

# Configuring the Unicast RIB and FIB

This chapter describes how to configure and manage routes in the unicast Routing Information Base (RIB) and the Forwarding Information Base (FIB) on Cisco NX-OS switches.

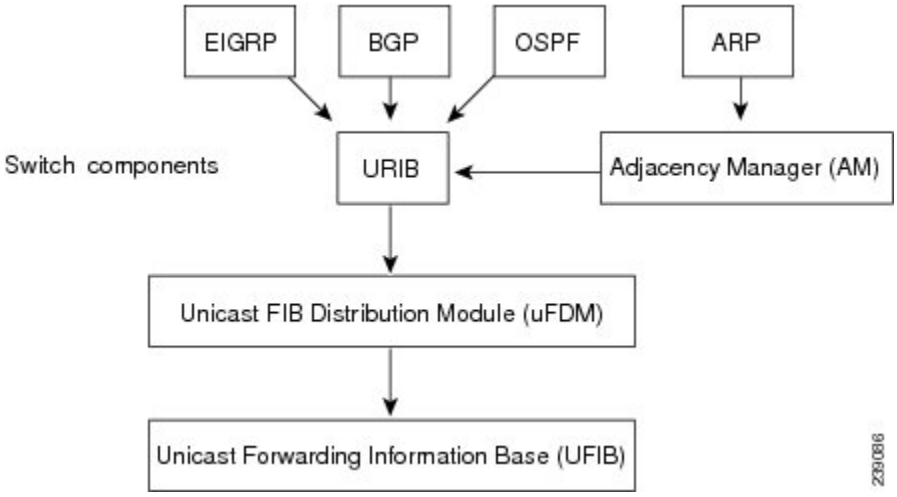
This chapter includes the following sections:


- Information About the Unicast RIB and FIB, on page 1
- Managing the Unicast RIB and FIB, on page 3
- Verifying the Unicast RIB and FIB Configuration, on page 8
- Additional References, on page 9


## Information About the Unicast RIB and FIB

The unicast RIB (IPv4 RIB) and FIB are part of the Cisco NX-OS forwarding architecture, as shown below.

***Figure 1: Cisco NX-OS Forwarding Architecture***




The unicast RIB maintains the routing table with directly connected routes, static routes, and routes learned from dynamic unicast routing protocols. The unicast RIB also collects adjacency information from sources such as the Address Resolution Protocol (ARP). The unicast RIB determines the best next hop for a given route and populates the unicast forwarding information base (FIBs) by using the services of the unicast FIB distribution module (FDM).

Each dynamic routing protocol must update the unicast RIB for any route that has timed out. The unicast RIB then deletes that route and recalculates the best next hop for that route (if an alternate path is available).

# Layer 3 Consistency Checker

In rare instances, an inconsistency can occur between the unicast RIB and the FIB on each module. Cisco NX-OS supports the Layer 3 consistency checker. This feature detects inconsistencies between the unicast IPv4 RIB and the FIB on each interface module. Inconsistencies include the following:

- Missing prefix
- Extra prefix
- Wrong next-hop address
- Incorrect Layer 2 rewrite string in the ARP or neighbor discovery (ND) cache

The Layer 3 consistency checker compares the FIB entries to the latest adjacency information from the Adjacency Manager (AM) and logs any inconsistencies. The consistency checker then compares the unicast RIB prefixes to the module FIB and logs any inconsistencies. See the Triggering the Layer 3 Consistency Checker section.

You can then manually clear any inconsistencies. See the Clearing Forwarding Information in the FIB section.

When the consistency checker is run before there is no consistency, consistency passes. But when more routes are learned exceeding the hardware limit of 4K and the **show consistency-checker forwarding ipv4** command is run, consistency passes. The same is true when it is transitioning from an inconsistent state to a consistent state. It still shows failure routes. Until and unless the **test forwarding ipv4 inconsistency route** command is run again, it doesn't leave this state. This is an expected behavior.

# FIB Tables

The following are the unicast routing table capacities for the Cisco Nexus 3548 switch when the switch is configured in the normal forwarding mode:

- Unicast Routing Host table = 64,000 hash table entries
- Unicast Routing LPM table = 16,000 TCAM entries
- ECMP members Table size: 16,000 entries

The following are the unicast routing table capacities for the Cisco Nexus 3548 switch when the switch is configured inthe warp mode:

- L3 Unicast Host table = 8000 TCAM entries
- L3 Unicast LPM table = 4000 TCAM entries

**Note** ECMP is not supported on warp mode.

**Note** In warp mode, when two equal cost paths are received in RIB, one of the paths is installed in the hardware. It is recommended to configure the maximum-path to one under the routing protocol configuration.

## Virtualization Support

The Unicast RIB and FIB support Virtual Routing and Forwarding instances (VRFs).

# Managing the Unicast RIB and FIB

**Note** If you are familiar with the Cisco IOS CLI, be aware that the Cisco NX-OS commands for this feature might differ from the Cisco IOS commands that you would use.

## Displaying Module FIB Information

You can display the FIB information on a switch.

**DETAILED STEPS**

To display the FIB information on a switch, use the following commands in any mode:

| Command | Purpose |
|---|---|
| **show ip fib adjacency**<br>**Example:**<br>`switch# show ip fib adjacency` | Displays the adjacency information for FIB. |
| **show forwarding ipv4 adjacency**<br>**Example:**<br>`switch# show forwarding ipv4 adjacency` | Displays the adjacency information for IPv4. |
| **show ip fib interfaces**<br>**Example:**<br>`switch# show ip fib interfaces` | Displays the FIB interface information for IPv4. |
| **show ip fib route**<br>**Example:**<br>`switch# show ip fib route` | Displays the route table for IPv4. |
| **show forwarding ipv4 route**<br>**Example:**<br>`switch# show forwarding ipv4 route` | Displays the route table for IPv4. |

This example shows the FIB contents on a switch:

```
switch# show ip fib route

IPv4 routes for table default/base

------------------+------------------+---------------------
Prefix | Next-hop | Interface
------------------+------------------+---------------------
0.0.0.0/32 Drop Null0
255.255.255.255/32 Receive sup-eth1
```

# Configuring Load Sharing in the Unicast FIB

Dynamic routing protocols, such as Open Shortest Path First (OSPF), support load balancing with equal-cost multipath (ECMP). The routing protocol determines its best routes based on the metrics configured for the protocol and installs up to the protocol-configured maximum paths in the unicast RIB. The unicast RIB compares the administrative distances of all routing protocol paths in the RIB and selects a best path set from all of the path sets installed by the routing protocols. The unicast RIB installs this best path set into the FIB for use by the forwarding plane.

The forwarding plane uses a load-sharing algorithm to select one of the installed paths in the FIB to use for a given data packet.

You can globally configure the following load-sharing settings:

- load-share mode—Selects the best path based on the destination address and port or the source and the destination address and port.
- Universal ID—Sets the random seed for the hash algorithm. You do not need to configure the Universal ID. Cisco NX-OS chooses the Universal ID if you do not configure it.

Load sharing uses the same path for all packets in a given flow. A flow is defined by the load-sharing method that you configure. For example, if you configure source-destination load sharing, then all packets with the same source IP address and destination IP address pair follow the same path.

To configure the unicast FIB load-sharing algorithm, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| **ip load-sharing address** { **destination port** *destination* \| **source-destination** [ **port source-destination** ]} [ **universal-id** *seed* ]<br>**Example:**<br>`switch(config)# ip load-sharing address source-destination` | Configures the unicast FIB load-sharing algorithm for data traffic. The *universal-id* range is from 1 to 4294967295. |

To display the unicast FIB load-sharing algorithm, use the following command in any mode:

| Command | Purpose |
|---|---|
| **show ip load-sharing**<br>**Example:**<br>`switch(config)# show ip load-sharing` | Displays the unicast FIB load-sharing algorithm for data traffic. |

To display the route that the unicast RIB and FIB use for a particular source address and destination address, use the following command in any mode:

| Command | Purpose |
|---|---|
| **show routing hash source-addr dest-addr** [ **source-port dest-port** ] [ **vrf** *vrf-name* ]<br>**Example:**s<br>`witch# show routing hash 192.0.2.1 10.0.0.1` | Displays the route that the unicast RIB FIB use for a source and destination address pair. The source address and destination address format is x.x.x.x. The source port and destination port range is from 1 to 65535. The VRF name can be any case-sensitive, alphanumeric string up to 64 characters. |

This example shows the route selected for a source/destination pair:

```
switch# show routing hash 10.0.0.5 30.0.0.2
Load-share parameters used for software forwarding:
load-share mode: address source-destination port source-destination
Universal-id seed: 0xe05e2e85
Hash for VRF "default"
Hashing to path *20.0.0.2 (hash: 0x0e), for route:
```

# Displaying Routing and Adjacency Information

ou can display the routing and adjacency information.

To display the routing and adjacency information, use the following commands in any mode:

| Command | Purpose |
|---|---|
| **show ip route** [ **route-type** \| **interface int-type number** \| **next-hop** ]<br>**Example:**<br>`switch# show ip route` | Displays the unicast route table. The *route-type* argument can be a single route prefix, direct, static, or a dynamic route protocol. Use the **?** keyword to see the supported interfaces. |
| **show ip adjacency** [ *prefix* \| *interface number* [ **summary** ] \| **non-best** ] [ **detail** ] [ **vrf** *vrf-id* ]<br>**Example:**<br>`switch# show ip adjacency` | Displays the adjacency table. The argument ranges are as follows:<br>• *prefix* —Any IPv4prefix address.<br>• *interface-type number* —Use the **?** keyword to see the supported interfaces.<br>• *vrf-id* —Any case-sensitive, alphanumeric string up to 32 characters. |
| **show ip routing** [ *route-type* \| **interface** *int-type number* \| **next-hop** \| **recursive-next-hop** \| **summary** \| **updated** { **since** \| **until** } *time* ]<br>**Example:**<br>`switch# show routing summary` | Displays the unicast route table. The *route-type* argument can be a single route prefix, direct, static, or a dynamic route protocol. Use the **?** keyword to see the supported interfaces. |

This example displays the unicast route table:

```
switch# show ip route
IP Route Table for VRF "default"
```

```
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]

192.168.0.2/24, ubest/mbest: 1/0, attached
*via 192.168.0.32, Eth1/5, [0/0], 22:34:09, direct
192.168.0.32/32, ubest/mbest: 1/0, attached
*via 192.168.0.32, Eth1/5, [0/0], 22:34:09, local
```

This example shows the adjacency information:

```
switch# show ip adjacency

IP Adjacency Table for VRF default
Total number of entries: 2
Address Age MAC Address Pref Source Interface Best
10.1.1.1 02:20:54 00e0.b06a.71eb 50 arp mgmt0 Yes
10.1.1.253 00:06:27 0014.5e0b.81d1 50 arp mgmt0 Yes
```

# Triggering the Layer 3 Consistency Checker

You can manually trigger the Layer 3 consistency checker.

To manually trigger the Layer 3 consistency checker, use the following commands in global configuration mode:

| Command | Purpose |
|---|---|
| **test** [ **ipv4** ] [ **unicast**] **forwarding inconsistency** [ **vrf** *vrf-name* ] [ **module** { *slot* \| **all** }] **Example:** `switch(config)# test forwarding inconsistency` | Starts a Layer 3 consistency check. The *vrf-name* can be any case-sensitive, alphanumeric string up to 32 characters. The *slot* range is from 1 to 10. |

To stop the Layer 3 consistency checker, use the following commands in global configuration mode:

| Command | Purpose |
|---|---|
| **test forwarding** [ **ipv4** ] [ **unicast** ] **inconsistency** [ **vrf** *vrf-name* ] [ **module** { *slot* \| **all** }] **stop** **Example:** `switch(config)# test forwarding inconsistency stop` | Stops a Layer 3 consistency check. The *vrf-name* can be any case-sensitive, alphanumeric string up to 64 characters. The *slot* range is from 1 to 10. |

To display the Layer 3 inconsistencies, use the following commands in any mode:

| Command | Purpose |
|---|---|
| **show forwarding** [ **ipv4** ] **inconsistency** [ **vrf** *vrf-name* ] [ **module** { *slot* \| **all** }] **Example:** `switch(config)# show forwarding inconsistency` | Displays the results of a Layer 3 consistency check. The *vrf-name* can be any case-sensitive, alphanumeric string up to 32 characters. The *slot* range is from 1 to 10. |

# Clearing Forwarding Information in the FIB

You can clear one or more entries in the FIB.

**Note** The **clear forwarding** command disrupts forwarding on the switch.

To clear an entry in the FIB, including a Layer 3 inconsistency, use the following command in any mode:

| Command | Purpose |
|---|---|
| **clear forwarding** { **ipv4** } **route** {* \| *prefix* } [ **vrf** *vrf-name* ] [ **module** { *slot* \| **all** }]<br>**Example:**<br>`switch(config)# clear forwarding ipv4 route *` | Clears one or more entries from the FIB. The route options are as follows:<br>• *—All routes.<br>• *prefix* —Any IPprefix.<br>The *vrf-name* can be any case-sensitive, alphanumeric string up to 32 characters. The **slot** range is from 1 to 10. |

**Note** Ensure you clear the RIB entry after you clear the FIB entry.

# Estimating Memory Requirements for Routes

You can estimate the memory that a number of routes and next-hop addresses will use.

To estimate the memory requirements for routes, use the following command in any mode:

| Command | Purpose |
|---|---|
| **show routing memory estimate routes** *num-routes* **next-hops** *num-nexthops*<br>**Example:**<br>`switch# show routing memory estimate routes 1000 next-hops 1` | Displays the memory requirements for routes. The *num-routes* range is from 1000 to 1000000. The *num-nexthops* range is from 1 to 16. |

# Clearing Routes in the Unicast RIB

You can clear one or more routes from the unicast RIB.

**Caution** The * keyword is severely disruptive to routing.

To clear one or more entries in the unicast RIB, use the following commands in any mode:

| Command | Purpose |
|---|---|
| **clear iproute** {* \| { *route* \| *prefix/length* }[ *next-hop interface* ]} [ **vrf** *vrf-name* ]<br>**Example:**<br>`switch(config)# clear ip route 10.2.2.2` | Clears one or more routes from both the unicast RIB and all the module FIBs. The route options are as follows:<br>• *—All routes.<br>• *route* —An individual IProute.<br>• *prefix/length* —Any IPprefix.<br>• *next-hop* —The next-hop address<br>• *interface* —The interface to reach the next-hop address.<br>The *vrf-name* can be any case-sensitive, alphanumeric string up to 32 characters. |
| **clear routing unicast** [ **ip** \| **ipv4** ] {***** \| { *route* \| *prefix/length* }[ *next-hop interface* ]} [ **vrf** *vrf-name* ]<br>**Example:**<br>`switch(config)# clear routing ip 10.2.2.2` | Clears one or more routes from the unicast RIB. The route options are as follows:<br>• *—All routes.<br>• *route* —An individual IProute.<br>• *prefix/length* —Any IPprefix.<br>• *next-hop* —The next-hop address<br>• *interface* —The interface to reach the next-hop address.<br>The *vrf-name* can be any case-sensitive, alphanumeric string up to 32 characters. |

# Verifying the Unicast RIB and FIB Configuration

To display the unicast RIB and FIB configuration information, perform one of the following tasks:

| Command | Purpose |
|---|---|
| **show forwarding adjacency** | Displays the adjacency table on a module. |
| **show forwarding distribution** { **clients** \| **fib-state** } | Displays the FIB distribution information. |
| **show forwarding interfaces module** *slot* | Displays the FIB information for a module. |
| **show forwarding ipv4route** | Displays routes in the FIB. |
| **show ip adjacency** | Displays the adjacency table. |
| **show ip route** | Displays IPv4routes from the unicast RIB. |
| **show routing** | Displays routes from the unicast RIB. |

# Additional References

For additional information related to managing unicast RIB and FIB, see the following sections:

- Related Documents

## Related Documents

| Related Topic | Document Title |
|---|---|
| Unicast RIB and FIB CLI commands | Cisco Nexus 3000 Series Command Reference |