Implementing BFD

- BFD Overview , on page 1
- BFD Session Types, on page 8
- BFD Singlepath Sessions, on page 8
- BFD Multipath Sessions, on page 25
- Seamless Bidirectional Forwarding Detection, on page 32
- Coexistence Between BFD over Bundle and BFD over Logical Bundle , on page 36
- BFD CPU Offload Support for IPv6, on page 42
- BFD Object Tracking, on page 44
- BFD Dampening, on page 45

BFD Overview

Bidirectional Forwarding Detection (BFD) provides low-overhead, short-duration detection of failures in the path between adjacent routers. BFD allows a single mechanism to be used for failure detection over any media and at any protocol layer, with a wide range of detection times and overhead. The fast detection of failures provides immediate reaction to failure in the event of a failed link or neighbor.

Tip You can programmatically configure BFD and retrieve operational data using `openconfig-bfd.yang` OpenConfig data model. To get started with using data models, see the *Programmability Configuration Guide*.

Features Unsupported

- BFD echo mode and encryption are not supported.
- BFD over MPLS tunnel interfaces is not supported.
- Dampening extensions for BFD are not supported.
- BFD down dampening is not supported.
- BFD IPv6 Dampening is not supported.
- SNMP traps are not supported for multipath BFD sessions.
- BFD Over GRE is not supported.

- BFD over PWHE is not supported.
- Seamless BFD is not supported.
- BFD over Satellite interface is not supported.
- BFD Authentication is not supported.

Supported Functionalities

- BFD hardware offload is supported for both IPv4 and IPv6.
- BFD is only supported in IP core. It cannot coexist with Label distribution Protocol, or Segment Routing, or Traffic Engineering in the core. This is applicable for releases prior to IOS XR Release 7.1.1.
- BFD over Bundle (BoB) over IPv6 is not supported with dyamically configured link-local address. It must be statically configured.
- Dampening extensions for BFD are not supported.
- Egress IPv4 ACLs block all traffic, including router-generated traffic for the following routers and line cards:
  - NC57-24DD
  - NC57-18DD-SE
  - NC57-36H-SE
  - NC57-36H6D-S
  - NC57-MOD-S
  - NCS-57B1-6D24-SYS
  - NCS-57B1-5DSE-SYS

  For all other routers and line cards, egress IPv4 ACLs do not block certain router-generated traffic, such as ICMP messages.

Feature Limitations

- Egress ACL with drop rule for src-ip equal to 0.0.0.0 will drop BFD-V4 Tx packets on that interface. This is because, BFD-V4 packets generated by OAMP will have src.ip 0.0.0.0 due to its limitation. And the actual source IP value is filled in ETPP block in pipeline before sending the packet. Since egress ACL is applied before ETPP, the BFD packets are dropped.
- BFD over Bundle (BOB) uses two entries in the MAC address table for each of IPv4 and IPv6, regardless of the number of BOB sessions on the Network Processing Unit (NPU). This MAC table is shared with other features such as Bridge Virtual Interface (BVI), Hot Standby Router Protocol (HSRP), Virtual Router Redundancy Protocol (VRRP), and DHCP, all of which also consume entries from the same MAC table. The MAC table has a limited capacity of 16 entries. If the MAC address for BOB cannot be programmed because the table is full with other MAC entries, the BOB session will fail to establish. This limitation means that the MAC table resource must be managed carefully to ensure successful BOB session establishment.
- BOB feature is supported only in IETF mode.

- If peer routers have mismatched BFD configurations, with one router configured with **bundle coexistence bob-blb logical** command and the other not, the BFD session may behave unexpectedly.

Note The **bundle coexistence bob-blb logical** command is specific to Cisco IOS XR. Therefore, for consistent BFD behaviour, this command must be configured identically across all peer nodes.

- The **bfd interface** *type interface-path-id* **disable** command is not supported.

BFD Timers

The BFD timers are applicable on the following NCS 540 variants:

Medium Density XR NCS 540 - N540-24Z8Q2C-SYS, N540-28Z4C-SYS, N540X-ACC-SYS, N540-ACC-SYS

Medium Density XR NCS 540 - N540-28Z4C-SYS-A, N540-28Z4C-SYS-D, N540X-16Z4G8Q2C-A, N540X-16Z4G8Q2C-D, N540X-16Z8Q2C-D, N540-12Z20G-SYS-A,N540-12Z20G-SYS-D, N540X-12Z16G-SYS-A, N540X-12Z16G-SYS-D

Small Density XR NCS 540 - N540X-6Z18G-SYS-A, N540X-6Z18G-SYS-D, N540X-8Z16G-SYS-A, N540X-8Z16G-SYS-D

Note If the timer is configured below the minimum timer supported, some undesirable behavior can be seen in BFD. customers some time will configure 3 msec as timer and will miss the minimum timer of 4 msec.

Note The router uses six unique timer profiles. Four timers profiles are available when you configure BFD over Bundle (BoB). Up to five timers profiles are available when you configure BoB.

Table 1: IPv4 BFD Timers

Type of BFD Session	Minimum Timer Supported	Minimum Multipliers Value	Supported Minimum-Interval Value (Up to 6 Unique Timers Profiles)
Single Hop	4ms	3	Any
BFD over Bundle Members (BoB)	4ms	3	Any
BFD over Logical bundle (BLB)	100ms (starting Release 24.3.1) 300ms (prior to Release 24.3.1)	3	Any

Type of BFD Session	Minimum Timer Supported	Minimum Multipliers Value	Supported Minimum-Interval Value (Up to 6 Unique Timers Profiles)
BGP Multi Hop	50ms	3	Any
BFD Over BVI	50ms	3	Any

Table 2: IPv6 BFD Timers

Type of BFD Session	Minimum Timer Supported	Minimum Multipliers Value	Supported Timer Profile (Up to 6 unique timer profiles)	Maximum Scale depending on Minimum Interval
Single Hop	4ms	3	Any	150 (with 8ms and above, all 256 sessions are configurable)
BFD over Bundle Members (BoB)	4ms	3	Any	150ms (with 8ms and above, all 256 sessions are configurable)
BFD over Logical bundle (BLB)	100ms (starting Release 24.3.1) 300ms (prior to Release 24.3.1)	3	Any	256
BGP Multi Hop	50ms	3	Any	256
BFD Over BVI	50ms	3	Any	250 or Max MP scale- whichever is lower

Enable and Disable IPv6 Checksum Calculations for BFD on a Router

Perform the following steps to configure IPv6 checksum calculations for BFD on a Router.

```
RP/0/RP0/CPU0:router(config)# bfd
RP/0/RP0/CPU0:router(config-bfd-if)# ipv6 checksum disable
RP/0/RP0/CPU0:router(config-bfd-if)# commit
```

Configure BFD Under a Dynamic Routing Protocol or Use a Static Route

To establish a BFD neighbor, complete at least one of the following procedures to configure BFD under a dynamic routing protocol or to use a static route:

Enable BFD for OSPF on an Interface

Perform the following steps to configure BFD for Open Shortest Path First (OSPF) on an interface. The steps in the procedure are common to the steps for configuring BFD on IS-IS; only the command mode differs.

Note BFD per interface configuration is supported for OSPF and IS-IS only.

```
Router# configure

/* Enter OSPF configuration mode to configure the OSPF routing process. */
Router(config)# router ospf 0

/* Set the BFD minimum interval. The range is from 15 to 30000 milliseconds. */
Router(config-ospf)# bfd minimum-interval 6500

/* Set the BFD multiplier. */
Router(config-ospf)# bfd multiplier 7

/* Configure an Open Shortest Path First (OSPF) area. */
Router(config-ospf)# area 0

/* Enter interface configuration mode. */
Router(config-ospf-ar)# interface gigabitEthernet 0/3/0/1

/* Enable BFD to detect failures in the path between adjacent forwarding engines. */
Router(config-ospf-ar-if)# bfd fast-detect
```

Running Configuration

```
configure
  router ospf 0
  bfd minimum-interval 6500
  bfd multiplier 7
   area 0
    interface gigabitEthernet 0/3/0/1
     bfd fast-detect
```

Verification

Verify that BFD is enabled on the appropriate interface.

```
Router(config-ospf-ar-if)# show run router ospf

 router ospf 0
 bfd minimum-interval 6500
 bfd multiplier 7
 area 0
 interface gigabitEthernet 0/3/0/1
 bfd fast-detect

/* Verify the details of the IPv4 BFD session in the source router. */

Router# show bfd session

Interface  Dest Addr  Local det time(int*mult)  State  Echo  Async   H/W   NPU
---------  ---------  --------  -------------   -----  ----  -----   ---   ---
```

```
Te0/0/0/0  10.23.1.2  0s(0s*0)  300ms(100ms*3)  UP     Yes               0/RP0/CPU0
BE3739     10.23.1.2  n/a       n/a             UP     No    n/a
```

Enable BFD over BGP

Perform the following steps to configure BFD over BGP. The following example shows how to configure BFD between autonomous system 65000 and neighbor 192.168.70.2:

```
Router# configure
Router(config)# router bgp 65000
Router(config-bgp)# bfd multiplier 2
Router(config-bgp)# bfd minimum-interval 20
Router(config-bgp)# neighbor 192.168.70.24
Router(config-bgp-nbr)# remote-as 2
Router(config-bgp-nbr)# bfd fast-detect
Router(config-bgp-nbr)# commit
Router(config-bgp-nbr)# end
```

Running Configuration

```
 router bgp 65000
  bfd multiplier 2
  bfd minimum-interval 20
  neighbor 192.168.70.24
  remote-as 2
  bfd fast-detect
  commit
  end
```

Verification

Verify that BFD has been enabled over BGP.

```
Router# show run router bgp
router bgp 65000
  bfd multiplier 2
  bfd minimum-interval 20
  neighbor 192.168.70.24
  remote-as 2
  bfd fast-detect
```

Enable BFD on an IPv4 Static Route

The following procedure shows how to enable BFD on an IPv4 static route.

```
RP/0/RSP0/CPU0:router# configure

/*Enter static route configuration mode to configure static routing. */
RP/0/RSP0/CPU0:router(config)# router static

/* Enable BFD fast-detection on the specified IPV4 unicast destination address prefix and
on the forwarding next-hop address.*/
RP/0/RSP0/CPU0:router(config-static)# address-family ipv4 unicast 10.2.2.0/24 10.6.0.1 bfd
 fast-detect minimum-interval 1000 multiplier 5

RP/0/RSP0/CPU0:router(config-static)# commit
```

Running Configuration

```
configure
 router static
  address-family ipv4 unicast 10.2.2.0/24 10.6.0.1 bfd fast-detect minimum-interval 1000
multiplier 5
  commit
```

Verification

Verify that BFD is enabled on the appropriate interface.

```
RP/0/RSP0/CPU0:router# show run router static address-family ipv4 unicast

router static
 address-family ipv4 unicast
  10.2.2.0/24 10.6.0.1 bfd fast-detect minimum-interval 1000 multiplier 5
   commit
 !
!
```

Enable BFD on an IPv6 Static Route

The following procedure describes how to enable BFD on a IPv6 static route.

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

/* Enter static route configuration mode to configure static routing.  */
RP/0/RP0/CPU0:router(config)# router static

/* Enable BFD fast-detection on the specified IPv6 unicast destination address prefix and
on the forwarding next-hop address. */
/* BFD sessions are established with the next hop 2001:0DB8:D987:398:AE3:B39:333:783 when
it becomes reachable. */

RP/0/RP0/CPU0:router(config-static)# address-family ipv6 unicast  2001:0DB8:C18:2:1::F/64
2001:0DB8:D987:398:AE3:B39:333:783 bfd fast-detect minimum-interval 150 multiplier 4

RP/0/RP0/CPU0:router(config-static-vrf)# commit
```

Running Configuration

```
configure
 router static
  address-family ipv6 unicast 2001:0DB8:C18:2:1::F/64 2001:0DB8:D987:398:AE3:B39:333:783
bfd fast-detect minimum-interval 150 multiplier 4
  commit
```

Verification

Verify that BFD is enabled on the appropriate interface.

```
RP/0/RP0/CPU0:router# show run router static address-family ipv6 unicast

 configure
 router static
 address-family ipv6 unicast 2001:0DB8:C18:2:1::F/64 2001:0DB8:D987:398:AE3:B39:333:783 bfd
 fast-detect minimum-interval 150 multiplier 4
  commit
```

Clear and Display BFD Counters

The following procedure describes how to display and clear BFD packet counters. You can clear packet counters for BFD sessions that are hosted on a specific node or on a specific interface.

```
RP/0/RP0/CPU0:router# show bfd counters all packet location 0/3/cpu0
RP/0/RP0/CPU0:router# clear bfd counters all packet location 0/3/cpu0
RP/0/RP0/CPU0:router# show bfd counters all packet location 0/3/cpu0
```

BFD Session Types

There are two types of BFD sessions:

- Single Path Sessions
- Multipath Sessions

BFD Singlepath Sessions

BFD over Bundle

BFD Over Bundle (BoB) (RFC 7130) has a BFD session on each bundle member. BOB verifies the ability for each member link to be able to forward Layer 3 packets.

The BoB feature enables BFD sessions to monitor the status of individual bundle member links. BFD notifies the bundle manager immediately when one of the member links goes down, and reduces the bandwidth used by the bundle.

For BoB, the BFD client is bundlemgr. When BFD detects a failure on a bundle member, bundlemgr removes that member from the bundle. If there are not enough members to keep the bundle up, then the main Bundle-Ether interface will go down so that all routing protocols running on the main bundle interface or a subinterface will detect an interface down.

BoB does not provide a true Layer 3 check and is not supported on subinterfaces. However, subinterfaces will go down at the same time as the main interface.

Limitations and Restrictions for BFD over Bundle

The following are the limitations and restrictions in using BoB feature:

- Each BoB session must use a unique destination IP address. If multiple BoB sessions across different bundles share the same destination IP address, BFD should not be enabled.
- It is only supported in IETF mode.
- It is only supported on the main bundle interface; it is not supported on bundle subinterfaces.
- It is not supported on routing protocols, such as OSPF, ISIS, and BGP.
- When the BFD timer is configured to 4 ms, which is the most aggressive timer, 256 sessions can be brought up.

- BFD echo mode and encryption is not supported.

Configure BFD over Bundle

Configuring BFD over bundle involves the following steps:

- Specify the mode, BFD packet transmission intervals, and failure detection times on a bundle.

Note Repeat the same configuration steps in the destination router.

```
/* Enable and Disable IPv6 checksum calculations for BFD on a router. */

Router(config-if)# bfd
Router(config-bfd-if)# dampening disable
Router(config-bfd-if)# commit

/* Specify the mode, BFD packet transmission intervals, and failure detection times on a
bundle */

Router(config)# interface Bundle-Ether 3739
Router(config-if)# bfd mode ietf
Router(config-if)# bfd address-family ipv4 multiplier 3
Router(config-if)# bfd address-family ipv4 destination 10.23.1.2
Router(config-if)# bfd address-family ipv4 fast-detect
Router(config-if)# bfd address-family ipv4 minimum-interval 100
Router(config-if)# bfd address-family ipv6 multiplier 3
Router(config-if)# bfd address-family ipv6 destination 2001:DB8:1::2
Router(config-if)# bfd address-family ipv6 fast-detect
Router(config-if)# bfd address-family ipv6 minimum-interval 100
Router(config-if)# ipv4 address 10.23.1.1 255.255.255.252
Router(config-if)# ipv6 address 2001:DB8:1::2/120
Router(config-if)# load-interval 30
Router(config-if)# commit
Router(config)# interface TenGigE 0/0/0/0
Router(config-if)# bundle id 3739 mode active
```

Running Configuration

```
bfd
 dampening disable!
 !

interface Bundle-Ether3739
 bfd mode ietf
 bfd address-family ipv4 multiplier 3
 bfd address-family ipv4 destination 10.23.1.2
 bfd address-family ipv4 fast-detect
 bfd address-family ipv4 minimum-interval 100
 bfd address-family ipv6 multiplier 3
 bfd address-family ipv6 destination 2001:DB8:1::2
 bfd address-family ipv6 fast-detect
 bfd address-family ipv6 minimum-interval 100
 ipv4 address 10.23.1.1 255.255.255.252
 ipv6 address 2001:DB8:1::2/120
 load-interval 30
```

```
 !

interface TenGigE 0/0/0/0
 bundle id 3739 mode active
```

Verification

The following show command outputs displays the status of BFD sessions on bundle members:

```
/* Verify the details of the IPv4 BFD session. */

Router# show bfd all session

Interface   Dest Addr   Local det  time(int*mult) State   Echo   Async       H/W   NPU
----------  ----------  --------   -------------- -----  ----   -----       ---   ---
Te0/5/0/6   10.10.10.1   0s         450ms(150ms*3)  UP    Yes    0/RP0/CPU0
Te0/5/0/6   10.10.10.1   0s(0s*0)   450ms(150ms*3)  UP    Yes    0/RP1/CPU0
BE5         10.10.10.1   n/a        n/a             UP    No     n/a


/* Verify the details of the IPv6 BFD session. */
Router# show bfd all session

Interface    Dest Addr   Local det  time(int*mult)  State   Echo     Async      H/W   NPU
----------   ----------  --------   --------------  -----   ----     ---------  ---   ---
Te0/5/0/6    10:10::10:1   0s       450ms(150ms*3)   UP      Yes    0/RP0/CPU0
Te0/5/0/6    10:10::10:1 0s(0s*0)   450ms(150ms*3)   UP      Yes    0/RP1/CPU0
BE5          10:10::10:1   n/a        n/a            UP       No    n/a
```

Configuration Example

Enter the OSPF configuration mode to configure the OSPF routing process.

```
Router(config)# router ospf 0
Router(config)# configure LDP-IGP synchronization
Router(config)# mpls ldp sync
Router(config)# enable LDP auto-configuration for a specified OSPF instance
Router(config)# mpls ldp auto-config
```

Configure a BFD over Bundle Session on an Unnumbered Interface.

```
Router(config)# interface Bundle-Ether1
```

Set the BFD minimum interval.

```
Router(config)# bfd minimum-interval 999
```

Enables BFD between the local networking devices and the neighbor whose IP address you configured to be a BGP peer.

```
Router(config)# bfd fast-detect
```

Set the BFD multiplier.

```
Router(config)# bfd multiplier 3
```

Running Configuration

```
!
router ospf 1
log adjacency changes
router-id 10.0.0.4
mpls ldp sync
mpls ldp auto-config
```

```
prefix-suppression
auto-cost reference-bandwidth 1000000
area 0
  interface Bundle-Ether1
   bfd minimum-interval 999
   bfd fast-detect
   bfd multiplier 3
   network point-to-point
  !
!
bfd
multipath include location 0/RP0/CPU0
!
```

Enabling BFD on a BGP Neighbor

BFD can be enabled per neighbor, or per interface. This task describes how to enable BFD for BGP on a neighbor router.

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters mode.
Step 2	**router bgp** *autonomous-system-number* **Example:** `RP/0/RP0/CPU0:router(config)# router bgp 120`	Enters BGP configuration mode, allowing you to configure the BGP routing process.
Step 3	**neighbor** *ip-address* **Example:** `RP/0/RP0/CPU0:router(config-bgp)# neighbor 172.168.40.24`	Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address as a BGP peer. This example configures the IP address 172.168.40.24 as a BGP peer.
Step 4	**remote-as** *autonomous-system-number* **Example:** `RP/0/RP0/CPU0:router(config-bgp-nbr)# remote-as 2002`	Creates a neighbor and assigns it a remote autonomous system. This example configures the remote autonomous system to be 2002.
Step 5	**bfd fast-detect** **Example:** `RP/0/RP0/CPU0:router(config-bgp-nbr)# bfd fast-detect`	Enables BFD between the local networking devices and the neighbor whose IP address you configured to be a BGP peer in Step 3. In the example in Step 3, the IP address 172.168.40.24 was set up as the BGP peer. In this example, BFD is enabled between the local

	Command or Action	Purpose
		networking devices and the neighbor 172.168.40.24.
Step 6	**bfd minimum-interval** *milliseconds* **Example:** `RP/0/RP0/CPU0:router(config-bgp-nbr)#bfd minimum-interval 6500`	Sets the BFD minimum interval. Range is 4-30000 milliseconds.
Step 7	**bfd multiplier** *multiplier* **Example:** `RP/0/RP0/CPU0:router(config-bgp-nbr)#bfd multiplier 7`	Sets the BFD multiplier. This is optional, the minimum is 3 and by default the multiplier will be 3 for all protocols
Step 8	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Enabling BFD for OSPF on an Interface

The following procedures describe how to configure BFD for Open Shortest Path First (OSPF) on an interface. The steps in the procedure are common to the steps for configuring BFD on IS-IS ; only the command mode differs.

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters mode.
Step 2	**router ospf** *process-name* **Example:**	Enters OSPF configuration mode, allowing you to configure the OSPF routing process. **Note**

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router(config)# router ospf 0`	To configure BFD for IS-IS, enter the corresponding configuration mode.
Step 3	**area** *area-id* **Example:** `RP/0/RP0/CPU0:router(config-ospf)# area 0`	Configures an Open Shortest Path First (OSPF) area. Replace *area-id* with the OSPF area identifier.
Step 4	**interface** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config-ospf-ar)# interface TengigabitEthernet 0/3/0/1`	Enters interface configuration mode and specifies the interface name.
Step 5	**bfd fast-detect** **Example:** `RP/0/RP0/CPU0:router(config-ospf-ar-if)# bfd fast-detect`	Enables BFD to detect failures in the path between adjacent routers.
Step 6	**bfd minimum-interval** *milliseconds* **Example:** `RP/0/RP0/CPU0:router(config-ospf-ar-if)# bfd minimum-interval 6500`	Sets the BFD minimum interval. Range is 4-30000 milliseconds. This example sets the BFD minimum interval to 6500 milliseconds.
Step 7	**bfd multiplier** *multiplier* **Example:** `RP/0/RP0/CPU0:router(config-ospf-ar-if)# bfd multiplier 7`	Sets the BFD multiplier. This is optional, the minimum is 3 and by default the multiplier will be 3 for all protocols. This example sets the BFD multiplier to 7.
Step 8	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Enabling BFD on a Static Route

The following procedure describes how to enable BFD on a static route.

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters mode.
Step 2	**router static** **Example:** `RP/0/RP0/CPU0:router(config)# router static`	Enters static route configuration mode, allowing you to configure static routing.
Step 3	**address-family ipv4 unicast** *address nexthop* **Example:** `RP/0/RP0/CPU0:router(config-static)# address-family ipv4 unicast 10.2.2.0/24 10.6.0.2`	Enables BFD fast-detection on the specified IPv4 unicast destination address prefix and on the forwarding next-hop address.
Step 4	**interface** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config-static)# interface TengigabitEthernet 0/3/0/1`	Enters interface configuration mode and specifies the interface name.
Step 5	**bfd fast-detect** **Example:** `RP/0/RP0/CPU0:router(config-static-if)# bfd fast-detect`	Enables BFD to detect failures in the path between adjacent forwarding engines.
Step 6	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes.

	Command or Action	Purpose
		• **Cancel** —Remains in the configuration session, without committing the configuration changes.

Enabling BFD Sessions on Bundle Members

To enable BFD sessions on bundle member links, complete these steps:

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters mode.
Step 2	**interface Bundle-Ether** *bundle-id* **Example:** `RP/0/RP0/CPU0:router(config)# interface Bundle-Ether 1`	Enters interface configuration mode for the specified bundle ID.
Step 3	**bfd address-family ipv4 fast-detect** **Example:** `RP/0/RP0/CPU0:router(config-if)# bfd address-family ipv4 fast-detect`	Enables IPv4 BFD sessions on bundle member links.
Step 4	**bfd mode ietf** **Example:** `RP/0/RP0/CPU0:router(config-if)# bfd mode ietf`	Enables IETF mode for BFD over bundle for the specified bundle.
Step 5	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Specifying the BFD Destination Address on a Bundle

To specify the BFD destination address on a bundle, complete these steps:

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters mode.
Step 2	**interface Bundle-Ether** *bundle-id* **Example:** `RP/0/RP0/CPU0:router(config)# interface Bundle-Ether 1`	Enters interface configuration mode for the specified bundle ID.
Step 3	**bfd address-family ipv4 destination** *ip-address* **Example:** `RP/0/RP0/CPU0:router(config-if)# bfd address-family ipv4 destination 10.20.20.1`	Specifies the primary IPv4 address assigned to the bundle interface on a connected remote system, where *ip-address* is the 32-bit IP address in dotted-decimal format (A.B.C.D).
Step 4	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configuring the Minimum Thresholds for Maintaining an Active Bundle

The bundle manager uses two configurable minimum thresholds to determine whether a bundle can be brought up or remain up, or is down, based on the state of its member links.

- Minimum active number of links
- Minimum active bandwidth available

Whenever the state of a member changes, the bundle manager determines whether the number of active members or available bandwidth is less than the minimum. If so, then the bundle is placed, or remains, in DOWN state. Once the number of active links or available bandwidth reaches one of the minimum thresholds, then the bundle returns to the UP state.

To configure minimum bundle thresholds, complete these steps:

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters mode.
Step 2	**interface Bundle-Ether** *bundle-id* **Example:** `RP/0/RP0/CPU0:router(config)# interface Bundle-Ether 1`	Enters interface configuration mode for the specified bundle ID.
Step 3	**bundle minimum-active bandwidth** *kbps* **Example:** `RP/0/RP0/CPU0:router(config-if)# bundle minimum-active bandwidth 580000`	Sets the minimum amount of bandwidth required before a bundle can be brought up or remain up. The range is from 1 through a number that varies depending on the platform and the bundle type.
Step 4	**bundle minimum-active links** *links* **Example:** `RP/0/RP0/CPU0:router(config-if)# bundle minimum-active links 2`	Sets the number of active links required before a bundle can be brought up or remain up. The range is from 1 to 32. **Note** When BFD is started on a bundle that is already active, the BFD state of the bundle is declared when the BFD state of all the existing active members is known.
Step 5	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configuring BFD Packet Transmission Intervals and Failure Detection Times on a Bundle

BFD asynchronous packet intervals and failure detection times for BFD sessions on bundle member links are configured using a combination of the **bfd address-family ipv4 minimum-interval** and **bfd address-family ipv4 multiplier** interface configuration commands on a bundle.

The BFD control packet interval is configured directly using the **bfd address-family ipv4 minimum-interval** command. The failure detection times are determined by a combination of the interval and multiplier values in these commands.

To configure the minimum transmission interval and failure detection times for BFD asynchronous mode control packets on bundle member links, complete these steps:

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters mode.
Step 2	**interface Bundle-Ether** *bundle-id* **Example:** `RP/0/RP0/CPU0:router(config)# interface Bundle-Ether 1`	Enters interface configuration mode for the specified bundle ID.
Step 3	**bfd address-family ipv4 minimum-interval** *milliseconds* **Example:** `RP/0/RP0/CPU0:router(config-if)#bfd address-family ipv4 minimum-interval 2000` **Note** Specifies the minimum interval, in milliseconds, for asynchronous mode control packets on IPv4 BFD sessions on bundle member links. The range is from 4 to 30000.	
Step 4	**bfd address-family ipv4 multiplier** *multiplier* **Example:** `RP/0/RP0/CPU0:router(config-if)#bfd address-family ipv4 multiplier 30`	Specifies a number that is used as a multiplier with the minimum interval to determine BFD control packet failure detection times and transmission intervals for IPv4 BFD sessions on bundle member links. The range is from 2 to 50. The default is 3. **Note** Although the command allows you to configure a minimum of 2, the supported minimum is 3.

	Command or Action	Purpose
Step 5	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configuring BFD over Bundle per Member Mode

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters mode.
Step 2	**bfd bundle per-member mode ietf** **Example:** `RP/0/RP0/CPU0:router(config)# bfd bundle per-member mode ietf`	Enables IETF mode for BFD over per-bundle member link.
Step 3	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configure BFD over Bundles IETF Mode Support on a Per Bundle Basis

To configure BFD over Bundles IETF mode support on a per bundle basis use these steps:

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters mode.
Step 2	**interface Bundle-Ether** *bundle-id* **Example:** `RP/0/RP0/CPU0:router(config)# interface Bundle-Ether 1`	Enters interface configuration mode for the specified bundle ID.
Step 3	**bfd mode ietf** **Example:** `RP/0/RP0/CPU0:router(config-if)# bfd mode ietf`	Enables IETF mode for BFD over bundle for the specified bundle.
Step 4	**bfd address-family ipv4 fast-detect** **Example:** `RP/0/RP0/CPU0:router(config-if)# bfd address-family ipv4 fast-detect`	Enables IPv4 BFD sessions on the specified bundle.
Step 5	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.
Step 6	**show bundle bundle-ether** *bundle-id*	Displays the selected bundle mode.

BoB Configuration for IPv4 and IPv6

Table 3: Feature History

Feature Name	Release Information	Feature Description
BFD v6 - HW Offload and IPv6 BFD/BoB (Bundle over Bundle)	Release 7.3.1	The Bidirectional Forwarding detection (BFD) Hardware Offload feature enables the offload of a BFD session in an IPv6 network. With this feature, each bundle member link with IPv6 address runs its own BFD session This feature improves scale and reduces the overall network convergence time by sending rapid failure detection packets to the routing protocols for recalculating the routing table.

The Bidirectional Forwarding detection (BFD) Hardware Offload feature enables the offload of a BFD session to the network processing units of the line cards, in an IPv4 network. BFD hardware offload improves scale and reduces the overall network convergence time by sending rapid failure detection packets to the routing protocols for recalculating the routing table.

Restrictions

- BFD over Bundle (BOB) over IPv6 is not supported with dynamically configured link-local address. It must be statically configured.
- In Cisco IOS XR Release 26.2.2 and Cisco IOS XR Release 26.3.1, **show bfd session** commands that include the **hardware** keyword can display this warning:

  ```
  sysdb_list failed 0x40818800,
          'sysdb' detected the 'warning' condition 'An invalid argument was passed to a
  SysDB function
          (maybe a NULL pointer?)'
  ```

  The output can be incomplete when hardware-offloaded sessions require multiple result pages. This warning does not affect BFD sessions or packet forwarding.

  To avoid the warning, run **show bfd session** command without the **hardware** keyword. Check the `H/W` and `NPU` columns to verify hardware-offload status.

Confiugration Example

Configuration example for IPv4

```
/* Configure BFD over Bundle(BOB) for hardware offload. */
Router# config
Router(config)# interface Bundle-Ether 1
Router(config-if)# bfd mode ietf
Router(config-if)# bfd address-family ipv4 multiplier 3
Router (config-if)# bfd address-family ipv4 destination 10.20.20.1
Router (config-if)# bfd address-family ipv4 fast-detect
Router(config-if)# bfd address-family ipv4 minimum-interval 2000
Router(config-if)# ipv4 address 10.20.20.2/30
```

Configuration example for IPv6

```
/* Configure BFD over Bundle(BOB) for hardware offload. */
Router# config
Router(config)# interface Bundle-Ether 1
Router(config-if)# bfd mode ietf
Router(config-if)# bfd address-family ipv6 multiplier 3
Router (config-if)# bfd address-family ipv6 destination 10.20:20::1
Router (config-if)# bfd address-family ipv6 fast-detect
Router(config-if)# bfd address-family ipv6 minimum-interval 2000
Router(config-if)# ipv6 address 10:20:20::2/64
```

Configuration Verification

Configuration example for IPv4

Use the **show bfd ipv4 session** command to verify the BoB Configuration for IPv4:

```
Router#show bfd ipv4 session
Interface           Dest Addr           Local det time(int*mult)      State
                                    Echo             Async   H/W   NPU
------------------- --------------- ---------------- ---------------- ----------
Hu0/0/0/22           10.20.20.1      0s(0s*0)         6s(2s*3)        UP
                                                             Yes     0/0/CPU0
BE1                 10.20.20.1      n/a              n/a              UP
                                                             No      n/a
```

Configuration example for IPv6

Use the **show bfd ipv6 session** command to verify the BoB Configuration for IPv6:

```
Router#show bfd ipv6 session
Interface           Dest Addr
                                        Local det time(int*mult)      State
H/W                 NPU             Echo             Async
------------------- --------------- ---------------- ---------------- ----------
Hu0/0/0/1           10.20:20::1
Yes                 0/0/CPU0        0s(0s*0)         6s(2s*3)             UP
BE1                 10.20:20::1
No                  n/a             n/a              n/a              UP
```

BFD Hardware Offload Support for IPv6

Table 4: Feature History

Feature Name	Release Information	Feature Description
BFD v6 - HW Offload and IPv6 BFD/BoB (Bundle over Bundle)	Release 7.3.1	The Bidirectional Forwarding detection (BFD) Hardware Offload feature enables the offload of a BFD session in an IPv6 network. With this feature, each bundle member link with IPv6 address runs its own BFD session This feature improves scale and reduces the overall network convergence time by sending rapid failure detection packets to the routing protocols for recalculating the routing table.

The Bidirectional Forwarding detection (BFD) Hardware Offload feature enables the offload of a BFD session to the network processing units of the line cards, in an IPv6 network. BFD hardware offload feature improves scale and reduces the overall network convergence time by sending rapid failure detection packets to the routing protocols for recalculating the routing table.

Restrictions

- This feature is not supported over MPLS LDP interfaces.
- This feature is not supported over MPLS TE or RSVP tunnel.
- BFD Dampening is not supported for BFD over IPv6.
- BFD over Bundle (BOB) over IPv6 is not supported with dynamically configured link-local address. It must be statically configured.
- BFD multihop will flap if underlay paths that consist of multiple bundle VLANs flap.

Configuration Example

```
/* Configure BFD over Bundle(BOB) for hardware offload. */
Router# config
Router(config)# interface Bundle-Ether 1
Router(config-if)# bfd mode ietf
Router(config-if)# bfd address-family ipv6 multiplier 3
Router (config-if)# bfd address-family ipv6 destination 10.20:20::1
Router (config-if)# bfd address-family ipv6 fast-detect
Router(config-if)# bfd address-family ipv6 minimum-interval 2000
Router(config-if)# ipv6 address 10:20:20::2/64

/* To define the line card to host BLB and BFD multihop sessions. */
Router(config)# bfd
Router(config-bfd)# multipath include location 0/RP0/CPU0

/* Configure BFD with a static route. */
Router(config)# router static
Router(config-static)# address-family ipv6 unicast 1011:17e4::1/128 ab11:15d2::2 bfd
fast-detect minimum-interval 50 multiplier 3

/* Configure BFD with IS-IS. */
Router(config)# router isis 65444
Router(config-isis)# address-family ipv6 unicast
Router(config-isis)# exit
Router(config-isis)# interface gigabitEthernet 0/3/0/1
Router(config-isis-if)#  bfd minimum-interval 6500
Router(config-isis-if)# bfd multiplier 7
Router(config-isis-if)# bfd fast-detect ipv6
Router(config-isis-if)# address-family ipv6 unicast

/* Configure BFDv6 with OSPFv3. */
Router(config)# router ospfv3 main
Router(config-ospfv3)# area 0
Router(config-ospfv3-ar)#  interface gigabitEthernet 1/0/0/1
Router(config-ospfv3-ar-if)# bfd multiplier 7
Router(config-ospfv3-ar-if)# bfd fast-detect
Router(config-ospfv3-ar-if)# bfd minimum-interval 6500

/* Configuring BFD over BGP. */
Router(config)# router bgp 120
Router(config-bgp)# neighbor 2001:DB8:1::1
Router(config-bgp-nbr)# bfd fast-detect
```

```
Router(config-bgp-nbr)# bfd multiplier 7
Router(config-bgp-nbr)# bfd minimum-interval 6500
```

Verification

Use the **show bfd ipv6 session** command to verify the configuration:

```
Router# show bfd ipv6 session
Interface           Dest Addr
                                      Local det time(int*mult)      State
H/W                 NPU             Echo             Async
------------------- --------------- ---------------- ---------------- ----------
BE7.2               fe80::28a:96ff:fed6:9cdb
Yes                 0/RP0/CPU0        0s(0s*0)         900ms(300ms*3)   UP
BE7.4               fe80::28a:96ff:fed6:9cdb
Yes                 0/RP0/CPU0        0s(0s*0)         900ms(300ms*3)   UP
```

BFD over Bundle with IPv4 Unnumbered Interfaces

BFD over Bundle with IPv4 Unnumbered Interfaces feature enables BFD to run on IP unnumbered interfaces, which take the IP address from the loopback address. The same loopback address is used on multiple interfaces. This saves IP addresses space or range.

BFD creates a session on the unnumbered interface for which the BFD clients provide the source and destination IP address along with the interface index. BFD establishes the session on the Layer 3 unnumbered link to which the interface index corresponds. The source address is derived from the Loopback interface at the source. The destination node also uses IP unnumbered interface with loopback address and that is used as destination IP address.

BFD sends control packets to the unnumbered interfaces. These control packets are the regular IP BFD packets. Address Resolution Protocol (ARP) resolves the destination loopback IP address to the destination node's router MAC address.

Restriction

Only Asynchronous mode is supported.

Configure BFD over Bundle with IPv4 Unnumbered Interface

- Configure loopback address
- Add physical interface to bundle
- Configure BOB session on an unnumbered interface

Configure Loopback Address

```
Router(config)# interface loopback 1
Router(config-if)# ipv4 address 10.1.1.1 255.255.255.0
```

Add Physical Interface to Bundle

```
Router(config)# interface HundredGigE0/0/1/0
Router(config-if)# bundle id 1 mode on
```

Configure a BFD over Bundle Session on an Unnumbered Interface

```
Router(config)# interface Bundle-Ether1
Router(config-if)# bfd address-family ipv4 destination 10.2.2.2
```

```
Router(config-if)# bfd address-family ipv4 fast-detect
Router(config-if)# ipv4 point-to-point
Router(config-if)# ipv4 unnumbered Loopback1
```

Running Configuration

```
interface Loopback1
ipv4 address 10.1.1.1 255.255.255.0
!
interface HundredGigE0/0/1/0
bundle id 1 mode on
!
interface Bundle-Ether1
bfd address-family ipv4 destination 10.2.2.2
bfd address-family ipv4 fast-detect
ipv4 point-to-point
ipv4 unnumbered Loopback1
```

Configuration Verification

```
show bfd session

Interface           Dest Addr           Local det time(int*mult)      State
                                      Echo              Async   H/W   NPU
------------------- --------------- ----------------- ---------------- ----------
Hu0/0/1/0           10.2.2.2      0s(0s*0)          450ms(150ms*3)   UP
                                                                 Yes   0/0/CPU0
BE1                 10.2.2.2      n/a               n/a               UP
                                                                 No    n/a
```

BFD Multipath Sessions

BFD can be applied over virtual interfaces such as GRE tunnel interfaces, PWHE interfaces, or between interfaces that are multihops away as described in the IPv4 Multihop BFD section. These types of BFD sessions are referred to BFD Multipath sessions.

As long as one path to the destination is active, these events may or may not cause the BFD Multipath session to fail as it depends on the interval negotiated versus the convergence time taken to update forwarding plane:

- Failure of a path
- Online insertion or removal (OIR) of a line card which hosts one or more paths
- Removal of a link (by configuration) which constitutes a path
- Shutdown of a link which constitutes a path

You must configure **bfd multipath include location** *location_id* command to enable at least one line card for the underlying mechanism that can be used to send and receive packets for the multipath sessions.

If a BFD multipath session is hosted on a line card that is being removed from the bfd multipath include configuration, online removed, or brought to maintenance mode, then BFD attempts to migrate all BFD Multipath sessions hosted on that line card to another one. In that case, static routes are removed from RIB and then the BFD session is established again and included to RIB.

In case of BFD multipath sessions, the input and output interface may change based on the routing table updates. If the multipath session BFD packets must get preferential treatment, then a QoS policy must be configured on the entire path, including the possible input and output interfaces of the router.

Note The CLI **bfd multipath include location** *location* is a mandatory configuration to download BFD sessions on a given location.

BFD over BVI

Table 5: Feature History

Feature Name	Release Information	Feature Description
BFD on BVI	Release 7.3.1	BFD can be configured on Bridge group Virtual Interface (BVI). BVI is a virtual interface within the router that acts like a normal routed interface that does not support bridging but represents the bridge group for the bridged physical interfaces. BFD detects the Layer3 fault over the BVI much quicker and informs the same to routing protocols.

In order for a VLAN to span a router, the router must be capable of forwarding frames from one interface to another, while maintaining the VLAN header. If the router is configured for routing a Layer 3 (network layer) protocol, it will terminate the VLAN and MAC layers at the interface on which a frame arrives. The MAC layer header can be maintained if the router bridges the network layer protocol. However, even regular bridging terminates the VLAN header.

Using the Integrated Routing Bridging (IRB) feature, a router can be configured for routing and bridging the same network layer protocol, on the same interface. This allows the VLAN header to be maintained on a frame while it transits a router from one interface to another. IRB provides the ability to route between a bridged domain and a routed domain with the Bridge Group Virtual Interface (BVI). The BVI is a virtual interface within the router that acts like a normal routed interface that does not support bridging, but represents the comparable bridge group to routed interfaces within the router. The interface number of the BVI is the number of the bridge group that the virtual interface represents. This number is the link between the BVI and the bridge group.

Because the BVI represents a bridge group as a routed interface, it must be configured only with Layer 3 (L3) characteristics, such as network layer addresses. Similarly, the interfaces configured for bridging a protocol must not be configured with any L3 characteristics.

BFD over IRB is a multipath single-hop session. BFD over IRB is supported on IPv4 address, IPv6 global address, and IPv6 link-local address. The BFD over IRB is supported only in asynchronous mode and does not support echo mode.

IPv4 Multihop BFD

IPv4 Multihop BFD is a BFD session between two addresses that are several hops away. An example of this feature is a BFD session between PE and CE loopback addresses or BFD sessions between routers that are several TTL hops away. The applications that support IPv4 Multihop BFD are external and internal BGP. IPv4 Multihop BFD feature supports BFD on arbitrary paths, which can span multiple networks hops.

A Virtual Routing and Forwarding (VRF) instance is a logical separation of a router's routing table. VRF allows you to have multiple routing tables on a single router, each with its own set of routes.

The default VRF is the first VRF that is created on a router. It is the VRF that is used by default for all routing protocols and interfaces.

Non-default VRFs must be explicitly configured.

The IPv4 Multihop BFD feature provides subsecond forwarding failure detection for a destination more than one hop, and up to 255 hops, away. IPv4 Multihop BFD feature is supported on all currently supported media-type for BFD single hop.

You can set up a BFD multihop session between a unique source-destination address pair that is provided by the client. You can set up a session two endpoints that have IP connectivity.

Multihop BFD feature runs on both default and non-default VRF.

Configure IPv4 Multihop BFD

This section describes how you can configure IPv4 Multihop BFD feature.

```
Router# configure
Router(config)# bfd
Router(config)# multipath include location 0/7/CPU0
Router(config)# router bgp 100
Router(config-bgp)# neighbor 209.165.200.225
Router(config-bgp-nbr)# remote-as 2000
Router(config-bgp-nbr)# update-source loopback 1
Router(config-bgp-nbr)# bfd fast-detect
Router(config-bgp-nbr)# bfd multiplier 3
Router(config-bgp-nbr)# bfd minimum-interval 300
Router(config-bgp-nbr-af)# commit
```

Running Configuration

```
bfd
 multipath include location 0/7/CPU0
router bgp 100
 neighbor 209.165.200.225
  remote-as 2000
  update-source loopback 1
  bfd fast-detect
  bfd multiplier 3
  bfd minimum-interval 300
address-family ipv4 unicast
```

Configure Multihop BFD on IPv4 Non-default VRFs

Configure multihop BFD on IPv4 or IPv6 non-default VRFs:

- Configure BGP with the Autonomous System Number (ASN) on the router.

- Define a BGP neighbor with the specified IPv4 or IPv6 address.
- Associate the neighbor with a non-default VRF named "vrf1."
- Assign a route distinguisher value to create a routing and forwarding table for a VRF.
- Configure the redistribution of connected routes.
- Establish and configure an eBGP session with the specified IPv4 or IPv6 neighbor.
- Configure the remote ASN.
- Enable BFD for fast link failure detection.
- Set the BFD detection time parameters.
- Configure eBGP sessions.
- Specify the primary IP address from a particular interface as the local address when forming an eBGP session with a neighbor.
- Apply a route-policy for both inbound and outbound traffic.

Configure the following steps to configure Multihop BFD on IPv4 nondefault VRF:

```
Router(config)# router bgp 100
Router(config-bgp)# neighbor 209.165.200.225
Router(config-bgp-nbr)#vrf vrf1
Router(config-bgp-nbr-vrf)# exit
Router(config-bgp-nbr)# rd auto
Router(config-bgp-nbr)#address-family ipv4 unicast
Router(config-bgp-nbr-af)#redistribute connected
Router(config-bgp-nbr-af)# exit
Router(config-bgp)# neighbor 209.165.200.225
Router(config-bgp-nbr)# remote-as 2000
Router(config-bgp-nbr)# bfd fast-detect
Router(config-bgp-nbr)# bfd multiplier 3
Router(config-bgp-nbr)# bfd minimum-interval 50
Router(config-bgp-nbr)# ebgp-multihop 255
Router(config-bgp-nbr)# update-source loopback 1
/* You can configure any interface here, including loopback or bvi */
Router(config-bgp-nbr)#address-family ipv4 unicast
Router(config-bgp-nbr-af)# route-policy pass-all in
Router(config-bgp-nbr-af)# route-policy pass-all out
Router(config-bgp-nbr-af)# commit
```

Running Configuration

```
router bgp 100
  neighbor 209.165.200.225
   vrf vrf1
    exit
     rd auto
      address-family ipv4 unicast
       redistribute connected
        exit
         neighbor 209.165.200.225
         remote-as 2000
         bfd fast-detect
         bfd multiplier 3
         bfd minimum-interval 50
```

```
ebgp-multihop 255
update-source loopback 1
 address-family ipv4 unicast
  route-policy pass-all in
  route-policy pass-all out
```

Verification

```
Router# show bfd session source 209.165.200.225
Thu Mar 10 10:13:43.305 IST
Src Addr         Dest Addr    VRF Name         H/W NPU
                               Local det time(int*mult)   State
                              Echo    Async
--------------   ---------   ------   ------ --------- ---------
209.165.200.225  192.0.2.254   vrf_1     Yes   0/0/CPU0
                              n/a  150ms(50ms*3)          UP
Router# show cef vrf vrf_1 209.165.200.225 location 0/0/CPU0
Thu Mar 10 10:24:13.372 IST
209.165.200.0/24, version 40, internal 0x5000001 0x30 (ptr 0x8ae26458) [1], 0x0 (0x0), 0xa08
 (0x8dc144a8)
 Updated Mar  9 15:09:43.398
 Prefix Len 24, traffic index 0, precedence n/a, priority 3
LDI Update time Mar  9 14:59:28.284
   via 1.1.1.1/32, 605 dependencies, recursive [flags 0x6000]
    path-idx 0 NHID 0x0 [0x8dd35988 0x0]
    recursion-via-/32
    next hop VRF - 'default', table - 0xe0000000
    next hop 10.1.1.1/32 via 24015/0/21
    next hop 192.0.2.255/32 Te0/0/0/3.1  labels imposed {ImplNull 24162}
```

Multihop BFD over BVI

Table 6: Feature History Table

Feature Name	Release Information	Feature Description
Multihop BFD over Bridge Group Virtual Interface (BVI)	Release 7.4.1	The multihop BFD over Bridge Group Virtual Interface (BVI) feature introduces support for multihop BFD over (BVI). You can set up a multihop BFD session between two endpoints that have IP connectivity. This session is between a unique source-destination address pair that the client provides. This feature allows you to extend BFD on arbitrary paths. These arbitrary paths can span multiple network hops, hence detecting link failures.

Multihop BFD over BVI feature allows you to configure both routing and bridging on the same interface using Integrated Routing Bridging (IRB). IRB enables you to route between a bridged domain and a routed domain with the Bridge Group Virtual Interface (BVI).

The BVI is a virtual interface within the router that acts like a normal, routed interface that does not support bridging, but represents the comparable bridge group to routed interfaces within the router.

Restrictions

- The minimum Multihop BFD timer for the BVI interface is 50 msec.

- The **multihop ttl-drop-threshold** command is not supported.
- The Multihop BFD over BVI or IRB functionality is supported only in asynchronous mode and does not support echo mode.
- The Multihop BFD over BVI feature is not supported over MPLS and SR core.

Supported Functionality

- This feature is supported in both IPv4 and IPv6.
- BFD Multihop over BVI feature supports on client BGP.
- BFD Multihop supports only over IP core.
- BFD Multihop supports on all currently supported media-type for BFD single-hop.

Configuration

```
/* Configure a BVI interface and assign an IP address */
Router(config)# interface BVI1
Router(config-if)# host-routing
Router(config-if)# mtu 8986
Router(config-if)# ipv4 address 10.1.1.1 255.255.255.0
Router(config-if)# ipv6 address 10:1:1::1/120

/* Configure the Layer 2 AC interface */
Router(config-if)# interface TenGigE0/5/0/6/0.1 l2transport
Router(config-subif)# encapsulation dot1q 1
Router(config-subif)# rewrite ingress tag pop 1 symmetric

/* Configure L2VPN Bridge Domain */
Router(config-subif)# l2vpn
Router(config-subif)# bridge group 1
Router(config-subif)# bridge-domain 1
Router(config-l2vpn-bg-bd)# interface TenGigE0/5/0/6/0.1
Router(config-l2vpn-bg-bd)# routed interface BVI1
```

Running Configuration

```
interface BVI1
 host-routing
 mtu 8986
 ipv4 address 10.1.1.1 255.255.255.0
 ipv6 address 10:1:1::1/120
!
interface TenGigE0/5/0/6/0.1 l2transport
 encapsulation dot1q 1
 rewrite ingress tag pop 1 symmetric
!
l2vpn
 bridge group 1
  bridge-domain 1
   interface TenGigE0/5/0/6/0.1
   !
   routed interface BVI1
   !
```

Repeat the configuration on the peer router.

```
/* Configure BGP as the routing protocol  */
Router(config)# router bgp 1
Router(config-bgp)# neighbor 2.2.1.1
Router(config-bgp-nbr)# remote-as 1
Router(config-bgp-nbr)# bfd fast-detect
Router(config-bgp-nbr)# bfd minimum-interval 300
Router(config-bgp-nbr)# update-source Loopback1
Router(config-bgp-nbr)# address-family ipv4 unicast

/* Configure reachability to the BGP neighbour IP either via static or IGP*/
Router(config-bgp-nbr-af)# router static
Router(config-static)# address-family ipv4 unicast
Router(config-static-afi)# 2.2.1.1/32 10.1.1.2

/* Configure the line cards to allow hosting of Multipath BFD sessions. */
Router(config-static-afi)# bfd
Router(config-bfd)#
 multipath include location 0/RP0/CPU0


router bgp 1
neighbor 2.2.1.1
  remote-as 1
  bfd fast-detect
  bfd minimum-interval 300
  update-source Loopback1
  address-family ipv4 unicast
  !
router static
 address-family ipv4 unicast
  2.2.1.1/32 10.1.1.2
 !
bfd
 multipath include location 0/RP0/CPU0!
```

Note To avoid the unsupported three-level recursion on BVI interfaces on the first and second generation of line cards, you must not configure the BVI interface as the next-hop in the static route configuration.

Verification

```
Router# show bfd session destination 2.2.1.1
Fri May 28 14:35:52.566 IST

Src Addr            Dest Addr       VRF Name                        H/W NPU
                                        Local det time(int*mult)        State
                                    Echo             Async
------------------- --------------- ---------------- ---------------- ----------
1.1.1.1             2.2.1.1         default                          Yes 0/RP0/CPU0
                                    n/a              900ms(300ms*3)   UP
```

Seamless Bidirectional Forwarding Detection

Table 7: Feature History Table

Feature Name	Release Information	Feature Description
Seamless Bidirectional Forwarding Detection	Release 24.2.11	This feature now extends support on the Cisco NCS 540 Series routers running on Cisco IOS XR7.
Seamless Bidirectional Forwarding Detection	Release 24.2.1	Introduced in this release on the following Cisco NCS 540 router variants running on Cisco IOS XR: • N540-ACC-SYS • N540X-ACC-SYS • N540-24Z8Q2C-SYS This feature introduces support for NCS 5500 routers as a Seamless BFD (S-BFD) reflector. Seamless BFD simplifies the negotiation and state establishment aspects of BFD by predetermining session discriminators and maintaining session state only at the headend. This approach ensures quicker connectivity tests and reduces complexity in session establishment. Previously, support for Seamless BFD reflector was not available. The feature introduces these changes: **CLI:** This feature introduces the **sbfd** command.

Advantages of SBFD over BFD

Seamless Bidirectional Forwarding Detection (S-BFD), is a simplified mechanism for using BFD with a large proportion of negotiation aspects eliminated, thus providing benefits such as quick provisioning, as well as improved control and flexibility for network nodes initiating path monitoring.

Components of S-BFD

S-BFD includes the following components:

- S-BFD discriminator
- Reflector BFD session
- S-BFD initiator

Each network node allocates one or more S-BFD discriminators for local entities and creates a reflector BFD session. The S-BFD initiator sends S-BFD control packets with the corresponding discriminator value. The reflector BFD session listens to incoming S-BFD control packets addressed to local entities and generates response S-BFD control packets.

Key differences between BFD and S-BFD

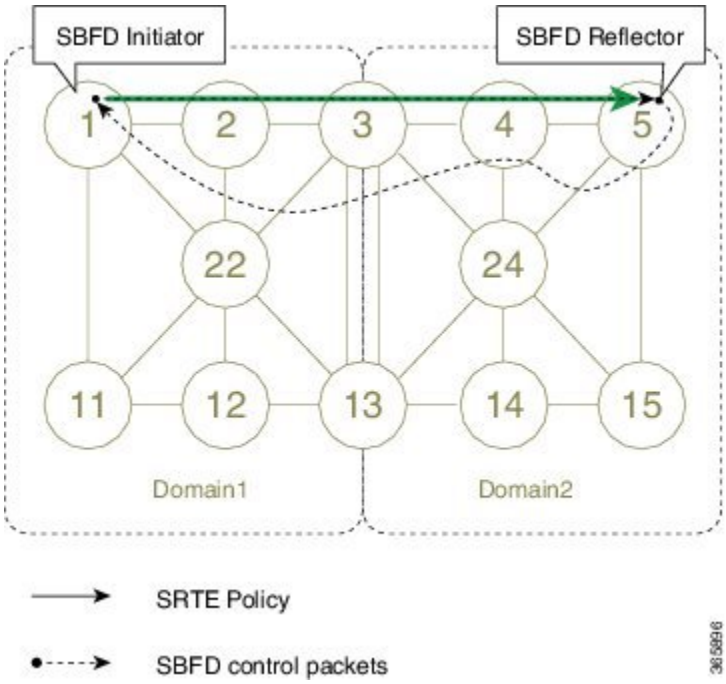
In BFD, each end of the connection maintains a BFD state and transmits packets periodically over a forwarding path. S-BFD is unidirectional, resulting in faster session activation than BFD. The BFD state and client context is maintained on the head-end (initiator) only. The tail-end (reflector) validates the BFD packet and responds, so there is no need to maintain the BFD state on the tail-end.

Initiator and Reflector Components of S-BFD

S-BFD runs in an asymmetric behavior, using initiators and reflectors.

The following figure represents the roles of the S-BFD initiator and reflector.

Figure 1: S-BFD Initiator and Reflector



The initiator is an S-BFD session on a network node that performs a continuity test to a remote entity by sending S-BFD packets. The initiator injects the S-BFD packets into the segment-routing traffic-engineering (SRTE) policy. The initiator triggers the S-BFD session and maintains the BFD state and client context. For more information about configuring SRTE policies, see the *Configure SR-TE Policies* chapter in the *Segment Routing Configuration Guide*.

The S-BFD reflector is an S-BFD session on a network node that listens for incoming S-BFD control packets to local entities and generates response S-BFD control packets. The reflector is stateless and only reflects the S-BFD packets back to the initiator.

Role of Discriminators in S-BFD Control Packet

The BFD control packet carries 32-bit discriminators (local and remote) to demultiplex BFD sessions. S-BFD requires globally unique S-BFD discriminators that are known by the initiator.

The S-BFD control packets contain the discriminator of the initiator, which is created dynamically by the initiator, and the discriminator of the reflector, which is configured as a local discriminator on the reflector.

Usage Guidelines and Limitations for S-BFD

The following usage guidelines and limitations apply:

- The NCS 5500 routers do not support initiator mode.
- The feature support is only for a global VRF and IPv4 addresses.
- The supported Packets Per Second (PPS) limit is up to 3000. Also, consider the jitter used by the initiator for accurate performance assessment.
- The network administrator configures reflector node discriminators at the initiator, allowing the initiator to know the globally unique discriminators of the reflector before the session starts.
- The S-BFD over SR policy is not supported on NCS 540 routers.

Configure the S-BFD Reflector

This section includes steps to configure the S-BFD reflector.

Before you begin

- Each reflector should have at least one globally unique discriminator, to ensure the S-BFD packet arrives on the intended reflector.
- An S-BFD reflector only accepts BFD control packets where "Your Discriminator" is the reflector discriminator.

Procedure

Step 1 Configure the local discriminators on the reflector using the **sbfd local-discriminator** {*ipv4-address* | *32-bit-value* | **dynamic** | **interface** *interface*} command.

You can configure a local discriminator in one of the following ways. For more information about configuring a local discriminator, see the *local-discriminator* command in the *Segment Routing Command Reference for Cisco 5500 Series Routers*.

- Configure an IPv4 address as the local discriminator.

```
Router(config)#sbfd
Router(config-sbfd)#local-discriminator 192.0.2.1
```

- Configure a unique 32-bit value as the local discriminator.

```
Router(config)#sbfd
Router(config-sbfd)#local-discriminator 987654321
```

- Configure an IPv4 address of the interface as the local discriminator.

```
Router(config)#sbfd
Router(config-sbfd)#local-discriminator interface Loopback0
```

- Configure a randomly generated value as the local discriminator.

```
Router(config)#sbfd
Router(config-sbfd)#local-discriminator dynamic
```

Step 2 Verify the configuration using the **show running-config** command.

Example:

```
 local-discriminator 10.1.1.5
 local-discriminator 987654321
 local-discriminator dynamic
 local-discriminator interface Loopback0
!
```

Step 3 Verify the configured BFD local discriminators using the **show bfd target-identifier** command.

Example:

```
Router#show bfd target-identifier local

Local Target Identifier Table
-----------------------------
Discr       Discr Src   VRF       Status   Flags
                        Name
-----       ---------   -------   -------- --------
16843013    Local       default   enable   -----ia-
987654321   Local       default   enable   ----v---
2147483649  Local       default   enable   -------d

Legend: TID - Target Identifier
        a   - IP Address mode
        d   - Dynamic mode
        i   - Interface mode
        v   - Explicit Value mode
```

Step 4 Verify the S-BFD reflector configuration using the **show bfd reflector** command.

Example:

```
Router#show bfd reflector info detail location 0/0/CPU0

Local Discr     : 2147483649
Remote Discr    : 65576
Source Address  : 1.1.1.1
Last DOWN received Time : (NA)
Last Rx packets timestamps before DOWN
  [NA                 ] [NA                 ] [NA                 ] [NA                 ]
```

```
  [NA                 ] [NA                 ] [NA                 ] [NA                 ]
  [NA                 ] [NA                 ]
Last Tx packets timestamps before DOWN
  [NA                 ] [NA                 ] [NA                 ] [NA                 ]
  [NA                 ] [NA                 ] [NA                 ] [NA                 ]
  [NA                 ] [NA                 ]
Last UP sent Time : (Jun  7 14:59:34.763)
Last recent Rx packets timestamps:
  [Jun 7 15:00:18.653 ] [Jun 7 15:00:18.751 ] [Jun 7 15:00:18.837 ] [Jun 7 15:00:18.927 ]
  [Jun 7 15:00:18.085 ] [Jun 7 15:00:18.185 ] [Jun 7 15:00:18.274 ] [Jun 7 15:00:18.372 ]
  [Jun 7 15:00:18.464 ] [Jun 7 15:00:18.562 ]
Last recent Tx packets timestamps:
  [Jun 7 15:00:18.653 ] [Jun 7 15:00:18.751 ] [Jun 7 15:00:18.837 ] [Jun 7 15:00:18.927 ]
  [Jun 7 15:00:18.085 ] [Jun 7 15:00:18.185 ] [Jun 7 15:00:18.274 ] [Jun 7 15:00:18.372 ]
  [Jun 7 15:00:18.464 ] [Jun 7 15:00:18.563 ]
```

Coexistence Between BFD over Bundle and BFD over Logical Bundle

The coexistence between BFD over Bundle (BoB) and BFD over Logical Bundle (BLB) feature allows you to monitor either physical bundle member for BOB, or logical interface for BLB, or both. This feature enables BFD to converge fast.

Difference between BoB and BLB

BFD over Bundle (BoB) (RFC 7130) has a BFD session on each bundle member. The client is the bundle manager. If a BFD session goes down on a specific member link, the whole bundle interface goes down. That is, when the member link goes down, the number of available links falls below the required minimum. Hence the routing session is brought down.

BFD over Logical Bundle (BLB) (RFC 5880) treats a bundle interface with all its members as a single interface. BLB is a multipath (MP) single-hop session. If BLB is configured on a bundle there is only one single BFD session that is active. This implies that only one bundle member is being monitored by BFD at any given time. The client is one of the routing protocols. When BFD detects a failure, the client brings down the routing session.

The mode (BoB or BLB) is determined by how you configure BFD:

- You can enable BoB by configuring BFD under a Bundle-Ether interface.
- You can enable BLB by configuring BFD under a Bundle-Ether interface on a routing client.

Coexistence modes for BoB and BLB

Logical coexistence mode:

When BoB and BLB coexistence is configured in logical mode, only BLB sessions using the main bundle interface's IPv4 address are inherited from the BoB session. BLB sessions configured on bundle sub-interfaces, each with a unique IPv4 address, will establish independent sessions. BLB sessions on loopback interfaces associated with the bundle also operate separately from the main bundle interface.

Inherit Coexistence Mode

In inherit mode, the system does not initiate a standalone BLB session on either the main interface or the sub-interfaces if BoB is not configured on the bundle. In this situation, the BLB session stays in a virtual *Down* state until a corresponding BoB session becomes available. In inherit mode, the BLB session always depends on the inherited BoB session and does not establish its own BFD session with real packets unless BoB is enabled on the bundle. Therefore, without BoB configured on the bundle interfaces, no BLB session will operate in inherit mode.

Functionality Comparison

Logical mode provides greater flexibility by allowing BLB sessions to be established independently on sub-interfaces or loopback interfaces. Inherit mode restricts BLB to inherit only from existing BoB sessions and does not allow independent BLB session establishment.

Note

- Use logical mode when independent BLB sessions per sub-interface or loopback are required.
- Use inherit mode when BLB session activation must be strictly dependent on the presence of a BoB session on the corresponding interface.

Configuration Example

Configure one or more linecards to allow hosting of MP BFD sessions. If no linecards are included, linecards groups are not formed, and consequently no BFD MP sessions are created. For default settings of group size and number, you must add at least two lines with the **bfd multiple-paths include location** *node-id* command and valid line cards to the configuration for the algorithm to start forming groups and BFD MP sessions to be established.

```
Router(config)# bfd multipath include location 0/RP0/CPU0
Router(config)# bfd multipath include location 0/1/CPU0

/* Configure inherited coexistence mode */
Router(config)# bfd
Router(config-bfd)# bundle coexistence bob-blb inherit

/* Configure logical coexistence mode */
Router(config)# bfd
Router(config-bfd)# bundle coexistence bob-blb logical
```

Running Configuration

Running confiiguration for inherited coexistence:

```
bfd
bundle coexistence bob-blb inherit
```

Running confiiguration for logical mode:

```
bfd
bundle coexistence bob-blb logical
```

Verification

Verify BOB and BLB coexistence inherited mode.

```
Router# show bfd session
Mon May 31 02:55:44.584 UTC
Interface           Dest Addr           Local det time(int*mult)      State
                                      Echo             Async   H/W   NPU
```

```
------------------- --------------- ---------------- ---------------- ----------
Te0/0/0/7           33.33.33.2      0s(0s*0)         450ms(150ms*3)   UP
                                                             Yes   0/RP0/CPU0
BE123               33.33.33.2      n/a              n/a              UP
                                                             No    n/a
BE123.1             34.34.34.2      n/a              n/a              UP
                                                             No    n/a

Router# show bfd session interface bundle-ether 123 detail
Fri May 28 13:49:35.268 UTC
I/f: Bundle-Ether123, Location: 0/RP0/CPU0
Dest: 33.33.33.2
Src: 33.33.33.1
 State: UP for 0d:0h:29m:50s, number of times UP: 1
 Session type: PR/V4/SH/BI/IB
Session owner information:
                          Desired               Adjusted
  Client                Interval   Multiplier Interval   Multiplier
  -------------------- ---------- ---------- ---------- ----------
  bundlemgr_distrib     150 ms     3          150 ms     3
Session association information:
  Interface             Dest Addr / Type
  --------------------- -----------------------------------
  Te0/0/0/7             33.33.33.2
                        BFD_SESSION_SUBTYPE_RTR_BUNDLE_MEMBER
  BE123.1               34.34.34.2
                        BFD_SESSION_SUBTYPE_STATE_INHERIT

Router# show bfd session interface bundle-ether 123.1 detail
Fri May 28 13:49:59.316 UTC
I/f: Bundle-Ether123.1, Location: 0/RP0/CPU0
Dest: 34.34.34.2
Src: 34.34.34.1
 State: UP for 0d:0h:12m:54s, number of times UP: 1
 Session type: PR/V4/SH/IH
Session owner information:
                          Desired               Adjusted
  Client                Interval   Multiplier Interval   Multiplier
  -------------------- ---------- ---------- ---------- ----------
  ipv4_static           100 ms     3          100 ms     3
Session association information:
  Interface             Dest Addr / Type
  --------------------- -----------------------------------
  BE123                 33.33.33.2
                        BFD_SESSION_SUBTYPE_RTR_BUNDLE_INTERFACE

Router# show bfd session interface tenGigE 0/0/0/7 detail
Mon May 31 03:00:04.635 UTC
I/f: TenGigE0/0/0/7, Location: 0/0/CPU0
Dest: 33.33.33.2
Src: 33.33.33.1
 State: UP for 2d:13h:40m:19s, number of times UP: 1
 Session type: PR/V4/SH/BM/IB
Received parameters:
 Version: 1, desired tx interval: 150 ms, required rx interval: 150 ms
 Required echo rx interval: 0 ms, multiplier: 3, diag: None
 My discr: 2147493276, your discr: 2147492184, state UP, D/F/P/C/A: 0/0/0/1/0
Transmitted parameters:
 Version: 1, desired tx interval: 150 ms, required rx interval: 150 ms
 Required echo rx interval: 0 ms, multiplier: 3, diag: None
 My discr: 2147492184, your discr: 2147493276, state UP, D/F/P/C/A: 0/0/0/1/0
Timer Values:
 Local negotiated async tx interval: 150 ms
 Remote negotiated async tx interval: 150 ms
 Desired echo tx interval: 0 s, local negotiated echo tx interval: 0 ms
```

```
 Echo detection time: 0 ms(0 ms*3), async detection time: 450 ms(150 ms*3)
Local Stats:
 Intervals between async packets:
   Tx: Number of intervals=4, min=5 ms, max=15 s, avg=6927 ms
       Last packet transmitted 222007 s ago
   Rx: Number of intervals=15, min=3 ms, max=1700 ms, avg=1133 ms
       Last packet received 222018 s ago
 Intervals between echo packets:
   Tx: Number of intervals=0, min=0 s, max=0 s, avg=0 s
       Last packet transmitted 0 s ago
   Rx: Number of intervals=0, min=0 s, max=0 s, avg=0 s
       Last packet received 0 s ago
 Latency of echo packets (time between tx and rx):
   Number of packets: 0, min=0 ms, max=0 ms, avg=0 ms
Session owner information:
                         Desired               Adjusted
  Client               Interval   Multiplier Interval   Multiplier
  -------------------- ---------------------- ----------------------
  bundlemgr_distrib    150 ms     3          150 ms     3
Session association information:
  Interface            Dest Addr / Type
  -------------------- -----------------------------------
  BE123                33.33.33.2
                       BFD_SESSION_SUBTYPE_RTR_BUNDLE_INTERFACE

H/W Offload Info:
 H/W Offload capability : Y, Hosted NPU     : 0/RP0/CPU0
 Async Offloaded        : Y, Echo Offloaded : N
 Async rx/tx            : 122/51

Platform Info:
NPU ID: 0
Async RTC ID        : 1          Echo RTC ID        : 0
Async Feature Mask  : 0x0        Echo Feature Mask  : 0x0
Async Session ID    : 0x2158     Echo Session ID    : 0x0
Async Tx Key        : 0x80002158  Echo Tx Key        : 0x0
Async Tx Stats addr : 0x0   Echo Tx Stats addr : 0x0
Async Rx Stats addr : 0x0   Echo Rx Stats addr : 0x0
```

Verify BOB and BLB coexistence logical mode.

```
show bfd session
Mon May 31 02:54:07.442 UTC
Interface           Dest Addr           Local det time(int*mult)      State
                                        Echo             Async   H/W   NPU
------------------- --------------- ---------------- ---------------- ----------
Te0/0/0/7           33.33.33.2      0s(0s*0)         450ms(150ms*3)   UP
                                                             Yes   0/0/CPU0
BE123.1             34.34.34.2      0s(0s*0)         300ms(100ms*3)   UP
                                                             Yes   0/0/CPU0
BE123               33.33.33.2      n/a              n/a              UP
                                                             No    n/a

Router# show bfd session interface bundle-ether 123 detail
Fri May 28 14:04:41.331 UTC
I/f: Bundle-Ether123, Location: 0/RP0/CPU0
Dest: 33.33.33.2
Src: 33.33.33.1
 State: UP for 0d:0h:44m:56s, number of times UP: 1
 Session type: PR/V4/SH/BI/IB
Session owner information:
                         Desired               Adjusted
  Client               Interval   Multiplier Interval   Multiplier
  -------------------- ---------------------- ----------------------
  bundlemgr_distrib    150 ms     3          150 ms     3
```

```
Session association information:
  Interface            Dest Addr / Type
  -------------------- ------------------------------------
  Te0/0/0/7            33.33.33.2
                       BFD_SESSION_SUBTYPE_RTR_BUNDLE_MEMBER

Router# show bfd session interface tenGigE 0/0/0/7 detail
Mon May 31 03:04:25.714 UTC
I/f: TenGigE0/0/0/7, Location: 0/RP0/CPU0Dest: 33.33.33.2
Src: 33.33.33.1
 State: UP for 2d:13h:44m:40s, number of times UP: 1
 Session type: PR/V4/SH/BM/IB
Received parameters:
 Version: 1, desired tx interval: 150 ms, required rx interval: 150 ms
 Required echo rx interval: 0 ms, multiplier: 3, diag: None
 My discr: 2147493276, your discr: 2147492184, state UP, D/F/P/C/A: 0/0/0/1/0
Transmitted parameters:
 Version: 1, desired tx interval: 150 ms, required rx interval: 150 ms
 Required echo rx interval: 0 ms, multiplier: 3, diag: None
 My discr: 2147492184, your discr: 2147493276, state UP, D/F/P/C/A: 0/0/0/1/0
Timer Values:
 Local negotiated async tx interval: 150 ms
 Remote negotiated async tx interval: 150 ms
 Desired echo tx interval: 0 s, local negotiated echo tx interval: 0 ms
 Echo detection time: 0 ms(0 ms*3), async detection time: 450 ms(150 ms*3)
Local Stats:
 Intervals between async packets:
   Tx: Number of intervals=4, min=5 ms, max=15 s, avg=6927 ms
       Last packet transmitted 222268 s ago
   Rx: Number of intervals=15, min=3 ms, max=1700 ms, avg=1133 ms
       Last packet received 222279 s ago
 Intervals between echo packets:
   Tx: Number of intervals=0, min=0 s, max=0 s, avg=0 s
       Last packet transmitted 0 s ago
   Rx: Number of intervals=0, min=0 s, max=0 s, avg=0 s
       Last packet received 0 s ago
 Latency of echo packets (time between tx and rx):
   Number of packets: 0, min=0 ms, max=0 ms, avg=0 ms
Session owner information:
                         Desired               Adjusted
  Client               Interval   Multiplier Interval   Multiplier
  -------------------- ---------------------- ---------------------
  bundlemgr_distrib    150 ms     3          150 ms     3
Session association information:
  Interface            Dest Addr / Type
  -------------------- ------------------------------------
  BE123                33.33.33.2
                       BFD_SESSION_SUBTYPE_RTR_BUNDLE_INTERFACE

H/W Offload Info:
 H/W Offload capability : Y, Hosted NPU     : 0/RP0/CPU0
 Async Offloaded        : Y, Echo Offloaded : N
 Async rx/tx            : 122/51

Platform Info:
NPU ID: 0
Async RTC ID        : 1          Echo RTC ID        : 0
Async Feature Mask  : 0x0        Echo Feature Mask  : 0x0
Async Session ID    : 0x2158     Echo Session ID    : 0x0
Async Tx Key        : 0x80002158  Echo Tx Key        : 0x0
Async Tx Stats addr : 0x0   Echo Tx Stats addr : 0x0
Async Rx Stats addr : 0x0   Echo Rx Stats addr : 0x0

Router# show bfd session interface bundle-ether 123.1 detail
Fri May 28 14:04:46.893 UTC
```

```
I/f: Bundle-Ether123.1, Location: 0/0/CPU0
Dest: 34.34.34.2
Src: 34.34.34.1
 State: UP for 0d:0h:5m:18s, number of times UP: 1
 Session type: SW/V4/SH/BL
Received parameters:
 Version: 1, desired tx interval: 100 ms, required rx interval: 100 ms
 Required echo rx interval: 0 ms, multiplier: 3, diag: None
 My discr: 984, your discr: 124, state UP, D/F/P/C/A: 0/0/0/1/0
Transmitted parameters:
 Version: 1, desired tx interval: 100 ms, required rx interval: 100 ms
 Required echo rx interval: 0 ms, multiplier: 3, diag: None
 My discr: 124, your discr: 984, state UP, D/F/P/C/A: 0/1/0/1/0
Timer Values:
 Local negotiated async tx interval: 100 ms
 Remote negotiated async tx interval: 100 ms
 Desired echo tx interval: 0 s, local negotiated echo tx interval: 0 ms
 Echo detection time: 0 ms(0 ms*3), async detection time: 300 ms(100 ms*3)
Label:
 Internal label: 24000/0x5dc0
Local Stats:
 Intervals between async packets:
   Tx: Number of intervals=3, min=103 ms, max=19 s, avg=7023 ms
       Last packet transmitted 318 s ago
   Rx: Number of intervals=15, min=1 ms, max=1704 ms, avg=1315 ms
       Last packet received 318 s ago
 Intervals between echo packets:
   Tx: Number of intervals=0, min=0 s, max=0 s, avg=0 s
       Last packet transmitted 0 s ago
   Rx: Number of intervals=0, min=0 s, max=0 s, avg=0 s
       Last packet received 0 s ago
 Latency of echo packets (time between tx and rx):
   Number of packets: 0, min=0 ms, max=0 ms, avg=0 ms
MP download state: BFD_MP_DOWNLOAD_ACK
State change time: May 28 13:59:07.124
Session owner information:
                          Desired               Adjusted
  Client               Interval   Multiplier Interval   Multiplier
  -------------------- ---------- ---------- ---------- ----------
  ipv4_static          100 ms     3          100 ms     3

H/W Offload Info:
 H/W Offload capability : Y, Hosted NPU     : 0/0/CPU0
 Async Offloaded        : Y, Echo Offloaded : N
 Async rx/tx            : 16/4

Platform Info:
NPU ID: 0
Async RTC ID        : 1          Echo RTC ID        : 0
Async Feature Mask  : 0x0        Echo Feature Mask  : 0x0
Async Session ID    : 0x7c       Echo Session ID    : 0x0
Async Tx Key        : 0x7c  Echo Tx Key        : 0x0
Async Tx Stats addr : 0x0   Echo Tx Stats addr : 0x0
Async Rx Stats addr : 0x0   Echo Rx Stats addr : 0x0
```

BFD CPU Offload Support for IPv6

Table 8: Feature History Table

Feature Name	Release Information	Feature Description
BFD CPU offload support for IPv6	Release 26.1.1	Introduced in this release on: NCS 5500 fixed port routers This functionality is now supported on all Cisco NCS 5500 router variants.
BFD CPU offload support for IPv6	Release 24.4.1	You can now enable CPU offloading for IPv6 BFD sessions, allowing the CPU to handle packet transmission and reception directly. This feature provides you the flexibility to choose between hardware-offloaded and CPU-offloaded IPv6 BFD sessions based on your requirements. This feature introduces these changes: **CLI:** • **hw-module profile bfd offload disable-v6**

The BFD CPU offload support for IPv6 feature enables the offload of a BFD session to the CPU, in an IPv6 network.

You can enable CPU offload for IPv6 BFD sessions by using the command **hw-module profile bfd offload disable-v6** and then restarting the router. When CPU offload functionality is enabled, the CPU directly handles BFD sessions, managing packet transmission and reception without offloading the sessions to hardware. If you do not enable CPU offload, BFD sessions are offloaded to the hardware by default.

Benefits of BFD CPU Offload Support for IPv6

The BFD CPU Offload Support for IPv6 feature provides the following benefit:

- **Flexibility**: Supports both hardware and CPU offloaded sessions, providing you with the flexibility to choose between hardware-offloaded and CPU-offloaded IPv6 BFD sessions based on your requirements.

Note IPv6 BFD sessions can be either hardware-offloaded or CPU-offloaded, but both types cannot exist simultaneously.

Limitations for BFD CPU offload support for IPv6

Review these limitations before you use BFD CPU offload support for IPv6.

- In Release 24.4.1, the feature supported only select variants of NCS 5500 fixed port routers and NCS 540 routers, where IPv6 BFD sessions were hosted on the ARM processor. From Release 26.1.1, Cisco extends support to all NCS 5500 fixed port routers, as well as the NCS 540 and NCS 560 routers.

- In Release 24.4.1, only BFD over physical or VLAN interfaces was supported in CPU mode. From Release 26.1.1, BFD over Bundle (BoB), BFD over logical bundle (BLB), Bridge-Group Virtual Interface (BVI), and multihop (MH) sessions are supported.
- Scale limits must be managed to ensure that the rate does not exceed 640 packets per second (PPS). You can check the actual PPS using the **show bfd summary** command.
- For single-path sessions, the minimum-interval value must be greater than or equal to 100 milliseconds.
- For multi-path sessions, including BLB, BVI, and MH, the minimum-interval value must be greater than or equal to 300 milliseconds.
- The maximum supported scale is 64 IPv6 BFD sessions.
- The BFD agent process is responsible for handling packets. If it crashes or restarts, it can cause BFD sessions to flap.
- When you configure **hw-module profile offload** to prioritize route download or performance-monitoring hardware offload along with Bsync, the router does not support BFD CPU offload sessions.

Configure BFD CPU Offload Support for IPv6

Follow these steps to enable CPU offload for BFD IPv6 sessions:

Procedure

Step 1 Enable CPU Offload.

Example:

```
Router# configure
Router(config)# hw-module profile bfd offload disable-v6
```

Note
Restart the router for the **hw-module** command configuration to take effect.

Step 2 Verify if CPU offload is enabled by using the **show bfd ipv6 session** command.

Example:

```
Router# show bfd ipv6 session
Interface           Dest Addr
                                        Local det time(int*mult)      State
H/W                 NPU             Echo              Async
------------------- --------------- ----------------- ----------------- ----------
Te0/0/0/0.501       2001:DB8::1:2
No                  n/a             0s(0s*0)          300ms(100ms*3)    UP
Te0/0/0/0.502       2001:DB8::2:2
No                  n/a             0s(0s*0)          300ms(100ms*3)    UP
Te0/0/0/0.503       2001:DB8::3:2
No                  n/a             0s(0s*0)          300ms(100ms*3)    UP
Te0/0/0/0.504       2001:DB8::4:2
No                  n/a             0s(0s*0)          300ms(100ms*3)    UP
Te0/0/0/0.505       2001:DB8::5:2
No                  n/a             0s(0s*0)          300ms(100ms*3)    UP
Te0/0/0/0.506       2001:DB8::6:2
No                  n/a             0s(0s*0)          300ms(100ms*3)    UP
```

```
Te0/0/0/0.507       2001:DB8::7:2
No                  n/a             0s(0s*0)          300ms(100ms*3)    UP
Te0/0/0/0.508       2001:DB8::8:2
No                  n/a             0s(0s*0)          300ms(100ms*3)    UP
Te0/0/0/0.509       2001:DB8::9:2
No                  n/a             0s(0s*0)          300ms(100ms*3)    UP
```

In this sample output, **No** indicates that CPU offload is enabled and hardware offload is disabled.

BFD Object Tracking

Object Tracking is enhanced to support BFD to track the reachability of remote IP addresses.This will enable complete detection and HSRP switch over to happen within a time of less than one second as BFD can perform the detection in the order of few milliseconds

Configuring BFD Object Tracking:

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters mode.
Step 2	**track** *track-name* **Example:** `RP/0/RP0/CPU0:router(config)# track track1`	Enters track configuration mode. • *track-name*—Specifies a name for the object to be tracked.
Step 3	**type bfdrtr rate** *tx-rate* **Example:** `RP/0/RP0/CPU0:router(config-track)# type bfdrtr rate 4`	tx_rate - time in msec at which the BFD should probe the remote entity
Step 4	**debounce***debounce* **Example:** `RP/0/RP0/CPU0:router(config-if)# debounce 10`	debounce - count of consecutive BFD probes whose status should match before BFD notifies OT
Step 5	**interface** *if-name* **Example:**	if_name - interface name on the source to be used by BFD to check the remote BFD status.

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router(config-track-line-prot)# interface GigabitEthernet0/0/0/4`	
Step 6	**destaddress** *dest_addr* **Example:** `RP/0/RP0/CPU0:router(config-if)#destaddress 1.2.3.4`	dest_addr - IPV4 address of the remote BFD entity being tracked.
Step 7	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

BFD Dampening

Table 9: Feature History Table

Feature Name	Release	Description
IPv6 BFD dampening on Cisco 5700 routers	Release 25.1.1	Introduced in this release on: NCS 5700 fixed port routers; NCS 5700 line cards [Mode: Compatibility; Native] You can now prevent excessive network instability and connectivity fluctuations by enabling BFD dampening on IPv6 sessions. With extending support for IPv6 in addition to IPv4, notifications on unnecessary routing updates and traffic disruptions during link flapping are minimized. This mechanism helps in overall network stability and improves router performance.

Bidirectional Forwarding Detection (BFD) is a mechanism used by routing protocols to quickly realize and communicate the reachability failures to their neighbors. When BFD detects a reachability status change of a client, its neighbors are notified immediately. Sometimes it might be critical to minimize changes in routing tables so as not to impact convergence, in case of a micro failure. An unstable link that flaps excessively can cause other devices in the network to consume substantial processing resources, and that can cause routing protocols to lose synchronization with the state of the flapping link.

The BFD Dampening feature introduces a configurable exponential delay mechanism. This mechanism is designed to suppress the excessive effect of remote node reachability events flapping with BFD. The BFD Dampening feature allows the network operator to automatically dampen a given BFD session to prevent excessive notification to BFD clients, thus preventing unnecessary instability in the network. Dampening the notification to a BFD client suppresses BFD notification until the time the session under monitoring stops flapping and becomes stable.

Configuring the BFD Dampening feature, especially on a high-speed interface with routing clients, improves convergence time and stability throughout the network. BFD dampening can be applied to all types of BFD sessions, including IPv4/single-hop, Multiprotocol Label Switching-Transport Profile (MPLS-TP), and Pseudo Wire (PW) Virtual Circuit Connection Verification (VCCV).

BFD Session Dampening

You can configure the BFD Dampening feature at the BFD template level (single-hop template). Dampening is applied to all the sessions that use the BFD template. If you choose not to have a session to be dampened, you should use a new BFD template without dampening for a new session.

Configuration Verification

```
/* Verify if the BFD session is up, and the timers are configured. */
RP/0//CPU0:router# show bfd session
Thu Jan 4 03:07:15.529 UTC
Interface Dest Addr Local det time(int*mult) State Echo Async H/W NPU
-------------- ---------- ------------------------ -------- ---- ----- ---
----
Te0/0/0/16.1 172.16.0.1 0s(0s*0) 200ms(100ms*2) UP Yes
```