when packet sizes exceed the maximum transmission units (MTUs) due to the added headers.

Here is a sample scenario that shows how the TCP MSS adjust feature works:

Figure 1: Sample TCP MSS Adjust

In this example, the HTTP client sends to the HTTP server a TCP synchronize (SYN) packet that signals an MSS value of 1300 (MTU) - 20 TCP - 20 IP header = 1260. On receiving it, the HTTP server acknowledges it with a SYN ACK message. The HTTP client confirms the TCP session with a single acknowledgment and opens up the TCP channel.

Note This is a sample scenario without PPPoE or L2TP.

When the HTTP server picks up a large file, it segments it into 1460 byte chunks (assuming that there are no http headers for now). When the HTTP server sends the packet, the first Cisco ASR 9000 Series Router (on the right) detects that the MTU is 576 downstream to the client and requires a 1300 byte packet to be fragmented.

If the server sets the DF ("don't fragment") bit, then the packet is dropped. And, if the packet does not have the DF bit set, then it gets fragmented, requiring the client to reassemble the packets. In digital subscriber line (DSL) or fibre-to-the-home (FTTH) like access, a CPE may block incoming fragments as a security mechanism, causing this transmission to be lost.

In a typical scenario, having packets that are dropped causes partial downloads, an obstruction, or a delay in displaying images in web pages. MSS adjust overcomes this scenario by intercepting the TCP SYN packet, reading the MSS option, and adjusting the value so that the server does not send packets larger than the configured size (plus headers).

Note that the TCP MSS value is only adjusted downward. If the clients request an MSS value lower than the configured value, then no action is taken.

In the case of PPPoE, an extra 8 bytes and in the case of L2TP, an extra 40 bytes is added to the packet. The recommended MSS adjust values are 1452 for PPPoE, and 1420 for L2TP scenarios, assuming a minimum MTU of 1500 end-to-end.

Separate unique global values for PTA and L2TP are supported, which once configured allows all future sessions to be TCP MSS adjustment; however, the sessions already established will not be TCP adjusted. If the global value is changed, then all new TCP subscriber sessions, will get the new global value.

For more information about configuring the TCP MSS value of packets, see Configuring the TCP MSS Value of TCP Packets, on page 19.

Note To disable this on a session, you must first disable the global configuration, then delete the session and recreate it.

TCP encapsulated in both IPv4 and IPv6 are supported.

Restrictions

These restrictions are applicable for TCP MSS Adjustment:

- Because the MSS is TCP-specific, the TCP MSS Adjustment feature is applicable only to (transit) TCP packets and the UDP packets are unaffected.
- TCP MSS Adjustment configuration affects only the PPPoE PTA and LAC sessions types. It does not affect IP sessions or any non-BNG interfaces.
- The MSS option must be the first option in the TCP header.
- The router uses the MSS value that the user configures for checking TCP/IPV4 packets. When checking TCP/IPv6 packets, the router automatically adjusts the configured MSS value down by 20 bytes to account for the larger IPv6 header. For example, if the TCP MSS value is configured to 1450, then the router adjusts the TCP MSS in an IPV4 packet down to 1450 and down to 1430 for an IPv6 packet.

Configuring the TCP MSS Value of TCP Packets

Perform this task to configure the TCP MSS value of TCP packets in order to prevent TCP sessions from being dropped.

SUMMARY STEPS

1. **configure**
2. **subscriber**
3. **pta tcp mss-adjust** *max-segment-size*
4. **commit**
5. **configure**
6. **vpdn**
7. **l2tp tcp-mss-adjust** *max-segment-size*
8. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**subscriber** **Example:** `RP/0/RSP0/CPU0:router(config)# subscriber`	Enables the subscriber configuration mode.
Step 3	**pta tcp mss-adjust** *max-segment-size* **Example:** `RP/0/RSP0/CPU0:router(config-subscriber)# pta tcp mss-adjust 1300`	Sets the MSS value of TCP packets going through a Cisco ASR 9000 Series Router for a PTA subscriber. The TCP MSS Adjust maximum segment size ranges from 1280 to 1536 (in bytes). **Note** The value represents the global value for the PTA sessions, when the feature is enabled it applies to all sessions.
Step 4	**commit**	
Step 5	**configure**	
Step 6	**vpdn** **Example:** `RP/0/RSP0/CPU0:router(config)# vpdn`	Enables the vpdn configuration mode.
Step 7	**l2tp tcp-mss-adjust** *max-segment-size* **Example:** `RP/0/RSP0/CPU0:router(config-vpdn)# l2tp tcp-mss-adjust 1300`	Sets the MSS value of TCP packets going through a Cisco ASR 9000 Series Router for a LAC subscriber. The TCP MSS Adjust maximum segment size ranges from 1280 to 1460 (in bytes).
Step 8	**commit**	

Configuring the TCP MSS Value of TCP Packets: Examples

```
//Example for configuring the TCP MSS value of TCP packets for a PPPoE PTA subscriber
session:

configure
subscriber
pta tcp mss-adjust 1280
!!


// Example for configuring the TCP MSS value of TCP packets for a PPPoE LAC subscriber
session:

configure
vpdn
l2tp tcp-mss-adjust 1460
!!
```

Linking to Subscriber Traffic in a Shared Policy Instance Group

You can associate the subscriber traffic belonging to a Shared Policy Instance (SPI) group of multiple subinterfaces with a link using a Cisco Vendor-Specific Attribute (VSA). When you apply member hash Cisco:Avpair from RADIUS for a SPI group, traffic for that group will not spill across members. You can identify hash to member mapping based on the bundle's Link Ordering Number (LON).

To enable this feature, configure the following Cisco VSA in the RADIUS profile of the subscriber:

```
Cisco-avpair = "subscriber:member-hash=XX"
```

where XX is the hash value.

Supported Features

- IPoE and PPPoE call flows
- IPv4 and IPv6
- Member hash can be downloaded from RADIUS server
- Traffic is programmed when a new hash value is downloaded and also when a bundle member is modified
- High availability scenarios such as Flap, LC OIR, Process restart, and RPFO
- Only route processor subscribers and with maximum scale

Verifying Hash Value

To display the hash value programmed for the subscriber session, refer to Flow-tag value in the **show route** *address* **detail** command output:

```
RP/0/0/CPU0:server#show route 10.0.0.1/32 detail
Mon Mar  2 20:08:29.079 IST

Routing entry for 10.0.0.1/32
  Known via "subscriber", distance 2, metric 0 (connected)
  Installed Mar  2 20:07:35.448 for 00:00:54
```

```
Routing Descriptor Blocks
    directly connected, via GigabitEthernet0/0/0/0.pppoe1
      Route metric is 0
      Label: 0x300 (768)
      Tunnel ID: None
      Extended communities count: 0
      NHID:0x0(Ref:0)
  Route version is 0x1 (1)
  No local label
  IP Precedence: Not Set
  QoS Group ID: Not Set
  Flow-tag: 33
  Route Priority: RIB_PRIORITY_RECURSIVE (9) SVD Type RIB_SVD_TYPE_LOCAL
  Download Priority 3, Download Version 5
  No advertising protos.
```

Subscriber Session on Ambiguous VLANs

Ambiguous VLAN enables you to create multiple subscriber sessions on a single access-interfaces. As a result, it increases the scalability of the access-interface. An ambiguous VLAN is an L3 interface on which either a VLAN ID range, or a group of individual VLAN IDs are specified. Instead of individually mapping each subscriber to a VLAN, an ambiguous VLAN configuration performs the mapping for a group. Multiple subscribers can be mapped on the ambiguous VLAN as long as they possess a unique MAC address. The subscriber sessions created over ambiguous VLANs are identical to the ones created over regular VLANs, and support all regular configurations such as policy-map, VRFs, QoS, access-control list, and so on.

For enabling IPoE subscriber session creation on an ambiguous VLAN, see Establishing Subscriber Session on Ambiguous VLANs, on page 21.

From Cisco IOS XR Release 5.1.3 and later, the DHCP offer can be send as Unicast (or as per the broadcast policy flag in the DHCP request) for ambiguous VLANs. The ambiguous VLAN configuration in this case, must use a range of VLAN tags (For example, **encapsulation ambiguous dot1q** *10, 100*).

For ambiguous VLAN dot1q configuration where the match criteria is explicitly configured for inner and outer VLAN tags or where a range is specified or where **any** is used for outer VLAN tag, the MTU is calculated by adding 8 bytes (2xdot1q tags) to the default MTU. That is, if default is 1514, the MTU is set to 1522 bytes in such scenarios. Whereas, for configurations where the match criteria for inner VLAN is specified as **any**, the MTU on the sub-interface is calculated by adding 4 (and not 8) bytes to the main interface MTU. That is, 1514 + 4 = 1518 bytes. This behavior is applicable for both physical interfaces and bundle sub-interfaces.

Restriction

The use of **any** tag in the ambiguous VLAN configuration is not supported for Unicast DHCP offers. The DHCP offer packets are not forwarded to the subscriber if **any** tag is used in the configuration.

A DHCP proxy debug error message saying, ARP is not supported on ambiguous VLAN interface, is logged in such failure scenarios.

Establishing Subscriber Session on Ambiguous VLANs

Perform this task to define an ambiguous VLAN and enable creation of IP subscriber session on it.

Note There is no DHCP-specific configuration required for ambiguous VLANs.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. Use any of these commands to configure encapsulated ambiguous VLANs:
   - **encapsulation ambiguous** { **dot1q** | **dot1ad** } {**any** | *vlan-range* }
   - **encapsulation ambiguous dot1q** *vlan-id* **second-dot1q** { **any** | *vlan-range* }
   - **encapsulation ambiguous dot1q any second-dot1q** { **any** | *vlan-id* }
   - **encapsulation ambiguous dot1ad** *vlan-id* **dot1q** { **any** | *vlan-range* }
   - **encapsulation ambiguous dot1q** *vlan-range* **second-dot1q any**
   - **encapsulation ambiguous dot1ad** *vlan-range* **dot1q any**
4. **ipv4 | ipv6address** *source-ip-address destination-ip-address*
5. **service-policy type control subscriber** *policy_name*
6. **ipsubscriber ipv4 l2-connected**
7. **initiator dhcp**
8. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**interface** *type interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface GigabitEthernet0/1/0/0.12`	Configures the interface and enters the interface configuration mode.
Step 3	Use any of these commands to configure encapsulated ambiguous VLANs: • **encapsulation ambiguous** { **dot1q**	**dot1ad** } {**any**	*vlan-range* } • **encapsulation ambiguous dot1q** *vlan-id* **second-dot1q** { **any**	*vlan-range* } • **encapsulation ambiguous dot1q any second-dot1q** { **any**	*vlan-id* } • **encapsulation ambiguous dot1ad** *vlan-id* **dot1q** { **any**	*vlan-range* } • **encapsulation ambiguous dot1q** *vlan-range* **second-dot1q any** • **encapsulation ambiguous dot1ad** *vlan-range* **dot1q any** **Example:**	Configures IEEE 802.1Q VLAN configuration. The *vlan-range* is given in comma-separated, or hyphen-separated format, or a combination of both, as shown in the examples. **Note** Although **encapsulation ambiguous dot1ad** is supported, it is not commonly used in BNG deployments. **encapsulation ambiguous dot1q any** is not supported for unicast DHCP offers. You must use **encapsulation ambiguous dot1q** *vlan-range* for such scenarios.

	Command or Action	Purpose

	`RP/0/RSP0/CPU0:router(config-if)# encapsulation ambiguous dot1q any`
	`RP/0/RSP0/CPU0:router(config-if)# encapsulation ambiguous dot1q 14 second-dot1q 100-200`
	`RP/0/RSP0/CPU0:router(config-if)# encapsulation ambiguous dot1q any second-dot1q any`
	`RP/0/RSP0/CPU0:router(config-if)# encapsulation ambiguous dot1ad 14 dot1q 100,200,300-400`
	`RP/0/RSP0/CPU0:router(config-if)# encapsulation ambiguous dot1q 1-1000 second-dot1q any`	

Step 4	**ipv4 | ipv6address** *source-ip-address destination-ip-address*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# ipv4 address 2.1.12.1 255.255.255.0`
	`RP/0/RSP0/CPU0:router(config-if)# ipv6 address 1:2:3::4 128`
		Configures the IPv4 or IPv6 protocol address.

Step 5	**service-policy type control subscriber** *policy_name* **Example:** `RP/0/RSP0/CPU0:router(config-if)# service-policy type control subscriber PL1`	Applies a policy-map to an access interface where the policy-map was previously defined with the specified PL1 policy_name.
Step 6	**ipsubscriber ipv4 l2-connected** **Example:** `RP/0/RSP0/CPU0:router(config-if)# ipsubscriber ipv4 l2-connected`	Enables l2-connected IPv4 IP subscriber.
Step 7	**initiator dhcp** **Example:** `RP/0/RSP0/CPU0:router(config-if)# initiator dhcp`	Enables initiator DHCP on the IP subscriber.
Step 8	**commit**	

Establishing Subscriber Session on Ambiguous VLANs: An example

```
configure
interface Bundle-Ether100.10
encapsulation ambiguous dot1q 14 second-dot1q any
ipv4 address 2.1.12.12 55.255.255.0
service-policy type control subscriber PL1
ipsubscriber ipv4 l2-connected
!
!
end
```

Outer VLAN Range

The Outer VLAN range is a BNG-specific feature that provides a more advanced VLAN encapsulation option of double-tagged VLANs, where the outer VLAN is specified as a range and the inner VLAN is specified as **any**.

The current BNG implementation supports a high scale of subscriber interface. However, due to QoS hardware limitation, the number of subscribers with QoS policies attached under a single L3 ambiguous VLAN sub-interface is limited to 8K. Therefore, in a large scale scenario, if QoS policies are to be attached to each of the subscribers and if the maximum scale per port is to be achieved, you must configure multiple L3 ambiguous VLAN sub-interfaces per port, with encapsulations that partition the subscribers among the VLAN sub-interfaces. The encapsulations used in such scenarios are:

- Single-tagged VLAN range encapsulations.
- Double-tagged encapsulation, with an inner VLAN range.
- Double-tagged encapsulations, with a fixed outer VLAN-ID and an inner VLAN match for **any**.

In certain scenarios, depending on how the VLAN-IDs are allocated for the subscribers, none of the above partitioning schemes may be suitable. In such scenarios, the L3 ambiguous encapsulation double tag that matches an outer VLAN range and **any** inner VLAN can be used.

The configuration options available for the Outer VLAN range feature are:

- **encapsulation ambiguous dot1q** *vlan range* **second-dot1q any**
- **encapsulation ambiguous dot1ad** *vlan range* **dot1q any**

Sample Configuration for Outer VLAN Range

The sample configuration listed in this section shows how to configure 32K subscribers for each physical interface, using a double-tagged encapsulation to partition the subscribers across four sub-interfaces. Here, 8K subscribers, each with a separate QoS policy applied, are configured for each VLAN sub-interface. Further, a total of four VLAN sub-interfaces are configured to support 32K subscribers for each physical interface.

Option 1: Four VLAN sub-interfaces

```
interface GigabitEthernet0/0/0/0.1
encapsulation ambiguous dot1q 1-1000 second-dot1q any
!
interface GigabitEthernet0/0/0/0.2
encapsulation ambiguous dot1q 1001-2000 second-dot1q any
!
interface GigabitEthernet0/0/0/0.3
encapsulation ambiguous dot1q 2001-3000 second-dot1q any
!
interface GigabitEthernet0/0/0/0.4
encapsulation ambiguous dot1q 3001-4000 second-dot1q any
!
```

Option 2: Nine VLAN configuration ranges

```
interface GigabitEthernet0/0/0/0.1
encapsulation ambiguous dot1q 9-18, 19-25, 26, 27-30, 32, 33-40, 42, 43-50, 52 second-dot1q
 any
```

```
!
```

Verification of Outer VLAN Range Configurations

These show commands can be used to verify the outer VLAN range configurations in BNG:

SUMMARY STEPS

1. **show interface***VLAN sub-interface*
2. **show ethernet tags** *VLAN sub-interface*
3. **show ethernet tags** *VLAN sub-interface* **detail**

DETAILED STEPS

Step 1 **show interface***VLAN sub-interface*

Displays VLAN sub-interface details, including encapsulations.

Example:

```
RP/0/RSP0/CPU0:router#
show interfaces GigabitEthernet 0/1/0/10.12
GigabitEthernet0/1/0/10.12 is up, line protocol is up
  Interface state transitions: 1
  Hardware is VLAN sub-interface(s), address is 0022.bde2.b222
  Internet address is Unknown
  MTU 1518 bytes, BW 1000000 Kbit (Max: 1000000 Kbit)
     reliability 255/255, txload 0/255, rxload 0/255
  Encapsulation 802.1Q Virtual LAN,
    Outer Match: Dot1Q VLAN 11-20,21-30,31-60,61-100,101-140,141-180,181-220,221-260,261-300
    Inner Match: Dot1Q VLAN any
    Ethertype Any, MAC Match src any, dest any
  loopback not set,
  Last input never, output never
  Last clearing of "show interface" counters never
  5 minute input rate 0 bits/sec, 0 packets/sec
- - - - -
- - - - -
```

Step 2 **show ethernet tags** *VLAN sub-interface*

Displays VLAN sub-interface outer tag information, including outer VLAN ranges.

Example:

```
RP/0/RSP0/CPU0:router#
show ethernet tags tengigE 0/0/0/0.1
St:    AD - Administratively Down, Dn - Down, Up - Up
Ly:    L2 - Switched layer 2 service, L3 = Terminated layer 3 service,
Xtra   C - Match on Cos, E  - Match on Ethertype, M - Match on source MAC
-,+:   Ingress rewrite operation; number of tags to pop and push respectively

Interface               St  MTU  Ly Outer            Inner            Xtra -,+
Te0/0/0/0.1             Up  1522 L3 .1Q:10           .1Q:100-200      -    0 0
- - - - -
- - - - -
```

Step 3 **show ethernet tags** *VLAN sub-interface* **detail**

Displays VLAN sub-interface outer tag information, including outer VLAN ranges, in detail.

Example:

```
RP/0/RSP0/CPU0:router#
show ethernet tags  GigabitEthernet 0/0/0/0.1 detail
GigabitEthernet0/1/0/10.12 is up, service is L3
    Interface MTU is 1518
    Outer Match: Dot1Q VLAN 11-20,21-30,31-60,61-100,101-140,141-180,181-220,221-260,261-300
    Inner Match: Dot1Q VLAN any
    Local traffic encap: -
    Pop 0 tags, push none
```

Limitations of Outer VLAN Range

The Outer VLAN Range feature is subjected to these restrictions:

- It is specific to BNG.
- The double-tagged L3 ambiguous encapsulation that matches an outer VLAN range and **any** inner VLAN, and an overlapping single tag encapsulation must not be configured at the same time under the same parent trunk interface. For example, the configurations listed here shows a double-tagged encapsulation configured under one sub-interface and a single-tagged encapsulation configured under another sub-interface of the same parent interface. Although it is not a valid configuration, the system does not reject it.

```
interface Bundle-ether 1.1
encapsulation ambiguous dot1q 2-100 second any
!
interface Bundle-ether 1.2
encapsulation ambiguous dot1q 3
```

- Network layer protocols must not be configured on L3 VLAN sub-interfaces configured with VLAN ranges or the **any** keyword. If they are configured in that manner, then any layer 3 traffic may be dropped. This is a limitation of generic ambiguous VLANs, and is applicable to BNG-specific outer VLAN range feature too.

uRPF

Unicast Reverse Path Forwarding (uRPF) is a feature in BNG that verifies whether the packets that are received on a subscriber interface are sent from a valid subscriber. uRPF only applies to subscribers using an L3 service.

For PPPoE subscribers, the uRPF check ensures that the source address in the arriving packet matches the set of addresses associated with the subscriber. The subscriber addresses are the IPCP assigned addresses, or any framed routed assigned through RADIUS. PPPoE subscribers are identified by session ID and VLAN keys. BNG performs the uRPF check to ensure that the source IP address in the arriving packets matches the expected session IDs and VLAN keys.

For IPoE subscribers, the subscriber addresses are the ones assigned through DHCP. IPoE subscribers are identified by the incoming MAC address. The uRPF check ensures that the source IP address is the one allocated by DHCP to the source MAC address.

uRPF is supported on both IPv4 and IPv6 subscribers and is enabled using a dynamic template. To define a dynamic template for enabling uRPF, see Creating Dynamic Template for IPv4 or IPv6 Subscriber Session.

Multicast Services

Multicast services enable multiple subscribers to be recipients of a single transmission from one source. For example, real-time audio and video conferencing makes good use of a multicast service. The multicast features applied on the PPPoE interfaces of BNG includes:

Multicast Coexistence

On BNG, the multicast services coexist with regular unicast services. The multicast feature on BNG is the same as the existing L3 multicast feature already supported on the Cisco ASR 9000 Series Routers. On BNG, multicast is enabled on the trunk interfaces, and the VLANs created over physical interfaces and bundles. Multicast co-existence works for PPPoE PTA subscriber sessions. For more details on multicast implementation on ASR9k, see *Implementing Layer-3 Multicast Routing on Cisco IOS XR Software* chapter in *Multicast Configuration Guide for Cisco ASR 9000 Series Routers*.

To enable multicast function on BNG, see Enabling Address Family for the VRF, on page 27.

Enabling Address Family for the VRF

Perform this task to enable multicast functions for the required address family.

SUMMARY STEPS

1. **configure**
2. **multicast-routing**
3. **vrf** *vrf_name*
4. **address-family ipv4**
5. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**multicast-routing** **Example:** `RP/0/RSP0/CPU0:router(config)# multicast routing`	Configures multicast-routing.
Step 3	**vrf** *vrf_name* **Example:** `RP/0/RSP0/CPU0:router(config)# vrf vrf1`	Configures the vrf name.

	Command or Action	Purpose
Step 4	**address-family ipv4** **Example:** `RP/0/RSP0/CPU0:router(config)# address-family ipv4`	Enables the multicast functions in the ipv4 address family.
Step 5	**commit**	

Enabling Address Family for the VRF: An example

```
multicast-routing
vrf vrf1
address-family ipv4
 !
!
end
```

Multicast Replication

BNG supports the multicast packet replication on PPPoE interfaces. It also supports multicast forwarding on subscriber interfaces, and transmission of multicast IP video content. When the multicast replication is enabled for a subscriber, BNG performs IGMP statistics gathering for that subscriber, and has the ability to export them. Multicast replication is supported on subscriber interfaces, which are configured in the passive mode.

HQoS Correlation

The Hierarchical quality of service (HQoS) correlation feature monitors every subscriber's multicast bandwidth usage through IGMP reports received on each subscriber's PPPoE session, and limits the unicast bandwidth usage, to leave enough bandwidth for multicast traffic. This is useful when the multicast traffic and unicast traffic share the same physical link to the subscriber in the last mile, when the multicast and unicast traffic are forwarded onto the last mile link by different devices. This feature is configured on BNG that forwards the unicast traffic to the subscriber. Based on the IGMP reports received, BNG informs the unicast QoS shaper on the PPPoE session to alter the bandwidth limit allowed for unicast traffic flows. Using this HQoS correlation feature, a service provider can protect the multicast traffic to the PPPoE subscriber from bursty unicast traffic. The bandwidth profiles for multicast flows need to be configured on BNG.

To define the bandwidth profile, see Configuring Minimum Unicast Bandwidth, on page 28.

To specify the mode for Multicast HQoS, see Configuring Multicast HQOS Correlation Mode or Passive Mode, on page 30.

Configuring Minimum Unicast Bandwidth

A minimum unicast bandwidth can be configured, to prevent unicast traffic from being completely cut off by oversubscribed multicast traffic. Perform this task to set the guaranteed minimum unicast bandwidth for a subscriber using QoS.

SUMMARY STEPS

1. **configure**
2. **dynamic-template**

3. **type** [**ppp** | **ip-subscriber** | **service**] *name*
4. **qos output minimum-bandwidth** *range*
5. **exit**
6. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**dynamic-template** **Example:** `RP/0/RSP0/CPU0:router(config)# dynamic-template`	Enters dynamic template configuration mode.
Step 3	**type** [**ppp**	**ip-subscriber**	**service**] *name* **Example:** `RP/0/RSP0/CPU0:router(config-dynamic-template)# type ppp p1`	. Specifies the type of dynamic template that needs to be applied. Three available types are: • PPP • IP-subscriber • Service
Step 4	**qos output minimum-bandwidth** *range* **Example:** `RP/0/RSP0/CPU0:router(config-dynamic-template-type)# qos output minimum-bandwidth 10`	Sets the guaranteed minimum bandwidth, in kbps, for a subscriber. Range is from 1 to 4294967295.
Step 5	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-dynamic-template-type)# exit`	Exits from the current mode.
Step 6	**commit**	

Configuring Minimum Bandwidth: An example

```
configure
dynamic-template
type ppp p1
service-policy output pmap
multicast ipv4 qos-correlation
qos output minimum-bandwidth 10
end
```

Configuring Multicast HQOS Correlation Mode or Passive Mode

Perform this task to configure multicast in HQoS correlation mode or passive mode to enable multicast replication over PPPoE interfaces.

SUMMARY STEPS

1. **configure**
2. **dynamic-template**
3. **type ppp** *dynamic-template name*
4. **multicast ipv4 <qos-correlation | passive>**
5. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**dynamic-template** **Example:** `RP/0/RSP0/CPU0:router(config)# dynamic-template`	Enter the dynamic-template configuration mode.
Step 3	**type ppp** *dynamic-template name* **Example:** `RP/0/RSP0/CPU0:router(config-dynamic-template)# type ppp foo`	Enters the ppp type mode to configure igmp for subscriber interfaces.
Step 4	**multicast ipv4 <qos-correlation	passive>** **Example:** `RP/0/RSP0/CPU0:router(config-dynamic-template-type)# multicast ipv4 qos-correlation`	Configures the subscriber either in the QoS-correlation mode (IGMP-HQOS correlation), or passive mode (multicast forwarding).
Step 5	**commit**	

Configuring Multicast HQOS Correlation Mode: An example

```
dynamic-template type ppp foo
multicast ipv4 qos-correlation
 !
!
end
```

IGMP to Unicast QoS Shaper Correlation

The Unicast QoS Shaper correlation feature configures the bandwidth profiles for the multicast flows and allows the IGMP messages to derive the multicast bandwidth usage for each subscriber. On the PPPoE subscriber sessions, the amount of multicast bandwidth that a subscriber uses is deducted from the unicast QoS shaper until a minimum threshold is reached.

For more information about configuring the IGMP QoS shaper, see Configuring the IGMP to HQoS Correlation Feature in a VRF, on page 31. For more information about configuring the IGMP for subscriber interfaces, see Configuring IGMP Parameters for Subscriber Interfaces, on page 33.

IGMP uses route-policies to distribute the absolute rate for all multicast flows. For more information for configuring the route-policy for unicast QoS shaper, see Configuring route-policy for Unicast QoS Shaper, on page 32.

Configuring the IGMP to HQoS Correlation Feature in a VRF

Perform this task to configure the IGMP to HQoS Correlation Feature in a VRF.

SUMMARY STEPS

1. **configure**
2. **router igmp**
3. **unicast-qos-adjust adjustment-delay** *time*
4. **unicast-qos-adjust download-interval** *time*
5. **unicast-qos-adjust holdoff** *time*
6. **vrf** *vrf-name*
7. **traffic profile** *profile-name*
8. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**router igmp** **Example:** `RP/0/RSP0/CPU0:router(config)# router igmp`	Enters the router process for IGMP configuration mode.
Step 3	**unicast-qos-adjust adjustment-delay** *time* **Example:** `RP/0/RSP0/CPU0:router(config-igmp)# unicast-qos-adjust adjustment-delay 1`	Configures the time to wait before programming rate in IGMP QoS shaper for subscriber unicast traffic. The time to wait ranges from 0 to 10 seconds.
Step 4	**unicast-qos-adjust download-interval** *time* **Example:** `RP/0/RSP0/CPU0:router(config-igmp)# unicast-qos-adjust download-interval 10`	Configures the time before downloading a batch of interfaces to IGMP QoS shaper for subscriber unicast traffic. The download interval time ranges from 10 to 500 milliseconds.
Step 5	**unicast-qos-adjust holdoff** *time* **Example:** `RP/0/RSP0/CPU0:router(config-igmp)# unicast-qos-adjust holdoff 5`	Configures the hold-off time before QoS clears the stale entries for the IGMP QoS shaper. The hold-off time ranges from 5 to 1800 seconds.
Step 6	**vrf** *vrf-name* **Example:**	Enters the VRF configuration mode.

	Command or Action	Purpose
	`RP/0/RSP0/CPU0:router(config-igmp)# vrf vrf1`	
Step 7	**traffic profile** *profile-name* **Example:** `RP/0/RSP0/CPU0:router(config-igmp-vrf1)# traffic profile routepolicy1`	Configures the route-policy to be used to map the bandwidth profile.
Step 8	**commit**	

Configuring the IGMP QoS Shaper: An Example

```
configure
router igmp
unicast-qos-adjust adjustment-delay 1
unicast-qos-adjust download-interval 10
unicast-qos-adjust holdoff 5
vrf vrf1
traffic profile routepolicy1
!
!
end
```

Configuring route-policy for Unicast QoS Shaper

Perform this task to configure route-policy for unicast QoS shaper.

SUMMARY STEPS

1. **configure**
2. **router igmp**
3. **vrf** *vrf-name*
4. **traffic profile** *profile-name*
5. **commit**
6. **show igmp unicast-qos-adjust statistics**
7. **show igmp unicast-qos-adjust statistics interface** *interface-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**router igmp** **Example:** `RP/0/RSP0/CPU0:router(config)# router igmp`	Enter the router process for igmp configuration mode.
Step 3	**vrf** *vrf-name* **Example:**	Enters the vrf configuration mode.

	Command or Action	Purpose
	`RP/0/RSP0/CPU0:router(config-igmp)# vrf vrf1`	
Step 4	**traffic profile** *profile-name* **Example:** `RP/0/RSP0/CPU0:router(config-igmp-vrf1)# traffic profile routepolicy1`	Configures the route-policy to be used to map the bandwidth profile.
Step 5	**commit**	
Step 6	**show igmp unicast-qos-adjust statistics** **Example:** `RP/0/RSP0/CPU0:router# show igmp unicast-qos-adjusted statistics`	(Optional) Displays the internal statistics of the feature, such as total number of interface groups under adjustment, uptime since last clear command, and total number of rate adjustment calls for unicast QoS shaper.
Step 7	**show igmp unicast-qos-adjust statistics interface** *interface-name* **Example:** `RP/0/RSP0/CPU0:router# show igmp unicast-qos-adjusted statistics interface interface1`	(Optional) Displays the interface name, number of flows adjusted, total rate adjusted, uptime after first adjustment for unicast QoS shaper.

Configuring route-policy for Unicast QoS Shaper: Examples

```
#Adding a route-policy for profile1

route-policy profile1
if destination in (239.0.0.0/8 le 32) then
set weight 1000
endif
end-policy

# Configuring profile1 for Unicast QoS Shaper
router igmp
vrf vrf1
traffic profile profile1
 !
!
end
```

Configuring IGMP Parameters for Subscriber Interfaces

Perform this task to configure IGMP parameters for subscriber interfaces.

SUMMARY STEPS

1. **configure**
2. **dynamic-template**
3. **type ppp** *dynamic-template name*
4. **igmp explicit-tracking**
5. **igmp query-interval** *value*
6. **igmp query-max-response-time** *query-response-value*

7. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**dynamic-template** **Example:** `RP/0/RSP0/CPU0:router(config)# dynamic-template`	Enter the dynamic-template configuration mode.
Step 3	**type ppp** *dynamic-template name* **Example:** `RP/0/RSP0/CPU0:router(config-dynamic-template)# type ppp foo`	Enters the ppp type mode to configure igmp for subscriber interfaces.
Step 4	**igmp explicit-tracking** **Example:** `RP/0/RSP0/CPU0:router(config-dynamic-template-type)# igmp explicit-tracking`	Enables IGMPv3 explicit host tracking.
Step 5	**igmp query-interval** *value* **Example:** `RP/0/RSP0/CPU0:router(config-dynamic-template-type)# igmp query-interval 60`	Sets the query-interval in seconds for igmp. **Note** The igmp query-interval value, in seconds, should be in the range from 1 to 3600. With 16000 PPPoE subscribers or less, the recommended value, that also the default, is 60 seconds.
Step 6	**igmp query-max-response-time** *query-response-value* **Example:** `RP/0/RSP0/CPU0:router(config-dynamic-template-type)# igmp query-max-response-time 4`	Sets the query-max-response-time in seconds for igmp. **Note** The igmp query-interval value, in seconds, is in the range from 1 to 12.
Step 7	**commit**	

Configuring IGMP for Subscriber Interfaces: An example

```
dynamic-template type ppp foo
igmp explicit-tracking
igmp query-interval 60
igmp query-max-response-time 4
 !
!
end
```

IGMP Accounting

The Internet Group Management Protocol (IGMP) accounting feature enables BNG to maintain a statistics file to log the instances of subscriber joining, or leaving a multicast group. The file's format is:

```
harddisk:/usr/data/igmp/accounting.dat.<Node ID>.<YYMMDD>
```

where

- Node ID is the name of the node that generates the file; for example, RP/0/RSP0/CPU0.
- YY is the year, MM is the month, and DD is the day.

An example of the statistics file name is:

```
harddisk:/usr/data/igmp/accounting.dat.RP_0_RSP0_CPU0.101225
```

The statistics file is stored on the route processor (RP) that is active. If a failover event occurs, then a new file is created on the new active RP, and no attempt is made to mirror the data between the active and the standby RP. Thus, the statistics files must be retrieved from both the active and standby RPs.

By default, the IGMP Accounting feature adds one file each day. To avoid exhausting disk space, you can specify in how many files, or for how many days, data should be retained, see Configuring IGMP Accounting, on page 35. Files older than the specified period are deleted, and the data is discarded from BNG. The maximum size of each file should be no more than 250 MB.

Configuring IGMP Accounting

Perform this task to configure the IGMP accounting.

SUMMARY STEPS

1. **configure**
2. **router igmp**
3. **accounting** [**max-history**] *days*
4. **commit**
5. **show igmp interface**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**router igmp** **Example:** `RP/0/RSP0/CPU0:router(config)# router igmp`	Enter the router process for IGMP configuration mode.
Step 3	**accounting** [**max-history**] *days* **Example:** `RP/0/RSP0/CPU0:router(config-igmp-vrf1)# accounting max-history 50`	Configures the IGMP accounting. The max-history parameter is optional and specifies how many files are kept; this number is equivalent to the number of days in the history.
Step 4	**commit**	

	Command or Action	Purpose
Step 5	**show igmp interface** **Example:** `RP/0/RSP0/CPU0:router# show igmp interface`	(Optional) Displays IGMP interface information.

Configuring IGMP Accounting: An example

```
configure
router igmp
accounting max-history 45
!
!
end
```

DAPS Support

A Distributed Address Pool Service (DAPS) allows address pools to be shared between DHCP processes that run on a line card (LC) and the route processor (RP). The DHCP Server and PPPoE subscribers are clients to DAPS, and are known as the DAPS client. DAPS is used to return IP address to clients only when the RADIUS attributes contain the attribute "Pool Name". If the RADIUS attribute for a subscriber contains a fixed address, then the client does not contact DAPS for its IP address.

DAPS runs in two forms, as DAPS server on the RP, and as DAPS-Proxy on the LC. The RP has an in-build DAPS-Proxy module. This model ensures that all DAPS clients always talk to the DAPS-Proxy. The DAPS-Proxy instances talk to the central DAPS-Server on the RP for address assignments and other requests. DAPS-Proxy runs on all the LCs in the system. The DAPS-Proxy running on an LC can service multiple clients, from that LC; for example, PPP, DHCPv6, IPv6ND. DAPS serves multiple DAPS clients on two or more nodes. A separate DAPS-Proxy process runs on each node and connects locally to each DAPS Client.

DAPS supports dynamic IPv4 and IPv6 address allocation by pool name. For more information about configuring IPv4 DAPS, see Configuring IPv4 Distributed Address Pool Service, on page 36. To create a configuration pool for IPv6, see Creating a Configuration Pool Submode, on page 37.

You can configure various DAPS IPv6 parameters in the IPv6 configuration submode. You can configure the subnet number and mask for an IPv6 address pool, for more information, see Configuring the Subnet Number and Mask for an Address Pool, on page 38. You can specify parameters such as a range of IPv6 addresses. For more information, see Specifying a Range of IPv6 Addresses, on page 40. To specify a utilization threshold, see Specifying a Utilization Threshold, on page 40. To specify a set of prefixes or addresses inside a subnet, see Specifying a Set of Addresses or Prefixes Inside a Subnet, on page 43. You can also specify the length of a prefix. For more information, see Specifying the Length of the Prefix, on page 42.

Configuring IPv4 Distributed Address Pool Service

Perform this task to configure IPv4 distributed address pool service (DAPS).

SUMMARY STEPS

1. **configure**

2. **pool ipv4** *ipv4-pool-name*
3. **address-range** *first_address second_address*
4. **pool vrf** *vrf-name* **ipv4** *ipv4-pool-name*{**address-range** *address-range*}
5. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**pool ipv4** *ipv4-pool-name* **Example:** `RP/0/RSP0/CPU0:router(config)# pool ipv4 pool1`	Configures IPv4 pool name.
Step 3	**address-range** *first_address second_address* **Example:** `RP/0/RSP0/CPU0:router(config-pool-ipv4)# address-range 1.1.1.1 9.8.9.8`	Configures the address range for allocation.
Step 4	**pool vrf** *vrf-name* **ipv4** *ipv4-pool-name*{**address-range** *address-range*} **Example:** `RP/0/RSP0/CPU0:router(config)# pool vrf vrf1 ipv4 pool1 address-range 1.1.1.1 9.8.9.8`	Configures IPv4 pool name.
Step 5	**commit**	

Configuring IPv4 Distributed Address Pool Service: An example

```
pool ipv4 pool1
address-range 1.1.1.1 9.8.9.8
pool vrf vrf1 ipv4 pool1 address-range 1.1.1.1 9.8.9.8
 !
!
end
```

Creating a Configuration Pool Submode

Perform this task to create and enable an IPv6 configuration pool submode for a default VRF and for a specific VRF.

SUMMARY STEPS

1. **configure**
2. **pool ipv6** *ipv6-pool-name*
3. **commit**

4. **configure**
5. **pool vrf** *vrf_name* **ipv6** *ipv6-pool-name*
6. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**pool ipv6** *ipv6-pool-name* **Example:** `RP/0/RSP0/CPU0:router(config)# pool ipv6 pool1`	Creates the IPv6 pool name for a default VRF and enters the pool IPv6 configuration submode.
Step 3	**commit**	
Step 4	**configure**	
Step 5	**pool vrf** *vrf_name* **ipv6** *ipv6-pool-name* **Example:** `RP/0/RSP0/CPU0:router(config)# pool vrf vrf1 ipv6 pool1`	Creates the IPv6 pool name for a specific VRF and enters the pool IPv6 configuration submode.
Step 6	**commit**	

Creating a Configuration Pool Submode: An example

```
configure
pool ipv6 pool1 (default vrf)
!
!
configure
pool vrf vrf1 ipv6 pool1 (for a specific vrf)
!
!
end
```

Configuring the Subnet Number and Mask for an Address Pool

Perform this task to create the subnet number and mask for an IPv6 address pool.

SUMMARY STEPS

1. **configure**
2. **pool vrf** *vrf_name* **ipv6** *ipv6-pool-name*
3. **prefix-length** *value*
4. **network** *subnet*
5. **utilization-mark high** *value* **low** *value*
6. **exclude** *low_ip_address high_ip_address*

7. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**pool vrf** *vrf_name* **ipv6** *ipv6-pool-name* **Example:** `RP/0/RSP0/CPU0:router(config)# pool vrf default ipv6 test`	Creates the IPv6 pool name for a specific VRF and enters the pool IPv6 configuration submode.
Step 3	**prefix-length** *value* **Example:** `RP/0/RSP0/CPU0:router(config-pool-ipv6)# prefix-length 120`	Specifies the length of the prefix that is assigned to the clients. The value of the prefix length ranges from 1 to 128.
Step 4	**network** *subnet* **Example:** `RP/0/RSP0/CPU0:router(config-pool-ipv6)# network 1101:1::/114`	Specifies a set of addresses or prefixes inside a subnet. **Note** The **prefix-length** command must be mandatorily configured whenever the **network** command is used.
Step 5	**utilization-mark high** *value* **low** *value* **Example:** `RP/0/RSP0/CPU0:router(config-pool-ipv6)# utilization-mark high 70 low 30`	Specifies a utilization threshold in the pool IPv6 submode. The high and low values are represented as percentages between 0 and 100.
Step 6	**exclude** *low_ip_address high_ip_address* **Example:** `RP/0/RSP0/CPU0:router(config-pool-ipv6)# exclude 1101:1::100 ::`	Specifies a range of IPv6 addresses or prefixes that DAPS must not assign to clients. The high and low values are represented as percentages between 0 and 100. **Note** Multiple exclude commands are allowed within a pool. To exclude a single address, <*high_ip_address*> can be omitted.
Step 7	**commit**	

Configuring the Subnet Number and Mask for an Address Pool: An example

```
configure
pool vrf default ipv6 test
prefix-length 120
network 1101:1::/114
utilization-mark high 70 low 30
exclude 1101:1::100 ::
!
```

```
!
end
```

Specifying a Range of IPv6 Addresses

Perform this task to specify a range of IPv6 addresses within a pool.

SUMMARY STEPS

1. **configure**
2. **pool vrf** *vrf_name* **ipv6** *ipv6-pool-name*
3. **address-range** *low_ip_address high_ip_address*
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**pool vrf** *vrf_name* **ipv6** *ipv6-pool-name* **Example:** `RP/0/RSP0/CPU0:router(config)# pool vrf vrf1 ipv6 addr_vrf`	Creates the IPv6 pool name for a specific VRF and enters the pool IPv6 configuration submode.
Step 3	**address-range** *low_ip_address high_ip_address* **Example:** `RP/0/RSP0/CPU0:router(config-pool-ipv6)# address-range 1234::2 1234::3e81`	Specifies the range of IPv6 addresses within a pool. Multiple address-ranges are allowed within a pool.
Step 4	**commit**	

Specifying a Range of IPv6 Addresses: An example

```
configure
pool vrf vrf1 ipv6 addr_vrf
address-range 1234::2 1234::3e81
!
!
end
```

Specifying a Utilization Threshold

Perform this task to specify a utilization threshold for a specific VRF in the pool IPv6 submode.

SUMMARY STEPS

1. **configure**

2. **pool vrf** *vrf_name* **ipv6** *ipv6-pool-name*
3. **prefix-length** *value*
4. **network** *subnet*
5. **utilization-mark high** *value* **low** *value*
6. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**pool vrf** *vrf_name* **ipv6** *ipv6-pool-name* **Example:** `RP/0/RSP0/CPU0:router(config)# pool vrf default ipv6 test`	Creates the IPv6 pool name for a specific VRF and enters the pool IPv6 configuration submode.
Step 3	**prefix-length** *value* **Example:** `RP/0/RSP0/CPU0:router(config-pool-ipv6)# prefix-length 120`	Specifies the length of the prefix that is assigned to the clients. The value of the prefix length ranges from 1 to 128.
Step 4	**network** *subnet* **Example:** `RP/0/RSP0/CPU0:router(config-pool-ipv6)# network 1101:1::/114`	Specifies a set of addresses or prefixes inside a subnet. **Note** The **prefix-length** command should be mandatorily configured whenever the **network** command is used.
Step 5	**utilization-mark high** *value* **low** *value* **Example:** `RP/0/RSP0/CPU0:router(config-pool-ipv6)# utilization-mark high 70 low 30`	Specifies a utilization threshold in the pool IPv6 submode. The high and low values are represented as percentages between 0 and 100.
Step 6	**commit**	

Specifying a Utilization Threshold: An example

```
configure
pool vrf default ipv6 test
prefix-length 120
network 1101:1::/114
utilization-mark high 70 low 30
!
!
end
```

Specifying the Length of the Prefix

Perform this task to specify the length of the prefix that is assigned to the clients.

SUMMARY STEPS

1. **configure**
2. **pool vrf** *vrf_name* **ipv6** *ipv6-pool-name*
3. **prefix-length** *value*
4. **prefix-range** *low_ipv6_prefix high_ipv6_prefix*
5. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**pool vrf** *vrf_name* **ipv6** *ipv6-pool-name* **Example:** `RP/0/RSP0/CPU0:router(config)# pool vrf vrf1 ipv6 prefix_vrf`	Creates the IPv6 pool name for a specific VRF and enters the pool IPv6 configuration submode.
Step 3	**prefix-length** *value* **Example:** `RP/0/RSP0/CPU0:router(config-pool-ipv6)# prefix-length 64`	Specifies the length of the prefix that is assigned to the clients. The value of the prefix length ranges from 1 to 128.
Step 4	**prefix-range** *low_ipv6_prefix high_ipv6_prefix* **Example:** `RP/0/RSP0/CPU0:router(config-pool-ipv6)# prefix-range 9fff:1:: 9fff:1:0:3e7f::`	Specifies a range of IPv6 address prefixes for a specific VRF in the pool IPv6 configuration mode. **Note** The **prefix-length** must be mandatorily configured whenever **prefix-range** is configured.
Step 5	**commit**	

Specifying the Length of the Prefix that is Assigned to the Clients: An example

```
configure
pool vrf vrf1 ipv6 prefix_vrf
prefix-length 64
prefix-range 9fff:1:: 9fff:1:0:3e7f::
!
!
end
```

Specifying a Set of Addresses or Prefixes Inside a Subnet

Perform this task to specify a set of addresses or prefixes inside a subnet in the pool IPv6 configuration submode.

SUMMARY STEPS

1. **configure**
2. **pool vrf** *vrf_name* **ipv6** *ipv6-pool-name*
3. **prefix-length** *value*
4. **network** *subnet*
5. **utilization-mark high** *value* **low** *value*
6. **exclude** *low_ip_address high_ip_address*
7. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**pool vrf** *vrf_name* **ipv6** *ipv6-pool-name* **Example:** `RP/0/RSP0/CPU0:router(config)# pool vrf default ipv6 test`	Creates the IPv6 pool name for a specific VRF and enters the pool IPv6 configuration submode.
Step 3	**prefix-length** *value* **Example:** `RP/0/RSP0/CPU0:router(config-pool-ipv6)# prefix-length 120`	Specifies the length of the prefix that is assigned to the clients. The value of the prefix length ranges from 1 to 128.
Step 4	**network** *subnet* **Example:** `RP/0/RSP0/CPU0:router(config-pool-ipv6)# network 1101:1::/114`	Specifies a set of addresses or prefixes inside a subnet. **Note** The **prefix-length** command should be mandatorily configured whenever the **network** command is used.
Step 5	**utilization-mark high** *value* **low** *value* **Example:** `RP/0/RSP0/CPU0:router(config-pool-ipv6)# utilization-mark high 70 low 30`	Specifies a utilization threshold in the pool IPv6 submode. The high and low values are represented as percentages between 0 and 100.
Step 6	**exclude** *low_ip_address high_ip_address* **Example:** `RP/0/RSP0/CPU0:router(config-pool-ipv6)# exclude 1101:1::100 ::`	Specifies a range of IPv6 addresses or prefixes that DAPS must not assign to clients. The high and low values are represented as percentages between 0 and 100.

	Command or Action	Purpose
		Note Multiple exclude commands are allowed within a pool. To exclude a single address, <high_ip_address> can be omitted.
Step 7	**commit**	

Specifying a Set of Addresses or Prefixes Inside a Subnet: An example

```
configure
pool vrf default ipv6 test
prefix-length 120
network 1101:1::/114
utilization-mark high 70 low 30
exclude 1101:1::100 ::
!
!
end
```

HTTP Redirect Using PBR

The HTTP Redirect (HTTPR) feature is used to redirect subscriber traffic to a destination other than the one to which it was originally destined. The HTTPR feature is implemented using Policy Based Routing (PBR) that makes packet forwarding decisions based on the policy configuration, instead of routing protocols. The HTTPR feature is implemented by sending an HTTP redirect response, which contains the redirect URL, back to the HTTP client that originally sent the request. Thereafter, the HTTP client sends requests to the redirected URL. HTTPR is supported for both IPv4 and IPv6 subscribers.

The most common use of HTTPR feature is for initial logon. In some cases, it is not possible to uniquely identify a subscriber and authorize them. This happens when the subscriber is using a shared network access medium to connect to the network. In such cases, the subscriber is allowed to access the network but restricted to what is known as an "open-garden". An open-garden is a collection of network resources that subscribers can access as long as they have physical access to the network. Subscribers do not have to provide authentication information before accessing the web sites in an open-garden.

When subscribers try to access resources outside the open-garden (which is called the "walled-garden"), they are redirected to a web logon portal. The walled-garden refers to a collection of web sites or networks that subscribers can access after providing minimal authentication information. The web logon portal requires the subscriber to login using a username and password. Thereafter, the web logon portal sends an account-logon CoA to BNG with user credentials. On successful authentication of these credentials, BNG disables the redirect and applies the correct subscriber policies for direct network access. Other uses of HTTPR include periodic redirection to a web portal for advertising reasons, redirection to a billing server, and so on.

The PBR function is configured in its own dynamic template. If the dynamic template contains other functions too, then the PBR policy that redirects packets must be deactivated using a CoA.

BNG maintains HTTP redirect statistics counters that track the number of packets that are being either redirected or dropped. The HTTP protocol uses some status codes to implement HTTPR. Currently, the redirect codes 302 (for HTTP version 1.0) and 307 (for HTTP version 1.1) are supported on BNG.

Note

- HTTP redirect applies only to HTTP packets. As a result, other services such as SMTP, FTP are not affected by this feature. Nevertheless, if these other services are part of the redirect classification rules, then the packets are dropped and not forwarded.
- HTTPS is not supported.
- Destination URL-based classification is not supported.
- HTTP redirect is supported only on subscriber interfaces.

The process of configuring HTTPR involves these stages:

- Creating access lists that define the redirected and open-garden permissions. See, Identifying HTTP Destinations for Redirection, on page 45.
- Creating the class-maps that uses the access list to classify the traffic as redirected, or permitted to access open-garden. See, Configuring Class Maps for HTTP Redirection, on page 48.
- Creating the policy-map to define the action to be performed on the traffic classified using class-maps. See, Configuring Policy Map for HTTP Redirect, on page 50.
- Creating the dynamic template to apply the service policy. See Configuring Dynamic Template for Applying HTTPR Policy, on page 52.

To configure a web logon that specifies a time limit to perform the authentication, see Configuring Web Logon, on page 53.

Identifying HTTP Destinations for Redirection

Perform this task to define access lists that identify http destinations that require redirection or are part of an open garden:

SUMMARY STEPS

1. **configure**
2. {**ipv4** | **ipv6**}**access-list** *redirect_acl_name*
3. Do one of the following:
   - [*sequence-number*]{**permit** | **deny**} *source source-wildcard destination destination-wildcard* [**precedence** *precedence*] [**dscp** *dscp*] [**fragments**] [*packet-length operator packet-length value*] [**log** | **log-input**]
   - [*sequence-number*] {**permit** | **deny**} *protocol* {*source-ipv6-prefix*/*prefix-length* | **any** | **host** *source-ipv6-address*} [*operator* {*port* | *protocol-port*}] {*destination-ipv6-prefix*/*prefix-length* | **any** | **host** *destination-ipv6-address*} [*operator* {*port* | *protocol-port*}] [**dscp** *value*] [*routing*] [**authen**] [**destopts**] [**fragments**] [*packet-length operator packet-length value*] [**log** | **log-input**]
4. Repeat Step 3 as necessary, adding statements by sequence number. Use the **no** *sequence-number* command to delete an entry.
5. {**ipv4** | **ipv6**}**access-list** *open_garden_acl*
6. Do one of the following:

- [*sequence-number*]{**permit** | **deny**} *source source-wildcard destination destination-wildcard* [**precedence** *precedence*] [**dscp** *dscp*] [**fragments**] [*packet-length operator packet-length value*] [**log** | **log-input**]
- [*sequence-number*] {**permit** | **deny**} *protocol* {*source-ipv6-prefix*/*prefix-length* | **any** | **host** *source-ipv6-address*} [*operator* {*port* | *protocol-port*}] {*destination-ipv6-prefix*/*prefix-length* | **any** | **host** *destination-ipv6-address*} [*operator* {*port* | *protocol-port*}] [**dscp** *value*] [*routing*] [**authen**] [**destopts**] [**fragments**] [*packet-length operator packet-length value*] [**log** | **log-input**]

7. Repeat Step 6 as necessary, adding statements by sequence number. Use the **no** *sequence-number* command to delete an entry.
8. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

Step 2	{**ipv4**	**ipv6**}**access-list** *redirect_acl_name*
		Example:
		`RP/0/RSP0/CPU0:router(config)# ipv4 access-lists redirect_acl`
		or
		`RP/0/RSP0/CPU0:router(config)# ipv6 access-lists redirect_acl`
			Enters either IPv4 or IPv6 access list configuration mode and configures the named access list.

Step 3	Do one of the following: • [*sequence-number*]{**permit**	**deny**} *source source-wildcard destination destination-wildcard* [**precedence** *precedence*] [**dscp** *dscp*] [**fragments**] [*packet-length operator packet-length value*] [**log**	**log-input**] • [*sequence-number*] {**permit**	**deny**} *protocol* {*source-ipv6-prefix*/*prefix-length*	**any**	**host** *source-ipv6-address*} [*operator* {*port*	*protocol-port*}] {*destination-ipv6-prefix*/*prefix-length*	**any**	**host** *destination-ipv6-address*} [*operator* {*port*	*protocol-port*}] [**dscp** *value*] [*routing*] [**authen**] [**destopts**] [**fragments**] [*packet-length operator packet-length value*] [**log**	**log-input**] **Example:** `RP/0/RSP0/CPU0:router(config-ipv4-acl)# 10 permit 172.16.0.0 0.0.255.255` `RP/0/RSP0/CPU0:router(config-ipv4-acl)# 20 deny 192.168.34.0 0.0.0.255` or `RP/0/RSP0/CPU0:router(config-ipv6-acl)# 20 permit`	Specifies one or more conditions allowed or denied in IPv4 or IPv6 access list redirect_acl. • The optional **log** keyword causes an information logging message about the packet that matches the entry to be sent to the console. • The optional **log-input** keyword provides the same function as the **log** keyword, except that the logging message also includes the input interface. or Specifies one or more conditions allowed or denied in IPv6 access list redirect_acl. • Refer to the **deny** (IPv6) and **permit** (IPv6) commands for more information on filtering IPv6 traffic based on based on IPv6 option headers and optional, upper-layer protocol type information. **Note** Every IPv6 access list has an implicit **deny ipv6 any any** statement as its last match condition. An IPv6 access list must contain at least one entry for the implicit **deny ipv6 any any** statement to take effect.

	Command or Action	Purpose
	icmp any any RP/0/RSP0/CPU0:router(config-ipv6-acl)# 30 deny tcp any any gt 5000	
Step 4	Repeat Step 3 as necessary, adding statements by sequence number. Use the **no** *sequence-number* command to delete an entry.	Allows you to revise an access list.
Step 5	{**ipv4** | **ipv6**}**access-list** *open_garden_acl* **Example:** RP/0/RSP0/CPU0:router(config)# ipv4 access-lists open_garden_acl or RP/0/RSP0/CPU0:router(config)# ipv6 access-lists open_garden_acl	Enters either IPv4 or IPv6 access list configuration mode and configures the named access list for open garden.
Step 6	Do one of the following: • [*sequence-number*]{**permit** | **deny**} *source source-wildcard destination destination-wildcard* [**precedence** *precedence*] [**dscp** *dscp*] [**fragments**] [*packet-length operator packet-length value*] [**log** | **log-input**] • [*sequence-number*] {**permit** | **deny**} *protocol* {*source-ipv6-prefix*/*prefix-length* | **any** | **host** *source-ipv6-address*} [*operator* {*port* | *protocol-port*}] {*destination-ipv6-prefix*/*prefix-length* | **any** | **host** *destination-ipv6-address*} [*operator* {*port* | *protocol-port*}] [**dscp** *value*] [*routing*] [**authen**] [**destopts**] [**fragments**] [*packet-length operator packet-length value*] [**log** | **log-input**] **Example:** RP/0/RSP0/CPU0:router(config-ipv4-acl)# 10 permit 172.16.0.0 0.0.255.255 RP/0/RSP0/CPU0:router(config-ipv4-acl)# 20 deny 192.168.34.0 0.0.0.255 or RP/0/RSP0/CPU0:router(config-ipv6-acl)# 20 permit icmp any any RP/0/RSP0/CPU0:router(config-ipv6-acl)# 30 deny tcp any any gt 5000	Specifies one or more conditions allowed or denied in IPv4 access list open_garden_acl. • The optional **log** keyword causes an information logging message about the packet that matches the entry to be sent to the console. • The optional **log-input** keyword provides the same function as the **log** keyword, except that the logging message also includes the input interface. or Specifies one or more conditions allowed or denied in IPv6 access list open_garden_acl. • Refer to the **deny** (IPv6) and **permit** (IPv6) commands for more information on filtering IPv6 traffic based on based on IPv6 option headers and optional, upper-layer protocol type information. **Note** Every IPv6 access list has an implicit **deny ipv6 any any** statement as its last match condition. An IPv6 access list must contain at least one entry for the implicit **deny ipv6 any any** statement to take effect.
Step 7	Repeat Step 6 as necessary, adding statements by sequence number. Use the **no** *sequence-number* command to delete an entry.	Allows you to revise an access list.
Step 8	**commit**	

Identifying HTTP Destinations for Redirection: An example

```
configure
 ipv4 access-list <redirect-acl>
   10 permit tcp any any syn eq www
   20 permit tcp any any ack eq www
   30 permit tcp any any eq www
  ipv4 access-group <allow-acl>
   10 permit tcp any 10.1.1.0 0.0.0.255 eq www
   20 permit tcp any 20.1.1.0 0.0.0.255 eq www
   30 permit tcp any 30.1.1.0 0.0.0.255 eq www
   40 permit udp any any eq domain
 !
!
!
end


configure
 ipv6 access-list <redirect-acl>
   10 permit tcp any any syn eq www
   20 permit tcp any any ack eq www
   30 permit tcp any any eq www
  ipv6 access-group <allow-acl>
   10 permit tcp any 10.1.1.0 0.0.0.255 eq www
   20 permit tcp any 20.1.1.0 0.0.0.255 eq www
   30 permit tcp any 30.1.1.0 0.0.0.255 eq www
   40 permit udp any any eq domain
 !
!
!
end
```

Configuring Class Maps for HTTP Redirection

Perform this task to configure the class maps for HTTP redirection. It makes use of previously defined ACLs.

Before you begin

The configuration steps mentioned in Identifying HTTP Destinations for Redirection, on page 45 has to be completed before performing the configuration of the HTTPR class maps.

SUMMARY STEPS

1. **configure**
2. **class-map type traffic match-all** *open-garden-class_name*
3. **match** [**not**] **access-group**{**ipv4** | **ipv6**} *open_garden_acl*
4. **end-class-map**
5. **class-map type traffic match-all** *http_redirect-class_name*
6. **match** [**not**] **access-group** {**ipv4** | **ipv6**} *redirect_acl*
7. **end-class-map**
8. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**class-map type traffic match-all** *open-garden-class_name* **Example:** `RP/0/RSP0/CPU0:router(config)# class-map type traffic match-all CL1`	Defines a traffic class and the associated rules that match packets to the class for a open garden class.

Step 3	**match** [**not**] **access-group**{**ipv4** | **ipv6**} *open_garden_acl*
	Example:
	`RP/0/RSP0/CPU0:router(config-cmap)# match not access-group ipv4 open_garden_acl`
	or
	`RP/0/RSP0/CPU0:router(config-cmap)# match not access-group ipv6 open_garden_acl`
		Identifies a specified access control list (ACL) number as the match criteria for a class map.
		Note The redirect acl name provided in this step is the one configured in the configuration step mentioned in the prerequisites.

Step 4	**end-class-map** **Example:** `RP/0/RSP0/CPU0:router(config-cmap)# end-class-map`	Ends the configuration of match criteria for the class and exits the class map configuration mode.
Step 5	**class-map type traffic match-all** *http_redirect-class_name* **Example:** `RP/0/RSP0/CPU0:router(config)# class-map type traffic match-all RCL1`	Defines a traffic class and the associated rules that match packets to the class for a open garden class.

Step 6	**match** [**not**] **access-group** {**ipv4** | **ipv6**} *redirect_acl*
	Example:
	`RP/0/RSP0/CPU0:router(config-cmap)# match not access-group ipv4 redirect-acl`
	or
	`RP/0/RSP0/CPU0:router(config-cmap)# match not access-group ipv6 redirect-acl`
		Identifies a specified access control list (ACL) number as the match criteria for a class map.
		Note The redirect acl name provided in this step is the one configured in the configuration step mentioned in the prerequisites.

Step 7	**end-class-map** **Example:** `RP/0/RSP0/CPU0:router(config-cmap)# end-class-map`	Ends the configuration of match criteria for the class and exits the class map configuration mode.
Step 8	**commit**	

Configuring Class Maps for HTTP Redirection: An example

```
configure
class-map type traffic [match-any | match-all] <open-garden-class>
match [not] access-group ipv4 allow-acl
end-class-map

class-map type traffic [match-any | match-all] <http-redirect-class>
match [not] access-group ipv4 redirect-acl
end-class-map
  !
 !
!
end


configure
class-map type traffic [match-any | match-all] <open-garden-class>
match [not] access-group ipv6 allow-acl
end-class-map

class-map type traffic [match-any | match-all] <http-redirect-class>
match [not] access-group ipv6 redirect-acl
end-class-map
  !
 !
!
end
```

Configuring Policy Map for HTTP Redirect

Perform this task to configure policy maps for http redirect.

Before you begin

The configuration steps mentioned in Identifying HTTP Destinations for Redirection, on page 45 and Configuring Class Maps for HTTP Redirection, on page 48 have to be completed before performing the configuration of the policy-map for HTTPR.

SUMMARY STEPS

1. **configure**
2. **policy-map type pbr** *http-redirect_policy_name*
3. **class type traffic** *open_garden_class_name*
4. **transmit**
5. **class type traffic** *http_redirect-class_name*
6. **http-redirect** *redirect_url*
7. **class class-default**
8. **drop**
9. **end-policy-map**
10. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**policy-map type pbr** *http-redirect_policy_name* **Example:** `RP/0/RSP0/CPU0:router(config)# policy-map type pbr RPL1`	Creates a policy map of type policy-based routing that can be attached to one or more interfaces to specify a service policy.
Step 3	**class type traffic** *open_garden_class_name* **Example:** `RP/0/RSP0/CPU0:router(config-pmap)# class type traffic CL1`	Specifies the name of the class whose policy you want to create or change. **Note** The open garden acl name provided in this step is the one configured in the configuration step mentioned in the prerequisites.
Step 4	**transmit** **Example:** `RP/0/RSP0/CPU0:router(config-pmap)# transmit`	Forwards the packet to the original destination.
Step 5	**class type traffic** *http_redirect-class_name* **Example:** `RP/0/RSP0/CPU0:router(config-pmap)# class type traffic RCL1`	Specifies the name of the class whose policy you want to create or change. **Note** The open garden acl name provided in this step is the one configured in the configuration step mentioned in the prerequisites.
Step 6	**http-redirect** *redirect_url* **Example:** `RP/0/RSP0/CPU0:router(config-pmap-c)# http-redirect redirect_url`	Specifies the URL to which the HTTP requests should be redirected.
Step 7	**class class-default** **Example:** `RP/0/RSP0/CPU0:router(config-pmap)# class class-default`	Configures default classes that cannot be used with user-defined classes.
Step 8	**drop** **Example:** `RP/0/RSP0/CPU0:router(config-pmap)# drop`	Drops the packet.
Step 9	**end-policy-map** **Example:**	Ends the configuration of a policy map and exits the policy map configuration mode.

	Command or Action	Purpose
	`RP/0/RSP0/CPU0:router(config-cmap)# end-policy-map`	
Step 10	**commit**	

Configuring Policy Map for HTTP Redirect: An example

```
configure
policy-map type pbr <http-redirect-policy>
class type traffic <open-garden-class>
transmit
   !
class type traffic <http-redirect-class>
http-redirect <redirect-url>
!
class class-default
drop
  !
end-policy-map
  !
 !
!
end
```

Configuring Dynamic Template for Applying HTTPR Policy

Perform this task to configure dynamic template for applying the HTTPR policy to subscriber sessions.

Before you begin

The configuration steps mentioned in Configuring Policy Map for HTTP Redirect, on page 50 have to be completed before defining the dynamic template that uses a previously defined policy-map.

Note Ensure that the Dynamic template contains only the Policy Based Routing policy, so it can be easily deactivated after web login.

SUMMARY STEPS

1. **configure**
2. **dynamic-template type ipsubscriber** *redirect_template_name*
3. **service-policy type pbr** *http-redirect-policy*
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

	Command or Action	Purpose
Step 2	**dynamic-template type ipsubscriber** *redirect_template_name* **Example:** `RP/0/RSP0/CPU0:router(config)# dynamic-template type ipsubscriber RDL1`	Creates a dynamic template of type "ipsubscriber".
Step 3	**service-policy type pbr** *http-redirect-policy* **Example:** `RP/0/RSP0/CPU0:router(config-pmap)# service-policy type pbr RPL1`	Attaches the service policy as a pbr type within a policy map created in the earlier configuration. **Note** The http redirect policy name provided in this step is the one configured in the configuration step mentioned in the prerequisites.
Step 4	**commit**	

Configuring Dynamic Template for Applying HTTPR Policy: An example

```
configure
dynamic-template type ip <redirect-template>
service-policy type pbr <http-redirect-policy>
  !
  !
!
end
```

Configuring Web Logon

Perform this task to configure Web Logon. As an example, a timer defines the maximum time permitted for authentication.

SUMMARY STEPS

1. **configure**
2. **class-map type control subscriber match-all** *classmap_name*
3. **match timer** *name*
4. **match authen-status authenticated**
5. **policy-map type control subscriber** *policymap_name*
6. **event session-start match-all**
7. **class type control subscriber** *class_name* **do-until-failure**
8. *sequence_number* **activate dynamic-template** *dt_name*
9. *sequence_number* **activate dynamic-template** *dt_name*
10. *sequence_number* **set-timer** *timer_name value*
11. **event account-logon match-all**
12. **class type control subscriber** *class_name* **do-until-failure**
13. *sequence_number* **authenticate aaa list default**
14. *sequence_number* **deactivate dynamic-template** *dt_name*

15. *sequence_number* **stop-timer** *timer_name*
16. **event time-expiry match-all**
17. **class type control subscriber** *class_name* **do-all**
18. *sequence_number* **disconnect**
19. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**class-map type control subscriber match-all** *classmap_name* **Example:** `RP/0/RSP0/CPU0:router(config)# class-map type control subscriber match-all IP_UNAUTH_COND`	Configures a subscriber control class-map with the match-all match criteria.
Step 3	**match timer** *name* **Example:** `RP/0/RSP0/CPU0:router(config-cmap)# match timer AUTH_TIMER`	Configures a match criteria for the class along with timer details.
Step 4	**match authen-status authenticated** **Example:** `RP/0/RSP0/CPU0:router(config-cmap)# match timer AUTH_TIMER`	Configures a match criteria for the class along with authentication status details.
Step 5	**policy-map type control subscriber** *policymap_name* **Example:** `RP/0/RSP0/CPU0:router(config)# class-map type control subscriber match-all RULE_IP_WEBSESSION`	Configures a subscriber control policy-map.
Step 6	**event session-start match-all** **Example:** `RP/0/RSP0/CPU0:router(config-pmap)# event session-start match-all`	Configures the session start policy event that runs all the matched classes.
Step 7	**class type control subscriber** *class_name* **do-until-failure** **Example:** `RP/0/RSP0/CPU0:router(config-pmap-e)# class type control subscriber class-default do-until-failure`	Configures the class to which the subscriber is to be matched. When there is a match, execute all actions that follow until a failure is encountered.
Step 8	*sequence_number* **activate dynamic-template** *dt_name* **Example:** `RP/0/RSP0/CPU0:router(config-pmap-c)# 10 activate dynamic-template DEFAULT_IP_SERVICE`	Activates the dynamic-template defined locally on the CLI with the specified dynamic template name.

	Command or Action	Purpose
Step 9	*sequence_number* **activate dynamic-template** *dt_name* **Example:** `RP/0/RSP0/CPU0:router(config-pmap-c)# 10 activate dynamic-template HTTP_REDIRECT`	Activates the dynamic-template defined locally on the CLI with the specified dynamic template name.
Step 10	*sequence_number* **set-timer** *timer_name value* **Example:** `RP/0/RSP0/CPU0:router(config-pmap-c)# 10 set-timer AUTH_TIMER 4567`	Sets a timer to run a rule on its expiry. The timer value, specified in minutes, ranges from 0 to 4294967295.
Step 11	**event account-logon match-all** **Example:** `RP/0/RSP0/CPU0:router(config-pmap)# event session-start match-all`	Configures the account logon policy event that runs all matched classes.
Step 12	**class type control subscriber** *class_name* **do-until-failure** **Example:** `RP/0/RSP0/CPU0:router(config-pmap-e)# class type control subscriber class-default do-until-failure`	Configures the class to which the subscriber is to be matched. When there is a match, execute all actions that follow, until a failure is encountered.
Step 13	*sequence_number* **authenticate aaa list default** **Example:** `RP/0/RSP0/CPU0:router(config-pmap-c)# 10 authenticate aaa list default`	Specifies and authenticates the default AAA method list.
Step 14	*sequence_number* **deactivate dynamic-template** *dt_name* **Example:** `RP/0/RSP0/CPU0:router(config-pmap-c)# 10 deactivate dynamic-template HTTP_REDIRECT`	Disables the timer before it expires.
Step 15	*sequence_number* **stop-timer** *timer_name* **Example:** `RP/0/RSP0/CPU0:router(config-pmap-c)# 20 stop-timer AUTH_TIMER`	Disables the timer before it expires.
Step 16	**event time-expiry match-all** **Example:** `RP/0/RSP0/CPU0:router(config-pmap)# event session-start match-all`	Configures the timer expiry policy event that runs all the matched classes.
Step 17	**class type control subscriber** *class_name* **do-all** **Example:** `RP/0/RSP0/CPU0:router(config-pmap-e)# class type control subscriber IP_UNAUTH_COND do-all`	Configures the class to which the subscriber has to be matched. When there is a match, execute all actions.

	Command or Action	Purpose
Step 18	*sequence_number* **disconnect** **Example:** `RP/0/RSP0/CPU0:router(config-pmap-c)# 10 disconnect`	Disconnects the session.
Step 19	**commit**	

Configuring Web Logon: An example

This example illustrates an IP session that is HTTP-redirected to an authentication web-portal for credentials. On successful authentication, the timer is unset. Otherwise, the subscriber gets disconnected when the timer window expires:

```
class-map type control subscriber match-all IP_UNAUTH_COND
   match timer AUTH_TIMER
   match authen-status unauthenticated

policy-map type control subscriber RULE_IP_WEBSESSION
    event session-start match-all
        class type control subscriber class-default do-until-failure
            10 activate dynamic-template DEFAULT_IP_SERVICE
            20 activate dynamic-template HTTP_REDIRECT
            30 set-timer AUTH_TIMER 5

    event account-logon match-all
        class type control subscriber class-default do-until-failure
            10 authenticate aaa list default
            15 deactivate dynamic-template HTTP_REDIRECT
            20 stop-timer AUTH_TIMER

    event timer-expiry match-all
        class type control subscriber IP_UNAUTH_COND do-all
            10 disconnect
```

Idle Timeout for IPoE and PPPoE Sessions

The Idle Timeout feature for IPoE and PPPoE sessions allows users to configure a maximum period of time that the subscriber sessions may remain idle. The subscriber sessions are terminated when this timeout period expires. The BNG monitors both the ingress and egress traffic for the determination of the idle time for the subscriber sessions. Control packets are not considered while determining session inactivity.

You can configure a threshold rate, and if packets sent or received by BNG in that interval is less than this threshold rate, then that particular session is considered idle. The threshold option allows you to consider low traffic rates as being idle and to exclude DHCP lease renewal packets from the statistics used for idle time determination. For instance, if you want to discount the DHCP short lease of 5 minutes, then you must configure the threshold as 5 packets per minute.

The dynamic template configuration of idle timeout is extended to also support **type ppp** templates. If idle timeout is enabled and if **monitor** action is not specified under the idle timeout event for a subscriber policy, then, by default, the sessions are disconnected. You can prevent the sessions from getting disconnected, by setting, for that particular subscriber policy, the policy action under the idle timeout event as **monitor**.

These Cisco VSAs are used to configure or update the idle timeout threshold and traffic direction from the RADIUS server:

```
idlethreshold = <mins/pkt>
idle-timeout-direction = <inbound | outbound | both>
```

For details on configuring idle timeout, see Creating Dynamic Template for IPv4 or IPv6 Subscriber Session.

For details on configuring a policy-map with the idle-timeout event, see Configuring a Policy-Map.

Routing Support on Subscriber Sessions

Routing support on subscriber sessions allows dynamic routes to be added on an individual subscriber basis for IPoE sessions. This allows to forward traffic from the default Virtual Routing and Forwarding (VRF) towards the subscriber, or to access the routes behind the subscriber. As opposed to static routes, dynamic routes must be added and removed when subscribers are created and deleted. Dynamic routes can belong to a VRF other than that of the subscriber and they are supported for IPv4 subscribers only.

Dynamic routes that are to be added for each subscriber are configured as part of the RADIUS profile of the subscriber. The subscriber sessions are not disconnected even if the dynamic route insertion fails. Instead, the route addition is re-tried at regular intervals.

The format of the Cisco:Avpair used for configuring the dynamic routes is:

```
Cisco:Avpair = "Framed-Route={vrf} [<destination_vrf>] {<prefix>} {<mask>} {vrf}
[<next_hop_vrf>] {<next_hop_ip_address>} [<admin_distance>] [tag <tag_value>]"
```

For example :

```
Cisco:Avpair = "Framed-Route=vrf vrfv1 10.121.1.254 255.255.255.255 vrf vrfv2 10.121.1.254
 30 tag 12"
```

In this example, the route for 10.121.1.254/32 is added to the vrfv1 with a next-hop of 10.212.1.254 in vrfv2. The route has an admin distance of 30 and a route tag value of 12.

Benefits of Routing Support on Subscriber Sessions

These are some of the benefits of routing support on subscriber sessions:

- Multiple dynamic routes for each subscriber are supported.
- The user can specify the destination VRF name and next-hop VRF name for each route to be added, and both can be different from the VRF of the subscriber. If the destination VRF is not specified, the VRF of the subscriber is taken as the default. If the next-hop VRF is not specified, the same VRF as that of the destination prefix is taken as the default.
- The user can specify the admin distance and tag to be used for the dynamic route.
- Dynamic routes are added as subscriber routes, with a default admin distance of 3.
- Dynamic routes are always recursive routes.
- Dynamic routes are CoA attribute and therefore, they can be changed while the subscriber is connected to the BNG router.

Traffic Mirroring on Subscriber Session

BNG supports the Traffic Mirroring feature on subscriber session. Traffic mirroring, also known as Switched Port Analyzer (SPAN), enables a user to monitor Layer 2 network traffic passing in or out of a set of Ethernet interfaces. This allows the mirroring of packets that pass through a source interface to a specified destination interface. The destination interface may then be attached to a network analyzer for debugging.

Traffic Mirroring or Switched Port Analyzer (SPAN) has these two distinct sets of configurations:

- Global configuration to create monitor sessions - A session is configured by specifying a session type and a destination that can be a local interface or a pseudo-wire interface.
- Source interface attachment configuration - This specifies how an interface should be attached to a monitor session.

For BNG, the source interface attachment configuration to a monitor session is through the use of dynamic templates. The subscriber is attached to the monitor session only when the template is applied to the subscriber. The template is applied or removed using the **activate-service** or **deactivate-service** CoA command sent from the RADIUS server or using the **test radius coa [activate | deactivate]** command.

For more information on Traffic Mirroring feature, see *Configuring Traffic Mirroring on the Cisco ASR 9000 Series Router* chapter in the *Interface and Hardware Component Configuration Guide for Cisco ASR 9000 Series Routers*. For complete command reference of the SPAN commands, see the *Traffic Mirroring Commands on the Cisco ASR 9000 Series Router* chapter in the *Interface and Hardware Component Command Reference for Cisco ASR 9000 Series Routers*.

For configuring traffic mirroring on BNG subscriber session, see Enabling Traffic Mirroring on Subscriber Session, on page 58.

Note

- It is recommended that a dynamic template is dedicated to SPAN configuration, so that SPAN can be enabled or disabled on a subscriber without any adverse impact.
- Modifications to SPAN configuration under a dynamic template, including the removal of configuration, have an immediate effect on all the subscribers to which that template is currently applied.

Enabling Traffic Mirroring on Subscriber Session

Perform this task to enable traffic mirroring on BNG subscriber session. These steps describe how to configure a dynamic template that references the monitor session and to associate or dis-associate it with a specific subscriber to enable or disable SPAN.

Before you begin

Create monitor sessions in global configuration mode using **monitor-session** command. Refer, Traffic Mirroring on Subscriber Session, on page 58

SUMMARY STEPS

1. **configure**

2. **dynamic-template type** {**ipsubscriber** | **ppp** | **service**} *dynamic-template-name*
3. Configure **monitor-session**, with optional **direction**, **acl** and **mirror first** options
4. **commit**
5. **test radius coa** {**activate** | **deactivate**} **service** *name* **acct-ses-id** *name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**dynamic-template type** {**ipsubscriber** | **ppp** | **service**} *dynamic-template-name* **Example:** `RP/0/RSP0/CPU0:router(config)# dynamic-template type ppp ppp_template`	Creates a dynamic-template of type **ppp**.

Step 3	Configure **monitor-session**, with optional **direction**, **acl** and **mirror first** options
	Example:
	`RP/0/RSP0/CPU0:router(config-dynamic-template-type)# monitor-session mon1 direction rx-only`
	`RP/0/RSP0/CPU0:router(config-dynamic-template-type)# acl`
	`RP/0/RSP0/CPU0:router(config-dynamic-template-type)# mirror first 100`
		Configures a dynamic template that references the monitor session.
		Note This syntax of monitor-session command for dynamic templates is same as the syntax for regular interfaces.

Step 4	**commit**	
Step 5	**test radius coa** {**activate** | **deactivate**} **service** *name* **acct-ses-id** *name* **Example:** `RP/0/RSP0/CPU0:router# test radius coa activate acct-ses-id 0x00000001 service service1`	If **activate** keyword is used, this command enables SPAN by associating a dynamic template with a specific subscriber. If **deactivate** keyword is used, this command disables SPAN by dis-associating a dynamic template with a specific subscriber.

Enabling Traffic Mirroring on Subscriber Session: An example

```
//Global configuration to create monitor sessions
configure
monitor-session mon1
destination interface gigabitethernet0/0/0/1
ethernet-services access-list tm_filter
 10 deny 0000.1234.5678 0000.abcd.abcd any capture
 !
!

//Configuring a dynamic template that references the monitor session
configure
dynamic-template type ppp ppp_template
 monitor-session mon1 direction rx-only
 acl
```

```
 mirror first 100
!
!

//Associating a dynamic-template with a specific subscriber to enable SPAN
test radius coa activate acct-ses-id 0x00000001 service service1
```

Randomization of Interim Timeout of Sessions or Services

The randomization feature distributes the interim timeouts in a relatively uniform manner and prevents accumulation of timeouts for interim accounts of sessions or services. This prevents a cycle where all messages are sent at once (this occurs if a primary link was recently restored and many dial-up users were directed to the same BNG at once). This is useful in scenarios such as churn scenarios of session bring up (that is, a small spurt with very high session bring up rate), subscriber redundancy group (SRG) slave to master switchover in BNG geo redundancy and so on.

For example, if a session is brought up at time 0, and it has an interim interval of 10 minutes (600 seconds), the first interim message is sent at time t1 = 600 seconds (this is without randomization enabled). With randomization enabled, a random number *x* which is less than 600 is selected and the first interim message is sent at that time, *x*. Use this command to specify the maximum variance allowed:

accounting interim variation

Sample configuration:

```
subscriber
 manager
  accounting interim variation 10
```

Additional References

These sections provide references related to implementing BNG subscriber features.

MIBs

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

Technical Assistance

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