Implementing Cisco Express Forwarding

Cisco Express Forwarding (CEF) is advanced, Layer 3 IP switching technology. CEF optimizes network performance and scalability for networks with large and dynamic traffic patterns, such as the Internet, on networks characterized by intensive web-based applications, or interactive sessions.

This module describes the tasks required to implement CEF on your Cisco ASR 9000 Series Aggregation Services Router.

Note For complete descriptions of the CEF commands listed in this module, refer to the *Cisco ASR 9000 Series Aggregation Services Router IP Addresses and Services Command Reference*.

Feature History for Implementing CEF

Release	Modification
Release 3.7.2	This feature was introduced.
Release 6.0.1	IPv6 Flow Label Field for Hashing feature was introduced.

Prerequisites for Implementing Cisco Express Forwarding

The following prerequisites are required to implement Cisco Express Forwarding:

- You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Information About Implementing Cisco Express Forwarding Software

To implement Cisco Express Forwarding features in this document you must understand the following concepts:

Key Features Supported in the Cisco Express Forwarding Implementation

The following features are supported for CEF on Cisco IOS XR software:

- Border Gateway Protocol (BGP) policy accounting

- Reverse path forwarding (RPF)

- Virtual interface support

- Multipath support

- Route consistency

- High availability features such as packaging, restartability, and Out of Resource (OOR) handling

- OSPFv2 SPF prefix prioritization

- BGP attributes download

Benefits of CEF

CEF offers the following benefits:

- Improved performance—CEF is less CPU-intensive than fast-switching route caching. More CPU processing power can be dedicated to Layer 3 services such as quality of service (QoS) and encryption.

- Scalability—CEF offers full switching capacity at each line card.

- Resilience—CEF offers an unprecedented level of switching consistency and stability in large dynamic networks. In dynamic networks, fast-switched cache entries are frequently invalidated due to routing changes. These changes can cause traffic to be process switched using the routing table, rather than fast switched using the route cache. Because the Forwarding Information Base (FIB) lookup table contains all known routes that exist in the routing table, it eliminates route cache maintenance and the fast-switch or process-switch forwarding scenario. CEF can switch traffic more efficiently than typical demand caching schemes.

CEF Components

Cisco IOS XR softwareCEF always operates in CEF mode with two distinct components: a Forwarding Information Base (FIB) database and adjacency table—a protocol-independent adjacency information base (AIB).

CEF is a primary IP packet-forwarding database for Cisco IOS XR software. CEF is responsible for the following functions:

- Software switching path

- Maintaining forwarding table and adjacency tables (which are maintained by the AIB) for software and hardware forwarding engines

The following CEF forwarding tables are maintained in Cisco IOS XR software:

- IPv4 CEF database

- IPv6 CEF database

- MPLS LFD database

- Multicast Forwarding Table (MFD)

The protocol-dependent FIB process maintains the forwarding tables for IPv4 and IPv6 unicast in the Route Switch Processor (RSP) and each MSC.

The FIB on each node processes Routing Information Base (RIB) updates, performing route resolution and maintaining FIB tables independently in the RSP and each MSC. FIB tables on each node can be slightly different. Adjacency FIB entries are maintained only on a local node, and adjacency entries linked to FIB entries could be different.

Border Gateway Protocol Policy Accounting

Border Gateway Protocol (BGP) policy accounting measures and classifies IP traffic that is sent to, or received from, different peers. Policy accounting is enabled on an individual input or output interface basis, and counters based on parameters such as community list, autonomous system number, or autonomous system path are assigned to identify the IP traffic.

Note There are two types of route policies. The first type (regular BGP route policies) is used to filter the BGP routes advertised into or out from the BGP links. This type of route policy is applied to the specific BGP neighbor. The second type (specific route policy) is used to set up a traffic index for the BGP prefixes. This route policy is applied to the global BGP IPv4 address family to set up the traffic index when the BGP routes are inserted into the RIB table. BGP policy accounting uses the second type of route policy.

Using BGP policy accounting, you can account for traffic according to the route it traverses. Service providers can identify and account for all traffic by customer and bill accordingly. In Figure 1: Sample Topology for BGP Policy Accounting, on page 4, BGP policy accounting can be implemented in Router A to measure packet and byte volumes in autonomous system buckets. Customers are billed appropriately for traffic that is routed from a domestic, international, or satellite source.
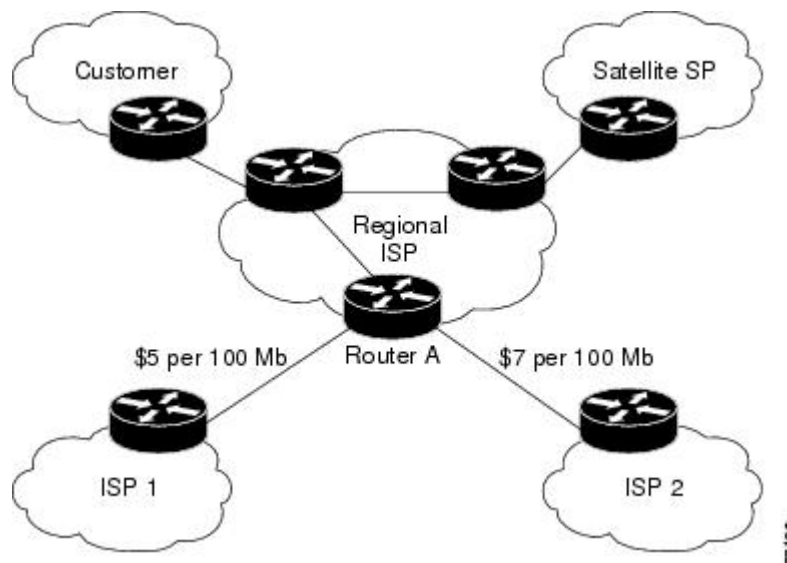
Note BGP policy accounting measures and classifies IP traffic for BGP prefixes only.

Figure 1: Sample Topology for BGP Policy Accounting

Based on the specified routing policy, BGP policy accounting assigns each prefix a traffic index (bucket) associated with an interface. BGP prefixes are downloaded from the RIB to the FIB along with the traffic index.

There are a total of 63 (1 to 63) traffic indexes (bucket numbers) that can be assigned for BGP prefixes. Internally, there is an accounting table associated with the traffic indexes to be created for each input (ingress) and output (egress) interface. The traffic indexes allow you to account for the IP traffic, where the source IP address, the destination IP address, or both are BGP prefixes.

Note Traffic index 0 contains the packet count using Interior Gateway Protocol (IGP) routes.

Reverse Path Forwarding (Strict and Loose)

Unicast IPv4 and IPv6 Reverse Path Forwarding (uRPF), both strict and loose modes, help mitigate problems caused by the introduction of malformed or spoofed IP source addresses into a network by discarding IP packets that lack a verifiable IP source address. Unicast RPF does this by doing a reverse lookup in the CEF table. Therefore, Unicast Reverse Path Forwarding is possible only if CEF is enabled on the router.

IPv6 uRPF is supported with ASR 9000-SIP-700 LC, ASR 9000 Ethernet LC and ASR 9000 Enhanced Ethernet LC.

Note Unicast RPF allows packets with 0.0.0.0 source addresses and 255.255.255.255 destination addresses to pass so that Bootstrap Protocol and Dynamic Host Configuration Protocol (DHCP) will function properly.

When strict uRPF is enabled, the source address of the packet is checked in the FIB. If the packet is received on the same interface that would be used to forward the traffic to the source of the packet, the packet passes the check and is further processed; otherwise, it is dropped. Strict uRPF should only be applied where there is natural or configured symmetry. Because internal interfaces are likely to have routing asymmetry, that is,

multiple routes to the source of a packet, strict uRPF should not be implemented on interfaces that are internal to the network.

> **Note** The behavior of strict RPF varies slightly by platform, number of recursion levels, and number of paths in Equal-Cost Multipath (ECMP) scenarios. A platform may switch to loose RPF check for some or all prefixes, even though strict RPF is configured.

When loose uRPF is enabled, the source address of the packet is checked in the FIB. If it exists and matches a valid forwarding entry, the packet passes the check and is further processed; otherwise, it is dropped.

Strict mode uRPF requires maintenance of uRPF interfaces list for the prefixes. The list contains only strict mode uRPF configured interfaces pointed by the prefix path. uRPF interface list is shared among the prefixes wherever possible. Size of this list is 12 for ASR 9000 Ethernet Line Cards and 64 for integrated 20G SIP cards. Strict to loose mode uRPF fallback happens when the list goes beyond the maximum supported value.

Loose and strict uRPF supports two options: **allow self-ping** and **allow default**. The **self-ping** option allows the source of the packet to ping itself. The **allow default** option allows the lookup result to match a default routing entry. When the **allow default** option is enabled with the strict mode of the uRPF, the packet is processed further only if it arrived through the default interface.

Per-Flow Load Balancing

Load balancing describes the functionality in a router that distributes packets across multiple links based on Layer 3 (network layer) and Layer 4 (transport layer) routing information. If the router discovers multiple paths to a destination, the routing table is updated with multiple entries for that destination.

Per-flow load balancing performs these functions:

- Incoming data traffic is evenly distributed over multiple equal-cost connections within a bundle interface.

- Layer 2 bundle and Layer 3 (network layer) load balancing decisions are taken on IPv4, IPv6, and MPLS flows which are supported for the 5-tuple hash algorithm.

- A 5-tuple hash algorithm provides more granular load balancing than the 3-tuple hash algorithm.

- The same hash algorithm (3-tuple or 5-tuple) is used for load balancing over multiple equal-cost Layer 3 (network layer) paths. The Layer 3 (network layer) path is on a physical interface or on a bundle interface. In addition, load balancing over member links can occur within a Layer 2 bundle interface.

- The **cef load-balancing fields L3 global** command allows you to select the 3-tuple hash algorithm.

- By default, 5-tuple hash algorithm is used for load balancing. If you use the **cef load-balancing fields L3 global** command, 3-tuple hash algorithm is enabled.

Layer 3 (Network Layer) Routing Information

The 3-tuple load-balance hash calculation contains these Layer 3 (Network Layer) inputs:

- Source IP address

- Destination IP address

- Router ID

Layer 4 (Transport Layer) Routing Information

The 5-tuple load-balance hash calculation contains 3-tuple inputs and these additional following Layer 4 (Transport Layer) inputs:

- Source port

- Destination port

Note In load-balancing scenarios, a line card may not use all output paths downloaded from routing protocols. This behavior varies with platform, number of recursion levels, and the fact whether MPLS is involved, or not.

Limitations for 3-Tuple Load Balance Hash Algorithm

- Load balancing is not symmetrical if the load balanced paths involve different speeds such as, 10 Gigabit Ethernet, 40 Gigabit Ethernet, or 100 Gigabit Ethernet ports.

- Because the 3-tuple hash algorithm excludes Layer 4 information and becomes dependent on Layer 3 information, load balancing is not symmetrical if the distribution of source and destination IP addresses is not varied enough, even if all ports operate at the same speed.

- You can configure 3-tuple hash algorithm only on Cisco ASR 9000 Enhanced Ethernet Line Cards.

IPv6 Flow Label Field for Hashing

You can use the 20-bit Flow Label field in the IPv6 header as an additional input field for hash algorithms to improve load balancing decisions. Use the **cef load-balancing fields ipv6 flow-label** command to enable the flow label field for IPv6 packets on Cisco ASR 9000 High Density 100GE Ethernet Line Cards and Cisco ASR 9000 Enhanced Ethernet Line Cards. The hash algorithms can use the flow label field in addition to inputs such as source IP address, destination IP address, router ID, source port, and destination port. See RFC 6437 for details about the IPv6 Flow Label Specification.

BGP Attributes Download

The BGP Attributes Download feature enables you to display the installed BGP attributes in CEF. Configure the **show cef bgp-attribute** command to display the installed BGP attributes in CEF. You can use the **show cef bgp-attribute attribute-id** command and the **show cef bgp-attribute local-attribute-id** command to look at specific BGP attributes by attribute ID and local attribute ID.

Verification

```
Router# show cef bgp-attribute
Wed Aug 21 14:05:51.772 UTC

VRF: default

Table ID: 0xe0000000. Total number of entries: 1
OOR state: GREEN. Number of OOR attributes: 0

BGP Attribute ID: 0x6, Local Attribute ID: 0x1
    Aspath      :    2
    Community   :
```

```
         Origin AS   :    2
         Next Hop AS :    2
```

How to Implement CEF

This section contains instructions for the following tasks:

Verifying CEF

This task allows you to verify CEF.

SUMMARY STEPS

1. **show cef** {**ipv4** | **ipv6**}
2. **show cef** {**ipv4** | **ipv6**} **summary**
3. **show cef** {**ipv4** | **ipv6**} **detail**
4. show adjacency detail

DETAILED STEPS

	Command or Action	Purpose
Step 1	**show cef** {**ipv4**	**ipv6**} **Example:** `RP/0/RSP0/CPU0:router# show cef ipv4`	Displays the IPv4 or IPv6 CEF table. The next hop and forwarding interface are displayed for each prefix. **Note** The output of the **show cef** command varies by location.
Step 2	**show cef** {**ipv4**	**ipv6**} **summary** **Example:** `RP/0/RSP0/CPU0:router# show cef ipv4 summary`	Displays a summary of the IPv4 or IPv6 CEF table.
Step 3	**show cef** {**ipv4**	**ipv6**} **detail** **Example:** `RP/0/RSP0/CPU0:router# show cef ipv4 detail`	Displays detailed IPv4 or IPv6 CEF table information.
Step 4	show adjacency detail **Example:** `RP/0/RSP0/CPU0:router# show adjacency detail`	Displays detailed adjacency information, including Layer 2 information for each interface. **Note** The output of the **show adjacency** command varies by location.

Configuring BGP Policy Accounting

This task allows you to configure BGP policy accounting.

Note There are two types of route policies. BGP policy accounting uses the type that is used to set up a traffic index for the BGP prefixes. The route policy is applied to the global BGP IPv4 address family to set up the traffic index when the BGP routes are inserted into the RIB table.

BGP policy accounting enables per interface accounting for ingress and egress IP traffic based on the traffic index assigned to the source IP address (BGP prefix) and destination IP address (BGP prefix). The traffic index of BGP prefixes can be assigned according to the following parameters using Routing Policy Language (RPL):

- prefix-set

- AS-path-set

- community-set

Note BGP policy accounting is supported on IPv4 prefixes only.

Two configuration tasks provide the ability to classify BGP prefixes that are in the RIB according to the prefix-set, AS-path-set, or the community-set parameters:

1. Use the **route-policy** command to define the policy for traffic index setup based on the prefix-set, AS-path-set, or community-set.

2. Use the BGP **table-policy** command to apply the defined route policy to the global BGP IPv4 unicast address family.

See the *Routing Command Reference for Cisco ASR 9000 Series Routers* for information on the **route-policy** and **table-policy** commands.

BGP policy accounting can be enabled on each interface with the following options:

- Use the ipv4 bgp policy accounting command with one of the following keyword options:

  - input source-accounting

  - input destination-accounting

  - input source-accounting destination-accounting

- Use the ipv4 bgp policy accounting command with one of the following keyword options:

  - output source-accounting

  - output destination-accounting

  - output source-accounting destination-accounting

- Use any combination of the keywords provided for the **ipv4 bgp policy accounting** command.

Before you begin

Before using the BGP policy accounting feature, you must enable BGP on the router (CEF is enabled by default). See the *Routing Configuration Guide for Cisco ASR 9000 Series Routers* for information on enabling BGP.

SUMMARY STEPS

1. configure
2. as-path-set
3. exit
4. **prefix-set** *name*
5. exit
6. **route-policy** *policy-name*
7. end
8. configure
9. **router bgp** *autonomous-system-number*
10. **address-family ipv4** {**unicast** | **multicast** }
11. **table policy** *policy-name*
12. end
13. configure
14. **interface** *type interface-path-id*
15. **ipv4 bgp policy accounting** {**input** | **output** {**destination-accounting** [**source-accounting**] | **source-accounting** [**destination-accounting**]}}
16. Do one of the following:

    - end
    - commit

DETAILED STEPS

	Command or Action	**Purpose**
Step 1	configure **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.

Step 2	as-path-set
	
	Example:
	
	`RP/0/RSP0/CPU0:router(config)# as-path-set as107`
	`RP/0/RSP0/CPU0:router(config-as)# ios-regex '107$'`
	`RP/0/RSP0/CPU0:router(config-as)# end-set`
	
	`RP/0/RSP0/CPU0:router(config)# as-path-set as108`
	`RP/0/RSP0/CPU0:router(config-as)# ios-regex '108$'`
	`RP/0/RSP0/CPU0:router(config-as)# end-set`
		Enters policy configuration mode.

Step 3	exit **Example:**	Returns to global configuration mode.

	Command or Action	Purpose
	RP/0/RSP0/CPU0:router(config-as)# exit	
Step 4	**prefix-set** *name* **Example:** RP/0/RSP0/CPU0:router(config)# prefix-set RT-65	Defines the prefix list.
Step 5	exit **Example:** RP/0/RSP0/CPU0:router(config-pfx)# exit	Returns to global configuration mode.
Step 6	**route-policy** *policy-name* **Example:** RP/0/RSP0/CPU0:router(config)# route-policy rp501b	Specifies the route-policy name.
Step 7	end **Example:** RP/0/RSP0/CPU0:router(config-rpl)# end	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: Uncommitted changes found, commit them before exiting(yes/no/cancel)?[cancel]: • Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
Step 8	configure **Example:** RP/0/RSP0/CPU0:router# configure	Enters global configuration mode.
Step 9	**router bgp** *autonomous-system-number* **Example:** RP/0/RSP0/CPU0:router(config)# router bgp 1	Allows you to configure the BGP routing process.
Step 10	**address-family ipv4** {**unicast**	**multicast** } **Example:**	Allows you to enter the address family configuration mode while configuring a BGP routing session.

	Command or Action	**Purpose**
	RP/0/RSP0/CPU0:router(config-bgp)# address-family ipv4 unicast	
Step 11	**table policy** *policy-name* **Example:** RP/0/RSP0/CPU0:router(config-bgp-af)# table-policy set-traffic-index	Applies a routing policy to routes being installed into the routing table.

Step 12	end
	
	Example:
	
	RP/0/RSP0/CPU0:router(config-bgp-af)# end
		Saves configuration changes.
		
		• When you issue the **end** command, the system prompts you to commit changes:
		
		   Uncommitted changes found, commit them before exiting(yes/no/cancel)?[cancel]:
		
		   • Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		   • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		   • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.

Step 13	configure **Example:** RP/0/RSP0/CPU0:router# configure	Enters global configuration mode.
Step 14	**interface** *type interface-path-id* **Example:** RP/0/RSP0/CPU0:router(config)# interface TenGigE0/1/0/2	Enters interface configuration mode.
Step 15	**ipv4 bgp policy accounting** {**input**	**output** {**destination-accounting** [**source-accounting**]	**source-accounting** [**destination-accounting**]}} **Example:** RP/0/RSP0/CPU0:router(config-if)# ipv4 bgp policy accounting output destination-accounting	Enables BGP policy accounting.

	Command or Action	**Purpose**

Step 16	Do one of the following:
	
	   • end
	   • commit
	
	Example:
	RP/0/RSP0/CPU0:router(config-if)# end
	
	or
	
	RP/0/RSP0/CPU0:router(config-if)# commit
		Saves configuration changes.
		
		• When you issue the **end** command, the system prompts you to commit changes:
		
		   Uncommitted changes found, commit them
		   before exiting(yes/no/cancel)?[cancel]:
		
		    • Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		
		    • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		
		    • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Verifying BGP Policy Accounting

This task allows you to verify BGP policy accounting.

> **Note** BGP policy accounting is supported on IPv4 prefixes.

Before you begin

BGP policy accounting must be configured. See the Configuring BGP Policy Accounting, on page 7.

SUMMARY STEPS

1. **show route bgp**
2. **show bgp summary**
3. **show bgp** *ip-address*
4. **show route ipv4** *ip-address*
5. **show cef ipv4** *prefix*
6. **show cef ipv4** *prefix* **detail**
7. **show cef ipv4 interface** *type* *interface-path-id* **bgp-policy-statistics**

DETAILED STEPS

	Command or Action	**Purpose**
Step 1	**show route bgp** **Example:** RP/0/RSP0/CPU0:router# show route bgp	Displays all BGP routes with traffic indexes.
Step 2	**show bgp summary** **Example:** RP/0/RSP0/CPU0:router# show bgp summary	Displays the status of all BGP neighbors.
Step 3	**show bgp** *ip-address* **Example:** RP/0/RSP0/CPU0:router# show bgp 40.1.1.1	Displays BGP prefixes with BGP attributes.
Step 4	**show route ipv4** *ip-address* **Example:** RP/0/RSP0/CPU0:router# show route ipv4 40.1.1.1	Displays the specific BGP route with the traffic index in the RIB.
Step 5	**show cef ipv4** *prefix* **Example:** RP/0/RSP0/CPU0:router# show cef ipv4 40.1.1.1	Displays the specific BGP prefix with the traffic index in the RP FIB.
Step 6	**show cef ipv4** *prefix* **detail** **Example:** RP/0/RSP0/CPU0:router# show cef ipv4 40.1.1.1 detail	Displays the specific BGP prefix with detailed information in the RP FIB.
Step 7	**show cef ipv4 interface** *type* *interface-path-id* **bgp-policy-statistics** **Example:** RP/0/RSP0/CPU0:router# show cef ipv4 interface TenGigE 0/2/0/4 bgp-policy-statistics	Displays the BGP Policy Accounting statistics for the specific interface.

Configuring a Route Purge Delay

This task allows you to configure a route purge delay. A purge delay purges routes when the RIB or other related process experiences a failure.

SUMMARY STEPS

1. **configure**

    2. **cef purge-delay** *seconds*

    3. **commit**

DETAILED STEPS

	Command or Action	**Purpose**
Step 1	**configure**	
Step 2	**cef purge-delay** *seconds* **Example:** RP/0/RSP0/CPU0:router(config)# cef purge-delay 180	Configures a delay in purging routes when the Routing Information Base (RIB) or other related processes experience a failure.
Step 3	**commit**	

Configuring Unicast RPF Checking

This task allows you to configure unicast Reverse Path Forwarding (uRPF) checking. Unicast RPF checking allows you to mitigate problems caused by malformed or forged (spoofed) IP source addresses that pass through a router. Malformed or forged source addresses can indicate denial-of-service (DoS) attacks based on source IP address spoofing.

SUMMARY STEPS

    1. **configure**

    2. **interface** *type interface-path-id*

    3. {**ipv4** | **ipv6**} **verify unicast source reachable-via** {**any** | **rx**} [**allow-default**] [**allow-self-ping**]

    4. **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 GigabitEthernet 0/1/0/0	Enters interface configuration mode.
Step 3	{**ipv4**	**ipv6**} **verify unicast source reachable-via** {**any**	**rx**} [**allow-default**] [**allow-self-ping**] **Example:** RP/0/RSP0/CPU0:router(config-if)# ipv4 verify unicast source reachable-via rx	Enables IPv4 or IPv6 uRPF checking. • The **rx** keyword enables strict unicast RPF checking. If strict unicast RPF is enabled, a packet is not forwarded unless its source prefix exists in the routing table and the output interface matches the interface on which the packet was received.

	Command or Action	Purpose
		• The **allow-default** keyword enables the matching of default routes. This option applies to both loose and strict RPF.
		• The **allow-self-ping** keyword enables the router to ping out an interface. This option applies to both loose and strict RPF.
		Note IPv6 uRPF checking is not supported on ASR 9000 Ethernet line cards.
Step 4	**commit**	

Configuring Modular Services Card-to-Route Processor Management Ethernet Interface Switching

This task allows you to enable MSC-to-RP management Ethernet interface switching.

SUMMARY STEPS

1. **configure**
2. **rp mgmtethernet forwarding**
3. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

Step 2	**rp mgmtethernet forwarding**
	
	Example:
	
	`RP/0/RSP0/CPU0:router(config)# rp mgmtethernet`
	`forwarding`
		Enables switching from the MSC to the route processor Management Ethernet interfaces.

Step 3	**commit**	

Configuring Per-Flow Load Balancing

This section describes the following tasks to configure per-flow load balancing:

Configuring 3-Tuple Hash Algorithm

This task allows you to configure per-flow load balancing for a 3-tuple hash algorithm.

SUMMARY STEPS

1. **configure**
2. **cef load-balancing fields L3 global**
3. **commit**
4. **show running-config**

DETAILED STEPS

	Command or Action	**Purpose**
Step 1	**configure**	

Step 2	**cef load-balancing fields L3 global**
	Example:
	
	`RP/0/RSP0/CPU0:router(config)# cef load-balancing fields L3 global`
		Configures the 3-tuple hashing algorithm for load balancing during forwarding.
		
		The **cef load-balancing fields l3 global** command configures the hash tuple with the following fields:
		
		• Source IP address.
		
		  Destination IP address.
		
		• Router ID.
		
		The following Layer 4 fields are ignored:
		
		• Source port.
		
		  Destination port.
		
		The **cef load-balancing fields l3 global** command is supported only on Cisco ASR 9000 Enhanced Ethernet Line Cards. This command is ignored on the other line cards.

Step 3	**commit**	
Step 4	**show running-config** **Example:** `RP/0/RSP0/CPU0:router# show running config`	Displays the running configuration which contains the load balancing information.

Configuring BGP Attributes Download

This task allows you to configure the BGP Attributes Download feature.

Configuring BGP Attributes Download

SUMMARY STEPS

1. **configure**
2. **cef bgp attribute** {*attribute-id* | *local-attribute-id* }
3. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**cef bgp attribute** {*attribute-id*	*local-attribute-id* } **Example:** `RP/0/RSP0/CPU0:router(config)# cef bgp attribute 508`	Configures a CEF BGP attribute.
Step 3	**commit**	

IPv6 Routing over IPv4 MPLS TE Tunnels

IPv6 routing over IPv4 Multiprotocol Label Switching with Traffic Engineering (MPLS TE) tunnels in the core is achieved by configuring the TE tunnels into the IPv6 Interior Gateway Protocol (IGP) topology as IPv6 forwarding adjacencies.

Figure 2: IPv6 Routing over IPv4 MPLS TE
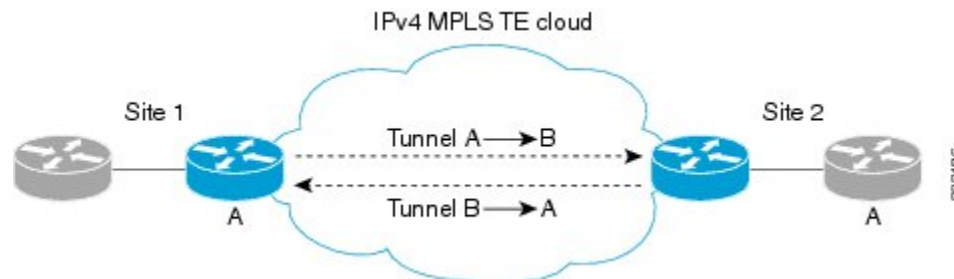
This figure shows two IPv4/IPv6-aware sites connected over a TE core, where TE is not IPv6-aware. Two tunnels are set up across the core, and are announced as forwarding adjacencies into the IPv6 topologies at Site 1 and Site 2. Routers at Site 1 and Site 2 can use these tunnels to compute the best IPv6 route to the other site within their IS-IS SPF.

Restrictions for Implementing IPv6 routing over IPv4 MPLS TE tunnels

The following restrictions apply to implementing IPv6 routing over IPv4 MPLS TE tunnels:

• It is supported for IS-IS only.

• IS-ISv4 and v6 must exist in a single topology.

• IS-ISv4 and v6 must be configured under the same IS-IS instance at the endpoints.

Configuring tunnel as IPV6 Forwarding-Adjacency

Perform this task to configure a tunnel as an IPv6 forwarding adjacency.

SUMMARY STEPS

1. **configure**
2. **interface tunnel-te** *n* **forwarding-adjacency include-ipv6**
3. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

Step 2	**interface tunnel-te** *n* **forwarding-adjacency include-ipv6**
	
	Example:
	
	`RP/0/RSP0/CPU0:router(config)# interface tunnel-te`
	`1 forwarding-adjacency include-ipv6`
		Configures tunnel as an IPV6 Forwarding-Adjacency.

Step 3	**commit**	

Configuring tunnel as IPV6 interface

Perform this task to configure a tunnel as an IPV6 interface.

SUMMARY STEPS

1. **configure**
2. **interface tunnel-te** *n* **ipv6 enable**
3. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

Step 2	**interface tunnel-te** *n* **ipv6 enable**
	
	Example:
	
	`RP/0/RSP0/CPU0:router(config)# interface tunnel-te`
	`1 ipv6 enable`
		Configures tunnel as an IPV6 interface.

Step 3	**commit**	

Configuration Examples for Implementing CEF on Routers Software

This section provides the following configuration examples:

Configuring BGP Policy Accounting: Example

The following example shows how to configure BGP policy accounting.

Configure loopback interfaces for BGP router-id:

```
interface Loopback1
ipv4 address 10
.1.1.1 255.255.255.255
```

Configure interfaces with the BGP policy accounting options:

```
interface TenGigE0/2/0/2
 mtu 1514
 ipv4 address 10
.1.0.1 255.255.255.0
 proxy-arp
 ipv4 directed-broadcast
 ipv4 bgp policy accounting input source-accounting destination-accounting
 ipv4 bgp policy accounting output source-accounting destination-accounting
!
interface TenGigE0/2/0/2.1
 ipv4 address 10
.1.1.1 255.255.255.0
 ipv4 bgp policy accounting input source-accounting destination-accounting
 ipv4 bgp policy accounting output source-accounting destination-accounting
 encapsulation dot1q 1
!
interface TenGigE0/2/0/4
 mtu 1514
 ipv4 address 10
.1.0.1 255.255.255.0
 proxy-arp
 ipv4 directed-broadcast
 ipv4 bgp policy accounting input source-accounting destination-accounting
 ipv4 bgp policy accounting output source-accounting destination-accounting
!
interface TenGigE0/2/0/4.1
 ipv4 address 10
.1.2
.1 255.255.255.0
 ipv4 bgp policy accounting input source-accounting destination-accounting
 ipv4 bgp policy accounting output source-accounting destination-accounting
 encapsulation dot1q 1
!
interface GigabitEthernet 0/0/0/4
 mtu 4474
 ipv4 address 10
.1.0.40
255.255.0.0
 ipv4 directed-broadcast
 ipv4 bgp policy accounting input source-accounting destination-accounting
```

```
 ipv4 bgp policy accounting output source-accounting destination-accounting
 encapsulation ppp
 GigabitEthernet
  crc 32
 !
 keepalive disable
!
interface GigabitEthernet 0/0/0/8
 mtu 4474
 ipv4 address 18
.8
.0.1 255.255.0.0
 ipv4 directed-broadcast
 ipv4 bgp policy accounting input source-accounting destination-accounting
 ipv4 bgp policy accounting output source-accounting destination-accounting
 GigabitEthernet
  crc 32
 !
 keepalive disable
!
```

Configure controller:

```
controller GigabitEthernet 0/0/0/4
 ais-shut
 path
  ais-shut
 !
 threshold sf-ber 5
!
controller SONET0/0/0/8
 ais-shut
 path
  ais-shut
 !
 threshold sf-ber 5
!
```

Configure AS-path-set and prefix-set:

```
as-path-set as107
  ios-regex '107$'
end-set

as-path-set as108
  ios-regex '108$'
end-set

prefix-set RT-65.0
  65.0.0.0/16 ge 16 le 32
end-set

prefix-set RT-66.0
  66.0.0.0/16 ge 16 le 32
end-set
```

Configure the route-policy (table-policy) to set up the traffic indexes based on each prefix, AS-path-set, and prefix-set:

```
route-policy bpa1

  if destination in  (10
.1.1.0/24) then
```

```
      set traffic-index 1
    elseif destination in  (10
.1.2.0/24) then
      set traffic-index 2
    elseif destination in  (10
.1.3.0/24) then
      set traffic-index 3
    elseif destination in  (10
.1.4.0/24) then
      set traffic-index 4
    elseif destination in  (10
.1.5.0/24) then
      set traffic-index 5
    endif

    if destination in  (10
.1.1.0/24) then
      set traffic-index 6
    elseif destination in  (10
.1.2.0/24) then
      set traffic-index 7
    elseif destination in  (10
.1.3.0/24) then
      set traffic-index 8
    elseif destination in  (10
.1.4.0/24) then
      set traffic-index 9
    elseif destination in  (10
.1.5.0/24) then
      set traffic-index 10
    endif

    if as-path in as107 then
        set traffic-index 7
    elseif as-path in as108 then
        set traffic-index 8
    endif

    if destination in RT-65.0 then
       set traffic-index 15
    elseif destination in RT-66.0 then
       set traffic-index 16
    endif

end-policy
```

Configure the regular BGP route-policy to pass or drop all the BGP routes:

```
route-policy drop-all
  drop
end-policy
!
route-policy pass-all
  pass
end-policy
!
```

Configure the BGP router and apply the table-policy to the global ipv4 address family:

```
router bgp 100
 bgp router-id Loopback1
 bgp graceful-restart
 bgp as-path-loopcheck
```

```
 address-family ipv4 unicast
  table-policy bpa1
  maximum-paths 8
  bgp dampening
 !
```

Configure the BGP neighbor-group:

```
neighbor-group ebgp-peer-using-int-addr
  address-family ipv4 unicast
   policy pass-all in
   policy drop-all out
  !
 !
 neighbor-group ebgp-peer-using-int-addr-121
  remote-as 121
  address-family ipv4 unicast
   policy pass-all in
   policy drop-all out
  !
 !
 neighbor-group ebgp-peer-using-int-addr-pass-out
  address-family ipv4 unicast
   policy pass-all in
   policy pass-all out
  !
 !
```

Configure BGP neighbors:

```
neighbor 10
.4
.0.2
  remote-as 107
  use neighbor-group ebgp-peer-using-int-addr
 !
 neighbor 10
.8
.0.2
  remote-as 108
  use neighbor-group ebgp-peer-using-int-addr
 !
 neighbor 10
.7
.0.2
  use neighbor-group ebgp-peer-using-int-addr-121
 !
 neighbor 10
.1.7
.2
  use neighbor-group ebgp-peer-using-int-addr-121
 !
neighbor 10
.18
.0.2
  remote-as 122
  use neighbor-group ebgp-peer-using-int-addr
 !
 neighbor 10
.18
.1.2
  remote-as 1221
  use neighbor-group ebgp-peer-using-int-addr
```

```
      !
      end
```

Verifying BGP Policy Statistics: Example

The following example shows how to verify the traffic index setup for each BGP prefix and BGP Policy Accounting statistics on ingress and egress interfaces. The following traffic stream is configured for this example:

- Traffic comes in from GigabitEthernet 0/2/0/4 and goes out to 5 VLAN subinterfaces under GigabitEthernet 0/2/0/2

- Traffic comes in from GigabitEthernet 0/0/08 and goes out to GigabitEthernet 0/0/0/4

```
show cef ipv4 interface GigabitEthernet 0/0/0/8 bgp-policy-statistics

GigabitEthernet0/0/0/8 is up
Input BGP policy accounting on dst IP address enabled
  buckets        packets         bytes
  7              5001160    500116000
  15             10002320  1000232000
Input BGP policy accounting on src IP address enabled
  buckets        packets         bytes
  8              5001160    500116000
  16             10002320  1000232000
Output BGP policy accounting on dst IP address enabled
  buckets        packets         bytes
  0                   15          790
Output BGP policy accounting on src IP address enabled
  buckets        packets         bytes
  0                   15          790

show cef ipv4 interface GigabitEthernet 0/0/0/4 bgp-policy-statistics

GigabitEthernet0/0/0/4 is up
Input BGP policy accounting on dst IP address enabled
  buckets        packets         bytes
Input BGP policy accounting on src IP address enabled
  buckets        packets         bytes
Output BGP policy accounting on dst IP address enabled
  buckets        packets         bytes
  0                   13          653
  7              5001160    500116000
  15             10002320  1000232000
Output BGP policy accounting on src IP address enabled
  buckets        packets         bytes
  0                   13          653
  8              5001160    500116000
  16             10002320  1000232000

show cef ipv4 interface GigabitEthernet 0/2/0/4 bgp-policy-statistics

GigabitEthernet0/2/0/4 is up
Input BGP policy accounting on dst IP address enabled
  buckets        packets         bytes
  1              3297102    329710200
  2              3297102    329710200
  3              3297102    329710200
  4              3297101    329710100
  5              3297101    329710100
Input BGP policy accounting on src IP address enabled
```

```
   buckets      packets        bytes
   6            3297102   329710200
   7            3297102   329710200
   8            3297102   329710200
   9            3297101   329710100
   10           3297101   329710100
Output BGP policy accounting on dst IP address enabled
   buckets      packets        bytes
   0                 15          733
Output BGP policy accounting on src IP address enabled
   buckets      packets        bytes
   0                 15          733


show cef ipv4 interface GigabitEthernet 0/2/0/2.1 bgp-policy-statistics

GigabitEthernet 0/2/0/2.1 is up
Input BGP policy accounting on dst IP address enabled
   buckets      packets        bytes
Input BGP policy accounting on src IP address enabled
   buckets      packets        bytes
Output BGP policy accounting on dst IP address enabled
   buckets      packets        bytes
   0                 15          752
   1            3297102   329710200
   2            3297102   329710200
   3            3297102   329710200
   4            3297101   329710100
   5            3297101   329710100
Output BGP policy accounting on src IP address enabled
   buckets      packets        bytes
   0                 15          752
   6            3297102   329710200
   7            3297102   329710200
   8            3297102   329710200
   9            3297101   329710100
   10           3297101   329710100
```

The following example show how to verify BGP routes and traffic indexes:

```
show route bgp

B     10
.1.1.0/24 [20/0] via 10
.17
.1.2, 00:07:09
      Traffic Index 1
B     10
.1.2.0/24 [20/0] via 10
.17
.1.2, 00:07:09
      Traffic Index 2
B     10
.1.3.0/24 [20/0] via 10
.17
.1.2, 00:07:09
      Traffic Index 3
B     10
.1.4.0/24 [20/0] via 10
.17
.1.2, 00:07:09
      Traffic Index 4
B     10
.1.5.0/24 [20/0] via 10
.17
```

```
       .1.2, 00:07:09
            Traffic Index 5
B     10
.18
.1.0/24 [20/0] via 10
.18
.1.2, 00:07:09
            Traffic Index 6
B     10
.18
.2.0/24 [20/0] via 10
.18
.1.2, 00:07:09
            Traffic Index 7
B     10
.18
.3.0/24 [20/0] via 10
.18
.1.2, 00:07:09
            Traffic Index 8
B     10
.28
.4.0/24 [20/0] via 10
.18
.1.2, 00:07:09
            Traffic Index 9
B     10
.28
.5.0/24 [20/0] via 10
.18
.1.2, 00:07:09
            Traffic Index 10
B     10
.65
.1.0/24 [20/0] via 10
.45
.0.2, 00:07:09
            Traffic Index 15
B     10
.65
.2.0/24 [20/0] via 10
.45
.0.2, 00:07:09
            Traffic Index 15
B     10
.65
.3.0/24 [20/0] via 10
.45
.0.2, 00:07:09
            Traffic Index 15
B     10
.65
.65
.0/24 [20/0] via 10
.45
.0.2, 00:07:09
            Traffic Index 15
B     10
.65
.5.0/24 [20/0] via 10
.45
.0.2, 00:07:09
            Traffic Index 15
B     10
```

```
.65
.6.0/24 [20/0] via 10
.45
.0.2, 00:07:09
     Traffic Index 15
B    10
.65
.7.0/24 [20/0] via 10
.45
.0.2, 00:07:09
     Traffic Index 15
B    10
.65
.8.0/24 [20/0] via 10
.45
.0.2, 00:07:09
     Traffic Index 15
B    10
.65
.9.0/24 [20/0] via 10
.45
.0.2, 00:07:09
     Traffic Index 15
B    10
.65
.10.0/24 [20/0] via 10
.45
.0.2, 00:07:09
     Traffic Index 15
B    10
.66
.1.0/24 [20/0] via 10
.32
.0.2, 00:07:09
     Traffic Index 16
B    10
.66
.2.0/24 [20/0] via 10
.32
.0.2, 00:07:09
     Traffic Index 16
B    10
.66
.3.0/24 [20/0] via 10
.32
.0.2, 00:07:09
     Traffic Index 16
B    10
.66
.4.0/24 [20/0] via 10
.32
.0.2, 00:07:09
     Traffic Index 16
B    10
.66
.5.0/24 [20/0] via 10
.32
.0.2, 00:07:09
     Traffic Index 16
B    10
.66
.6.0/24 [20/0] via 10
.32
.0.2, 00:07:09
```

```
        Traffic Index 16
B    10
.66
.7.0/24 [20/0] via 10
.32
.0.2, 00:07:09
        Traffic Index 16
B    10
.66
.8.0/24 [20/0] via 10
.32
.0.2, 00:07:09
        Traffic Index 16
B    10
.66
.9.0/24 [20/0] via 10
.32
.0.2, 00:07:09
        Traffic Index 16
B    10
.66
.10.0/24 [20/0] via 10
.32
.0.2, 00:07:09
        Traffic Index 16
B    10
.67
.1.0/24 [20/0] via 10
.32
.0.2, 00:07:09
        Traffic Index 7
B    10
.67
.2.0/24 [20/0] via 10
.32
.0.2, 00:07:09
        Traffic Index 7
B    10
.67
.3.0/24 [20/0] via 10
.32
.0.2, 00:07:09
        Traffic Index 7
B    10
.67
.4.0/24 [20/0] via 10
.32
.0.2, 00:07:09
        Traffic Index 7
B    10
.67
.5.0/24 [20/0] via 10
.32
.0.2, 00:07:09
        Traffic Index 7
B    10
.67
.6.0/24 [20/0] via 10
.32
.0.2, 00:07:09
        Traffic Index 7
B    10
.67
.7.0/24 [20/0] via 10
```

```
.32
.0.2, 00:07:09
      Traffic Index 7
B    10
.67
.8.0/24 [20/0] via 10
.32
.0.2, 00:07:09
      Traffic Index 7
B    10
.67
.9.0/24 [20/0] via 10
.32
.0.2, 00:07:09
      Traffic Index 7
B    10
.67
.10.0/24 [20/0] via 10
.32
.0.2, 00:07:09
      Traffic Index 7
B    10
.68
.1.0/24 [20/0] via 10
.8
.0.2, 00:07:09
      Traffic Index 8
B    10
.68
.2.0/24 [20/0] via 10
.8
.0.2, 00:07:09
      Traffic Index 8
B    10
.68
.3.0/24 [20/0] via 10
.8
.0.2, 00:07:09
      Traffic Index 8
B    10
.68
.4.0/24 [20/0] via 10
.8
.0.2, 00:07:09
      Traffic Index 8
B    10
.68
.5.0/24 [20/0] via 10
.8
.0.2, 00:07:09
      Traffic Index 8
B    10
.68
.6.0/24 [20/0] via 10
.8
.0.2, 00:07:09
      Traffic Index 8
B    10
.68
.7.0/24 [20/0] via 10
.8
.0.2, 00:07:09
      Traffic Index 8
B    10
```

```
.68
.8.0/24 [20/0] via 10
.8
.0.2, 00:07:09
      Traffic Index 8
B     10
.68
.9.0/24 [20/0] via 10
.8
.0.2, 00:07:09
      Traffic Index 8
B     10
.68
.10.0/24 [20/0] via 10
.8
.0.2, 00:07:09
      Traffic Index 8

show bgp summary

BGP router identifier 192
.0
.2
.0
, local AS number 100
BGP generic scan interval 60 secs
BGP main routing table version 151
Dampening enabled
BGP scan interval 60 secs
BGP is operating in STANDALONE mode.
```

Process	RecvTblVer	bRIB/RIB	SendTblVer
Speaker	151	151	151

Neighbor	Spk	AS	MsgRcvd	MsgSent	TblVer	InQ	OutQ	Up/Down	St/PfxRcd
10.4.0.2		0	107	54	53	151	0	0 00:25:26	20
10.1.0.2		0	108	54	53	151	0	0 00:25:28	20
10.1.0.2	0	121	53	54	151	0	0 00:25:42	0
10.1.1.2	0	121	53	53	151	0	0 00:25:06	5
10.1.2.2	0	121	52	54	151	0	0 00:25:04	0
10.1.3.2	0	121	52	53	151	0	0 00:25:26	0
10.1.4.2	0	121	53	54	151	0	0 00:25:41	0
10.1.5.2	0	121	53	54	151	0	0 00:25:43	0
10.1.6.2	0	121	51	53	151	0	0 00:24:59	0
10.1.7.2	0	121	51	52	151	0	0 00:24:44	0
10.1.8.2	0	121	51	52	151	0	0 00:24:49	0
10.2.0.2	0	122	52	54	151	0	0 00:25:21	0
10.2.1.2	0	1221	54	54	151	0	0 00:25:43	5

```
10
.2
.2.2          0  1222     53       54     151   0     0 00:25:38         0
10
.2
.3.2          0  1223     52       53     151   0     0 00:25:17         0
10
.2
.4.2          0  1224     51       52     151   0     0 00:24:57         0
10
.2
.5.2          0  1225     52       53     151   0     0 00:25:14         0
10
.2
.6.2          0  1226     52       54     151   0     0 00:25:04         0
10
.2
.7.2          0  1227     52       54     151   0     0 00:25:13         0
10
.2
.8.2          0  1228     53       54     151   0     0 00:25:36         0

show bgp 27.1.1.1

BGP routing table entry for 27.1.1.0/24
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker                102         102
Paths: (1 available, best #1)
  Not advertised to any peer
  Received by speaker 0
  121
    10
.1.1.2 from 10
.1.1.2 (10
.1.1.2)
      Origin incomplete, localpref 100, valid, external, best
      Community: 27:1 121:1

show bgp 10
.1.1.1

BGP routing table entry for 10
.1.1.0/24
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker                107         107
Paths: (1 available, best #1)
  Not advertised to any peer
  Received by speaker 0
  1221
    10
.2
.1.2 from 10
.2
.1.2 (18.1.1.2)
      Origin incomplete, localpref 100, valid, external, best
      Community: 28:1 1221:1

show bgp 10
.0.1.1

BGP routing table entry for 10
.0.1.0/24
```

```
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker              112         112
Paths: (1 available, best #1)
  Not advertised to any peer
  Received by speaker 0
  107
    10
.1.0.2 from 10
.1.0.2 (10
.1.0.2)
      Origin incomplete, localpref 100, valid, external, best
      Community: 107:65

show bgp 10
.2
.1.1

BGP routing table entry for 10
.2
.1.0/24
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker              122         122
Paths: (1 available, best #1)
  Not advertised to any peer
  Received by speaker 0
  108
    8.1.0.2 from 8.1.0.2 (8.1.0.2)
      Origin incomplete, localpref 100, valid, external, best
      Community: 108:66

show bgp 67.0.1.1

BGP routing table entry for 67.0.1.0/24
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker              132         132
Paths: (1 available, best #1)
  Not advertised to any peer
  Received by speaker 0
  107
    4.1.0.2 from 4.1.0.2 (4.1.0.2)
      Origin incomplete, localpref 100, valid, external, best
      Community: 107:67

show bgp 68.0.1.1

BGP routing table entry for 68.0.1.0/24
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker              142         142
Paths: (1 available, best #1)
  Not advertised to any peer
  Received by speaker 0
  108
    8.1.0.2 from 8.1.0.2 (8.1.0.2)
      Origin incomplete, localpref 100, valid, external, best
      Community: 108:68

show route ipv4 27.1.1.1

Routing entry for 27.1.1.0/24
  Known via "bgp 100", distance 20, metric 0
```

```
      Tag 121, type external, Traffic Index 1
      Installed Nov 11 21:14:05.462
      Routing Descriptor Blocks
        17.1.1.2, from 17.1.1.2
          Route metric is 0
      No advertising protos.

show route ipv4 28.1.1.1

Routing entry for 28.1.1.0/24
  Known via "bgp 100", distance 20, metric 0
  Tag 1221, type external, Traffic Index 6
  Installed Nov 11 21:14:05.462
  Routing Descriptor Blocks
    18.1.1.2, from 18.1.1.2
      Route metric is 0
  No advertising protos.

show route ipv4 65.0.1.1

Routing entry for 65.0.1.0/24
  Known via "bgp 100", distance 20, metric 0
  Tag 107, type external, Traffic Index 15
  Installed Nov 11 21:14:05.462
  Routing Descriptor Blocks
    4.1.0.2, from 4.1.0.2
      Route metric is 0
  No advertising protos.

show route ipv4 66.0.1.1

Routing entry for 66.0.1.0/24
  Known via "bgp 100", distance 20, metric 0
  Tag 108, type external, Traffic Index 16
  Installed Nov 11 21:14:05.462
  Routing Descriptor Blocks
    8.1.0.2, from 8.1.0.2
      Route metric is 0
  No advertising protos.

show route ipv4 67.0.1.1

Routing entry for 67.0.1.0/24
  Known via "bgp 100", distance 20, metric 0
  Tag 107, type external, Traffic Index 7
  Installed Nov 11 21:14:05.462
  Routing Descriptor Blocks
    4.1.0.2, from 4.1.0.2
      Route metric is 0
  No advertising protos.

show route ipv4 68.0.1.1

Routing entry for 68.0.1.0/24
  Known via "bgp 100", distance 20, metric 0
  Tag 108, type external, Traffic Index 8
  Installed Nov 11 21:14:05.462
  Routing Descriptor Blocks
    8.1.0.2, from 8.1.0.2
      Route metric is 0
  No advertising protos.

show cef ipv4 27.1.1.1
```

```
27.1.1.0/24, version 263, source-destination sharing
Prefix Len 24, Traffic Index 1, precedence routine (0)
  via 17.1.1.2, 0 dependencies, recursive
    next hop 17.1.1.2/24, GigabitEthernet 0/2/0/2.1 via 17.1.1.0/24
    valid remote adjacency
  Recursive load sharing using 17.1.1.0/24

show cef ipv4 28.1.1.1

28.1.1.0/24, version 218, source-destination sharing
Prefix Len 24, Traffic Index 6, precedence routine (0)
  via 18.1.1.2, 0 dependencies, recursive
    next hop 18.1.1.2/24, GigabitEthernet0/2/0/4.1 via 18.1.1.0/24
    valid remote adjacency
  Recursive load sharing using 18.1.1.0/24

show cef ipv4 65.0.1.1

65.0.1.0/24, version 253, source-destination sharing
Prefix Len 24, Traffic Index 15, precedence routine (0)
  via 4.1.0.2, 0 dependencies, recursive
    next hop 4.1.0.2/16, GigabitEthernet0/0/0/4 via 4.1.0.0/16
    valid remote adjacency
  Recursive load sharing using 4.1.0.0/16

show cef ipv4 66.0.1.1

66.0.1.0/24, version 233, source-destination sharing
Prefix Len 24, Traffic Index 16, precedence routine (0)
  via 8.1.0.2, 0 dependencies, recursive
    next hop 8.1.0.2/16, GigabitEthernet 0/0/0/8 via 8.1.0.0/16
    valid remote adjacency
  Recursive load sharing using 8.1.0.0/16

show cef ipv4 67.0.1.1

67.0.1.0/24, version 243, source-destination sharing
Prefix Len 24, Traffic Index 7, precedence routine (0)
  via 4.1.0.2, 0 dependencies, recursive
    next hop 4.1.0.2/16, GigabitEthernet 0/0/0/4 via 4.1.0.0/16
    valid remote adjacency
  Recursive load sharing using 4.1.0.0/16

show cef ipv4 68.0.1.1

68.0.1.0/24, version 223, source-destination sharing
Prefix Len 24, Traffic Index 8, precedence routine (0)
  via 8.1.0.2, 0 dependencies, recursive
    next hop 8.1.0.2/16, GigabitEthernet0/0/0/8 via 8.1.0.0/16
    valid remote adjacency
  Recursive load sharing using 8.1.0.0/16

show cef ipv4 27.1.1.1 detail

27.1.1.0/24, version 263, source-destination sharing
Prefix Len 24, Traffic Index 1, precedence routine (0)
  via 17.1.1.2, 0 dependencies, recursive
    next hop 17.1.1.2/24, GigabitEthernet 0/2/0/2.1 via 17.1.1.0/24
    valid remote adjacency

  Recursive load sharing using 17.1.1.0/24
  Load distribution: 0 (refcount 6)

  Hash  OK  Interface                Address  Packets
```

```
 1    Y   GigabitEthernet 0/2/0/2.1          (remote)               0

show cef ipv4 28.1.1.1 detail

28.1.1.0/24, version 218, source-destination sharing
Prefix Len 24, Traffic Index 6, precedence routine (0)
  via 18.1.1.2, 0 dependencies, recursive
    next hop 18.1.1.2/24, GigabitEthernet 0/2/0/4.1 via 18.1.1.0/24
    valid remote adjacency

  Recursive load sharing using 18.1.1.0/24
  Load distribution: 0 (refcount 6)

  Hash  OK  Interface              Address  Packets
  1     Y   GigabitEthernet 0/2/0/4.1          (remote)              0

show cef ipv4 65.0.1.1 detail

65.0.1.0/24, version 253, source-destination sharing
Prefix Len 24, Traffic Index 15, precedence routine (0)
  via 4.1.0.2, 0 dependencies, recursive
    next hop 4.1.0.2/16, GigabitEthernet0/0/0/4 via 4.1.0.0/16
    valid remote adjacency

  Recursive load sharing using 4.1.0.0/16
  Load distribution: 0 (refcount 21)

  Hash  OK  Interface              Address  Packets
  1     Y   GigabitEthernet0/0/0/4              (remote)             0

show cef ipv4 66.0.1.1 detail

66.0.1.0/24, version 233, source-destination sharing
Prefix Len 24, Traffic Index 16, precedence routine (0)
  via 8.1.0.2, 0 dependencies, recursive
    next hop 8.1.0.2/16, GigabitEthernet0/0/0/8 via 8.1.0.0/16
    valid remote adjacency

  Recursive load sharing using 8.1.0.0/16
  Load distribution: 0 (refcount 21)

  Hash  OK  Interface              Address  Packets
  1     Y   GigabitEthernet 0/0/0/8              (remote)             0

show cef ipv4 67.0.1.1 detail

67.0.1.0/24, version 243, source-destination sharing
Prefix Len 24, Traffic Index 7, precedence routine (0)
  via 4.1.0.2, 0 dependencies, recursive
    next hop 4.1.0.2/16, GigabitEthernet 0/0/0/4 via 4.1.0.0/16
    valid remote adjacency

  Recursive load sharing using 4.1.0.0/16
  Load distribution: 0 (refcount 21)

  Hash  OK  Interface              Address  Packets
  1     Y   GigabitEthernet 0/0/0/4              (remote)             0

show cef ipv4 68.0.1.1 detail

68.0.1.0/24, version 223, source-destination sharing
Prefix Len 24, Traffic Index 8, precedence routine (0)
  via 8.1.0.2, 0 dependencies, recursive
    next hop 8.1.0.2/16, GigabitEthernet 0/0/0/8 via 8.1.0.0/16
```

```
  valid remote adjacency

Recursive load sharing using 8.1.0.0/16
Load distribution: 0 (refcount 21)

Hash  OK  Interface               Address  Packets
1     Y   GigabitEthernet 0/0/0/8           (remote)           0
```

Configuring Unicast RPF Checking: Example

The following example shows how to configure unicast RPF checking:

```
configure
interface GigabitEthernet 0/0/0/1
ipv4 verify unicast source reachable-via rx
end
```

Configuring the Switching of Modular Services Card to Management Ethernet Interfaces on the Route Processor: Example

The following example shows how to configure the switching of the MSC to Management Ethernet interfaces on the route processor:

```
configure
rp mgmtethernet forwarding
end
```

Configuring Per-Flow Load Balancing: Example

The following examples show how to configure Layer 3 load-balancing for the hash algorithm from the **cef load-balancing fields L3 global** command, and how to verify summary information for the CEF table from the **show cef summary** command:

Configuring Layer 3 load-balancing

```
configure
 cef load-balancing fields L3 global
 end
 !
show cef summary
Router ID is 10.6.6.6

IP CEF with switching (Table Version 0) for node0_RSP0_CPU0

  Load balancing: L3
  Tableid 0xe0000000 (0x9cbb51b0), Vrfid 0x60000000, Vrid 0x20000000, Flags 0x2031
  Vrfname default, Refcount 577
  300 routes, 0 protected, 0 reresolve, 0 unresolved (0 old, 0 new), 21600 bytes
  212 load sharing elements, 62576 bytes, 324 references
  19 shared load sharing elements, 5388 bytes
  193 exclusive load sharing elements, 57188 bytes
  0 route delete cache elements
  622 local route bufs received, 1 remote route bufs received,  0 mix bufs received
  176 local routes, 0 remote routes
  4096 total local route updates processed
```

```
     0 total remote route updates processed
     0 pkts pre-routed to cust card
     0 pkts pre-routed to cust card
     0 pkts received from core card
     0 CEF route update drops, 96 revisions of existing leaves
     0 CEF route update drops due to version mis-match
     Resolution Timer: 15s
     0 prefixes modified in place
     0 deleted stale prefixes
     82 prefixes with label imposition, 107 prefixes with label information
    95 next hops
     0 incomplete next hops

   0 PD backwalks on LDIs with backup path
```

Configuring BGP Attributes Download: Example

The following example shows how to configure the BGP Attributes Download feature:

```
router configure
show cef bgp attribute {attribute-id| local-attribute-id}
```

Additional References

The following sections provide references related to implementing CEF.

Related Documents

Related Topic	Document Title
CEF commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco Express Forwarding Commands* module in *IP Addresses and Services Command Reference for Cisco ASR 9000 Series Routers*
BGP commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*BGP Commands* module in the *Routing Command Reference for Cisco ASR 9000 Series Routers*
Link Bundling Commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Link Bundling Commands* module in the *Interface and Hardware Component Command Reference for Cisco ASR 9000 Series Routers*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

MIBs	MIBs Link
—	To locate and download MIBs, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: https://mibs.cloudapps.cisco.com/ITDIT/MIBS/servlet/index

RFCs

RFCs	Title
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	—

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport