Implementing BFD

BFD Overview

Table 1: Feature History Table

Feature Name	Release Information	Feature Description
Multihop Bidirectional Forwarding Detection on IPv4 and IPv6 Non-DefaultVRFs	Release 7.10.1	Introduced in this release on: NCS 5500 fixed port routers; NCS 5700 fixedport routers; NCS 5500 modular routers (NCS 5500 line cards; NCS 5700 line cards [Mode: Compatibility; Native]) (select variants only*) You can now improve the reliability of your network by providing earlynotification of failures. This is made possible due to extension of Multihop Bidirectional Forwarding Detection (BFD) for IPv4 and IPv6 non-default VRFs to process BFD packets encapsulated with MPLS label for a particular VRF. *NCS 5700 line cards in compatibility and native modes support this feature for IPv4 non-default VRFs. Other variants listed here already support this feature for IPv4 non-default VRFs.

Multihop BFD for IPv4 Non-default VRF	Release 7.7.1	BFD provides fast forwarding path failure detection between two routingdevices that are connected by a network link. BFD Multihop enables you to detect connectivity between routers that span multiple network hops and follow unpredictable paths. Prior to this release, BFD Multihop was supported on default VRFs only. This feature provides you the flexibility to extend BFD Multihop for IPv4 non-defaultVRFs.
BFD Support for VRRP	Release 7.5.1	This feature is now supported on routers that have the Cisco NC57 line cards installed and operate in native and compatible modes. This feature introduces BFD support over VRRP interfaces.
IPv4 BFD Multihop over MPLS Core and Segment Routing	Release 7.5.1	Thisfeature is now supported on routers that have the Cisco NC57 line cards that are installed and operate in native and compatible modes. This feature enables you to configure IPv4 Multihop BFD on MPLS LDP and Segment Routing.
Coexistence of BFD Over Bundle and BFD over Logical Bundle	Release 7.4.1	Thisfeature provides the benefits of both BFD over bundle (BOB) and BFD over logical bundle (BLB). This feature enables you to configure both BOB and BLB over physical bundle interfaces and subinterfaces. BOB functionality allows you to detect failures in physical bundle interfaces. BLB functionality allows you to detect failures in the client protocols configured on the subinterfaces.
BFD over Logical Bundle	Release 7.4.1	This feature is now supported on routers that have Cisco NC57 line cards installed and operate in compatibility mode. The BLB feature implements and deploys BFD over bundle VLAN interfaces. The advantage of BFD, low-overhead and short-duration detection of path failures between routers, is extended to bundle VLAN interfaces.

IPv4 and IPv6 Multihop BFD Support	Release 7.4.1	BFD IPv6 Multihop feature enables IPv6 Multihop BFD sessions whereBFD neighbors can be multiple hops away, either physically or logically. It removes the restriction of a single path IPv6 BFD session, where theBFD neighbor is always one hop away, and the BFD Agent in the line card always receives or transmits BFD packets over a local interfaceon the same line card. Thus, the advantage of BFD, low-overhead and short-duration detectionof path failures between (IPv6) routers, is extended to a multihop scenario. Thisfeature is also supported on routers that have Cisco NC57 line cards installed and operate in native and compatibility modes. IPv4 Multihop BFD is a BFD session between two nodes, such as a PEand CE node, or between routers that are several TTL hops away. This feature provides subsecond forwarding failure detection for a destination more than one hop, and up to 255 hops away. Thus, the advantage of BFD, low-overhead, and short-duration detectionof path failures between routers, is extended to a multihop scenario.
BFD Over BVI	Release 7.4.1Release 7.3.1Release 7.5.1	This feature is now supported on routers that have Cisco NC57 line cards installed and operate in native and compatibility modes. This feature is also supported on routers that have Cisco NCS550x and Cisco NCS55Ax line cards installed and operate in native and compatibility modes. BFD over IRB, using a BVI, is a multipath single-hop session. In a BFD multipath session, BFD can be applied over virtual interfaces or between interfaces that are multihops away. The advantage of BFD, low-overhead and short-duration detection of path failures between routers, is extended to an IRB deployment scenario. 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 inform the same to routing protocols. Starting from Cisco IOS XR Release 7.1.1, BFD over BVI is supported on fixed NCS5500 platforms.

BFD for BoB with IPv4 Unnumbered	Release 7.3.1	This feature is now supported on routers that have Cisco NC57 line cards installed and operate in the native mode.
BFDv6 - HW Offload and IPv6 BFD/BoB (Bundle over Bundle)	Release 6.6.1	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.

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.
- Starting from IOS XR Release 6.3.2, BFD dampening for IPv4 is supported.
- Starting from IOS XR Release 6.3.2, BFD multihop over an IP core is supported.

- BFD is only supported in IP core. It cannot coexist with Label distribution Protocol, or Segment Routing, or Traffic Engineering in the core.
- BFD over Bundle (BoB) over IPv6 is not supported with dyamically configured link-local address. It must be statically configured.
- 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.

BFD Packet Injection

BFD enables egress packet processing injection which results in egress feature processing and ACL processing. The manner in which the egress ACL processes the locally generated BFD packets is different from how the routing protocol processes them. Routing protocols use TX inject to inject packets from the IOS-XR software stack where most locally-generated packets are not subject to egress ACL.

The router injects the source IP address 0.0.0.0 with tunnel encapsulation. The BFD programmable editor overwrites the source IP address with the actual IP address using tunnel encapsulation in the egress transmit packet processor (ETPP). However, as this occurs after the egress IPv4 ACL processing, the egress ACL processing is aware of the SRC IP address of 0.0.0.0. This could result in unexpected matches. A bearing condition monitoring (BCM) limitation causes the aforesaid unexpected mismatches because operations, administration, maintenance, and provisioning (OAMP) supports only 16 different SRC IP addresses. To overcome this limitation, you need to permit BFD packets specifically.

To know more about protecting locally originated BFD packets, see Protecting Locally Originated BFD Packets.

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 feature is supported only in IETF mode.

BFD Timers

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 BFD HW (OAMP) supports 8 timer profiles, each profile is assigned to unique bfd minimum-interval value. Out of these, two profiles are reserved and another profile is reserved if BFD over Bundle (BoB) is configured.

So, 6 profiles are available without BoB config and 5 profiles with BoB.

Table 2: 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
BGP Multi Hop	50ms	3	Any
BFD Over BVI	50ms	3	Any

Table 3: 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)

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
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

BFD Session Types

There are two types of BFD sessions:

- Single Path Sessions
- Multipath Sessions

BFD Singlepath Sessions

SH Sessions over Physical or Physical Sub Interface

Enabling BFD on a Static Route

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

```
RP/0/RP0/CPU0:router(config)# configure

/* Enter static route configuration mode, and configure static routing. */
Router(config)# router static

/* Enter address family configuration mode. */
Router(config-static)# address-family ipv4 unicast

/* Specify an unicast destination address and next-hop IPv4 address.
Enable BFD fast-detection on the specified IPv4 unicast destination address */
Router(config-static)# 192.168.2.2/32 HundredGigE0/0/0/2 192.168.6.2 bfd fast-detect
minimum-interval 4 multiplier 3
```

Configuration example of a BFD single-hop scenario:

```
!
router
```

```
 address-family ipv4 unicast
  10.2.150.193/32 BVI1252 10.2.153.1 bfd fast-detect
```

Note The next-hop IPv4 address (10.2.153.1) is determined from the IP address of the directly connected interface.

Configuration example of a BFD multi-hop scenario:

```
!
router static
address-family ipv4 unicast
 10.10.10.0/24 10.20.20.20 bfd fast-detect multihop 10.30.30.30
```

Note In a BFD multi-hop scenario:

- The next-hop (10.20.20.20) must be reachable.
- The next-hop must be resolved through the connected interface if the interface is provided as part of the configuration. If the interface is provided, the 'multihop' option will not be available.

Configuration example of BFD on directly-connected host routes (/32 or /128):

```
!
router static
address-family ipv4 unicast
 10.102.134.140/32 TenGigE0/0/0/1 10.102.134.140 bfd fast-detect
```

Note In this BFD single-hop specific scenario:

- The prefix and next-hop are the same (this makes nexthop a static nexthop). The prefix length is either 32 or 128.
- Next-hop (10.102.134.140) is resolved using static routing.

Running Configuration

```
router static
 address-family ipv4 unicast
  192.168.2.2/32 HundredGigE0/0/0/2 192.168.6.2 bfd fast-detect minimum-interval 4 multiplier
 3
 !
!
```

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.

SUMMARY STEPS

1. **configure**

2. **router ospf** *process-name*
3. **area** *area-id*
4. **interface** *type interface-path-id*
5. **bfd fast-detect**
6. **bfd minimum-interval** *milliseconds*
7. **bfd multiplier** *multiplier*
8. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

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

	Command or Action	Purpose
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.

Enable BFD for IS-IS on an Interface

Perform the following steps to configure BFD for Integrated Intermediate System-to-Intermediate System (IS-IS) on an interface.

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

```
Router# configure

/* Enter IS-IS configuration mode to configure the IS-IS routing process. */
Router(config)# router isis 65444

/* Set the system type (area or backbone router). Each IS-IS instance can support either a
 single Level 1 or Level 2, or one of each.*/
Router(config-isis)#is-type level-2-only

/* Specify a NET for each routing instance if you are configuring multi-instance IS-IS.*/
Router(config-isis)# net 49.0001.0840.3803.4088.00

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

/* Set the BFD minimum interval. */
Router(config-isis-if)# bfd minimum-interval 6500

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

/* Enable BFD to detect failures in the path between adjacent forwarding engines. Only IPv4
 is supported.*/
Router(config-isis-if)# bfd fast-detect ipv4

/* Specify the IPv4 address family and enters router address family configuration mode. */
```

```
Router(config-isis-if)# address-family ipv4 unicast

!
```

Running Configuration

```
configure
  router isis 65444
   is-type level-2-only
   net 49.0001.0840.3803.4088.00
    interface gigabitEthernet 0/3/0/1
     bfd minimum-interval 6500
     bfd multiplier 7
     bfd fast-detect ipv4
     address-family ipv4 unicast
  !
!
```

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.

SUMMARY STEPS

1. **configure**
2. **router bgp** *autonomous-system-number*
3. **neighbor** *ip-address*
4. **remote-as** *autonomous-system-number*
5. **bfd fast-detect**
6. **bfd minimum-interval** *milliseconds*
7. **bfd multiplier** *multiplier*
8. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration 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.

	Command or Action	Purpose
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 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.

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.

BoB implementation is a standard-based fast failure detection of link aggregation (LAG) member links that is interoperable between different platforms. NCS 5500 platforms only support the IETF mode. When BoB is configured, the nodes should be directly connected.

Restrictions for BFD over Bundle

The following are the restrictions in using BoB feature:

- 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

Configure BFD IPv4 Over Bundle

Perform the following tasks to configure the BOB feature for IPv4:

- Enable BFD sessions on bundle members
- Specify the BFD destination address on a bundle
- Configure BFD packet transmission intervals and failure detection times on a bundle
- Configure BFD over bundles IETF mode support on a per-bundle basis

```
/* Enable BFD sessions on bundle members */
Router(config)# interface Bundle-Ether 1
Router(config-if)# bfd address-family ipv4 fast-detect
Router(config-if)# bfd mode ietf

/* Specify the BFD destination address on a bundle */
Router(config)# interface Bundle-Ether 1
Router(config-if)# bfd address-family ipv4 destination 10.20.20.1

/* Configure BFD packet transmission intervals and failure detection times on a bundle */
Router(config)# interface Bundle-Ether 1
Router(config-if)# bfd address-family ipv4 minimum-interval 2000
Router(config-if)# bfd address-family ipv4 multiplier 3

/* Configure BFD over bundles IETF mode support on a per-bundle basis */
Router(config)# interface Bundle-Ether 1
Router(config-if)# bfd mode ietf
```

Configure BFD IPv6 Over Bundle

Configure BFD over Bundle(BOB) for hardware offload to configure the BOB feature 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
```

Enabling BFD Sessions on Bundle Members

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

SUMMARY STEPS

1. **configure**
2. **interface Bundle-Ether** *bundle-id*
3. **bfd address-family ipv4 fast-detect**
4. **bfd mode ietf**
5. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration 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.

	Command or Action	Purpose
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:

SUMMARY STEPS

1. **configure**
2. **interface Bundle-Ether** *bundle-id*
3. **bfd address-family ipv4 destination** *ip-address*
4. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface Bundle-Ether** *bundle-id* **Example:**	Enters interface configuration mode for the specified bundle ID.

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router(config)# interface Bundle-Ether 1`	
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 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:

SUMMARY STEPS

1. **configure**
2. **interface Bundle-Ether** *bundle-id*
3. **bfd address-family ipv4 minimum-interval** *milliseconds*
4. **bfd address-family ipv4 multiplier** *multiplier*
5. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure**	Enters global configuration mode.

	Command or Action	Purpose
	Example: `RP/0/RP0/CPU0:router# configure`	
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.
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.

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:

SUMMARY STEPS

1. **configure**

2. **interface Bundle-Ether** *bundle-id*
3. **bfd mode ietf**
4. **bfd address-family ipv4 fast-detect**
5. Use the **commit** or **end** command.
6. **show bundle bundle-ether** *bundle-id*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration 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.

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:

SUMMARY STEPS

1. **configure**
2. **interface Bundle-Ether** *bundle-id*
3. **bundle minimum-active bandwidth** *kbps*
4. **bundle minimum-active links** *links*
5. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration 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:**	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**

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router(config-if)# bundle minimum-active links 2`	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.

BoB Configuration for IPv4 and IPv6

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.

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 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
```

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.

The QoS policy must classify ingress and egress BFD packets into priority level 1 or priority level 2 queue. Similar approach applies to BFD sessions on BVI and "BFD Over VLAN Over Bundle" (that is, BLB).

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

Bidirectional Forwarding Detection over Logical Bundle

BFD over Logical Bundle

The BLB feature implements and deploys BFD over bundle interfaces based on RFC 5880. In the BLB, the bundle interface is a single interface, whereas, in BOB, BFD is implemented per member link. BLB is a multipath (MP) single-hop session so at least one line card must be configured under the **bfd multipath include location** *location* command before a BLB session can come up. Because BFD treats the bundle as a single big interface, BLB requires limited knowledge of the bundle interfaces on which the sessions run. BLB requires information about IP addresses, interface types, and caps on bundle interfaces only. Information such as a list of bundle members, member states, and configured minimum or maximum bundle managers are not required. In the case of BLB, the BFD client is not the bundle manager, but protocols running over the bundle manager. BLB is supported on IPv4 address, IPv6 global address, and IPv6 link-local address.

Configuration Example

1. Configure multipath capability under BFD
2. Create VLAN subinterface under bundle interface
3. Enable BFD on a static route
4. Enable BFD on IS-IS
5. Enable BFD for OSPF on an interface
6. Enable BFD on a BGP neighbor

```
/* Configure a specific LC (or LCs) to host BLB sessions. The BLB sessions and bundle member
 links need not be configured on the same LC. For example, you can configure the bundle
member links on LC slot 2 and slot 3 while you configure BLB sessions to be hosted on LC
slot 5. */
Router(config)# bfd
Router(config-bfd)# multipath include location 0/6/CPU0
Router(config-bfd)# multipath include location 0/2/CPU0

/* Create VLAN subinterface under bundle interface */
Router# configure
Router(config)# interface Bundle-Ether 2.1
Router(config-if)# ipv4 address 10.1.1.1 255.255.255.0
Router(config-if)# encapsulation dot1q 1
Router(config-if)# end

/* Enable BFD on a static route. */
```

```
Router# configure
Router(config)# router static
Router(config-static)# address-family ipv4 unicast
Router(config-static)# 10.158.3.13/32 10.1.1.2 bfd fast-detect minimum-interval 300 multiplier
 3

/* Enable BFD on IS-IS. */
Router# configure
Router(config)# router isis cybi
Router(config-isis)# interface Bundle-Ether 2.1
Router(config-isis-if)# bfd minimum-interval 300
Router(config-isis-if)# bfd multiplier 3
Router(config-isis-if)# bfd fast-detect ipv4
Router(config-isis-if)# address-family ipv4 unicast
Router(config-isis-if-af)# end

/* Enable BFD for OSPF on an interface. */
Router# configure
Router(config)# router ospf cybi
Router(config-ospf)# area 0
Router(config-ospf)# interface Bundle-Ether 2.1
Router(config-ospf-if)# bfd fast-detect
Router(config-ospf-if)# bfd minimum-interval 300
Router(config-ospf-if)# bfd multiplier 3
Router(config-ospf-if)# end

/* Enable BFD on a BGP neighbor.*/
Router# configure
Router(config)# router bgp 4787
Router(config-bgp)# neighbor 10.158.1.1
Router(config-bgp-nbr)# remote-as 4787
Router(config-bgp-nbr)# update-source Bundle-Ether 2.1
Router(config-bgp-nbr)# bfd fast-detect
Router(config-bgp-nbr)# bfd minimum-interval 300
Router(config-bgp-nbr)# bfd multiplier 3
Router(config-bgp-nbr)# address-family ipv4 unicast
Router(config-bgp-nbr-af)# exit
Router(config-bgp-nbr)# commit
```

Configuration Verification

Configuration verification for OSPF:

```
Router# show bfd all session
Interface           Dest Addr           Local det time(int*mult)      State
                                    Echo             Async   H/W   NPU
------------------- --------------- ---------------- ---------------- ----------
BE2.1                 10.1.1.2      0s(0s*0)         300ms(100ms*3)     UP
                                                             Yes   0/6/CPU0
```

Configuration verification for IS-IS:

```
Router# show bfd all session
Interface           Dest Addr           Local det time(int*mult)      State
                                    Echo             Async   H/W   NPU
------------------- --------------- ---------------- ---------------- ----------
BE2.1              10.1.1.2       0s(0s*0)         900ms(300ms*3)     UP
                                                             Yes   0/6/CPU0
```

Configuration verification for BGP:

```
Router# show bfd all session
Interface           Dest Addr           Local det time(int*mult)      State
                                    Echo             Async   H/W   NPU
```

```
------------------- --------------- ---------------- ---------------- ----------
BE2.1                 10.158.1.1      0s(0s*0)         900ms(300ms*3)     UP
                                                               Yes   0/6/CPU0
```

Configuration verification for Static:

```
Router# show bfd all session
Interface           Dest Addr           Local det time(int*mult)      State
                                    Echo             Async   H/W   NPU
------------------- --------------- ---------------- ---------------- ----------
BE2.1                 10.1.1.2      0s(0s*0)         900ms(300ms*3)     UP
                                                               Yes   0/6/CPU0
```

BFD over BVI

Table 4: Feature History Table

Feature Name	Release Information	Feature Description
BFD Over BVI	Release 7.4.1	This feature is now supported on routers that have Cisco NC57 line cards installed and operate in native and compatibility modes. This feature is also supported on routers that have Cisco NCS550x and Cisco NCS55Ax line cards installed and operate in native and compatibility modes. BFD over IRB, using a BVI, is a multipath single-hop session. In a BFD multipath session, BFD can be applied over virtual interfaces or between interfaces that are multihops away. The advantage of BFD, low-overhead and short-duration detection of path failures between routers, is extended to an IRB deployment scenario.

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.

In Releases 24.2.1 to 24.4.1, if the configured number of IPv4 BFD sessions exceed any of the following thresholds, the limitation applies.

- 500 BFD Single-path IPv4 sessions
- 250 BFD Multi-path IPv4 sessions
- 500 BFD IPv4 sessions (Single-path and Multi-path together)
- The BFD timers must be configured to a minimum of 300 milliseconds. This measure is crucial to maintain system stability.

For more information on IRB feature, see *Integrated Routing and Bridging* Chapter in *Interface and Hardware Component Configuration Guide for Cisco NCS 5500 Series Routers*.

Configure BFD Over BVI

These steps allow you to configure BFD Over BVI:

1. Configure the line cards to allow hosting of Multipath BFD sessions
2. Configure a BVI interface and assign an IP address
3. Configure the Layer 2 AC interface
4. Configure the BFD client (OSPF, IS-IS, BGP, Static, and so on)
5. Configure BFD on BVI
6. Configure the line cards to allow hosting of Multipath BFD sessions
7. Configure BVI on the peer nodes

Configuration Example

```
/* Configure the line cards to allow hosting of Multipath BFD sessions */
Router# configure
Router(config)# bfd multipath include location 0/5/CPU0

/* Configure a BVI interface and assign an IP address */
Router(config)# interface bvi 1
Router(config-if)# ipv4 address 192.168.1.1 255.255.255.0
Router(config-if)# exit

/* Configure the Layer 2 AC interface */
Router(config)# interface HundredGigE 0/0/1/3
Router(config-if)# l2transport
Router(config-if)# exit

/* Configure BVI on the peer nodes */
Router:ios(config)# l2vpn
Router:ios(config-l2vpn)# bridge group bg1
Router:ios(config-l2vpn-bg)# bridge-domain bfd
Router:ios(config-l2vpn-bg-bd)# interface HundredGigE 0/0/1/3
Router:ios(config-l2vpn-bg-bd-ac)# exit
Router:ios(config-l2vpn-bg-bd)# routed interface BVI1

/* Configure OSPF as the routing protocol */
Router:DUT2(config)# router ospf 100
```

```
Router:DUT2(config-ospf)# router-id 192.168.1.1
Router:DUT2(config-ospf)# area 0
Router:DUT2(config-ospf-ar)# interface BVI1
Router:DUT2(config-ospf-ar-if)# exit
Router:DUT2(config-ospf-ar)#

/* Configure BFD on BVI */
Router(config)# interface bvi1
Router(config-if)# bfd minimum-interval 100
Router(config-if)# bfd fast-detect
Router(config-if)# bfd multiplier 3

/* Configure the line cards to allow hosting of Multipath BFD sessions */
Router# configure
Router(config)# bfd multipath include location 0/5/CPU0
```

Running Configuration

```
interface BVI1
 ipv4 address 192.168.1.1 255.255.255.0
!
interface HundredGigE0/0/1/3
 l2transport
 !
!
router ospf 100
 router-id 192.168.1.1
 area 0
  interface Loopback100
  !
  interface BVI1
   bfd minimum-interval 100
   bfd fast-detect
   bfd multiplier 3
  !
 !
bfd multipath include location 0/5/CPU0
!
l2vpn
 bridge group 1
  bridge-domain 1
   interface HundredGigE0/0/1/3
   !
   routed interface BVI1
   !
```

Configuration Verification

```
Router# show bfd all session
Interface           Dest Addr           Local det time(int*mult)      State
                                    Echo             Async   H/W   NPU
------------------- --------------- ---------------- ---------------- ----------
BV1                 192.168.1.2      0s(0s*0)         150ms(50ms*3)    UP
                                                             Yes   0/1/CPU0
```

IPv4 and IPv6 Multihop BFD Support

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.

The IPv4 BFD Multihop feature enables you to configure IPv4 Multihop BFD on MPLS LDP and Segment Routing in the core.

For information on configuring MPLS LDP, see the MPLS Configuration Guide for Cisco NCS 5500 Series Routers.

For information on configuring Segment Routing, see the Segment Routing Configuration Guide for Cisco NCS 5500 Series Routers.

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
```

IPv6 Multihop BFD

Table 5: Feature History Table

Feature Name	Release Information	Feature Description
IPv6 Multihop BFD support	Release 7.4.1	BFD IPv6 Multihop feature enables IPv6 Multihop BFD sessions where BFD neighbors can be multiple hops away, either physically or logically. It removes the restriction of a single path IPv6 BFD session, where the BFD neighbor is always one hop away, and the BFD Agent in the line card always receives or transmits BFD packets over a local interface on the same line card. Thus, the advantage of BFD, low-overhead and short-duration detection of path failures between (IPv6) routers, is extended to a multihop scenario. This feature is also supported on routers that have Cisco NC57 line cards installed and operate in native and compatibility modes.

In a Multihop Bidirectional Forwarding Detection (BFD) scenario, more than one path is available for a node to reach its IPv6 BFD neighbor. BFD packets are received on a line card that may or may not host the respective BFD session. The BFD Agent in one line card may need to transmit BFD packets out of an egress interface on a different line card. The BFD switching mechanism for IPv6 Multihop link is used when the BFD packets are transmitted from one end point node to the other. The BFD punting mechanism is employed when BFD packets are received at the remote end point node.

Configuration Example

/* BFD IPv6 Multihop for an eBGP Neighbor */

```
Router# configure
```

Specify a line card to host a BFD multihop session.

```
Router(config)# bfd multipath include location 0/7/CPU0
```

Enable multihop peering with the eBGP neighbor, and enable BFD fast detection.

```
Router(config)# router bgp 65001
Router(config-bgp)#neighbor 21:1:1:1:1:1:1:2 ebgp-multihop 255
Router(config-bgp)#neighbor 21:1:1:1:1:1:1:2 bfd fast-detect
Router(config-bgp)#commit
```

/* BFD IPv6 Multihop for an iBGP Neighbor */

```
Router# configure
```

Specify a line card to host a BFD multihop session.

```
Router(config)# bfd multipath include location 0/0/CPU0
Router(config)# router bgp 65001
```

Specifies an iBGP neighbor and enable BFD fast detection.

```
Router(config-bgp)#neighbor 21:1:1:1:1:1:1:2
Router(config-bgp-nbr)#commit
```

Running Configuration

/* BFD IPv6 Multihop for an eBGP Neighbor */

```
bfd
 multipath include location 0/7/CPU0
!
router bgp 65001
 neighbor 21:1:1:1:1:1:1:2
  bfd fast-detect
  ebgp-multiphop 255
```

/* BFD IPv6 Multihop for an iBGP Neighbor */

```
bfd
 multipath include location 0/0/CPU0
!
router bgp 65001
 neighbor 21:1:1:1:1:1:1:2
  bfd fast-detect
```

Multihop BFD on IPv4 and IPv6 Non-Default VRFs

Overview

Bidirectional Forwarding Detection (BFD) is a network protocol that quickly detects link failures within a network. This is achieved through the periodic transmission of BFD packets between two routers. By configuring a specific time interval, if a receiving node fails to receive a BFD packet within that timeframe, it concludes that a link failure has occurred.

When BFD is configured on a router, the router establishes BFD sessions, exchanges lightweight control packets, monitors packet reception, and rapidly detects link or path failures. The router responds quickly to failures contributing to better network resilience and faster convergence times.

When Multihop BFD is configured on a router, the router accomodates multihop environment. With Multihop BFD, the router establishes BFD sessions with neighbors that are not directly connected. These sessions extend over multiple hops, allowing the router to monitor the health of paths that involve intermediate routers. The router continues to exchange BFD control packets at a specified interval. However, in a multihop scenario, the control packets traverse multiple routers along the path, ensuring that the path's overall health is monitored. In addition to monitoring the the reception of control packets, in a multihop environment, this monitoring accounts for the additional latency introduced by the intermediate routers. The quick response to failures in multihop paths further accelerates convergence times for routes that span intermediate devices.

The router utilizes MPLS label stack encapsulation to include IPv4 or IPv6 packets within BFD sessions. The Multihop BFD for IPv4 or IPv6 non-default VRF feature enables BFD to effectively monitor the status of individual Label Switched Paths (LSPs), enabling rapid detection of link failures as and when they happen.

When you are configuring a Bidirectional Forwarding Detection (BFD) session between a router and a remote peer, which resides within an IPv4 or IPv6 non-default Virtual Routing and Forwarding (VRF) instance and is connected to a core network, the router employs MPLS label stack encapsulation. This encapsulation enables BFD to effectively detect link failures between any two MPLS routers. The MPLS label stack plays a vital role in this process, as it furnishes a distinctive identifier for each Label Switched Path (LSP) within the network. BFD leverages this unique identifier to facilitate efficient link failure detection and monitoring.

The Multihop BFD for IPv4 or IPv6 non-default VRF feature preserves the connectivity across the core network while ensuring accurate data-path programming for packets associated with the non-default VRF. This is accomplished by generating packets that incorporate the required MPLS label stack specific to the targeted VRF. By encapsulating the MPLS label stack, Multihop BFD enables the correct handling of packets within the non-default VRF, thus ensuring seamless connectivity and accurate routing throughout the network infrastructure.

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}
```

Configure Multihop BFD on IPv6 Non-default VRFs

Configure the following steps to configure Multihop BFD on IPv6 nondefault VRF:

```
Router(config)# router bgp 4134
Router(config-bgp)# neighbor 2001:DB8:1::1
Router(config-bgp-nbr)#vrf vrf1
Router(config-bgp-nbr-vrf)# exit
Router(config-bgp-nbr)# rd auto
Router(config-bgp-nbr)#address-family ipv6 unicast
Router(config-bgp-nbr-af)#redistribute connected
Router(config-bgp-nbr-af)# exit
Router(config-bgp)# neighbor 2001:DB8:1::1
Router(config-bgp-nbr)# remote-as 1
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 ipv6 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 4134
  neighbor 2001:DB8:1::1
   vrf vrf1
   !
   rd auto
   address-family ipv6 unicast
   redistribute connected
   !
  neighbor 2001:DB8:1::1
    remote-as 2000
    bfd fast-detect
    bfd multiplier 3
    bfd minimum-interval 50
    ebgp-multihop 255
    update-source loopback 1
    address-family ipv6 unicast
     route-policy pass-all in
     route-policy pass-all out
```

Verification

Verify that the status, configuration, and details of the BFD IPv6 multihop session display per your configuration.

```
Router# show bfd ipv6 multihop session

Src Addr         Dest Addr       VRF Name       Local det time(int*mult)      State
                                                     Echo     Async           H/W      NPU
--------------- -------------  ----------- ------     ------    ------          ------
2001:DB8:1::1   2001:DB8:1::2  vrf1  0s(0s*0) 900ms(300ms*3)   UP  Yes  /0/CPU0
```

Verify that the IPv6 packet has reached the destination.

```
Router# show cef vrf vrf1 ipv6 2001:DB8:1::1

2001:DB8:1::1/28, version 17, internal 0x5000001 0x30 (ptr 0xbc0c1b48) [1], 0x0 (0x0), 0xa08
 (0x9a7d6268)
 Updated Feb  5 21:49:34.252
 Prefix Len 128, traffic index 0, precedence n/a, priority 3
  gateway array (0xbbd6a3c8) reference count 1, flags 0x2038, source rib (7), 0 backups
               [1 type 1 flags 0x48441 (0xbabe1078) ext 0x0 (0x0)]
  LW-LDI[type=0, refc=0, ptr=0x0, sh-ldi=0x0]
  gateway array update type-time 1 Feb  5 21:49:34.252
 LDI Update time Feb  5 21:49:34.252
   via ::ffff:100.100.100.1/128, 501 dependencies, recursive [flags 0x6000]
    path-idx 0 NHID 0x0 [0xba5c9a08 0x0]
    recursion-via-/128
    next hop VRF - 'default', table - 0xe0000000
    next hop ::ffff:100.100.100.1/128 via 16100/0/21
    next hop 2001:DB8:0:ABCD::1 BE10000  labels imposed {ImplNull 24254} --> This indicates
that the IPv6 packet has reached the destination.
```

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 0/5/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 0/5/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 0/5/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 on NCS 5700 Fixed Port Routers	Release 24.2.11	Introduced in this release on: NCS 5700 fixed port routers The feature support is now extended to NCS 5700 fixed port routers.

Feature Name	Release Information	Feature Description
Seamless Bidirectional Forwarding Detection	Release 24.2.1	Introduced in this release on: NCS 5500 fixed port routers; NCS 5500 modular routers (NCS 5500 line cards; NCS 5700 line cards [Mode: Native]). 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

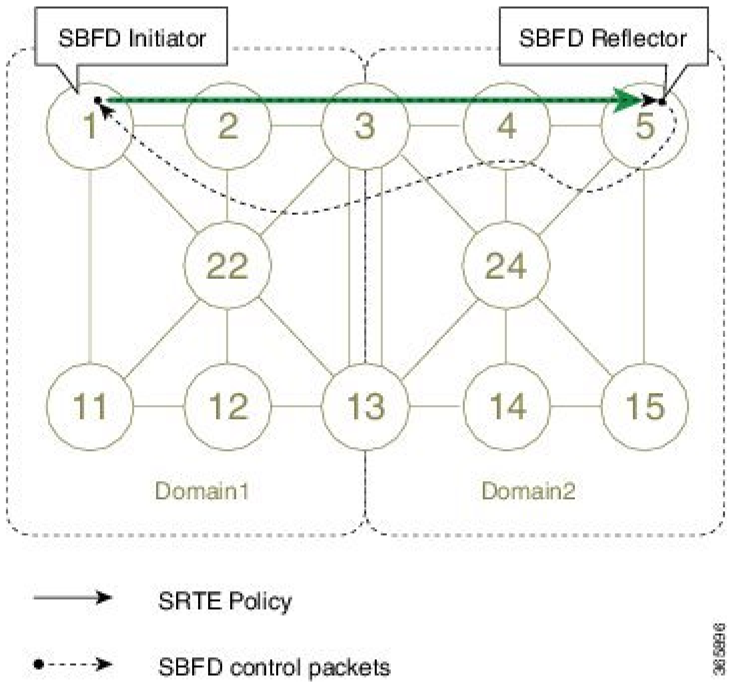
In BFD, each end of the connection maintains a BFD state and transmits packets periodically over a forwarding path. S-BFD is unidirectional, resulting in faster session activation than BFD. The BFD state and client context is maintained on the head-end (initiator) only. The tail-end (reflector) validates 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.

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.

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 0/0/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   0/0/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     : 0/0/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: 0/0/CPU0
Dest: 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     : 0/0/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 Transparency

Bidirectional Forwarding Detection(BFD) protocol is a simple hello mechanism that detects failures in a network in less than one second, depending on the timer value configured.

Both endpoints of a BFD Session periodically send Hello packets to each other. If a number of those packets are not received, the session is considered down. BFD provides fast BFD peer failure detection times independently of all media types, encapsulations, topologies, and routing protocols BGP, IS-IS, and OSPF.

BFD Transparency feature enables you to configure BFD Sessions between customer edge devices connected over an L2VPN network. These BFD sessions are transparent to the core. For example, BFD packets being exchanged between CEs are neither dropped on any router in the core, nor punted on any core device.

In this section, you will learn how to configure BFD Transparency in Ethernet VPN (EVPN) Virtual Private Wire Service (VPWS).

Ethernet VPN Virtual Private Wire Service

EVPN VPWS (Ethernet VPN Virtual Private Wire Service) is a BGP control plane solution for point-to-point services. It implements signaling and encapsulation techniques for establishing an EVPN instance between a pair of provider edge devices.

EVPN VPWS supports both single-homing and multi-homing.

Configuration

The following sections describes the procedure for configuring IP Fast Reroute with Remote LFA.

- Configure L2VPN on the provide edge router
- Configure BFD on the customer edge router

Configure L2VPN on the Provide Edge Router

Configure L2VPN on the provider edge router.

```
/* Enable IS-IS and configure routing level for an area. */
RP/0//CPU0:router# configure
RP/0//CPU0:router(config)# interface tengige 0/1/0/8/2.1
RP/0//CPU0:router(config-subif)# exit
RP/0//CPU0:router(config)# router isis
RP/0//CPU0:router(config-isis)# is-type level-2-only
RP/0//CPU0:router(config-isis)# net 49.1234.2222.2222.2222.00
RP/0//CPU0:router(config-isis)# nsr
RP/0//CPU0:router(config-isis)# nsf cisco
RP/0//CPU0:router(config-isis)# address-family ipv4 unicast
RP/0//CPU0:router(config-isis-af)# metric style wide
RP/0//CPU0:router(config-isis)# end
RP/0//CPU0:router(config)# interface Bundle-Ether 199
RP/0//CPU0:router(config-if)# address-family ipv4 unicast
RP/0//CPU0:router(config-if)# end
RP/0//CPU0:router(config)# interface Loopback 0
```

```
RP/0//CPU0:router(config-if)# end
RP/0//CPU0:router(config-if)# address-family ipv4 unicast
RP/0//CPU0:router(config-if)# exit

/* Configure L2VPN EVPN address family. */
RP/0//CPU0:router(config)# router bgp 100
RP/0//CPU0:router(config-bgp)# bgp router-id 10.10.10.1
RP/0//CPU0:router(config-bgp)# address-family l2vpn evpn
RP/0//CPU0:router(config-bgp)# neighbor 192.0.2.1
RP/0//CPU0:router(config-bgp-nbr)# remote-as 100
RP/0//CPU0:router(config-bgp-nbr)# update-source Loopback 0
RP/0//CPU0:router(config-bgp-nbr)# address-family l2vpn evpn

/* Configure MPLS LDP for the physical core interface. */
RP/0//CPU0:router(config-bgp-nbr-af)# mpls ldp
RP/0//CPU0:router(config-bgp-nbr-af)# exit
RP/0//CPU0:router(config-bgp-nbr)# exit
RP/0//CPU0:router(config-bgp)# exit
RP/0//CPU0:router(config)# interface Bundle-Ether 199
RP/0//CPU0:router(config-if)# exit

/* Configure L2VPN Xconnect. */
RP/0//CPU0:router(config)# l2vpn
RP/0//CPU0:router(config-l2vpn)# router-id 10.10.10.1
RP/0//CPU0:router(config-l2vpn)# xconnect group bfdtr
RP/0//CPU0:router(config-l2vpn-xc)# p2p vpws-ce
RP/0//CPU0:router(config-l2vpn-xc-p2p)# interface TenGigE 0/4/0/2/0.1
RP/0//CPU0:ios(config-l2vpn-xc-p2p)# neighbor evpn evi 100 target 3 source 4
```

Configure BFD on the Customer Edge Router

Configure BFD on the customer edge router.

```
RP/0//CPU0:router# configure
RP/0//CPU0:router(config)#  router bgp 100
RP/0//CPU0:router(config-bgp)# bgp router-id 10.10.10.1
RP/0//CPU0:router(config-bgp)# address-family ipv4 unicast
RP/0//CPU0:router(config-bgp-af)# exit
RP/0//CPU0:router(config-bgp)# neighbor 172.16.0.1
RP/0//CPU0:router(config-bgp)# address-family ipv4 unicast
RP/0//CPU0:router(config-bgp-nbr)# remote-as 100
RP/0//CPU0:router(config-bgp-nbr)# bfd fast-detect
RP/0//CPU0:router(config-bgp-nbr)# bfd multiplier 2
RP/0//CPU0:router(config-bgp-nbr)# bfd minimum-interval 100
RP/0//CPU0:router(config-bgp-nbr)# update-source TenGigE 0/0/0/16.1
RP/0//CPU0:router(config-bgp-nbr)# address-family ipv4 unicast
RP/0//CPU0:router(config-bgp-nbr-af)#
```

Running Configuration

This section shows the BFD Transparency configuration.

```
!
interface TenGigE 0/4/0/2/0.1
  l2transport
router isis 1
   is-type level-2-only
   net 49.0000.1000.0000.0001.00
   nsr
   nsf cisco
   address-family ipv4 unicast
```

```
    metric-style wide
!
  interface Bundle-Ether199
     address-family ipv4 unicast
  interface Loopback0
      address-family ipv4 unicast
router bgp 100
  bgp router-id 10.10.10.1
   address-family l2vpn evpn
    neighbor 192.0.2.1
     remote-as 100
     update-source Loopback 0
    address-family l2vpn evpn
 !
   mpls ldp
   interface Bundle-Ether199
 !
  l2vpn
   router-id 10.10.10.1
    xconnect group bfdtr
   p2p vpws-ce
    interface TenGigE 0/4/0/2/0.1
     neighbor evpn evi 100 target 3 source 4

         router bgp 100
          bgp router-id 10.10.10.1
           address-family ipv4 unicast
           !
           neighbor 172.16.0.1
            address-family ipv4 unicast
            remote-as 100
            bfd fast-detect
            bfd multiplier 2
            bfd minimum-interval 100
            update-source TenGigE0/0/0/16.1
            address-family ipv4 unicast
```

Verification

The show outputs given in the following section display the details of the configuration of the BFD transparency, and the status of their configuration.

```
/* 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

-------------- ---------- -----------------------  --------     ----  -----   ---
----
Te0/0/0/4.1    10.1.1.1   0s(0s*0)                  20ms(10ms*2) UP    Yes     0/RP0/CPU0
                                                     Yes   0/RP0/CPU0


/* Verify if the BFD session is up and check the timer value, numbers of hellos exchanged,
 and information
about last packet. */

RP/0//CPU0:router# show bfd session destination 10.1.1.1 detail
Thu Jan  4 03:09:48.573 UTC
I/f: TenGigE0/0/0/4.1, Location: 0/RP0/CPU0
Dest: 10.1.1.1
```

```
Src:  10.1.1.2
 State: UP for 0d:0h:9m:27s, number of times UP: 1
 Session type: PR/V4/SH
Received parameters:
 Version: 1, desired tx interval: 10 ms, required rx interval: 10 ms
Required echo rx interval: 0 ms, multiplier: 2, diag: None
 My discr: 2147483898, your discr: 2147483899, state UP, D/F/P/C/A: 0/0/0/1/0
Transmitted parameters:
 Version: 1, desired tx interval: 10 ms, required rx interval: 10 ms
 Required echo rx interval: 0 ms, multiplier: 2, diag: None
 My discr: 2147483899, your discr: 2147483898, state UP, D/F/P/C/A: 0/1/0/1/0
Timer Values:
 Local negotiated async tx interval: 10 ms
 Remote negotiated async tx interval: 10 ms
 Desired echo tx interval: 0 s, local negotiated echo tx interval: 0 ms
 Echo detection time: 0 ms(0 ms*2), async detection time: 20 ms(10 ms*2)
Local Stats:
 Intervals between async packets:
   Tx: Number of intervals=100, min=6 ms, max=6573 ms, avg=1506 ms
       Last packet transmitted 186 s ago
   Rx: Number of intervals=100, min=4 ms, max=5 s, avg=575 ms
       Last packet received 184 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
  -------------------- --------------------- ---------------------
  bgp-default          10 ms      2          10 ms      2

H/W Offload Info:
 H/W Offload capability : Y, Hosted NPU     : 0//CPU0
 Async Offloaded        : Y, Echo Offloaded : N
 Async rx/tx            : 344/209

Platform Info:
NPU ID: 0
Async RTC ID        : 1          Echo RTC ID        : 0
Async Feature Mask  : 0x0        Echo Feature Mask  : 0x0
Async Session ID    : 0xfb       Echo Session ID    : 0x0
Async Tx Key        : 0x800000fb  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 the complete history including session state, type, transitions, offload history,
 last down reason if any,
 received and transmitted packets, rx/tx intervals, location, timestamp, and local and
remote descriptors. */

RP/0/RP0/CPU0:router# show bfd session status history destination 10.1.10.1 location
0/RP0/CPU0

Thu Jan  4 03:45:18.768 UTC
I/f: TenGigE0/0/0/4.10, Location: 0//CPU0 table_id:0xe0000000
State: UP, flags:0x80040
Iftype: 0x19, basecaps: 107
Async dest addr: 10.1.10.1
Async src addr: 10.1.10.2
Echo dest addr: 10.1.10.2
Echo src addr: 192.0.2.1
Additional info from Flags:
 FIB is READY
 Session Active on 0/RP0/CPU0
Platform Info: 0x0, Mac Length: 18
Redundancy session info:
 Created from active BFD server
Last Down Diag: None
Last UP Time: Jan  4 03:00:19.272

Received parameters:
 Version: 1, desired tx interval: 10 ms, required rx interval: 10 ms
 Required echo rx interval: 0 ms, multiplier: 2, diag: None
 My discr: 2147483747, your discr: 2147483751, state UP, D/F/P/C/A: 0/0/0/1/0

Transmitted parameters:
 Version: 1, desired tx interval: 10 ms, required rx interval: 10 ms
 Required echo rx interval: 0 ms, multiplier: 2, diag: None
 My discr: 2147483751, your discr: 2147483747, state UP, D/F/P/C/A: 0/1/0/1/0

Tx Echo pkt :
 Version: 0, Local Discr: 2147483751, Sequence No: 0

History:
[Jan  4 03:00:19.272] Session (v1) state change, triggered by event 'Remote
    state init', from INIT to UP with current diag being None
[Jan  4 03:00:16.851] Session (v1) state change, triggered by event 'Remote
    state down', from DOWN to INIT with current diag being None
[Jan  4 03:00:16.509] Session (v1) state change, triggered by event 'Session
    create', from Unknown to DOWN with current diag being None
[Jan  4 03:00:16.509] Flag cleared: session creation is in-progress, currently
    set flags (0x80040)

Offload history:
[Jan  4 03:06:42.013] Packet punted to sw: Packet word0 : (0x20c80218),
desired_min_tx_interval 10000, required_min_rx_interval 10000, Last punted pkt
    required_min_rx_interval 10000
[Jan  4 03:06:42.003] Packet punted to sw: Packet word0 : (0x20d80218),
desired_min_tx_interval 10000, required_min_rx_interval 10000, Last punted pkt
    required_min_rx_interval 10000
[Jan  4 03:06:41.989] Packet punted to sw: Packet word0 : (0x20c80218),
desired_min_tx_interval 10000, required_min_rx_interval 10000, Last punted pkt
    required_min_rx_interval 10000
[Jan  4 03:06:41.980] Packet punted to sw: Packet word0 : (0x20d80218),
desired_min_tx_interval 10000, required_min_rx_interval 10000, Last punted pkt
    required_min_rx_interval 10000

Rx Counters and Timestamps :
Async valid packets received: count 5280
  [Jan  4 03:06:42.013]  [Jan  4 03:06:42.003]  [Jan  4 03:06:41.989]
Async valid packets while session is not in Up state: count 3
  [Jan  4 03:00:19.272]  [Jan  4 03:00:18.030]  [Jan  4 03:00:16.851]
```

BFD Dampening

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.

BFD-Triggered FRR

The BFD-triggered Fast Reroute (FRR) feature allows you to obtain link and node protection using Bidirectional Forwarding Detection (BFD) protocol. This feature provides fast forwarding path failure detection for the following:

- All media types
- Encapsulations
- Topologies
- Routing protocols

In addition to fast forwarding path failure detection, BFD provides a consistent failure detection method for network administrators.

When you enable FRR on Interior Gateway Protocol (IGP) in an IP network, BFD triggers FRR. BFD switches to the backup path when either the primary or the secondary link fails.

When you enable FRR on IGP in an MPLS TE network, you can enable BFD on single-hop IGP links. MPLS TE uses these links to define the protected TE tunnel paths. During traffic disruption on either the primary or the secondary links, BFD triggers a link-down event that triggers FRR.

When you enable FRR on IGP in a segment routing network, BFD triggers FRR.

This feature complements the link-scan failure detection feature. Among these two features, whichever detects the link down event first, triggers the FRR.

Restrictions

BFD-triggered FRR feature has the following restrictions:

- You cannot enable BFD on TE tunnels.

- You cannot enable BFD for multihop.
- You cannot enable BFD on logical bundles.

Configuration Example

```
Router# config
Router(config)# router ospf 100
Router(config-ospf)# router-id 10.32.32.32
Router(config-ospf)# area 0
Router(config-ospf)# exit
Router(config)# interface TenGigE 0/4/0/0.1
Router(config-subif)# bfd minimum-interval 3
Router(config-subif)# bfd fast-detect
Router(config-if)# exit
Router(config)# interface TenGigE 0/4/0/3
Router(config-if)# bfd minimum-interval 3
Router(config-if)# bfd fast-detect
Router(config-if)# exit
Router(config)# mpls traffic-eng
Router(config-mpls)# interface TenGigE 0/4/0/0.1
Router(config-mpls-if)# backup-path tunnel-te 5000
Router(config-mpls-if)# exit
Router(config-mpls)# interface TenGigE0/4/0/3
Router(config-mpls-if)# interface TenGigE0/4/0/3
Router(config-if)# exit
Router(config)# mpls ldp
Router(config-ldp)# router-id 10.32.32.32
Router(config-ldp)# interface Bundle-Ether1
Router(config-ldp-if)# exit
Router(config-ldp)# interface TenGigE0/4/0/0.1
Router(config-ldp-if)# exit
Router(config-ldp-if)# interface TenGigE0/4/0/3
Router(config-ldp-if)# exit
Router(config-ldp)# exit
Router(config)# rsvp
Router(config-rsvp)# interface TenGigE0/4/0/0.1
Router(config-rsvp-if)# bandwidth 100000000
Router(config-rsvp-if)# exit
Router(config-rsvp)# interface TenGigE0/4/0/3
Router(config-rsvp-if)# bandwidth 100000000
Router(config-rsvp-if)# exit
Router(config-rsvp)# exit
Router(config)# exit
Router(config)# interface tunnel-te5000
Router(config-if)# ipv4 unnumbered Loopback0
Router(config-if)# destination 10.31.31.31
Router(config-if)# path-option 1 explicit name ind1
Router(config-if)# exit
Router(config)# exit
Router(config)# explicit-path name direct1-sub
Router(config-expl-path)# index 1 next-address strict ipv4 unicast 10.2.36.2
Router(config-expl-path)# destination 10.31.31.31
Router(config-expl-path)# exit
Router(config)# explicit-path name ind1
Router(config-expl-path)# index 1 next-address strict ipv4 unicast 10.1.33.2
Router(config-expl-path)# index 1 next-address strict ipv4 unicast 10.1.30.1
Router(config-expl-path)# exit
```

Running Configuration

```
router ospf 100
 router-id 32.32.32.32
 area 0
interface TenGigE0/4/0/0.1
   bfd minimum-interval 3
   bfd fast-detect
interface TenGigE0/4/0/3
   bfd minimum-interval 3
   bfd fast-detect
!
mpls traffic-eng
interface TenGigE0/4/0/0.1
  backup-path tunnel-te 5000
interface TenGigE0/4/0/3
 !
mpls ldp
 router-id 32.32.32.32
 interface Bundle-Ether1
 !
 interface TenGigE0/4/0/0.1
 interface TenGigE0/4/0/3
 !
rsvp
interface TenGigE0/4/0/0.1
  bandwidth 100000000
interface TenGigE0/4/0/3
  bandwidth 100000000
!
interface tunnel-te1
 ipv4 unnumbered Loopback0
 destination 10.31.31.31
 fast-reroute
 path-option 1 explicit name direct1-sub
 !
interface tunnel-te5000
 ipv4 unnumbered Loopback0
 destination 10.31.31.31
 path-option 1 explicit name ind1
!
 !
explicit-path name direct1-sub
 index 1 next-address strict ipv4 unicast 10.2.36.2
 !
explicit-path name ind1
 index 1 next-address strict ipv4 unicast 10.1.33.2
 index 2 next-address strict ipv4 unicast 10.1.30.1
```

BFD CPU Offload Support for IPv6

Table 8: Feature History Table

Feature Name	Release Information	Feature Description

BFD CPU offload support for IPv6	Release 24.4.1	Introduced in this release on: NCS 5500 fixed port routers (select variants only*). 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 functionality is supported only on NCS-55A2-MOD-HD-S routers. 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

These limitations apply to the BFD CPU Offload Support for IPv6 feature:

- This feature is supported only on NCS 5500 fixed port routers and NCS 540, where IPv6 BFD sessions are hosted on the ARM processor.
- Only BFD over physical or VLAN interfaces are supported in CPU mode.
- BFD over Bundle (BoB), BFD over logical bundle (BLB), Bridge-Group Virtual Interface (BVI), and multihop (MH) sessions are not 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.
- The minimum-interval value must be greater than or equal to100 ms.

- 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.

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.

Monitor the Per-Session BFD Hardware Offload Statistics

Table 9: Feature History Table

Feature Name	Release Name	Description
Monitor the Per-Session BFD Hardware Offload Statistics	Release 24.1.1	Introduced in this release on:NCS 5700 fixed port routers (NCS 5700 line cards [Mode: Native]) We've improved the ability to diagnose scenarios related to Bidirectional Forwarding Detection (BFD) sessions, enabling more effective troubleshooting and optimization by allowing detailed insights into the packets sent and received by hardware for transmission (Tx) and reception (Rx) per BFD. session. This feature is supported on both IPv4 and IPv6 networks. The feature introduces these changes: **CLI:** • **hw-module profile bfd statistics singlepath** **YANG Data Models:** • New XPaths for `Cisco-IOS-XR-ip-bfd-oper.yang` (See GitHub, YANG Data Models Navigator)

Overview

In hardware offloading for BFD, dedicated hardware components handle packet processing and session management instead of relying solely on software. BFD endpoints are programmed and managed through the Operations, Administration, Maintenance, and Provisioning (OAMP) framework. By leveraging OAMP resources, including configuration settings and monitoring tools, hardware offloading enables the device to effectively manage and monitor BFD sessions. This hardware offloading enhances BFD performance by using the dedicated capabilities of the OAMP framework for configuration and management tasks.

Previously, although Cisco NCS 5700 Series line cards had built-in provision for collecting BFD per-session statistics, there was no supporting infrastructure in place. Starting from Cisco IOS XR Release 24.1.1, this feature allocates the necessary infrastructure as follows:

- Dedicated databases and counter engines, crucial for efficiently storing packet counts associated with each BFD endpoint.
- A comprehensive OAMP framework is essential for effectively facilitating the packet counting process
- Activation of Tx and Rx flags within the OAMP framework, completing a setup that ensures smooth and dependable tracking of BFD endpoint statistics.

You can enable the Monitor the Per-Session BFD Hardware Offload Statistics feature using the **hw-module profile bfd statistics singlepath** command. However, before you configure the **hw-module profile bfd statistics singlepath**command, you must do the following:

- Configure Cisco NCS 5700 Series line cards in native mode using the **hw-module profile npu native-mode-enable** command.
- Enable the hardware modular database profile using either of the following commands:
    - For networks primarily handling Layer 2 traffic, configure the **hw-module profile mdb l2max-se** command.
    - For networks primarily handling Layer 3 traffic. configure the **hw-module profile mdb l3max-se** command.

This feature enables you to access detailed packet statistics for each BFD session, including the count of packets sent and received by hardware.. The router allocates dedicated databases and engines within the counter resource processor to efficiently store packet counts associated with each BFD endpoint. The OAMP, by its directly interacting with the counter resource processor, access, and provides packet transmission records in BFD transmissions. This significantly enhances your ability to diagnose scenarios related to BFD sessions for more effective troubleshooting and optimization.

The feature gathers key operational statistics, including Tx and Rx packet counts per BFD session, which can be utilized to infer potential problems like packet drops or reachability issues on one side of the connection if significant discrepancies or unusual counts are observed.