



## Configure Performance Measurement

Network performance metrics is a critical measure for traffic engineering (TE) in service provider networks. Network performance metrics include the following:

- Packet loss
- Delay
- Delay variation
- Bandwidth utilization

These network performance metrics provide network operators information about the performance characteristics of their networks for performance evaluation and help to ensure compliance with service level agreements. The service-level agreements (SLAs) of service providers depend on the ability to measure and monitor these network performance metrics. Network operators can use Segment Routing Performance Measurement (SR-PM) feature to monitor the network metrics for links and end-to-end TE label switched paths (LSPs).

The following table explains the functionalities supported by performance measurement feature for measuring delay for links or SR policies.

**Table 1: Performance Measurement Functionalities**

Functionality	Details
Profiles	You can configure different profiles for different types of delay measurements. Delay profile type interfaces is used for link-delay measurement. Delay profile type sr-policy is used for SR policy delay measurements. Delay profile allows you to schedule probe and configure metric advertisement parameters for delay measurement.
Protocols	Two-Way Active Measurement Protocol (TWAMP) Light (using RFC 5357 with IP/UDP encap).
Probe and burst scheduling	Schedule probes and configure metric advertisement parameters for delay measurement.
Metric advertisements	Advertise measured metrics periodically using configured thresholds. Also supports accelerated advertisements using configured thresholds.
Measurement history and counters	Maintain packet delay and loss measurement history and also session counters and packet advertisement counters.

- [Liveness Monitoring](#), on page 2
- [Delay Measurement](#), on page 25
- [Path Tracing in SRv6 Network](#), on page 53
- [Two-Way Active Measurement Protocol Light Source Address Filtering](#), on page 63
- [Synthetic Loss Measurement](#), on page 67

## Liveness Monitoring

Liveness refers to the ability of the network to confirm that a specific path, segment, or a node is operational and capable of forwarding packets. Liveness checks are essential for maintaining network availability and reliability.

### Benefits

- **Fault Detection:** You can quickly identify if a device is down, which allows for immediate response and troubleshooting.
- **Load Balancing:** You can identify if the devices in a network are live, so work can be distributed more evenly across the network, preventing overloading of specific components and improving overall performance.
- **System Health:** You can provide an ongoing snapshot of a system's health, helping to identify potential issues before they become significant problems.
- **Maintenance Planning:** Liveness information can also help with maintenance planning, as system administrators can understand which components are live or down and plan maintenance and downtime accordingly without significant disruption to services.
- **Security:** Regular liveness checks can also play a role in maintaining network security. Administrators can take proactive steps to mitigate the damage and prevent future incidents by identifying unusual activity that might indicate a security breach or attack.

You can determine liveness for SR Policy and IP Endpoint.

# IP Endpoint Liveness Monitoring

Table 2: Feature History Table

Feature Name	Release Information	Feature Description
Liveness Monitoring for IP Endpoint over SRv6 Network	Release 24.2.11	<p>In Segment Routing over an IPv6 network (SRv6), you can keep track of the operational status of both the forward and reverse paths of a particular node or IP endpoint. You can use this information for troubleshooting, network maintenance, and optimizing network performance.</p> <p>Additionally, you can use flow labels to verify the liveness of each subsequent hop path toward the IP endpoint of that path. So that, when network traffic is distributed across multiple available paths towards an IP endpoint, liveness detection tracks the operational status of each of these paths towards the IP endpoint.</p> <p>The feature introduces these changes:</p> <p><b>CLI:</b></p> <ul style="list-style-type: none"> <li>The <b>reverse-path</b> and <b>segment-list name</b> keywords are introduced in the <b>segment-routing traffic-eng explicit</b> command.</li> <li>The <b>source-address ipv6</b> is introduced in the <b>performance-measurement endpoint</b> command.</li> </ul> <p><b>YANG Data Model:</b></p> <ul style="list-style-type: none"> <li><code>Cisco-IOS-XR-unperformance-measurement-cfg</code></li> <li><code>Cisco-IOS-XR-perf-meas-oper.yang</code></li> </ul> <p>(see <a href="#">GitHub</a>, <a href="#">YANG Data Models Navigator</a>)</p>

Feature Name	Release Information	Feature Description
IP Endpoint Liveness Monitoring	Release 7.4.1	<p>This feature measures the end-to-end delay and monitors liveness of a specified IP endpoint node, including VRF-aware (awareness of multiple customers belonging to different VRFs).</p> <p>This feature is supported on IPv4, IPv6, and MPLS data planes.</p>

The Segment Routing Performance Measurement (SR-PM) for IP endpoint liveness is a type of node liveness that involves testing whether an IP endpoint or a device identified by an IP address is available to send and receive data.

IP endpoint liveness is verified by sending a request to the IP address of the endpoint and waiting for a response. The probe could be an ICMP echo request (Ping), a TCP packet, a UDP packet, or any other type of packet that the endpoint would respond to.

- If a response is received, the endpoint is considered *live*.
- If no response is received within a certain time frame, the endpoint is considered *down* or *unreachable*.

IP endpoint dynamically measures the liveness towards a specified IP endpoint. IP endpoints can be located in a default or nondefault VRFs. IP endpoint is any device in the network a device identified by an IP address.

Liveness of an IP endpoint is verified by sending a request to the IP address of the endpoint and waiting for a response, which is referred to as a probe.

The endpoint of a probe is defined by an IP address, which can be either IPv4 or IPv6. This IP address can be any address that the sender can reach, such as a local interface or a remote node or host, either within an operator's network or accessible via a VRF.

The endpoint of a probe can be any IP address reachable by the sender. For example, a local interface or a remote node or host located within an operator's network or reachable through a VRF.

The IP address of the endpoint can be reached through an IP path, MPLS, LSP SRV6, or IP tunnel (GRE).

- When the endpoint is reachable using an MPLS LSP (for example, SR, LDP, RSVP-TE, SR Policy), the forwarding stage imposes the corresponding MPLS transport labels.
- When the endpoint is reachable via a GRE tunnel, the forwarding stage imposes the corresponding GRE header.
- When the endpoint is reachable via a VRF in an MPLS network, the forwarding stage imposes the corresponding MPLS service labels. In the forward path, the sender node uses the configured VRF for the endpoint address. In the return path, the reflector node derives the VRF based on which incoming VRF label the probe packet is received with.
- When the endpoint is reachable using SRv6, the forwarding stage imposes the SRv6 encapsulation.

You can configure the following parameters in the **performance-measurement** command:

- **Endpoint:** The endpoint of a probe is defined by an IP address, which can be either IPv4 or IPv6. This IP address can be any address that the sender can reach, such as a local interface or a remote node or

host, either within an operator's network or accessible via a VRF. The endpoint's IP address can be located in the global routing table or under a user-specified VRF routing table.

The endpoint of a probe can be any IP address reachable by the sender. For example, a local interface or a remote node or host located within an operator's network or reachable through a VRF.

Use the **performance-measurement endpoint** command to configure a probe endpoint source and destination addresses on a sender node.

- **VRF:** You can define the endpoint IP address belonging to a specific VRF. Use the **performance-measurement endpoint {ipv4 | ipv6} ip\_addr [vrf WORD]** command to configure an endpoint to define the VRF. Endpoint segment list configuration is not supported under nondefault VRF.
  - VRF-awareness allows operators to deploy probes in the following scenarios:
    - Managed Customer Equipment (CE) scenarios:
      - PE to CE probes
      - CE to CE probes
    - Unmanaged Customer Equipment (CE) scenarios:
      - PE to PE probes
      - PE to PE (source from PE-CE interface) probes

- **Source address:** You can define the source of the endpoint using the endpoint specific source address and the global source address.

Global source address configuration is applied to all the endpoints when the endpoint specific source address configuration isn't specified. endpoint specific configuration overrides all the global source address configuration for those specific endpoints for which source addresses are configured.

For Micro-SID configuration for IPv4 endpoint sessions, if IPv6 global source address is configured, then it applies the configured global IPv6 source address for the IPv6 header in the SRv6 packet. If IPv6 global address is not configured, then It does not form a valid SRv6 packet.

You can use the **source-address** keyword under the **performance-measurement** command to define the global source address or use the keyword under **performance-measurement endpoint** to define endpoint specific source address.

- **Reverse Path:** To detect the liveness of the reverse of the segment, you can configure the reverse path using the **reverse-path** command.

The default reverse path configured under the endpoint submode is only used for sessions with segment list. The endpoint session without a segment list does not support reverse path configuration and will not use this reverse path.

The **reverse-path** under the **performance-measurement endpoint** is used as the default reverse path if there are no reverse paths configured under the segment list.

Use the **reverse-path** under the **performance-measurement endpoint segment-routing traffic-eng explicit segment-list name** to configure the reverse path under segment list.

The reverse type must be the same as the forward path. Using different types for forward and reverse paths is not supported. For example, uSID forward path and uSID reverse path; MPLS forward path and MPLS reverse path.

User-configured segment-list can also represent the reverse path (reflector to sender) when probe is configured in liveness detection mode. Up to 128 segment-lists can be configured under a probe. An additional PM session is created for each segment-list. Segment-lists are configured under **segment-routing traffic-eng segment-list** submode. See [SR-TE Policy with Explicit Path](#) for details about configuring segment lists.

- **Flow Label:** The flow label field in the IPv6 header is used to carry information that helps distribute traffic across multiple network paths. The flow label is a 20-bit field in the IPv6 header designed to carry information about the flow of packets, which routers can use to identify and differentiate between different traffic flows. Flow label sweeping uses a flow label to distribute the traffic load across multiple paths to the endpoint.

Use the **flow-label** keyword to configure flow label.

### Supported and Unsupported Features

The table lists supported and unsupported features for Liveness Monitoring for IP Endpoint over SRv6 Network.

**Table 3: Supported and Unsupported Features for Liveness Monitoring for IP Endpoint over SRv6 Network**

Supported Features	Unsupported Features
SRv6 Endpoint Liveness in Default VRF Example: <i>endpoint ipv6 fccc:2:: liveness-detection</i> In this example, the endpoint with the IPv6 address fccc:2:: is the SRv6 uSID format.	IPv6 Endpoint Liveness in Default VRF (over SRv6) Example: <i>endpoint ipv6 10::2 liveness-detection</i> In this example, the endpoint with the IPv6 address 2::2 is part of an underlay network using SRv6.
IPv6 Endpoint Liveness in VRF (static uDT6) Example: <i>endpoint ipv6 fccc:2:fe02:: liveness-detection</i> In this example, the endpoint with the IPv6 address fccc:2:fe02:: is the static uDT6 uSID carrier	IPv6 Endpoint Liveness in VRF (dynamic DT6 encap) Example: <i>endpoint ipv6 10::1 vrf purple source-address ipv6 10::2 liveness-detection</i>
IPv4 Endpoint Liveness in VRF or GRT (static uDT4) Example: <i>endpoint ipv4 10.5.56.1 source-address ipv4 10.1.17.1 segment-routing traffic-eng explicit segment-list name vrf-2-3-5-udt4 liveness-detection</i> The segment-list <i>vrf-2-3-5-udt4</i> here has static uDT4 SID.	IPv4 Endpoint Liveness in VRF or GRT (dynamic uDT4 encap) Example: <i>endpoint ipv4 10.0.0.1 vrf purple source-address ipv4 10.0.0.2 liveness-detection</i>
	IPv6 address over SRv6 underlay Example: <i>endpoint ipv6 10::1 source-address ipv6 10::2 liveness-detection</i>

### Usage Guidelines and Limitations

- For liveness detection, the session fails to come up when the endpoint address is a regular IPv4 address in a default VRF and that is a normal loopback IP address that uses IGP path. Packets get dropped with the following message. However, this issue does not apply if a segment list is configured.

```
GRE IPv4 decap qualification failed
```

To mitigate this issue, you must configure the GRE tunnel on responder. The following example shows how to configure GRE tunnel:

```
/*Tunnel config on responder*\ninterface tunnel-ipl\n tunnel model ipv4 decap\n tunnel source 10.3.1.1\n tunnel destination 10.1.1.1
```

- Liveness session without segment list for an endpoint in a non-default VRF is not supported.
- SR Performance Measurement endpoint session over BVI interface is not supported.
- SRv6 locator prefix and VRF SRv6 locator/function (uDT4/uDT6) as IPv6 endpoint of a probe is not supported.
- IPv6 Endpoint Liveness in Default VRF is not supported over SRv6.
- PM probe over GREv4 is supported.

## IP Endpoint Liveness Detection in an SR MPLS Network

IP endpoint liveness detection leverages the loopback measurement-mode. The following workflow describes the sequence of events.

1. The sender creates and transmits the PM probe packets.

The IP destination address (DA) on the probe packets is set to the loopback value of the sender itself.

The transmit timestamp (T1) is added to the payload.

The probe packet is encapsulated with the label corresponding to the endpoint.

2. The network delivers the PM probe packets following the LSP toward the endpoint.

3. The end-point receives the PM probe packets.

Packets are forwarded back to the sender based on the forwarding entry associated with the IP DA of the PM probe packet. If an LSP exists, the probe packet is encapsulated with the label of the sender.

4. The sender node receives the PM probe packets.

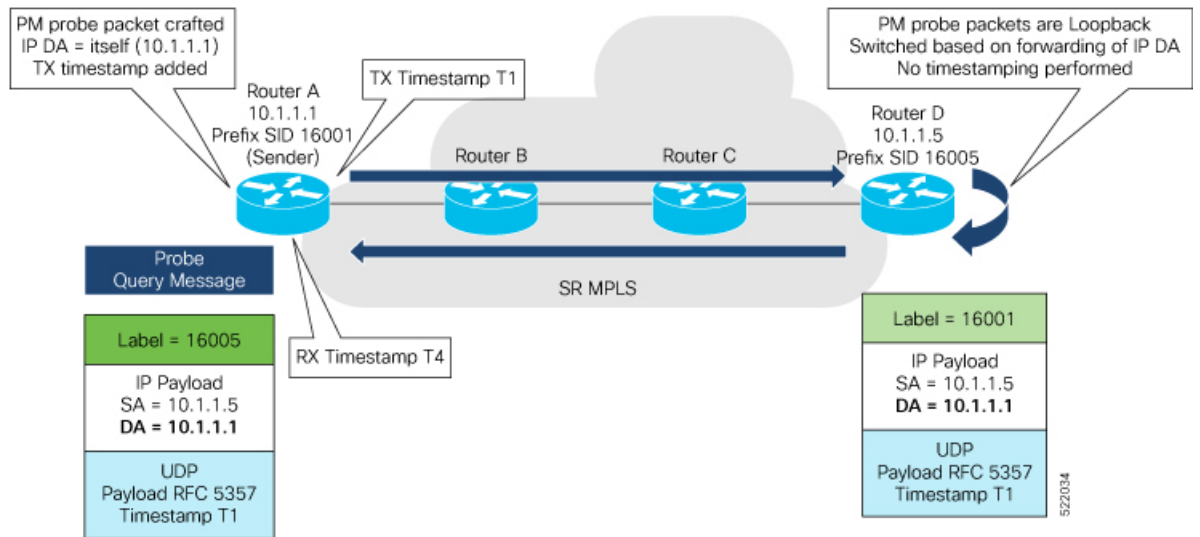
The received timestamp (T4) stored.

If the sender node doesn't receive the specified number of probe packets (based on the configured multiplier), the sender node declares the PM session as down.

The following figure illustrates a liveness detection probe toward an IP endpoint learned by the IGP. The network interconnecting the sender and reflector provides MPLS connectivity with Segment Routing.

The liveness detection multiplier is set to 5 to specify the number of consecutive missed probe packets before the PM session is declared as down.

Figure 1: IP Endpoint Liveness Detection



### Configuration Example

```

RouterA(config)# performance-measurement
RouterA(config-perf-meas)# endpoint ipv4 1.1.1.5
RouterA(config-pm-ep)# source-address ipv4 1.1.1.1
RouterA(config-pm-ep)# liveness-detection
RouterA(config-pm-ep-ld)# exit
RouterA(config-pm-ep)# exit
RouterA(config-perf-meas)# liveness-profile endpoint default
RouterA(config-pm-ld-ep)# liveness-detection
RouterA(config-pm-ld-ep-ld)# multiplier 5
RouterA(config-pm-ld-ep-ld)# exit
RouterA(config-pm-ld-ep)# probe
RouterA(config-pm-ld-ep-probe)# measurement-mode loopback

```

### Running Configuration

```

performance-measurement
 endpoint ipv4 1.1.1.5
  source-address ipv4 1.1.1.1
  liveness-detection
  !
 !
 liveness-profile endpoint default
  liveness-detection
  multiplier 5
  !
  probe
  measurement-mode loopback
  !
 !
 !
end

```

### Verification

```

RouterA# show performance-measurement endpoint ipv4 1.1.1.5

```



```

-----
0/RSP0/CPU0
-----

Endpoint name: IPv4-1.1.1.5-vrf-default
Source address      : 1.1.1.1
VRF name           : default
Liveness Detection  : Enabled
Profile Keys:
  Profile name      : default
  Profile type      : Endpoint Liveness Detection

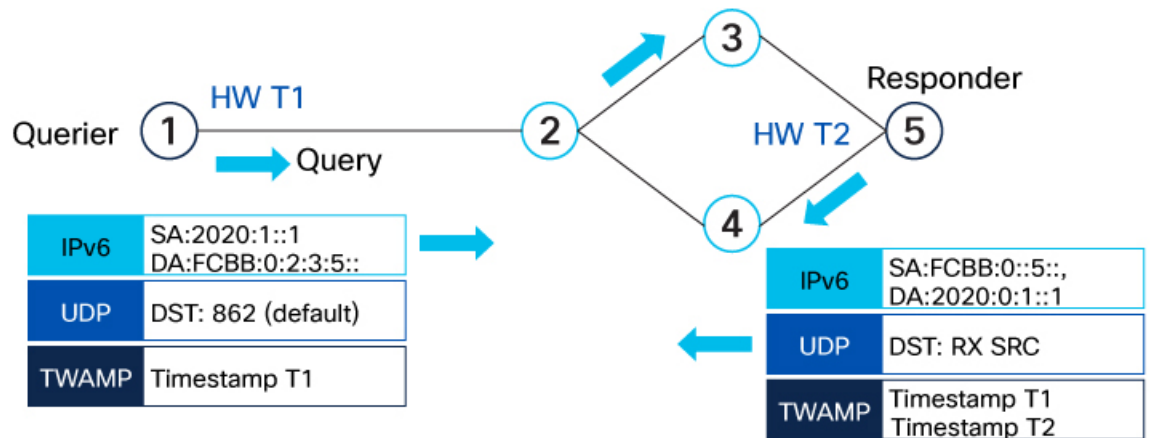
Segment-list       : None
Session State: Down
Missed count: 0
    
```

## IP Endpoint Liveness in an SRv6 Network

IP endpoint liveness detection leverages the loopback measurement-mode. The following workflow describes the sequence of events.

1. The querier creates and transmits the PM TWAMP probe packets based on the endpoint configuration.  
The packet is formed so it could reach the responder and return to the querier node over the SRv6 transport network.
2. The querier node declares the session up once it receives the probe packet back.  
If the sender node doesn't receive the specified number of probe packets consecutively based on the configured multiplier, it declares the PM session down.

Figure 2: IP Endpoint Liveness In an SRv6 Network



523654



**Note** Liveness is not supported for non-default VRF.

### Configuration Example

```

Router(config)#performance-measurement
Router(config-perf-meas)#source-address ipv6 2020:1::1
    
```

```

Router(config-perf-meas)#endpoint ipv6 FCBB:0::5::
Router(config-pm-ep)#exit
Router(config-perf-meas)#liveness-profile endpoint default
Router(config-pm-ld-ep)#probe
Router(config-pm-ld-ep-probe)#exit
Router(config-pm-ld-ep)#liveness-detection
Router(config-pm-ld-ep-ld)#multiplier 3
Router(config-pm-ld-ep-ld)#

```

The following example shows how to configure liveness with segment list and reverse path.

```

Router(config-sr)#traffic-eng
Router(config-sr-te)#segment-lists
Router(config-sr-te-segment-lists)#srv6
Router(config-sr-te-sl-global-srv6)#sid-format usid-f3216
Router(config-sr-te-sl-global-srv6)#exit
Router(config-sr-te-sl-global)#segment-list test
Router(config-sr-te-sl)#srv6
Router(config-sr-te-sl-srv6)#index 10 sid ff::2
Router(config-sr-te-sl-srv6)#index 20 sid ff::3

```

The following example shows how to configure liveness reverse path under segment list and under endpoint:

```

Router(config)#performance-measurement
Router(config-perf-meas)#endpoint ipv6 ff::2

/* Configure reverse path under segment list name */
Router(config-pm-ep)#segment-routing traffic-eng explicit segment-list name fwd-path
Router(config-pm-ep-sl)#reverse-path segment-list name rev-path
Router(config-pm-ep-sl)#exit

/* Configure reverse path under performance measurement endpoint */
Router(config-pm-ep)# segment-routing traffic-eng explicit reverse-path segment-list name
rev-path-name

```

The following example shows how to configure liveness with flow label:

```

Router(config-perf-meas)#liveness-profile endpoint default
Router(config-pm-ld-ep)#probe
Router(config-pm-ld-ep-probe)#flow-label from 1000 to 20000 increment 16
Router(config-pm-ld-ep-probe)#liveness-detection
Router(config-pm-ld-ep-ld)#multiplier 3

```

The following example shows how to configure liveness with flow label sweeping:

```

Router#configure
Router(config)#performance-measurement
Router(config-perf-meas)#liveness-profile name profile-sweeping
Router(config-pm-ld-profile)# flow-label from 1000 to 20000 increment 16
Router(config-pm-ld-profile)#commit

```

## Verification

```

Router# show performance-measurement endpoint detail
Endpoint name: IPv6-FCBB:0::5::-vrf-default
Source address          : 2020:1::1
VRF name                : default
Liveness Detection      : Enabled
Profile Keys:
  Profile name          : default
  Profile type          : Endpoint Liveness Detection
Segment-list            : None
Liveness Detection session:
  Session ID           : 4109
  Flow-label           : 1000

```

```
Session State: Up
Last State Change Timestamp: Jan 23 2024 16:06:01.214
Missed count: 0

Liveness Detection session:
  Session ID           : 4110
  Flow-label          : 2000
  Session State: Up
  Last State Change Timestamp: Jan 23 2024 16:06:01.214
  Missed count: 0

Segment-list           : test-dm-two-carrier-s12
FCBB:0::5:2:e004::/64
  Format: f3216
FCBB:0::5:3:e000::/64
  Format: f3216
FCBB:0::5:2:e004::/64
  Format: f3216
FCBB:0::5:2:e000::/64
  Format: f3216
FCBB:0::5:1:e000::/64
  Format: f3216
FCBB:0::5:1:e004::/64
  Format: f3216
FCBB:0::5:4:e000::/64
  Format: f3216
FCBB:0::5:4::/48
  Format: f3216

Liveness Detection session:
  Session ID           : 4111
  Flow-label          : 1000
  Session State: Up
  Last State Change Timestamp: Jan 23 2024 16:06:01.217
  Missed count: 0

Liveness Detection session:
  Session ID           : 4112
  Flow-label          : 2000
  Session State: Up
  Last State Change Timestamp: Jan 23 2024 16:06:01.217
  Missed count: 0
```

## SR Policy Liveness Monitoring

Table 4: Feature History Table

Feature Name	Release Information	Feature Description
SR Policy Liveness Monitoring on Segment Routing over IPv6 (SRv6)	Release 7.11.1	In segment routing over IPv6 (SRv6), you can now verify end-to-end traffic forwarding over an SR policy candidate path by periodically sending probe messages. Performance monitoring on an SRv6 network enables you to track and monitor traffic flows at a granular level.  Earlier releases supported SR policy liveness monitoring over an SR policy candidate path on MPLS.
SR Policy Liveness Monitoring	Release 7.5.2	This feature allows you to verify end-to-end traffic forwarding over an SR Policy candidate path by periodically sending performance monitoring packets.

SR Policy liveness monitoring allows you to verify end-to-end traffic forwarding over an SR Policy candidate path by periodically sending probe messages. The head-end router sends PM packets to the SR policy's endpoint router, which sends them back to the head-end without any control-plane dependency on the endpoint router.

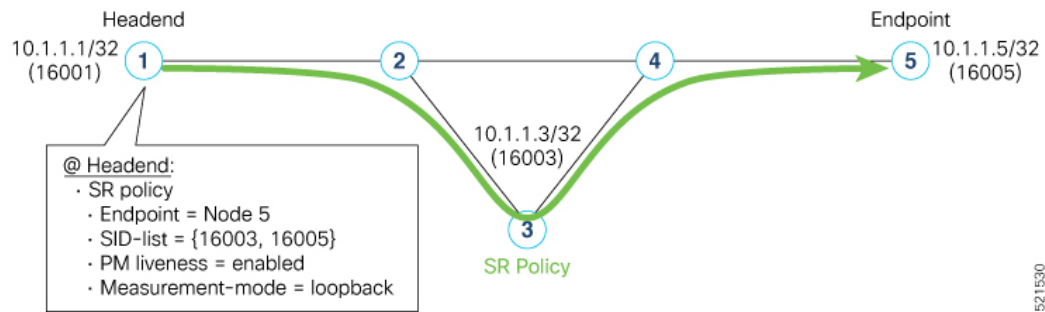
For more information about the segment routing over IPv6, see [Segment Routing over IPv6](#) topic.

The following are benefits to using SR-PM liveness monitoring:

- Allows both liveness monitoring and delay measurement using a single-set of PM packets as opposed to running separate monitoring sessions for each purpose. This improves the overall scale by reducing the number of PM sessions required.
- Eliminates network and device complexity by reducing the number of monitoring protocols on the network (for example, no need for Bidirectional Failure Detection [BFD]). It also simplifies the network and device operations by not requiring any signaling to bootstrap the performance monitoring session.
- Improves interoperability with third-party nodes because signaling protocols aren't required. In addition, it leverages the commonly supported TWAMP protocol for packet encoding.
- Improves liveness detection time because PM packets aren't punted on remote nodes
- Provides a common solution that applies to data-planes besides MPLS, including IPv4, IPv6, and SRv6.

The workflow associated with liveness detection over SR policy is described in the following sequence.

Consider an SR policy programmed at head-end node router 1 towards end-point node router 5. This SR policy is enabled for liveness detection using the loopback measurement-mode.



521530

- **A:** The head-end node creates and transmits the PM probe packets.

The IP destination address (DA) on the probe packets is set to the loopback value of the head-end node itself.

A transmit (Tx) timestamp is added to the payload.

Optionally, the head-end node may also insert extra encapsulation (labels) to enforce the reverse path at the endpoint node.

Finally, the packet is injected into the data-plane using the same encapsulation (label stack) of that of the SR policy being monitored.

- **B:** The network delivers the PM probe packets as it would user packet for the SR policy.
- **C:** The end-point node receives the PM probe packets.

Packets are switched back based on the forwarding entry associated with the IP DA of the packet. This would typically translate to the end-point node pushing the prefix SID label associated with the head-end node.

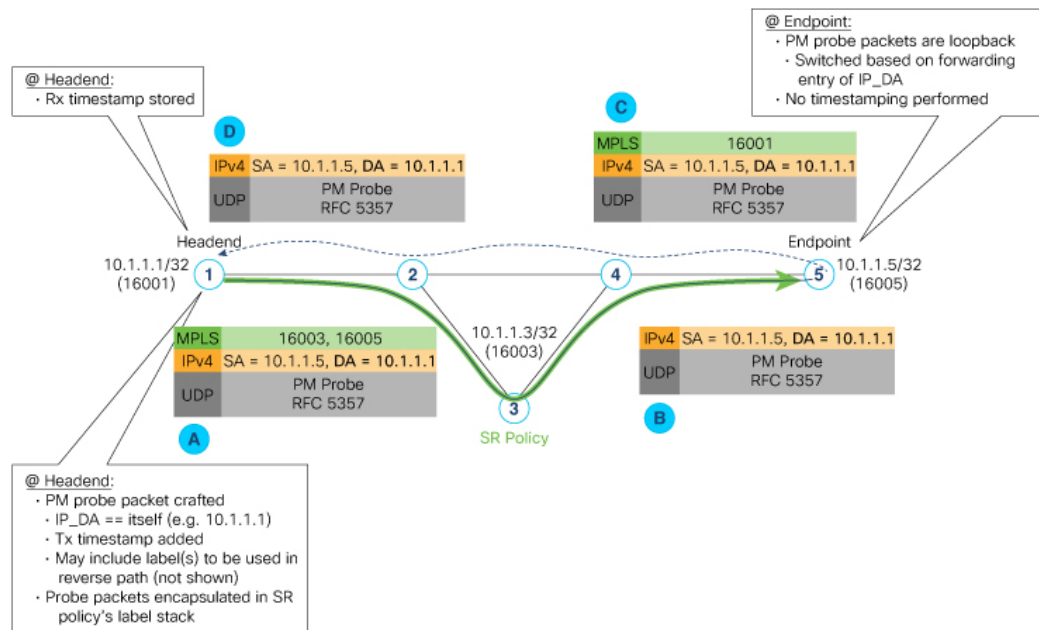
If the head-end node inserted label(s) for the reverse path, then the packets are switched back at the end-point node based on the forwarding entry associated with the top-most reverse path label.

- **D:** Headend node receives the PM probe packets.

A received (Rx) timestamp stored.

If the head-end node receives the PM probe packets, the head-end node assume that the SR policy active candidate path is up and working.

If the head-end node doesn't receive the specified number of consecutive probe packets (based on configured multiplier), the head-end node assumes the candidate path is down and a configured action is triggered.



521531

## Usage Guidelines and Limitations

The following usage guidelines and limitations apply:

- Liveness-detection and delay-measurement aren't supported together
- When liveness-profile isn't configured, SR Policies use the default values for the liveness-detection profile parameters.

## Configuring SR Policy Liveness Monitoring

Configuring SR Policy liveness monitoring involves the following steps:

- Configuring a performance measurement liveness profile to customize generic probe parameters
- Enabling liveness monitoring under SR Policy by associating a liveness profile, and customizing SR policy-specific probe parameters

## Configuring Performance Measurement Liveness Profile

Liveness monitoring parameters are configured under `sr-policy` sub-mode. The following parameters are configurable:

- **liveness-profile** `{sr-policy default | name name}`
- **probe**: Configure the probe parameters.
- **measurement-mode**: Liveness detection must use loopback mode (see Measurement Mode topic within this guide).
- **tx-interval**: Interval for sending probe packet. The default value is 3000000 microseconds and the range is from 3300 to 15000000 microseconds.
- **tos dscp value**: The default value is 48 and the range is from 0 to 63. You can modify the DSCP value of the probe packets, and use this value to prioritize the probe packets from headend to tailend.

- **sweep destination ipv4 127.x.x.x range** *range*: Configure SR Policy ECMP IP-hashing mode. Specify the number of IP addresses to sweep. The range is from 0 (default, no sweeping) to 128. The option is applicable to IPv4 packets.




---

**Note** The destination IPv4 headend address 127.x.x.x – 127.y.y.y is used in the Probe messages to take advantages of 3-tuple IP hashing (source-address, destination-address, and local router ID) for ECMP paths of SR-MPLS Policy.

The destination IPv4 address must be 127/8 range (loopback), otherwise it will be rejected.

---




---

**Note** One PM session is always created for the actual endpoint address of the SR Policy.

---

- **liveness-detection**: Configure the liveness-detection parameters:
- **multiplier**: Number of consecutive missed probe packets before the PM session is declared as down. The range is from 2 to 10, and the default is 3.




---

**Note** The detection-interval is equal to (tx-interval \* multiplier).

---

### Enabling Liveness Monitoring under SR Policy

Enable liveness monitoring under SR Policy, associate a liveness-profile, and configure SR Policy-specific probe parameters under the **segment-routing traffic-eng policy performance-measurement** sub-mode. The following parameters are configurable:

- **liveness-detection**: Enables end-to-end SR Policy Liveness Detection for all segment-lists of the active and standby candidate-path that are in the forwarding table.
- **liveness-profile name** *name*: Specifies the profile name for named profiles.
- **invalidation-action {down | none}**:
  - **Down (default)**: When the PM liveness session goes down, the candidate path is immediately operationally brought down.
  - **None**: When the PM liveness session goes down, no action is taken. If logging is enabled, the failure is logged but the SR Policy operational state is not modified.
- **logging session-state-change**: Enables Syslog messages when the session state changes.
- **reverse-path label** {*BSID-value* | *NODE-SID-value* | *ADJACENCY-SID-value*}: Specifies the MPLS label to be used for the reverse path for the reply. If you configured liveness detection with ECMP hashing, you must specify the reverse path. The default reverse path uses IP Reply.
  - *BSID-value*: The Binding SID (BSID) label for the reverse SR Policy. (This is practical for manual SR policies with a manual BSID.)

- *NODE-SID-value*: The Node SID is a segment type that represents the ECMP-aware shortest path to reach a particular IP prefix from any IGP topology location
- *ADJACENCY-SID-value*: The absolute SID label of the (local) Sender Node to be used for the reverse path for the reply.

## Configuration Examples

### Configure a Default SR-Policy PM Liveness-Profile

The following example shows a default sr-policy liveness-profile:

```
RP/0/RSP0/CPU0:ios(config)# performance-measurement
RP/0/RSP0/CPU0:ios(config-perf-meas)# liveness-profile sr-policy default
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy)# probe

RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe)# tx-interval 150000
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe)# tos dscp 52
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe)# exit
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy)# liveness-detection
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-ld)# multiplier 5
```

### Running Configuration:

```
performance-measurement
 liveness-profile sr-policy default
  liveness-detection
   multiplier 5
 !
 probe
  tos dscp 52
  tx-interval 150000
 !
 !
end
```

### Configure a Named (Non-Default) SR-Policy PM Liveness-Profile

The following example shows a named sr-policy liveness-profile:

```
Router(config)# performance-measurement
Router(config-perf-meas)# liveness-profile name sample-profile
Router(config-pm-ld-profile)# probe
Router(config-pm-ld-probe)# tx-interval 150000
Router(config-pm-ld-probe)# tos dscp 52
Router(config-pm-ld-probe)# exit
Router(config-pm-ld-profile)# liveness-detection
Router(config-pm-ld-profile-ld)# multiplier 5
Router(config-pm-ld-profile-ld)#commit
```

### Running Configuration:

```
performance-measurement
 liveness-profile name sample-profile
  liveness-detection
   multiplier 5
 !
 probe
  tos dscp 52
  tx-interval 150000
```



```

!
!
!
end

```

### Configure a SR-Policy PM Liveness-Profile with Sweep Parameters

The following example shows a named sr-policy liveness-profile with sweep parameters:

```

Router(config)# performance-measurement
Router(config-perf-meas)# liveness-profile name sample-profile
Router(config-pm-ld-profile)# probe
Router(config-pm-ld-probe)# tx-interval 150000
Router(config-pm-ld-probe)# tos dscp 52
Router(config-pm-ld-probe)# sweep
Router(config-pm-ld-probe-sweep)# destination ipv4 127.0.0.1 range 25
Router(config-pm-ld-probe-sweep)# exit
Router(config-pm-ld-probe)# exit

Router(config-pm-ld-profile)# liveness-detection
Router(config-pm-ld-profile-ld)# multiplier 5
Router(config-pm-ld-profile-ld)#commit

```

### Running Configuration

```

performance-measurement
  liveness-profile name sample-profile
  liveness-detection
    multiplier 5
  !
  probe
    tos dscp 52
    sweep
    destination ipv4 127.0.0.1 range 25
  !
  tx-interval 150000
  !
!
end

```

### Enable Liveness Monitoring Under SR Policy

The following example shows how to enable liveness monitoring under SR Policy, associate a liveness-profile, and configure the invalidation action:

```

RP/0/RSP0/CPU0:ios(config)# segment-routing traffic-eng
RP/0/RSP0/CPU0:ios(config-sr-te)# policy FOO
RP/0/RSP0/CPU0:ios(config-sr-te-policy)# performance-measurement
RP/0/RSP0/CPU0:ios(config-sr-te-policy-perf-meas)# liveness-detection
RP/0/RSP0/CPU0:ios(config-sr-te-policy-live-detect)# liveness-profile name sample-profile
RP/0/RSP0/CPU0:ios(config-sr-te-policy-live-detect)# invalidation-action none

```

### Running Config

```

segment-routing
  traffic-eng
  policy FOO
  performance-measurement
    liveness-detection
      liveness-profile name sample-profile
      invalidation-action none
  !
!
!
!

```

```
!
end
```

### Enable Liveness Monitoring under SR Policy with Optional Parameters

The following example shows how to enable liveness monitoring under SR Policy, associate a liveness-profile, and configure reverse path label and session logging:

```
RP/0/RSP0/CPU0:ios(config)# segment-routing traffic-eng
RP/0/RSP0/CPU0:ios(config-sr-te)# policy BAA
RP/0/RSP0/CPU0:ios(config-sr-te-policy)# performance-measurement
RP/0/RSP0/CPU0:ios(config-sr-te-policy-perf-meas)# liveness-detection
RP/0/RSP0/CPU0:ios(config-sr-te-policy-live-detect)# liveness-profile name sample-profile
RP/0/RSP0/CPU0:ios(config-sr-te-policy-live-detect)# invalidation-action down
RP/0/RSP0/CPU0:ios(config-sr-te-policy-live-detect)# logging session-state-change
RP/0/RSP0/CPU0:ios(config-sr-te-policy-live-detect)# exit
RP/0/RSP0/CPU0:ios(config-sr-te-policy-perf-meas)# reverse-path label 16001
```

### Running Config

```
segment-routing
 traffic-eng
  policy BAA
  performance-measurement
   liveness-detection
    logging
     session-state-change
    !
   liveness-profile name sample-profile
   invalidation-action down
  !
  reverse-path
   label 16001
  !
 !
 !
 !
end
```

## Configure Segment Lists to Activate Candidate Paths in SRv6 for PM Liveness

Table 5: Feature History Table

Feature Name	Release Information	Feature Description
Configure Segment Lists to Activate Candidate Paths in SRv6 for PM Liveness	Release 7.11.1	<p>You can now enable a candidate path to be up by configuring the minimum number of active segment lists associated with the candidate path. The head-end router determines that a candidate path is up based on the minimum number of active segment lists configured.</p> <p>In earlier releases, the head-end router identified a candidate path as up only when all the segment lists associated with the path were active.</p> <p>The feature introduces these changes:</p> <p><b>CLI:</b></p> <ul style="list-style-type: none"> <li>The <b>validation-cp minimum-active segment-lists</b> option is introduced in the <b>performance-measurement liveness-detection</b> command.</li> </ul> <p><b>YANG Data Models:</b></p> <ul style="list-style-type: none"> <li><a href="#">Cisco-IOS-XR-infra-xtc-agent-cfg.yang</a></li> </ul> <p>See (<a href="#">GitHub</a>, <a href="#">Yang Data Models Navigator</a>)</p>

The state of the segment lists in a candidate path determines whether a candidate path is up or down. You can now configure the minimum number of active segment lists associated with a candidate path. The head-end router identifies a candidate path as up when one or more segment lists are active.



**Note** If the configured minimum number of active segment lists is greater than the number of available segment lists in a candidate path, the head-end router determines the candidate path as up only when all the segment lists are active.

In earlier releases, the router identified a candidate path as up only when all the segment lists associated with the path were active.

### Configure the minimum number of segment lists in SRv6

Perform this task to activate three segment lists to have the PM liveness session up:

```
Router(config)#segment-routing
Router(config-sr)#traffic-eng
Router(config-sr-te)#policy po-103
Router(config-sr-te-policy)#performance-measurement
Router(config-sr-te-policy-perf-meas)#liveness-detection
Router(config-sr-te-policy-live-detect)#validation-cp minimum-active segment-lists 3
```

### Running Configuration

```
segment-routing
traffic-eng
policy po-103
performance-measurement
liveness-detection
validation-cp minimum-active segment-lists 3
!
!
!
!
!
```

### Verification

The following example shows three active segment-lists to have the PM liveness session up:

```
Router#show performance-measurement sr-policy liveness color 103 detail verbose private
Mon Oct 30 15:10:51.863 EDT
```

---

```
0/1/CPU0
```

---

```
SR Policy name: srte_c_103_ep_3::1
Color : 103
SRv6 Encap Source Address : 1::1
Endpoint : 3::1
Handle : 0x00000000
Policy to be deleted : False
Number of candidate-paths : 1

Candidate-Path:
Instance : 5
Preference : 300
Protocol-origin : Configured
Discriminator : 300
Profile Keys:
Profile name : default
Profile type : SR Policy Liveness Detection
Candidate path to be deleted: False
Source address : 1::1
Local label : Not set
Fast notification for session down: Disabled
No fast notifications have been sent
Number of segment-lists : 3
Liveness Detection: Enabled
Minumum SL Up Required: 1
Session State: Up
Last State Change Timestamp: Oct 30 2023 15:10:16.322
Missed count: 0

Segment-List : s1-1041
fccc:cc00:1:fe10:: (Local Adjacency SID)
fccc:cc00:2:fe41::/64
```

```

Format: f3216
Segment List ID: 0
Reverse path segment-List: Not configured
Segment-list to be deleted: False
Number of atomic paths : 1
Liveness Detection: Enabled
  Session State: Up
    Last State Change Timestamp: Oct 30 2023 15:10:16.322
    Missed count: 0

```

```

Atomic path:
  Flow Label : 0
  Session ID : 4198
  Trace ID : 738913600
  Atomic path to be deleted: False
  NPU Offloaded session : False
  Timestamping Enabled : True
  Liveness Detection: Enabled
    Session State: Up
      Last State Change Timestamp: Oct 30 2023 15:10:16.322
      Missed count: 0
  Responder IP : 1::1
  Number of Hops : 3

```

```

Segment-List : s1-1042
fcc:cc00:1:fe10:: (Local Adjacency SID)
fcc:cc00:2:fe42::/64
Format: f3216
Segment List ID: 0
Reverse path segment-List: Not configured
Segment-list to be deleted: False
Number of atomic paths : 1
Liveness Detection: Enabled
  Session State: Up
    Last State Change Timestamp: Oct 30 2023 15:10:16.322
    Missed count: 0

```

```

Atomic path:
  Flow Label : 0
  Session ID : 4199
  Trace ID : 954039677
  Atomic path to be deleted: False
  NPU Offloaded session : False
  Timestamping Enabled : True
  Liveness Detection: Enabled
    Session State: Up
      Last State Change Timestamp: Oct 30 2023 15:10:16.322
      Missed count: 0
  Responder IP : 1::1
  Number of Hops : 3

```

```

Segment-List : s1-1043
fcc:cc00:1:fe10:: (Local Adjacency SID)
fcc:cc00:2:fe43::/64
Format: f3216
Segment List ID: 0
Reverse path segment-List: Not configured
Segment-list to be deleted: False
Number of atomic paths : 1
Liveness Detection: Enabled
  Session State: Up
    Last State Change Timestamp: Oct 30 2023 15:10:16.322
    Missed count: 0

```

```

Atomic path:
  Flow Label           : 0
  Session ID          : 4200
  Trace ID            : 1119107116
  Atomic path to be deleted: False
  NPU Offloaded session : False
  Timestamping Enabled : True
  Liveness Detection: Enabled
    Session State: Up
    Last State Change Timestamp: Oct 30 2023 15:10:16.322
    Missed count: 0
  Responder IP        : 1::1
  Number of Hops      : 3

```

---

0/RSP0/CPU0

---

## Configure Flow Labels in SRv6 Header for PM Liveness

**Table 6: Feature History Table**

Feature Name	Release Information	Feature Description
Configure Flow Labels in SRv6 Header for PM Liveness	Release 7.11.1	<p>You can now monitor the activeness of multiple paths for a given segment list using flow labels in the SRv6 header.</p> <p>In earlier releases, the SRv6 header didn't include flow labels.</p> <p>The feature introduces these changes:</p> <p><b>CLI:</b></p> <ul style="list-style-type: none"> <li>The <b>flow-label</b> keyword is introduced in the <b>performance-measurement liveness-profile</b> command.</li> </ul> <p><b>YANG Data Models:</b></p> <ul style="list-style-type: none"> <li><a href="#">Cisco-IOS-XR-performance-measurement-ctrl.yang</a></li> <li><a href="#">Cisco-IOS-XR-perf-meas-oper.yang</a></li> </ul> <p>See (<a href="#">GitHub</a>, <a href="#">Yang Data Models Navigator</a>)</p>

To monitor the activeness of multiple paths for a given a segment list, you can configure the SRv6 header to include flow labels as the packet travels in the network. When there are multiple paths, different traffic flows may use different paths. A flow label is a flow identifier and you can use different flow labels to monitor different ECMP paths. It's only used for IPv6 probe packets. Flow labels are 20-bit fields in the SRv6 header.

### Configure flow labels in the SRv6 header

Perform the following task in the global configuration mode to configure flow labels in the SRv6 header:

```
Router#configure
Router(config)#performance-measurement
Router(config-perf-meas)#liveness-profile name name1
Router(config-pm-ld-profile)#probe flow-label from 0 to 1000000 increment 10
```

### Running Configuration

```
performance-measurement
  liveness-profile name name1
  probe
    flow-label from 0 to 1000000 increment 10
  !
!
!
```

### Verification

The following example shows an SR-policy configured with flow labels:

```
Router#show performance-measurement sr-policy liveness color 1001 detail verbose private
Mon Oct 30 15:25:55.241 EDT
```

---

0/1/CPU0

---

```
SR Policy name: srte_c_1001_ep_3::1
Color                : 1001
SRv6 Encap Source Address : 1::1
Endpoint              : 3::1
Handle                : 0x00000000
Policy to be deleted   : False
Number of candidate-paths : 1

Candidate-Path:
Instance              : 3
Preference            : 300
Protocol-origin       : Configured
Discriminator         : 300
Profile Keys:
  Profile name        : profile-scale
  Profile type        : Generic Liveness Detection
Candidate path to be deleted: False
Source address        : 1::1
Local label           : Not set
Fast notification for session down: Disabled
  No fast notifications have been sent
Number of segment-lists : 2
Liveness Detection: Enabled
  Mininum SL Up Required: 2
  Session State: Up
  Last State Change Timestamp: Oct 26 2023 15:31:43.478
  Missed count: 0

Segment-List          : sl-1041
  fccc:cc00:1:fe10:: (Local Adjacency SID)
  fccc:cc00:2:fe41::/64
  Format: f3216
  Segment List ID: 0
  Reverse path segment-List: Not configured
  Segment-list to be deleted: False
  Number of atomic paths : 2
```

```

Liveness Detection: Enabled
  Session State: Up
  Last State Change Timestamp: Oct 26 2023 15:31:43.478
  Missed count: 0

```

```

Atomic path:
  Flow Label           : 0
  Session ID            : 4178
  Trace ID              : 280178832
  Atomic path to be deleted: False
  NPU Offloaded session : False
  Timestamping Enabled  : True
  Liveness Detection: Enabled
    Session State: Up
    Last State Change Timestamp: Oct 26 2023 15:31:43.478
    Missed count: 0
  Responder IP         : 1::1
  Number of Hops       : 3

```

```

Atomic path:
  Flow Label           : 10
  Session ID            : 4179
  Trace ID              : 1866227171
  Atomic path to be deleted: False
  NPU Offloaded session : False
  Timestamping Enabled  : True
  Liveness Detection: Enabled
    Session State: Up
    Last State Change Timestamp: Oct 26 2023 15:31:43.478
    Missed count: 0
  Responder IP         : 1::1
  Number of Hops       : 3

```

```

Segment-List           : sl-scale
fccc:cc00:1:fe10:: (Local Adjacency SID)
fccc:cc00:2:fed1::/64
  Format: f3216
Segment List ID: 0
Reverse path segment-List: Not configured
Segment-list to be deleted: False
Number of atomic paths : 2
Liveness Detection: Enabled
  Session State: Up
  Last State Change Timestamp: Oct 26 2023 15:31:43.478
  Missed count: 0

```

```

Atomic path:
  Flow Label           : 0
  Session ID            : 4180
  Trace ID              : 2609815826
  Atomic path to be deleted: False
  NPU Offloaded session : False
  Timestamping Enabled  : True
  Liveness Detection: Enabled
    Session State: Up
    Last State Change Timestamp: Oct 26 2023 15:31:43.478
    Missed count: 0
  Responder IP         : 1::1
  Number of Hops       : 3

```

```

Atomic path:
  Flow Label           : 10
  Session ID            : 4181
  Trace ID              : 170501506

```



```

Atomic path to be deleted: False
NPU Offloaded session : False
Timestamping Enabled : True
Liveness Detection: Enabled
  Session State: Up
  Last State Change Timestamp: Oct 26 2023 15:31:43.478
  Missed count: 0
Responder IP           : 1::1
Number of Hops         : 3

```

---

0/RSP0/CPU0

---

## Delay Measurement

Delay measurement is a mechanism used to measure the latency or delay experienced by data packets when they traverse a network.

The PM for delay measurement uses the IP/UDP packet format defined in RFC 5357 (TWAMP-Light) for probes. Two-Way Active Measurement Protocol (TWAMP) adds two-way or round-trip measurement capabilities. TWAMP employs time stamps applied at the echo destination (reflector) to enable greater accuracy. In the case of TWAMP Light, the Session-Reflector doesn't necessarily know about the session state. The Session-Reflector simply copies the Sequence Number of the received packet to the Sequence Number field of the reflected packet. The controller receives the reflected test packets and collects two-way metrics. This architecture allows for collection of two-way metrics.

### Benefits

- **Network Troubleshooting:** You can quickly and easily identify areas in your network with high delay and resolve network problems using delay measurement.
- **Network Planning and Optimization:** You can easily understand the performance of your network under various conditions and design a network that can handle expected traffic loads.
- **Quality of Service (QoS):** You can ensure quality of service standards are being met by continuously monitoring the delay in your network.

### Supported Delay Measurement Methods

You can measure delay using the following methods:

- Use to monitor delay experienced by data packets in a single link or path between two nodes in a network.
- Use to monitor the amount of time it takes for a data packet to travel from a source device to a specific IP endpoint within a network.
- Use to monitor the end-to-end delay experienced by the traffic sent over an SR policy.

## Measurement Modes

The following table compares the different hardware and timing requirements for the measurement modes supported in SR PM.

Feature Name	Release	Description
SR Performance Measurement: Loopback Measurement Mode	Release 7.5.2	<p>Loopback measurement mode provides two-way and one-way measurements. PTP-capable hardware and hardware timestamping are required on the Sender but are not required on the Reflector.</p> <p>Liveness monitoring uses "self-addressed" PM IP packets crafted by the sender (where the destination address is the sender's own IP address); this mode of operation is referred as "loopback mode".</p>

Table 7: Measurement Mode Requirements

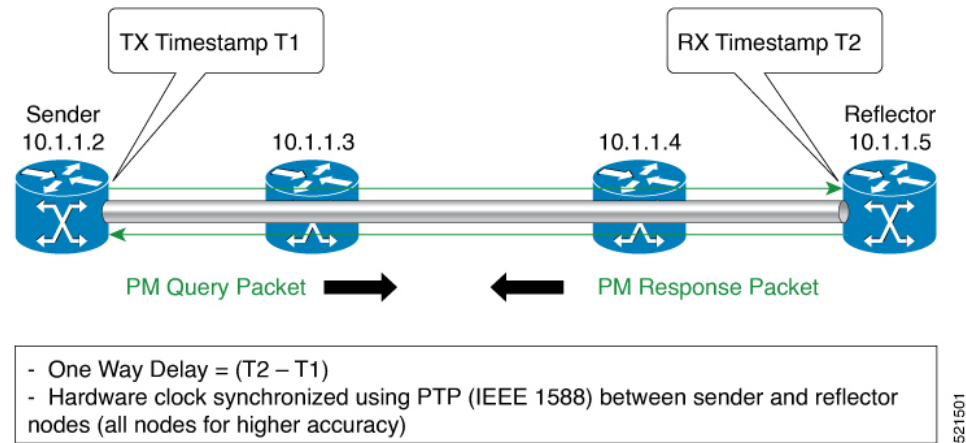
Measurement Mode	Sender: PTP-Capable HW and HW Timestamping	Reflector: PTP-Capable HW and HW Timestamping	PTP Clock Synchronization between Sender and Reflector
One-way	Required	Required	Required
Two-way	Required	Required	Not Required
Loopback	Required	Not Required	Not Required

### One-Way Measurement Mode

One-way measurement mode provides the most precise form of one-way delay measurement. PTP-capable hardware and hardware timestamping are required on both Sender and Reflector, with PTP Clock Synchronization between Sender and Reflector.

Delay measurement in one-way mode is calculated as  $(T2 - T1)$ .

Figure 3: One-Way



The PM query and response for one-way delay measurement can be described in the following steps:

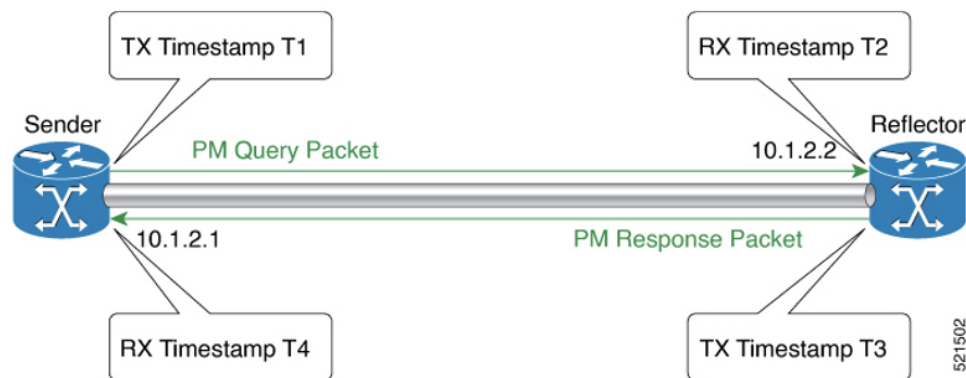
1. The local-end router sends PM query packets periodically to the remote side once the egress line card on the router applies timestamps on packets.
2. The ingress line card on the remote-end router applies time-stamps on packets as soon as they are received.
3. The remote-end router sends the PM packets containing time-stamps back to the local-end router.
4. One-way delay is measured using the time-stamp values in the PM packet.

**Two-Way Measurement Mode**

Two-way measurement mode provides two-way measurements. PTP-capable hardware and hardware timestamping are required on both Sender and Reflector, but PTP clock synchronization between Sender and Reflector is not required.

Delay measurement in two-way mode is calculated as  $((T4 - T1) - (T3 - T2))/2$ .

Figure 4: Two-Way



The PM query and response for two-way delay measurement can be described in the following steps:

1. The local-end router sends PM query packets periodically to the remote side once the egress line card on the router applies timestamps on packets.

2. Ingress line card on the remote-end router applies time-stamps on packets as soon as they are received.
3. The remote-end router sends the PM packets containing time-stamps back to the local-end router. The remote-end router time-stamps the packet just before sending it for two-way measurement.
4. The local-end router time-stamps the packet as soon as the packet is received for two-way measurement.
5. Delay is measured using the time-stamp values in the PM packet.

### Loopback Measurement Mode

Loopback measurement mode provides two-way and one-way measurements. PTP-capable hardware and hardware timestamping are required on the Sender, but are not required on the Reflector.

Delay measurements in Loopback mode are calculated as follows:

- Round-Trip Delay =  $(T4 - T1)$
- One-Way Delay = Round-Trip Delay/2

**Figure 5: Loopback**



The PM query and response for Loopback delay measurement can be described in the following steps:

1. The local-end router sends PM probe packets periodically on the SR Policy.
2. The probe packets are loopback on the endpoint node (not punted), with no timestamping on endpoint node.
3. Round-trip Delay =  $T4 - T1$ .

# Link Delay Measurement

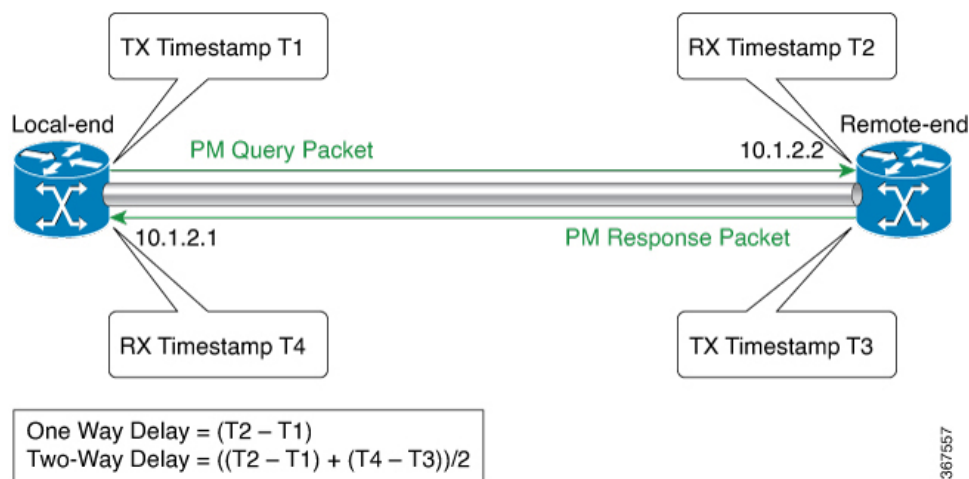
Table 8: Feature History Table

Feature Name	Release Information	Feature Description
Link Delay Measurement using TWAMP Light Encoding	Release 7.3.1	The PM for link delay uses the IP/UDP packet format defined in RFC 5357 (TWAMP-Light) for probes. Two-Way Active Measurement Protocol (TWAMP) adds two-way or round-trip measurement capabilities. TWAMP employs time stamps applied at the echo destination (reflector) to enable greater accuracy.

The PM for link delay uses the IP/UDP packet format defined in RFC 5357 (TWAMP-Light) for probes. Two-Way Active Measurement Protocol (TWAMP) adds two-way or round-trip measurement capabilities. TWAMP employs time stamps applied at the echo destination (reflector) to enable greater accuracy. In the case of TWAMP Light, the Session-Reflector doesn't necessarily know about the session state. The Session-Reflector simply copies the Sequence Number of the received packet to the Sequence Number field of the reflected packet. The controller receives the reflected test packets and collects two-way metrics. This architecture allows for collection of two-way metrics.

The following figure explains the PM query and response for link delay.

Figure 6: Performance Measurement for Link Delay



The PM query and response for link delay can be described in the following steps:

1. The local-end router sends PM query packets periodically to the remote side once the egress line card on the router applies timestamps on packets.
2. Ingress line card on the remote-end router applies time-stamps on packets as soon as they are received.
3. The remote-end router sends the PM packets containing time-stamps back to the local-end router. The remote-end router time-stamps the packet just before sending it for two-way measurement.

4. The local-end router time-stamps the packet as soon as the packet is received for two-way measurement.
5. One-way delay and optionally two-way delay is measured using the time-stamp values in the PM packet.

### Restrictions and Usage Guidelines for PM for Link Delay

The following restrictions and guidelines apply for the PM for link delay feature for different links.

- For LSPs, remote-end line card needs to be MPLS and multicast MAC address capable.
- For broadcast links, only point-to-point (P2P) links are supported. P2P configuration on IGP is required for flooding the value.
- For link bundles, the hashing function may select a member link for forwarding but the reply may come from the remote line card on a different member link of the bundle.
- For one-way delay measurement, clocks should be synchronized on two end-point nodes of the link using PTP.
- Link delay measurement is supported on IPv4 unnumbered interfaces. An IPv4 unnumbered interface is identified by a node ID (a loopback address) and the local SNMP index assigned to the interface. Note that the reply messages could be received on any interface, since the packets are routed at the responder based on the loopback address used to identify the link.

### Configuration Example: PM for Link Delay

This example shows how to configure performance-measurement functionalities for link delay as a global default profile. The default values for the different parameters in the PM for link delay is given as follows:

- **probe measurement mode:** The default measurement mode for probe is two-way delay measurement. If you are configuring one-way delay measurement, hardware clocks must be synchronized between the local-end and remote-end routers using precision time protocol (PTP).
- **protocol:** Interface delay measurement uses TWAMP-Light (RFC5357) with IP/UDP encaps.
- **tx-interval:** Interval for sending probe packet. The default value is 3000000 microseconds and the range is from 3300 to 15000000 microseconds.
- **computation interval:** Interval for metric computation. Default is 30 seconds; range is 1 to 3600 seconds.
- **periodic advertisement:** Periodic advertisement is enabled by default.
- **periodic-advertisement interval:** The default value is 120 seconds and the interval range is from 30 to 3600 seconds.
- **periodic-advertisement threshold:** The default value of periodic advertisement threshold is 10 percent.
- **periodic-advertisement minimum change:** The default value is 1000 microseconds (usec) and the range is from 0 to 10000 microseconds.
- **accelerated advertisement:** Accelerated advertisement is disabled by default.
- **accelerated-advertisement threshold:** The default value is 20 percent and the range is from 0 to 100 percent.
- **accelerated-advertisement minimum change:** The default value is 1000 microseconds and the range is from 1 to 100000 microseconds.

```
RP/0/0/CPU0:router(config)# performance-measurement delay-profile interfacesdefault
RP/0/0/CPU0:router(config-pm-dm-intf)# probe
RP/0/0/CPU0:router(config-pm-dm-intf-probe)# measurement-mode one-way
RP/0/0/CPU0:router(config-pm-dm-intf-probe)# tx-interval 30000
RP/0/0/CPU0:router(config-pm-dm-intf-probe)# exit

RP/0/0/CPU0:router(config-pm-dm-intf)# advertisement periodic
RP/0/0/CPU0:router(config-pm-dm-intf-adv-per)# interval 120
RP/0/0/CPU0:router(config-pm-dm-intf-adv-per)# threshold 20
RP/0/0/CPU0:router(config-pm-dm-intf-adv-per)# minimum-change 1000
RP/0/0/CPU0:router(config-pm-dm-intf-adv-per)# exit

RP/0/0/CPU0:router(config-pm-dm-intf)# advertisement accelerated
RP/0/0/CPU0:router(config-pm-dm-intf-adv-acc)# threshold 30
RP/0/0/CPU0:router(config-pm-dm-intf-adv-acc)# minimum-change 1000
RP/0/0/CPU0:router(config-pm-dm-intf-adv-per)# exit
```

### Configure the UDP Destination Port

Configuring the UDP port for TWAMP-Light protocol is optional. By default, PM uses port 862 as the TWAMP-reserved UDP destination port.

The UDP port is configured for each PM measurement probe type (delay, loss, protocol, authentication mode, etc.) on querier and responder nodes. If you configure a different UDP port, the UDP port for each PM measurement probe type must match on the querier and the responder nodes.



**Note** The same UDP destination port is used for delay measurement for links and SR Policy.

This example shows how to configure the UDP destination port.

```
Router(config)# performance-measurement
Router(config-perf-meas)# protocol twamp-light
Router(config-pm-protocol)# measurement delay unauthenticated
Router(config-pm-proto-mode)# querier-dst-port 12000
```

### Enable PM for Link Delay Over an Interface

This example shows how to enable PM for link delay over an interface.

```
RP/0/0/CPU0:router(config)# performance-measurement
RP/0/0/CPU0:router(config-perf-meas)# interface TenGigE0/0/0/0
RP/0/0/CPU0:router(config-pm-intf)# delay-measurement
RP/0/0/CPU0:router(config-pm-dm-intf)# exit
```

### Verification

```
RP/0/0/CPU0:router# show performance-measurement profile default interface
Thu Dec 12 14:13:16.029 PST
```

```
-----
0/0/CPU0
-----
```

```
Interface Delay-Measurement:
```

```

Profile configuration:
  Measurement Type                : Two-Way
  Probe computation interval       : 30 (effective: 30) seconds
  Type of services                 : Traffic Class: 6, DSCP: 48
  Burst interval                   : 3000 (effective: 3000) mSec
  Burst count                      : 10 packets
  Encap mode                       : UDP
  Payload Type                     : TWAMP-light
  Destination sweeping mode        : Disabled
  Periodic advertisement           : Enabled
    Interval                       : 120 (effective: 120) sec
    Threshold                       : 10%
    Minimum-Change                  : 500 uSec
  Advertisement accelerated        : Disabled
  Threshold crossing check         : Minimum-delay

```

```
RP/0/0/CPU0:router# show performance-measurement summary detail location 0/2/CPU0
```

```
Thu Dec 12 14:09:59.162 PST
```

```
-----
0/2/CPU0
-----
```

```

Total interfaces                : 1
Total SR Policies               : 0
Total RSVP-TE tunnels           : 0
Total Maximum PPS               : 2000 pkts/sec
Total Interfaces PPS            : 0 pkts/sec
Maximum Allowed Multi-hop PPS   : 2000 pkts/sec
Multi Hop Requested PPS         : 0 pkts/sec (0% of max allowed)
Dampened Multi Hop Requested PPS : 0% of max allowed
Inuse Burst Interval Adjustment Factor : 100% of configuration

```

```
Interface Delay-Measurement:
```

```

Total active sessions           : 1
Counters:
  Packets:
    Total sent                   : 26
    Total received                : 26
  Errors:
    TX:
      Reason interface down      : 0
      Reason no MPLS caps        : 0
      Reason no IP address       : 0
      Reason other                : 0
    RX:
      Reason negative delay      : 0
      Reason delay threshold exceeded : 0
      Reason missing TX timestamp : 0
      Reason missing RX timestamp : 0
      Reason probe full          : 0
      Reason probe not started   : 0
      Reason control code error  : 0
      Reason control code notif  : 0
  Probes:
    Total started                 : 3
    Total completed               : 2
    Total incomplete              : 0
    Total advertisements         : 0

```

```
SR Policy Delay-Measurement:
```

```

Total active sessions           : 0
Counters:

```



```

Packets:
  Total sent                : 0
  Total received            : 0
Errors:
  TX:
    Reason interface down   : 0
    Reason no MPLS caps     : 0
    Reason no IP address    : 0
    Reason other             : 0
  RX:
    Reason negative delay   : 0
    Reason delay threshold exceeded : 0
    Reason missing TX timestamp : 0
    Reason missing RX timestamp : 0
    Reason probe full       : 0
    Reason probe not started : 0
    Reason control code error : 0
    Reason control code notif : 0
Probes:
  Total started             : 0
  Total completed           : 0
  Total incomplete          : 0
  Total advertisements      : 0

RSVP-TE Delay-Measurement:
Total active sessions      : 0
Counters:
  Packets:
    Total sent              : 0
    Total received          : 0
  Errors:
    TX:
      Reason interface down : 0
      Reason no MPLS caps   : 0
      Reason no IP address  : 0
      Reason other           : 0
    RX:
      Reason negative delay  : 0
      Reason delay threshold exceeded : 0
      Reason missing TX timestamp : 0
      Reason missing RX timestamp : 0
      Reason probe full     : 0
      Reason probe not started : 0
      Reason control code error : 0
      Reason control code notif : 0
  Probes:
    Total started           : 0
    Total completed         : 0
    Total incomplete        : 0
    Total advertisements    : 0

Global Delay Counters:
  Total packets sent        : 26
  Total query packets received : 26
  Total invalid session id  : 0
  Total missing session     : 0
  
```

```

RP/0/0/CPU0:router# show performance-measurement interfaces detail
Thu Dec 12 14:16:09.692 PST
  
```

```

-----
0/0/CPU0
-----
  
```

0/2/CPU0

```

-----
Interface Name: GigabitEthernet0/2/0/0 (ifh: 0x1004060)
Delay-Measurement           : Enabled
Loss-Measurement           : Disabled
Configured IPv4 Address    : 10.10.10.2
Configured IPv6 Address    : 10:10:10::2
Link Local IPv6 Address    : fe80::3a:6fff:fec9:cd6b
Configured Next-hop Address : Unknown
Local MAC Address          : 023a.6fc9.cd6b
Next-hop MAC Address       : 0291.e460.6707
Primary VLAN Tag           : None
Secondary VLAN Tag         : None
State                       : Up

Delay Measurement session:
  Session ID                : 1

Last advertisement:
  Advertised at: Dec 12 2019 14:10:43.138 (326.782 seconds ago)
  Advertised reason: First advertisement
  Advertised delays (uSec): avg: 839, min: 587, max: 8209, variance: 297

Next advertisement:
  Threshold check scheduled in 1 more probe (roughly every 120 seconds)
  Aggregated delays (uSec): avg: 751, min: 589, max: 905, variance: 112
  Rolling average (uSec): 756

Current Probe:
  Started at Dec 12 2019 14:15:43.154 (26.766 seconds ago)
  Packets sent: 9, received: 9, lost: 0
  Measured delays (uSec): avg: 795, min: 631, max: 1199, variance: 164
  Next probe scheduled at Dec 12 2019 14:16:13.132 (in 3.212 seconds)
  Next burst packet will be sent in 0.212 seconds
  Burst packet sent every 3.0 seconds
  Probe samples:
    Packet Rx Timestamp      Measured Delay (nsec)
    Dec 12 2019 14:15:43.156  689223
    Dec 12 2019 14:15:46.156  876561
    Dec 12 2019 14:15:49.156  913548
    Dec 12 2019 14:15:52.157  1199620
    Dec 12 2019 14:15:55.156  794008
    Dec 12 2019 14:15:58.156  631437
    Dec 12 2019 14:16:01.157  656440
    Dec 12 2019 14:16:04.157  658267
    Dec 12 2019 14:16:07.157  736880

```

You can also use the following commands for verifying the PM for link delay on the local-end router.

Command	Description
<b>show performance-measurement history probe interfaces</b> [ <i>interface</i> ]	Displays the PM link-delay probe history for interfaces.
<b>show performance-measurement history aggregated interfaces</b> [ <i>interface</i> ]	Displays the PM link-delay aggregated history for interfaces.
<b>show performance-measurement history advertisement interfaces</b> [ <i>interface</i> ]	Displays the PM link-delay advertisement history for interfaces.

Command	Description
<b>show performance-measurement counters</b> [interface <i>interface</i> ] [location <i>location-name</i> ]	Displays the PM link-delay session counters.

You can also use the following commands for verifying the PM for link-delay configuration on the remote-end router.

Command	Description
<b>show performance-measurement responder summary</b> [location <i>location-name</i> ]	Displays the PM for link-delay summary on the remote-end router (responder).
<b>show performance-measurement responder interfaces</b> [ <i>interface</i> ]	Displays PM for link-delay for interfaces on the remote-end router.
<b>show performance-measurement responder counters</b> [interface <i>interface</i> ] [location <i>location-name</i> ]	Displays the PM link-delay session counters on the remote-end router.

### SR Performance Measurement Named Profiles

Feature	Release Information	Feature Description
SR Performance Measurement Named Profiles	Release 7.5.2	<p>You can use this feature to create specific performance measurement delay and liveness profiles, and associate it with an SR policy.</p> <p>You can use the delay or liveness profile to be associated with an interface or network area, where the performance management probes are enabled, and performance measurement is precise and enhanced.</p>

You can create a named performance measurement profile for delay or liveness.

#### Delay Profile

This example shows how to create a named SR performance measurement delay profile.

```

Router(config)# performance-measurement delay-profile name profile2
Router(config-pm-dm-profile)# probe
Router(config-pm-dm-probe)# tx-interval 60000
Router(config-pm-dm-probe)# computation-interval 60
Router(config-pm-dm-probe)# protocol twamp-light
Router(config-pm-dm-probe)# tos dscp 63
Router(config-pm-dm-probe)# exit

Router(config-pm-dm-profile)# advertisement
Router(config-pm-dm-adv)# periodic
Router(config-pm-dm-adv-per)# interval 60
Router(config-pm-dm-adv-per)# minimum-change 1000

```

```
Router(config-pm-dm-adv-per)# threshold 20
Router(config-pm-dm-adv-per)# commit
```

Apply the delay profile for an SR Policy.

```
Router(config)# segment-routing traffic-eng
Router(config-sr-te)# policy TEST
Router(config-sr-te-policy)# color 4 end-point ipv4 10.10.10.10
Router(config-sr-te-policy)# performance-measurement
Router(config-sr-te-policy-perf-meas)# delay-measurement delay-profile name profile2
```

```
Router(config-sr-te-policy)#candidate-paths
Router(config-sr-te-policy-path)#preference 100
Router(config-sr-te-policy-path-pref)#explicit segment-list LIST1
Router(config-sr-te-pp-info)#weight 2
```

```
Router(config-sr-te-policy-path-pref)#explicit segment-list LIST2
Router(config-sr-te-pp-info)#weight 3
```

### Running Configuration

```
Router# show run segment-routing traffic-eng policy TEST
```

```
segment-routing
traffic-eng
  policy TEST
    color 4 end-point ipv4 10.10.10.10
    candidate-paths
      preference 100
      explicit segment-list LIST1
      weight 2
    !
    explicit segment-list LIST2
      weight 3
    !
  !
  performance-measurement
    delay-measurement
      delay-profile name profile2
```

### Verification

```
Router# show performance-measurement profile named-profile delay
```

```
-----
0/RSP0/CPU0
-----
SR Policy Delay Measurement Profile Name: profile2
Profile configuration:
  Measurement mode           : One-way
  Protocol type              : TWAMP-light
  Encap mode                 : UDP
  Type of service:
    PM-MPLS traffic class    : 6
    TWAMP-light DSCP         : 63
  Probe computation interval  : 60 (effective: 60) seconds
  Burst interval             : 60 (effective: 60) mSec
  Packets per computation interval : 1000
  Periodic advertisement     : Enabled
    Interval                 : 60 (effective: 60) sec
    Threshold                 : 20%
    Minimum-change           : 1000 uSec
  Advertisement accelerated  : Disabled
  Advertisement logging:
    Delay exceeded           : Disabled (default)
```

```

Threshold crossing check           : Maximum-delay
Router alert                       : Disabled (default)
Destination sweeping mode         : Disabled
Liveness detection parameters:
Multiplier                         : 3
Logging state change              : Disabled

```

### On-Demand SR Policy

```

Router(config-sr-te)# on-demand color 20
Router(config-sr-te-color)# performance-measurement delay-measurement
Router(config-sr-te-color-delay-meas)# delay-profile name profile2
Router(config-sr-te-color-delay-meas)# commit

```

### Running Configuration

```

Router# show run segment-routing traffic-eng on-demand color 20

segment-routing
 traffic-eng
  on-demand color 20
  performance-measurement
  delay-measurement
  delay-profile name profile2

```

### Liveness Profile

This example shows how to create a *named* SR performance measurement liveness profile.

```

Router(config)# performance-measurement liveness-profile name profile3
Router(config-pm-ld-profile)# probe
Router(config-pm-ld-probe)# tx-interval 60000
Router(config-pm-ld-profile)# probe
Router(config-pm-ld-probe)# tx-interval 60000
Router(config-pm-ld-probe)# tos dscp 10
Router(config-pm-ld-probe)# exit

Router(config-pm-ld-profile)# liveness-detection
Router(config-pm-ld-profile-ld)# multiplier 5
Router(config-pm-ld-profile-ld)# commit

```

### Apply the liveness profile for the SR policy

This example shows how to enable PM for SR policy liveness for a specific policy.

For the same policy, you cannot enable delay-measurement (delay-profile) and liveness-detection (liveness-profile) at the same time. For example, if delay measurement is enabled, use the **no delay-measurement** command to disable it, and then enable the following command for enabling liveness detection.

```

Router(config)# segment-routing traffic-eng
Router(config-sr-te)# policy TRST2
Router(config-sr-te-policy)# color 40 end-point ipv4 20.20.20.20
Router(config-sr-te-policy)#candidate-paths
Router(config-sr-te-policy-path)#preference 50
Router(config-sr-te-policy-path-pref)#explicit segment-list LIST3
Router(config-sr-te-pp-info)#weight 2

Router(config-sr-te-policy-path-pref)#explicit segment-list LIST4
Router(config-sr-te-pp-info)#weight 3

Router(config-sr-te-policy)# performance-measurement
Router(config-sr-te-policy-perf-meas)# liveness-detection liveness-profile name profile3

```

### Running Configuration

```

Router# show run segment-routing traffic-eng policy TRST2

segment-routing
traffic-eng
policy TRST2
color 40 end-point ipv4 20.20.20.20
candidate-paths
preference 50
explicit segment-list LIST3
weight 2
!
explicit segment-list LIST4
weight 3
!
!
!
performance-measurement
liveness-detection
liveness-profile name profile3
!

```

### Verification

```
Router# show performance-measurement profile named-profile delay sr-policy
```

```

-----
0/RSP0/CPU0
-----

SR Policy Liveness Detection Profile Name: profile1
Profile configuration:
  Measurement mode           : Loopback
  Protocol type              : TWAMP-light
  Type of service:
    TWAMP-light DSCP         : 10
  Burst interval             : 60 (effective: 60) mSec
  Destination sweeping mode  : Disabled
  Liveness detection parameters:
    Multiplier                : 3
    Logging state change      : Disabled

SR Policy Liveness Detection Profile Name: profile3
Profile configuration:
  Measurement mode           : Loopback
  Protocol type              : TWAMP-light
  Type of service:
    TWAMP-light DSCP         : 10
  Burst interval             : 60 (effective: 60) mSec
  Destination sweeping mode  : Disabled
  Liveness detection parameters:
    Multiplier                : 3
    Logging state change      : Disabled

```

### On-Demand SR Policy

For the same policy, you cannot enable delay-measurement (delay-profile) and liveness-detection (liveness-profile) at the same time. For example, to disable delay measurement, use the **no delay-measurement** command, and then enable the following command for enabling liveness detection.

```

Router(config-sr-te)#on-demand color 30
Router(config-sr-te-color)#performance-measurement
Router(config-sr-te-color-pm)# liveness-detection liveness-profile name profile1
Router(config-sr-te-color-delay-meas)# commit

```

### Running Configuration

```
Router# show run segment-routing traffic-eng on-demand color 30

segment-routing
 traffic-eng
  on-demand color 30
  performance-measurement
  liveness-detection
  liveness-profile name profile1
  !
```

### Verification

```
Router# show performance-measurement profile named-profile liveness

-----
0/RSP0/CPU0
-----
SR Policy Liveness Detection Profile Name: profile1
Profile configuration:
  Measurement mode           : Loopback
  Protocol type              : TWAMP-light
  Type of service:
    TWAMP-light DSCP         : 10
  Burst interval             : 60 (effective: 60) mSec
  Destination sweeping mode  : Disabled
  Liveness detection parameters:
    Multiplier                : 3
    Logging state change     : Disabled
```

## Delay Normalization

Performance measurement (PM) measures various link characteristics like packet loss and delay. Such characteristics can be used by IS-IS as a metric for Flexible Algorithm computation. Low latency routing using dynamic delay measurement is one of the primary use cases for Flexible Algorithm technology.

Delay is measured in microseconds. If delay values are taken as measured and used as link metrics during the IS-IS topology computation, some valid ECMP paths might be unused because of the negligible difference in the link delay.

The Delay Normalization feature computes a normalized delay value and uses the normalized value instead. This value is advertised and used as a metric during the Flexible Algorithm computation.

The normalization is performed when the delay is received from the delay measurement component. When the next value is received, it is normalized and compared to the previous saved normalized value. If the values are different, then the LSP generation is triggered.

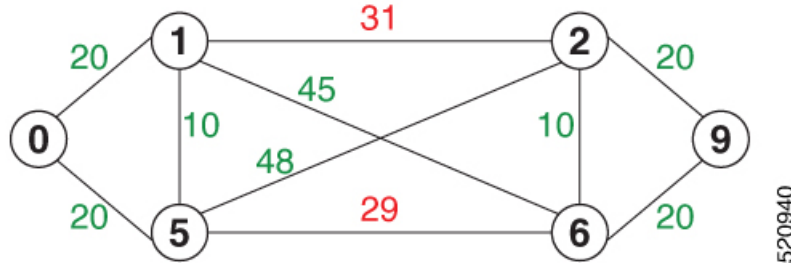
The following formula is used to calculate the normalized value:

- **Dm** – measured Delay
- **Int** – configured normalized Interval
- **Off** – configured normalized Offset (must be less than the normalized interval Int)
- **Dn** – normalized Delay
- **a** =  $Dm / Int$  (rounded down)
- **b** =  $a * Int + Off$

If the measured delay (**Dm**) is less than or equal to **b**, then the normalized delay (**Dn**) is equal to **b**. Otherwise, **Dn** is **b + Int**.

**Example**

The following example shows a low-latency service. The intent is to avoid high-latency links (1-6, 5-2). Links 1-2 and 5-6 are both low-latency links. The measured latency is not equal, but the difference is insignificant.



We can normalize the measured latency before it is advertised and used by IS-IS. Consider a scenario with the following:

- Interval = 10
- Offset = 3

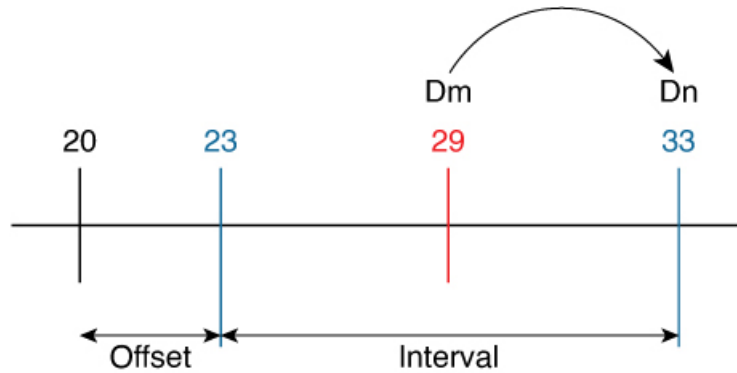
The measured delays will be normalized as follows:

- **Dm** = 29

**a** = 29 / 10 = 2 (2.9, rounded down to 2)

**b** = 2 \* 10 + 3 = 23

In this case, **Dm** (29) is greater than **b** (23); so **Dn** is equal to **b+I** (23 + 10) = **33**



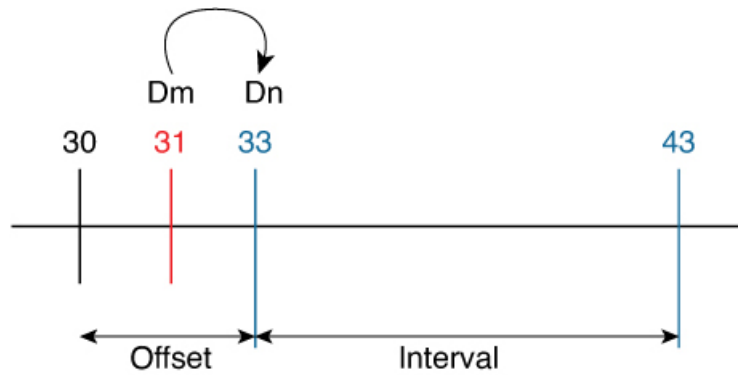
- **Dm** = 31

**a** = 31 / 10 = 3 (3.1, rounded down to 3)

**b** = 3 \* 10 + 3 = 33

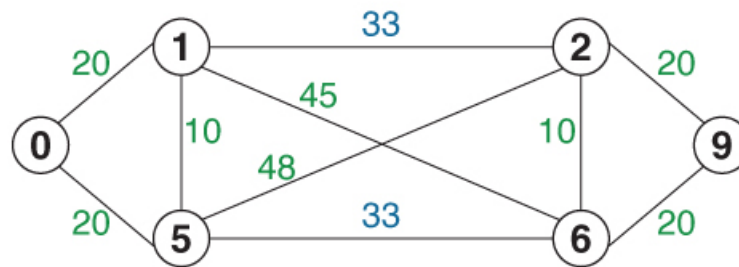
In this case, **Dm** (31) is less than **b** (33); so **Dn** is **b** = **33**





520943

The link delay between 1-2 and 5-6 is normalized to 33.



520941

**Configuration**

Delay normalization is disabled by default. To enable and configure delay normalization, use the **delay normalize interval interval [offset offset]** command.

- *interval* – The value of the normalize interval in microseconds.
- *offset* – The value of the normalized offset in microseconds. This value must be smaller than the value of normalized interval.

**IS-IS Configuration**

```
router isis 1
interface GigEth 0/0/0/0
  delay normalize interval 10 offset 3
address-family ipv4 unicast
metric 77
```

**OSPF Configuration**

```
router ospf 1
area 0
interface GigabitEthernet0/0/0/0
  delay normalize interval 10 offset 3
!
```

## Link Anomaly Detection with IGP Penalty

Table 9: Feature History Table

Feature Name	Release Information	Feature Description
Link Anomaly Detection with IGP Penalty	Release 7.4.1	This feature allows you to define thresholds above the measured delay that is considered “anomalous” or unusual. When this threshold is exceeded, an anomaly (A) bit/flag is set along with link delay attribute that is sent to clients.

Customers might experience performance degradation issues, such as increased latency or packet loss on a link. Degraded links might be difficult to troubleshoot and can affect applications, especially in cases where traffic is sent over multiple ECMP paths where one of those paths is degraded.

The Anomaly Detection feature allows you to define a delay anomaly threshold to identify unacceptable link delays. Nodes monitor link performance using link delay monitoring probes. The measured value is compared against the delay anomaly threshold values. When the upper bound threshold is exceeded, the link is declared “abnormal”, and performance measurement sets an anomaly bit (A-bit). When IGP receives the A-bit, IGP can automatically increase the IGP metric of the link by a user-defined amount to make this link undesirable or unusable. When the link recovers (lower bound threshold), PM resets the A-bit.

### Usage Guidelines and Limitations

This feature is not active when narrow metrics are configured because the performance measurement advertisement requires the “wide” metric type length values.

### Configuration Example

The following example shows how to configure the upper and lower anomaly thresholds. The range for *upper\_bound* and *lower\_bound* is from 1 to 200,000 microseconds. The *lower\_bound* value must be less than the *upper\_bound* value.

```
RP/0/0/CPU0:router(config)# performance-measurement delay-profile interfaces default
RP/0/0/CPU0:router(config-pm-dm-intf)# advertisement
RP/0/0/CPU0:router(config-pm-dm-intf-adv)# anomaly-check upper-bound 5000 lower-bound 1000
RP/0/0/CPU0:router(config-pm-dm-intf-adv)# commit
```

### Running Configuration

```
performance-measurement
 delay-profile interfaces default
  advertisement
  anomaly-check
  upper-bound 5000 lower-bound 1000
  !
!
!
end
```

## Delay Measurement for IP Endpoint

Table 10: Feature History Table

Feature Name	Release Information	Feature Description
Delay Measurement for IP Endpoint over SRv6 Network	Release 24.2.11	<p>In Segment Routing over an IPv6 network (SRv6), you can measure packet delay from the source to a specific IP endpoint. You can use this information for troubleshooting, network maintenance, and optimizing network performance.</p> <p>Additionally, you can use flow labels to verify the delay of each subsequent hop path towards the IP endpoint of that path. So that, when network traffic is distributed across multiple available paths towards an IP endpoint, delay measurement tracks the delay of each of these paths towards the IP endpoint.</p> <p>The feature introduces these changes:</p> <p><b>CLI:</b></p> <ul style="list-style-type: none"> <li>• The <b>source-address ipv6</b> keyword is introduced in the <b>performance-measurement endpoint</b> command.</li> <li>• The <b>segment-list name</b> keyword is introduced in the <b>segment-routing traffic-eng explicit</b> command.</li> <li>• The <b>flow-label</b> keyword is introduced in the <b>performance-measurement delay-profile name</b> command.</li> </ul> <p><b>YANG Data Model:</b></p> <ul style="list-style-type: none"> <li>• <code>Cisco-IOS-XR-unperformance-measurement-cfg</code></li> <li>• <code>Cisco-IOS-XR-perf-meas-oper.yang</code></li> </ul> <p>(See <a href="#">GitHub</a>, <a href="#">YANG Data Models Navigator</a>)</p>

Feature Name	Release Information	Feature Description
IP Endpoint Delay Measurement Monitoring	Release 7.4.1	This feature measures the end-to-end delay and monitors liveness of a specified IP endpoint node, including VRF-aware (awareness of multiple customers belonging to different VRFs).  This feature is supported on IPv4, IPv6, and MPLS data planes.

Delay for an IP endpoint is the amount of time it takes for a data packet to travel from a source device to a specific IP endpoint within a network.

To measure a delay for a packet, also called a probe, is sent from a source device to the target IP endpoint.

The time from when the packet leaves the source to when it arrives at the endpoint is measured and recorded as the delay.

You can measure one-way delay, Two-way delay, and Roundtrip delay or delay in loop-back mode. For more information on Delay measurement, see Link Delay Measurement and Measurement Modes.

### Collecting IP Endpoint Probe Statistics

- Statistics associated with the probe for delay metrics are available via Histogram and Streaming Telemetry.
- Model Driven Telemetry (MDT) is supported for the following data:
  - Summary, endpoint, session, and counter show command bags.
  - History buffers data
- Model Driven Telemetry (MDT) and Event Driven Telemetry (EDT) are supported for the following data:
  - Delay metrics computed in the last probe computation-interval (event: probe-completed)
  - Delay metrics computed in the last aggregation-interval; that is, end of the periodic advertisement-interval (event: advertisement-interval expired)
  - Delay metrics last notified (event: notification-triggered)
- The following xpaths for MDT/EDT is supported:
  - `Cisco-IOS-XR-perf-meas-oper:performance-measurement/nodes/node/endpoints/endpoint-delay/endpoint-last-probes`
  - `Cisco-IOS-XR-perf-meas-oper:performance-measurement/nodes/node/endpoints/endpoint-delay/endpoint-last-aggregations`
  - `Cisco-IOS-XR-perf-meas-oper:performance-measurement/nodes/node/endpoints/endpoint-delay/endpoint-last-advertisements`

### Supported Features

- IPv6 Endpoint Delay in Default VRF (over SRv6)
- SRv6 Endpoint Delay in Default VRF (Endpoint can be Node SID, Flex-Algo SID, Packed uSID carrier)
- IPv6 Endpoint Delay in VRF (static uDT6)
- IPv6 Endpoint Delay in VRF (dynamic uDT6 encap)
- IPv4 Endpoint Delay in VRF or GRT (static uDT4)
- IPv4 Endpoint Delay in VRF or GRT (dynamic uDT4 encap)

### Guidelines and Limitations

You can specify a custom labeled path through one or more user-configured segment-lists. User-configured segment-list represents the forwarding path from sender to reflector when the probe is configured in delay-measurement mode.

- SR PM is supported on hardware that supports Precision Time Protocol (PTP). This requirement applies to both one-way and two-way delay measurement.  
  
See the "**Configuring Precision Time Protocol**" chapter in the *System Management Configuration Guide for Cisco 8000 Series Routers* for Restrictions for PTP and the Timing Hardware Support Matrix.
- Examples of the custom segment-list include:
  - Probe in delay-measurement mode with a segment-list that includes Flex-Algo prefix SID of the endpoint
  - Probe in delay-measurement mode with a segment-list that includes a SID-list with labels to reach the endpoint or the sender (forward direction)
  - Probe in delay-measurement mode with a segment-list that includes BSID associated with SR policy to reach the end point.
- Endpoint segment list configuration is not supported under nondefault VRF.
- Liveness session without segment list for an endpoint in a non-default VRF is not supported.
- SR Performance Measurement endpoint session over BVI interface is not supported.

## IP Endpoint Liveness Detection in an SR MPLS Network

IP endpoint liveness detection leverages the loopback measurement-mode. The following workflow describes the sequence of events.

1. The sender creates and transmits the PM probe packets.  
The IP destination address (DA) on the probe packets is set to the loopback value of the sender itself.  
The transmit timestamp (T1) is added to the payload.  
The probe packet is encapsulated with the label corresponding to the endpoint.
2. The network delivers the PM probe packets following the LSP toward the endpoint.
3. The end-point receives the PM probe packets.

Packets are forwarded back to the sender based on the forwarding entry associated with the IP DA of the PM probe packet. If an LSP exists, the probe packet is encapsulated with the label of the sender.

#### 4. The sender node receives the PM probe packets.

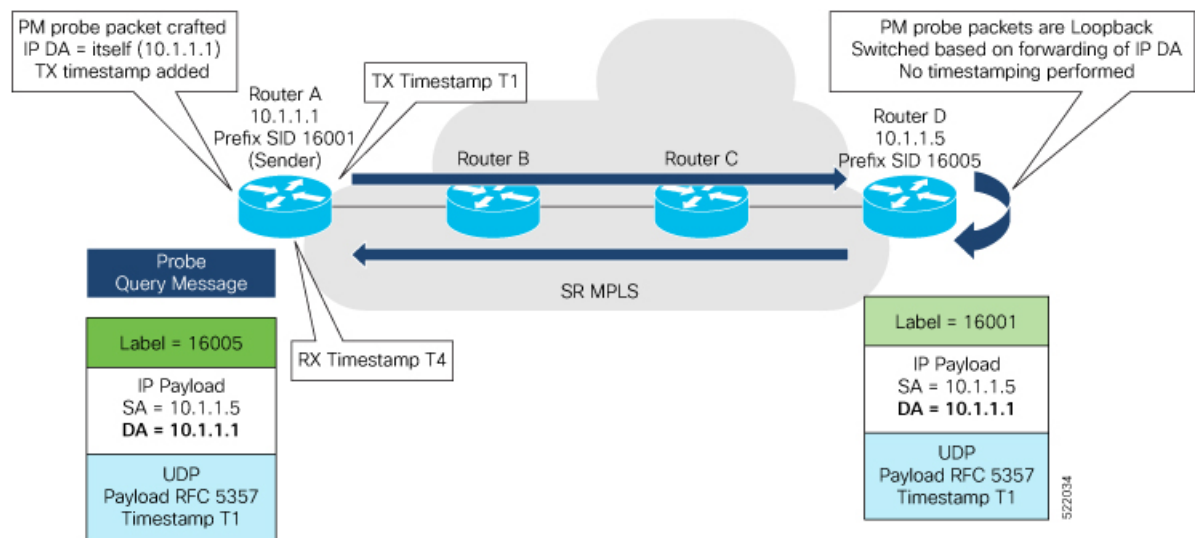
The received timestamp (T4) stored.

If the sender node doesn't receive the specified number of probe packets (based on the configured multiplier), the sender node declares the PM session as down.

The following figure illustrates a liveness detection probe toward an IP endpoint learned by the IGP. The network interconnecting the sender and reflector provides MPLS connectivity with Segment Routing.

The liveness detection multiplier is set to 5 to specify the number of consecutive missed probe packets before the PM session is declared as down.

**Figure 7: IP Endpoint Liveness Detection**



### Configuration Example

```
RouterA(config)# performance-measurement
RouterA(config-perf-meas)# endpoint ipv4 1.1.1.5
RouterA(config-pm-ep)# source-address ipv4 1.1.1.1
RouterA(config-pm-ep)# liveness-detection
RouterA(config-pm-ep-ld)# exit
RouterA(config-pm-ep)# exit
RouterA(config-perf-meas)# liveness-profile endpoint default
RouterA(config-pm-ld-ep)# liveness-detection
RouterA(config-pm-ld-ep-ld)# multiplier 5
RouterA(config-pm-ld-ep-ld)# exit
RouterA(config-pm-ld-ep)# probe
RouterA(config-pm-ld-ep-probe)# measurement-mode loopback
```

### Running Configuration

```
performance-measurement
 endpoint ipv4 1.1.1.5
 source-address ipv4 1.1.1.1
 liveness-detection
```

```

!
!
liveness-profile endpoint default
liveness-detection
  multiplier 5
!
probe
  measurement-mode loopback
!
!
end
    
```

**Verification**

```
RouterA# show performance-measurement endpoint ipv4 1.1.1.5
```

```
-----
0/RSP0/CPU0
-----
```

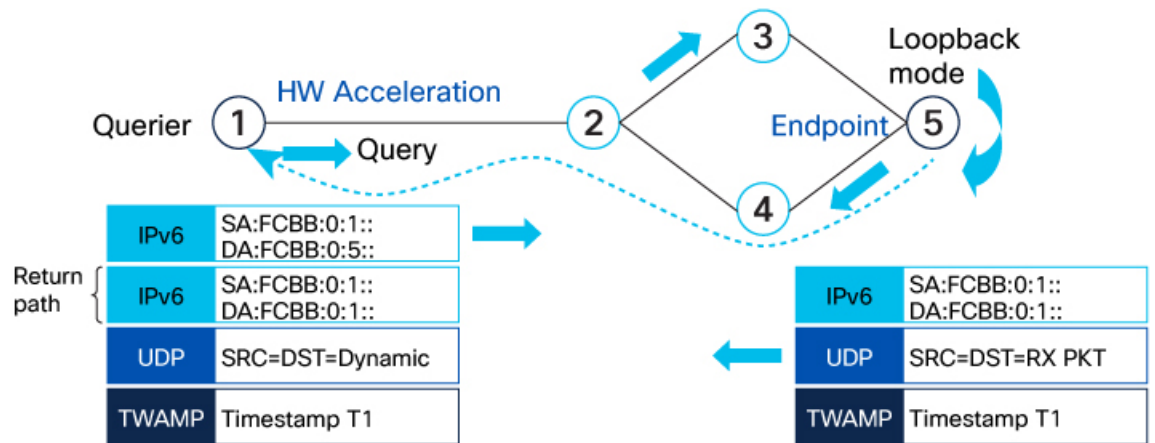
```

Endpoint name: IPv4-1.1.1.5-vrf-default
Source address      : 1.1.1.1
VRF name            : default
Liveness Detection  : Enabled
Profile Keys:
  Profile name      : default
  Profile type      : Endpoint Liveness Detection

Segment-list       : None
Session State: Down
Missed count: 0
    
```

**IP Endpoint Delay Measurement over SRv6 Network Usecase**

The following figure illustrates a delay measurement probe toward an IP endpoint over the SRv6 network. The network interconnecting the sender and the reflector provides plain IP connectivity.



```

RP/0/RSP0/CPU0:ios#configure
RP/0/RSP0/CPU0:ios(config)#performance-measurement
RP/0/RSP0/CPU0:ios(config-perf-meas)#endpoint ipv6 FCBB:0:1::
RP/0/RSP0/CPU0:ios(config-pm-ep)#delay-measurement
RP/0/RSP0/CPU0:ios(config-pm-ep-dm)#delay-profile name test
    
```

523655

```

RP/0/RSP0/CPU0:ios(config-pm-ep-dm)#exit
RP/0/RSP0/CPU0:ios(config-pm-ep)#exit
RP/0/RSP0/CPU0:ios(config-perf-meas)#liveness-profile name test
RP/0/RSP0/CPU0:ios(config-pm-ld-profile)#probe
RP/0/RSP0/CPU0:ios(config-pm-ld-probe)#flow-label explicit 100 200 300
RP/0/RSP0/CPU0:ios(config-pm-ld-probe)#

```

The following example shows how to use flow label for delay profile for a default endpoint:

```

RP/0/RSP0/CPU0:ios#configure
RP/0/RSP0/CPU0:ios(config)#performance-measurement
RP/0/RSP0/CPU0:ios(config-perf-meas)#delay-profile endpoint default
RP/0/RSP0/CPU0:ios(config-pm-dm-ep)#probe
RP/0/RSP0/CPU0:ios(config-pm-dm-ep-probe)#flow-label explicit 100 200 300

```

### Show Running Configuration

```

performance-measurement
 endpoint ipv6 FCBB:0:1::
   delay-measurement
     delay-profile name test
   !
 !
 liveness-profile name test
 probe
   flow-label explicit 100 200 300
 !
 !
 !

```

### Verification

The show output displays the delay information for the endpoint.

```

Router# show performance-measurement endpoint detail
Endpoint name: IPv6-FCBB:0:1::-vrf-default
Source address      : 192::2
VRF name            : default
Liveness Detection  : Enabled
Profile Keys:
  Profile name      : default
  Profile type      : Endpoint Liveness Detection
Segment-list        : None
Liveness Detection session:
  Session ID        : 4109
  Flow-label        : 1000
  Session State: Up
  Last State Change Timestamp: Jan 23 2024 16:06:01.214
  Missed count: 0

Liveness Detection session:
  Session ID        : 4110
  Flow-label        : 2000
  Session State: Up
  Last State Change Timestamp: Jan 23 2024 16:06:01.214
  Missed count: 0

Segment-list        : test-dm-two-carrier-sl2
FCBB:0:1:2:e004::/64
  Format: f3216
FCBB:0:1:3:e000::/64
  Format: f3216
FCBB:0:1:2:e004::/64

```



```

Format: f3216
FCBB:0:1:2:e000::/64
Format: f3216
FCBB:0:1:1:e000::/64
Format: f3216
FCBB:0:1:1:e004::/64
Format: f3216
FCBB:0:1:4:e000::/64
Format: f3216
FCBB:0:1:4::/48
Format: f3216
Liveness Detection session:
Session ID           : 4111
Flow-label           : 1000
Session State: Up
Last State Change Timestamp: Jan 23 2024 16:06:01.217
Missed count: 0

Liveness Detection session:
Session ID           : 4112
Flow-label           : 2000
Session State: Up
Last State Change Timestamp: Jan 23 2024 16:06:01.217
Missed count: 0

```

## SR Policy End-to-End Delay Measurement

Feature Name	Release	Description
SR Policy End-to-End Delay Measurement	Release 7.5.2	This feature allows you to monitor the end-to-end delay experienced by the traffic sent over an SR policy to ensure that the delay does not exceed the requested “upper-bound” and violate SLAs.

The PM for SR Policy uses IP/UDP packet format defined in RFC 5357 (TWAMP-Light) for probes.

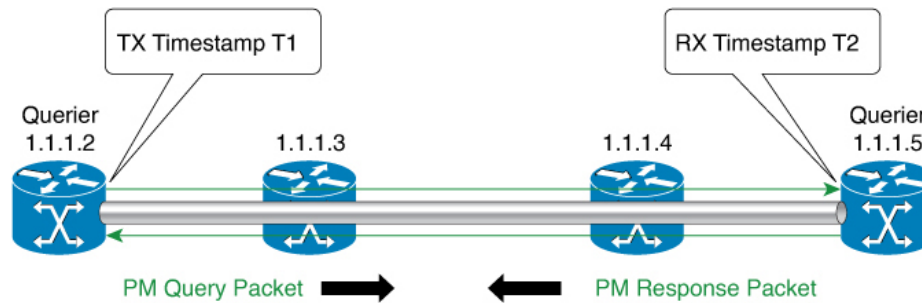
The extended TE link delay metric (minimum-delay value) can be used to compute paths for SR policies as an optimization metric or as an accumulated delay bound.

There is a need to monitor the end-to-end delay experienced by the traffic sent over an SR policy to ensure that the delay does not exceed the requested “upper-bound” and violate SLAs. You can verify the end-to-end delay values before activating the candidate-path or the segment lists of the SR policy in forwarding table, or to deactivate the active candidate-path or the segment lists of the SR policy in forwarding table.



**Note** The end-to-end delay value of an SR policy will be different than the path computation result (for example, the sum of TE link delay metrics) due to several factors, such as queuing delay within the routers.

Figure 8: Performance Measurement for SR Policy End-to-End Delay



- One Way Delay = (T2 – T1)
- Hardware clock synchronized using PTP (IEEE 1588) between querier and responder nodes (all nodes for higher accuracy)

3609825

The PM query and response for end-to-end SR Policy delay can be described in the following steps:

1. The local-end router sends PM query packets periodically to the remote side once the egress line card on the router applies timestamps on packets.
2. The ingress line card on the remote-end router applies time-stamps on packets as soon as they are received.
3. The remote-end router sends the PM packets containing time-stamps back to the local-end router.
4. One-way delay is measured using the time-stamp values in the PM packet.

### Restrictions and Usage Guidelines for PM for SR Policy Delay

Hardware clocks must be synchronized between the querier and the responder nodes of the link using PTP for one-way delay measurement.

### Enable One-Way Delay Mode

This example shows how to enable one-way delay mode.

When one-way delay mode is enabled, an IP/UDP TLV (defined in RFC 7876) is added in the query packet to receive the PM reply via IP/UDP. Hardware clocks must be synchronized between querier and responder nodes (using PTP).

```
Router(config)# performance-measurement delay-profile sr-policy default
Router(config-pm-dm-intf)# probe measurement-mode one-way
Router(config-perf-meas)# exit
```

### Configuring Performance Measurement Parameters

This example shows how to configure performance-measurement parameters for SR policy delay as a global default profile. The default values for the different parameters in the PM for SR policy delay is given as follows:

- **probe**: The default mode for probe is one-way delay measurement.
- **tx-interval**: Interval for sending probe packet. The default value is 3000000 microseconds and the range is from 3300 to 15000000 microseconds.
- **computation interval**: Interval for metric computation. Default is 30 seconds; range is 1 to 3600 seconds.

- **protocol:**
  - **twamp-light:** SR Policy delay measurement using RFC 5357 with IP/UDP encaps. This is the default protocol.
- **tos:** Type of Service
  - **dscp value:** The default value is 0 and the range is from 0 to 63.
  - **traffic-class value:** The default value is 0 and the range is from 0 to 7.
- **advertisement threshold-check:** The advertisement threshold-check has three types:
  - **average-delay:** Enable average-delay threshold-check.
  - **maximum-delay:** Enable maximum-delay threshold-check.
  - **minimum-delay:** Enable minimum-delay threshold-check.




---

**Note** The default value of periodic **advertisement threshold-check** is **maximum-delay**.

---

- **periodic advertisement:** Periodic advertisement is enabled by default.
- **periodic-advertisement interval:** The default value is 120 seconds and the interval range is from 30 to 3600 seconds.
- **periodic-advertisement threshold:** The default value of periodic advertisement threshold is 10 percent and the range is from 0 to 100 percent.
- **periodic-advertisement minimum-change:** The default value is 500 microseconds (usec) and the range is from 0 to 100000 microseconds.
- **accelerated advertisement:** Accelerated advertisement is disabled by default.
- **accelerated-advertisement threshold:** The default value is 20 percent and the range is from 0 to 100 percent.
- **accelerated-advertisement minimum:** The default value is 500 microseconds and the range is from 1 to 100000 microseconds.

```
Router(config)# performance-measurement delay-profile sr-policy default
Router(config-pm-dm-srpolicy)# probe
Router(config-pm-dm-srpolicy-probe)# tx-interval 60000
Router(config-pm-dm-srpolicy-probe)# computation-interval 60
Router(config-pm-dm-srpolicy-probe)# protocol twamp-light
Router(config-pm-dm-srpolicy-probe)# tos dscp
Router(config-pm-dm-srpolicy-probe)# exit

Router(config-pm-dm-srpolicy)# advertisement
Router(config-pm-dm-srpolicy-adv)# periodic
Router(config-pm-dm-srpolicy-adv-per)# interval 60
Router(config-pm-dm-srpolicy-adv-per)# minimum-change 1000
Router(config-pm-dm-srpolicy-adv-per)# threshold 20
Router(config-pm-dm-srpolicy-adv-per)# exit

Router(config-pm-dm-srpolicy-adv)# accelerated
```

```

Router(config-pm-dm-srpolicy-adv-acc) # minimum-change 1000
Router(config-pm-dm-srpolicy-adv-acc) # threshold 10
Router(config-pm-dm-srpolicy-adv-acc) # exit

Router(config-pm-dm-srpolicy-adv) # threshold-check minimum-delay
Router(config-pm-dm-srpolicy-adv) # exit
Router(config-pm-dm-srpolicy) #

```

### Configure the UDP Destination Port

Configuring the UDP port for TWAMP-Light protocol is optional. By default, PM uses port 862 as the TWAMP-reserved UDP destination port.

The UDP port is configured for each PM measurement probe type (delay, loss, protocol, authentication mode, etc.) on querier and responder nodes. If you configure a different UDP port, the UDP port for each PM measurement probe type must match on the querier and the responder nodes.




---

**Note** The same UDP destination port is used for delay measurement for links and SR Policy.

---

This example shows how to configure the UDP destination port.

```

Router(config) # performance-measurement
Router(config-perf-meas) # protocol twamp-light
Router(config-pm-protocol) # measurement delay unauthenticated
Router(config-pm-proto-mode) # querier-dst-port 12000

```

### Enable Performance Measurement for SR Policy

This example shows how to enable PM for SR policy delay for a specific policy.

```

Router(config) # segment-routing traffic-eng
Router(config-sr-te) # policy foo
Router(config-sr-te-policy) # performance-measurement
Router(config-sr-te-policy-perf-meas) # delay-measurement

```

### SR Policy Probe IP/UDP ECMP Hashing Configuration

This example shows how to configure SR Policy ECMP IP-hashing mode.

- The destination IPv4 address 127.x.x.x – 127.y.y.y is used in the Probe messages to take advantages of 3-tuple IP hashing (source-address, destination-address, and local router ID) for ECMP paths of SR-MPLS Policy.




---

**Note** The destination IPv4 address must be 127/8 range (loopback), otherwise it will be rejected.

---

- One PM session is always created for the actual endpoint address of the SR Policy.
- You can specify the number of IP addresses to sweep. The range is from 0 (default, no sweeping) to 128.
- Platforms may have a limitation for large label stack size to not check IP address for hashing.

```
Router(config)# performance-measurement delay-profile sr-policy default
Router(config-pm-dm-srpolicy)# probe
Router(config-pm-dm-srpolicy-probe)# sweep
Router(config-pm-dm-srpolicy-probe-sweep)# destination ipv4 127.0.0.1 range 28
```

## Path Tracing in SRv6 Network

Table 11: Feature History Table

Feature Name	Release	Description
Path Tracing Source and Sink Nodes	Release 24.1.1	<p>You can now view the Path Tracing Source and Sink nodes, which are responsible for handling IPv6 transit traffic. This feature also provides full characterization of the packet delivery path which includes real-time information to check the current status of the network, such as whether packets are being diverted due to a breach. It also allows for the pinpointing of the exact location of problems between routers, and ensures that traffic flows according to specified priorities for Quality of Service (QoS) enforcement.</p> <p>This feature introduces a new behavior keyword <b>utef</b> under the <b>route (static)</b> command.</p>
Path Tracing Midpoint Node	Release 7.8.1	<p>Path Tracing (PT) provides a log or record of the packet path as a sequence of interface IDs along with its time stamp. In Path Tracing, a node can behave as a source, midpoint, or sink node.</p> <p>The Path Tracing Midpoint feature is implemented in this release which measures the hop-by-hop delay, traces the path in the network and collects egress interface load information and interface Id, and stores them in the Midpoint Compressed Data (MCD) section of Hop-by-Hop Path Tracing (HbH-PT) header.</p> <p>This feature provides visibility to the Path Tracing Midpoint node that handles IPv6 transit in Path Tracing and full characterization of the packet delivery path. It provides real time information and the current status of the network.</p> <p>This feature introduces the following command:</p> <ul style="list-style-type: none"> <li>• <b>performance-measurement interface</b></li> </ul>

Operators do not know the actual path that the packets take within their network. This makes operations, such as troubleshooting routing problems, or verifying Equal-Cost Multipath (ECMP), a complex problem. Also, operators want to characterize the network in terms of delay and load on a per-hop basis.

Knowledge of the Path Tracing Midpoint helps the operators to troubleshoot the routing problems faster.

This feature allows the operators to:

- Detect the actual path the packet takes between any two nodes in network (A and Z).
- Measure the end-to-end delay from A to Z.
- Measure the per-hop delay at each node on the path from A to Z.
- Detects the load on each router that forwards the packet from A to Z

Path Tracing (PT) provides a log or record of the packet path as a sequence of interface IDs along with its time-stamp. In addition, it provides a record of end-to-end delay, per-hop delay, and load on each egress interface along the packet delivery path.

In Path Tracing, a node can behave as a source, midpoint, or a sink node.

The source node generates and injects probe packets toward a destination node to trace the time-stamp and interface ID along the path of the probe packet. The Interface ID value of 0 means that Path Tracing (PT) is disabled on the interface.

Path Tracing (PT) Midpoint: It is a transit node that performs IPv6 routing. In addition, it records the PT information (MCD) in the HbH-PT.




---

**Note** There is no support for Path Tracing Midpoint on transit nodes that perform SRH operations or SRv6 endpoint operations. Support starts from Cisco IOS XR Release 24.1.1.

---

- Midpoint Compressed Data (MCD): The PT Midpoint along the packet delivery path from the Source to Sink node, stores its PT information into the HbH-PT header. This PT information is called Midpoint Compressed Data (MCD).
- Hop-by-Hop Path Tracing (HbH-PT): In IPv6 The HbH PT Options header is used to carry optional information that is examined and processed by every node along a packet's delivery path. It contains a stack of MCDs.
- PT-Aware Midpoint: A midpoint node that performs plain IPv6 routing or SR Endpoint processing and in addition stores the Path Tracing information in HbH-PT.
- PT-Unaware Midpoint: A midpoint node that performs plain IPv6 routing or SR Endpoint processing and is not capable of performing Path Tracing.
- PT-Legacy Midpoint: A midpoint node that performs plain IPv6 routing or SR Endpoint processing and is not capable of recording Path Tracing information in the HBH-PT. However, it is capable of exporting Path Tracing information directly to the collector, using the node telemetry system.
- PT Source: A Source node is the one that starts a PT session and generates PT probes. Support starts from Cisco IOS XR Release 24.1.1.
- PT Sink: A node that receives the PT probes sent from the Source node containing the information recorded by every PT Midpoint along the path and forwards them to the collector after recording its Path Tracing information. Support starts from Cisco IOS XR Release 24.1.1.
- RC: Regional collector that receives PT probes, parses, and stores them in Timeseries DB

The destination or sink node that receives the PT probes generated by the PT source node, stores PT related info into PT-TLV and forwards them to a Regional Collector (RC). This Regional Collector (RC) parses and stores them in the TimeSeries Database. It uses the information in the Hop-by-Hop Path Tracing (HbH-PT) to construct the packet delivery path and the timestamps at each node.

## Limitations and Guidelines

This section lists the limitations of this feature.

- PT Source and Sink nodes are not supported yet. The system can still work as PT midpoint for other devices acting as Source or Sink in the PT network path.
- No support for interface load calculation and recording on IPv6 Path Tracing MidPoint Node. MCD contains interface load value of 0.
- SRv6 Segment Endpoint Midpoint PT (Update DA from SRH.SL and PT MCD update) at midpoint node is not supported. SRv6 endpoint function will not execute properly.
- IPv6 and SRv6 Path Tracing Midpoint Node are supported. SRv6 PT midpoint support Micro-SID (uSID) Shift and Forward action with MCD update.
- Path tracing on Bundled Interfaces and subinterfaces is supported by configuring path-tracing interface-id on physical ports.
- PT unaware IPv6 and SRv6 midpoint forwards transparently without PT update or may punt the packet locally and the control-plane drops the packet.
- PT unaware SRv6 Segment Endpoint Midpoint Node will not execute SRv6 endpoint function. PT packet is forwarded transparently without PT update or punted locally and the control-plane drops the packet.

## Configuration Steps



**Note** These configurations must be done on the Source, Midpoint and Sink routers as shown in the following configuration examples.

- Configuration example of Source node:

Configure the endpoint with the probe profile name on the source node:

```
Router(config)# performance-measurement
Router(config-perf-meas)# endpoint ipv6 fccc:cc00:9000:fef1::
Router(config-pm-ep)# path-tracing
Router(config-pm-ep-ptrace)# session-id 1011
Router(config-pm-ep-ptrace-sid)# segment-routing traffic-eng seg$
Router(config-pm-ep-ptrace-sid)# probe-profile name PP_12_1
Router(config-pm-ep-ptrace-sid)# source-address ipv6 1::1
Router(config-pm-ep)# path-assurance
Router(config-pm-ep-passurance)# session-id 1111
Router(config-pm-ep-passurance-sid)# segment-routing traffic-eng$
Router(config-pm-ep-passurance-sid)# probe-profile name PP_12_1
```

Configure probe profile parameters:

```

Router(config)# performance-measurement
Router(config-perf-meas)# path-tracing
Router(config-pm-pttrace)# probe-profile name PP_12_1
Router(config-pm-pr-profile)# tx-interval 3000
Router(config-pm-pr-profile)# flow-label explicit 1000 2000 4000 8$
Router(config-pm-pr-profile)# traffic-class from 16 to 128 increme$

```

Configure Interface ID under Path-tracing for the Source node and for it to participate in the MCD updates inside the probe packets:

```

Router(config)# performance-measurement
Router(config-pm)# interface FourHundredGigE0/0/0/1
Router(config-pm-interf)# path-tracing
Router(config-pm-interf-interf-id)# interface-id 200
Router(config-pm-interf-time)# exit

```

- Configuration example of Midpoint node:

Configure the Interface ID under Path-tracing for the Midpoint node and for it to participate in the MCD updates inside the probe packets:

```

Router(config)# performance-measurement
Router(config-pm)# interface FourHundredGigE0/0/0/1
Router(config-pm-interf)# path-tracing
Router(config-pm-interf-interf-id)# interface-id 200
Router(config-pm-interf-time)# exit

```

- Configuration example of Sink node.

Configure Router Static:

```

Router(config)# router static
Router(config-static)# address-family ipv6 unicast
Router(config-static-afi)# fccc:cc00:9000:fe1::/64 segment-routing srv6 endpoint
behavior utef controller-address fccc:cc00:7::
!

```

Configure Interface ID under Path-tracing for the Sink node and for it to participate in the MCD updates inside the probe packets:

```

Router(config)# performance-measurement
Router(config-pm)# interface FourHundredGigE0/0/0/1
Router(config-pm-interf)# path-tracing
Router(config-pm-interf-interf-id)# interface-id 200
Router(config-pm-interf-time)# exit

```

## Running Configuration

- Running configuration example of Source node:

Configure the endpoint with the probe profile name on the source node:

```

performance-measurement
endpoint ipv6 fccc:cc00:9000:fe1::
path-tracing
session-id 1011
segment-routing traffic-eng seg$

```



```

        probe-profile name PP_12_1
        source-address ipv6 1::1
    path-assurance
        session-id 1111
        segment-routing traffic-eng$
        probe-profile name PP_12_1
    !
    !
!
!
!

```

Configure probe profile parameters:

```

performance-measurement
  path-tracing
    probe-profile name PP_12_1
    tx-interval 3000
    flow-label explicit 1000 2000 4000 8$
    traffic-class from 16 to 128 increme$
  !
!
!

```

Configure Interface ID under Path-tracing for the Source node and for it to participate in the MCD updates inside the probe packets:

```

performance-measurement
  interface FourHundredGigE0/0/0/1
    path-tracing
      interface-id 200
    exit
  !
!
!
!
!

```

- Running configuration example of Midpoint node:

Configure the Interface ID under Path-tracing for the Midpoint node and for it to participate in the MCD updates inside the probe packets:

```

performance-measurement
  interface FourHundredGigE0/0/0/1
    path-tracing
      interface-id 200
    exit
  !
!
!
!
!

```

- Running configuration example of Sink node.

Configure Router Static:

```

router static
  address-family ipv6 unicast
    fccc:cc00:9000:fef1::/64 segment-routing srv6 endpoint behavior utef controller-address
    fccc:cc00:7::
  !
!
!
!

```

Configure Interface ID under Path-tracing for the Sink node and for it to participate in the MCD updates inside the probe packets:

```
performance-measurement
 interface FourHundredGigE0/0/0/1
   path-tracing
     interface-id 200
   exit
 !
 !
 !
 !
```

## Verification

It is good to check the target interface configuration and performance-measurement configuration for that interface.

Verify using the show commands listed below to check if the PT configuration is applied to the interface properly.

### Source Node Verification

```
Router# sh run performance-measurement
performance-measurement
probe-profile name foo
 tx-interval 6000
 flow-label from 100 to 300 increment 10
 !
 !
```

```
Router# sh performance-measurement profile named-profile
Endpoint Probe Measurement Profile Name: foo
Profile configuration:
Measurement mode           : One-way
Protocol type              : TWAMP-light
Type of service:
  TWAMP-light DSCP         : 48
TX interval                : 6000000 (effective: 6000000) uSec
Destination sweeping mode  : Disabled
Liveness detection parameters:
  Multiplier               : 3
  Logging state change     : Disabled

Hop Limit                  : 255
Flow Label Count           : 21
  Flow Labels: 100, 110, 120, 130, 140, 150, 160, 170, 180, 190, 200, 210, 220, 230,
240,
                        250, 260, 270, 280, 290, 300
Packet Size Count          : 0
Traffic Class Count        : 0
```

```
Router#sh run performance-measurement
performance-measurement
endpoint ipv6 bbbb::
 path-assurance
   session-id 11
 !
 !
 !
source-address ipv6 aaaa::
```

!

```
Router# sh performance-measurement endpoint
Endpoint name: IPv6-bbbb::-vrf-default
Source address           : Unknown
VRF name                 : default
Probe Measurement      : Enabled
Profile Keys:
  Profile name           : default
  Profile type         : Endpoint Probe Measurement
```

Run this show command to verify the probe sessions:

```
Router# show performance-measurement probe-sessions
Transport type           : Endpoint
Measurement type        : Probe
Endpoint name           : IPv6-bbbb:bbbb:2::-vrf-default
endpoint                : bbbb:bbbb:2::
source                  : bbbb:bbbb:1::
vrf                     : default
Segment-list            :
Path Tracing session:
  Session ID             : 10
  Profile Keys:
    Profile name         : pt1
    Profile type         : Probe
  Current status:
    Packet sent every 0.30000 seconds (value stretched for rate-limiting)
    Next packet will be sent in 0.20 seconds
```

```
Transport type           : Endpoint
Measurement type        : Probe
Endpoint name           : IPv6-bbbb:bbbb:2::-vrf-default
endpoint                : bbbb:bbbb:2::
source                  : bbbb:bbbb:1::
vrf                     : default
Segment-list            :
Path Tracing session:
  Session ID             : 11
  Profile Keys:
    Profile name       : pt2 (Profile not found)
    Profile type       : N/A
  Current status:
    Not running: Profile is not configured
```

```
Transport type           : Endpoint
Measurement type        : Probe
Endpoint name           : IPv6-bbbb:bbbb:2::-vrf-default
endpoint                : bbbb:bbbb:2::
source                  : bbbb:bbbb:1::
vrf                     : default
Segment-list            :
Path Assurance session:
  Session ID             : 20
  Profile Keys:
    Profile name         : pal
    Profile type         : Probe
  Current status:
    Packet sent every 0.30000 seconds (value stretched for rate-limiting)
    Next packet will be sent in 0.24 seconds
```

Run this show command to view the summary of all the probe sessions:

```
Router# show performance-measurement summary
Measurement Information:
```

```

Total interfaces with PM sessions      : 0
Total SR Policies with PM sessions    : 0
Total Endpoints with PM sessions      : 1
Total RSVP-TE tunnels with PM sessions : 0

```

```

Global Counters:
  Total packets sent      : 0
  Total query packets received : 0
  Total invalid session id : 0
  Total missing session   : 0

```

```

Probe sessions:
  Total sessions      : 3
  Path-tracing sessions:
    Total running sessions      : 1
    Total running error sessions : 0
  Path-assurance sessions:
    Total running PA sessions   : 1
    Total running error PA sessions : 0
Counters:
  Path-tracing packets:
    Total sent      : 3063
    Total sent errors : 0
  Path-assurance packets:
    Total sent      : 470
    Total sent errors : 0

```

```

Router# show cef interface fourHundredGigE 0/0/0/1
FourHundredGigE0/0/0/1 is up if_handle 0x0f000208 if_type IFT_FOURHUNDREDGE(0xcd)
  idb info 0x94dfbf88 flags 0x30001 ext 0x0
  Vrf Local Info (0x0)
  Interface last modified, create
  Reference count 1      Next-Hop Count 0
  PT (path tracing) is enabled: id:0xC8 load_in:0x0 load_out:0x0 tts:0x3
  Protocol Reference count 0
  Protocol ipv4 not configured or enabled on this card
  Primary IPV4 local address NOT PRESENT

```

This is an example of Show CLI with Interface ID:

```

Router# show run performance-measurement
performance-measurement
probe-profile name foo
tx-interval 6000
flow-label from 100 to 300 increment 10
!
!
Router# sh performance-measurement profile named-profile
Endpoint Probe Measurement Profile Name: foo
Profile configuration:
  Measurement mode      : One-way
  Protocol type         : TWAMP-light
  Type of service:
    TWAMP-light DSCP    : 48
  TX interval          : 6000000 (effective: 6000000) uSec
  Destination sweeping mode : Disabled
  Liveness detection parameters:
    Multiplier          : 3
    Logging state change : Disabled

Hop Limit              : 255
Flow Label Count       : 21
  Flow Labels: 100, 110, 120, 130, 140, 150, 160, 170, 180, 190, 200, 210, 220, 230,

```

```

240,
           250, 260, 270, 280, 290, 300
Packet Size Count           : 0
Traffic Class Count        : 0

Router# show cef interface GigabitEthernet 0/2/0/0
GigabitEthernet0/2/0/0 is up if_handle 0x01000020 if_type IFT_GETHERNET(0xf)
  idb info 0x619f16f0 flags 0x30101 ext 0x627ef180 flags 0x30050
  Vrf Local Info (0x626510f0)
  Interface last modified Mar  4, 2022 13:34:43, modify
  Reference count 1          Next-Hop Count 3
  PT (path tracing) is enabled: id:0x40 load_in:0x0 load_out:0x0 tts:0x1
  Forwarding is enabled
  ICMP redirects are never sent
  ICMP unreachable are enabled
  Protocol MTU 1500, TableId 0xe0000000(0x61ccf768)
  Protocol Reference count 4
  Primary IPV4 local address 10.10.10.1

```

```

Router# show performance-measurement interfaces
Interface Name: GigabitEthernet0/2/0/0 (ifh: 0x1000020)
  Delay-Measurement           : Disabled
  Loss-Measurement           : Disabled
  Path-Tracing                : Enabled
  Configured IPv4 Address     : 10.10.10.1
  Configured IPv6 Address     : 10:10:10::1
  Link Local IPv6 Address     : fe80::91:e4ff:fe60:6707
  Configured Next-hop Address : Unknown
  Local MAC Address          : 0291.e460.6707
  Next-hop MAC Address       : 023a.6fc9.cd6b
  In-use Source Address      : 10.10.10.1
  In-use Destination Address  : 10.10.10.2
  Primary VLAN Tag           : None
  Secondary VLAN Tag         : None
  State                       : Up

  Path-Tracing:
    Interface ID              : 64
    Load IN                   : 0
    Load OUT                  : 0
    Load Interval             : 60
  Last FIB Update:
    Updated at: Mar 04 2022 13:34:43.112 (0.392 seconds ago)
    Update reason: Path tracing config
    Update status: Done

```

This is an example of Show CLI without InterfaceID, which means PT is disabled on the target interface. So, you can configure timestamp template:

```

Router# show cef interface GigabitEthernet 0/2/0/0
GigabitEthernet0/2/0/0 is up if_handle 0x01000020 if_type IFT_GETHERNET(0xf)
  idb info 0x619f16f0 flags 0x30101 ext 0x627ef180 flags 0x30050
  Vrf Local Info (0x626510f0)
  Interface last modified Mar  4, 2022 13:49:37, modify
  Reference count 1          Next-Hop Count 3
  Forwarding is enabled
  ICMP redirects are never sent
  ICMP unreachable are enabled
  Protocol MTU 1500, TableId 0xe0000000(0x61ccf768)
  Protocol Reference count 4
  Primary IPV4 local address 10.10.10.1

```

```

Router# sh performance-measurement interfaces
Interface Name: GigabitEthernet0/2/0/0 (ifh: 0x1000020)
  Delay-Measurement           : Disabled
  Loss-Measurement           : Disabled
  Path-Tracing                : Enabled
  Configured IPv4 Address     : 10.10.10.1
  Configured IPv6 Address     : 10:10:10::1
  Link Local IPv6 Address     : fe80::91:e4ff:fe60:6707
  Configured Next-hop Address : Unknown
  Local MAC Address           : 0291.e460.6707
  Next-hop MAC Address        : 023a.6fc9.cd6b
  In-use Source Address       : 10.10.10.1
  In-use Destination Address  : 10.10.10.2
  Primary VLAN Tag            : None
  Secondary VLAN Tag          : None
  State                       : Up

Path-Tracing:
  Interface ID                 : 0
  Timestamp Template       : 3
  Load IN                     : 0
  Load OUT                     : 0
  Load Interval                : 60
Last FIB Update:
  Updated at: Mar 04 2022 13:49:37.492 (176.418 seconds ago)
  Update reason: Path tracing config
  Update status: Done

```

### Sink Node Verification

```

Router# sh segment-routing srv6 sid fccc:cc00:1:fef1:: detail

SID                               Behavior      Context
Owner                             State        RW
fccc:cc00:1:fef1::                uTEF        [fccc:cc01:7::, default]:fccc:cc00:1:fef1::
ip_static_srv6                    InUse       Y
SID Function: 0xfef1
SID context: { controller=fccc:cc01:7::, table-id=0xe0800000 ('default':IPv6/Unicast),
differentiator=fccc:cc00:1:fef1:: }
Locator: 'locator0'
Allocation type: Explicit

Router# sh segment-routing srv6 sid fccc:cc00:1:fef3:: detail

SID                               Behavior      Context
Owner                             State        RW
fccc:cc00:1:fef3::                uTEF        [fccc:cc00:7::, default]:fccc:cc00:1:fef3::
ip_static_srv6                    InUse       Y
SID Function: 0xfef3
SID context: { controller=fccc:cc00:7::, table-id=0xe0800000 ('default':IPv6/Unicast),
differentiator=fccc:cc00:1:fef3:: }
Locator: 'locator0'
Allocation type: Explicit

Router# sh segment-routing srv6 sid fccc:cc01:1:fef2:: detail

SID                               Behavior      Context                               Owner
State RW
fccc:cc01:1:fef2::                uTEF        [fccc:cc00:7::, default]:fccc:cc01:1:fef2::
ip_static_srv6                    InUse       Y
SID Function: 0xfef2
SID context: { controller=fccc:cc00:7::, table-id=0xe0800000 ('default':IPv6/Unicast),
differentiator=fccc:cc01:1:fef2:: }
Locator: 'locator1'

```

Allocation type: Explicit  
 Created: Feb 27 11:06:54.999 (00:10:05 ago)

# Two-Way Active Measurement Protocol Light Source Address Filtering

Table 12: Feature History Table

Feature Name	Release Information	Feature Description
Two-Way Active Measurement Protocol Light Source Address Filtering	Release 7.11.1	<p>You can now restrict unauthorized users from sending packets to the network and prevent compromising the network security and reliability. For a destination UDP port, you can configure the list of IP addresses that can send Two-Way Active Measurement Protocol (TWAMP)-light packets to responder or querier nodes.</p> <p>In earlier releases, the responder or querier node accepted TWAMP-light packets from all IP addresses.</p> <p>The feature introduces these changes:</p> <p><b>CLI:</b></p> <ul style="list-style-type: none"> <li>The <b>querier</b> and <b>responder</b> keywords are introduced in the <a href="#">performance-measurement protocol twamp-light measurement delay</a> command.</li> </ul> <p><b>YANG Data Models:</b></p> <ul style="list-style-type: none"> <li><a href="#">Cisco-IOS-XR-performance-measurement-cfg.yang</a></li> <li><a href="#">Cisco-IOS-XR-perf-meas-oper.yang</a></li> </ul> <p>See (<a href="#">GitHub</a>, <a href="#">Yang Data Models Navigator</a>)</p>

Earlier, the responder node scanned all IP addresses of a querier on the destination UDP port. In other words, the responder node accepted packets from any IP address. See [Measurement Modes](#) for more information about querier and responder nodes.




---

**Note** The responder is also called the reflector, and the querier is also called the sender.

---

With this configuration, you can specify the source IP addresses on both the responder and querier nodes. The responder or querier nodes accept packets only from the IP addresses configured in the TWAMP-light protocol, and reject the packets from an IP address that isn't included in the configured list.

All the configured addresses are available for use on all interfaces in the Local Packet Transport Services (LPTS). The configured address filter applies to both default and nondefault VRFs. The TWAMP delay measurement sessions use the configured addresses.

## Usage Guidelines and Limitations

The following usage guidelines and limitations apply:

- When you configure the prefix entries on the responder or querier nodes, the PM adds the responder or querier node source IP address to the LPTS. For each prefix, a new LPTS entry is added or created.
- For TWAMP liveness sessions, the PM automatically adds the source IP addresses to the LPTS for if you have configured the prefix entries on the responder or querier nodes.
- As the maximum number of LPTS hardware entries are limited, ensure that enough LPTS entries are allocated for the IP addresses on a line card. You can scale the LPTS configuration to maximum LPTS entries for the PM flow-type. For more details on configuring the LPTS entries for PM flow-type, refer to the *IP Addresses and Services Configuration Guide*.




---

**Note** The PM UDP port accepts all incoming IPv4 or IPv6 packets when there are no IPv4 or IPv6 prefix entries configured.

---

## Configure IP address on querier and responder nodes

- The length of the IPv4 and IPv6 prefixes must be less than 32 and 128 respectively.
- The length or mask of the source IP address must be:
  - For IPv4: 0–31
  - For IPv6: 0–127

### Configure the IP address on a responder

Perform this task to configure the IP address of a querier on a responder node for delay measurement.

```
Router#configure
Router(config)#performance-measurement
Router(config-perf-meas)#protocol twamp-light
Router(config-pm-protocol)#measurement delay
Router(config-pm-proto-meas)#responder
Router(config-pm-proto-responder)#allow-querier
Router(config-pm-allowed-querier)#address ipv4 10.10.10.1
```



### Running Configuration

```
performance-measurement
  protocol twamp-light
    measurement delay
      responder
        allow-querier
          address ipv4 10.10.10.1
        !
      !
    !
  !
End
```

### Verification

The following example shows output from the IP address of a querier, which is configured on a responder node for delay measurement.

```
Router#show performance-measurement allowed-querier summary
Wed Oct 11 10:41:43.268 UTC
```

```
-----
0/RP0/CPU0
-----
Allowed-querier IPv4 prefix           : 1
   10.10.10.1/32
Allowed-querier IPv6 prefix           : 0

RX UDP port status:
TWAMP-Light Default Unauthenticated responder port : 862
  Opened IPv4 port                               : 862
  IPv4 Port Update Time                          : Oct 11 2023 10:37:48.118
  Opened IPv6 port                               : 862
  IPv6 Port Update Time                          : Oct 11 2023 10:37:47.778
```

### Configure the source IP address on a querier

Perform this task to configure the IP address of a responder on a querier node for delay measurement.

```
Router#configure
Router(config)#performance-measurement
Router(config-perf-meas)#protocol twamp-light
Router(config-pm-protocol)#measurement delay
Router(config-pm-proto-meas)#querier
Router(config-pm-proto-querier)#allow-responder
Router(config-pm-allowed-responder)#address ipv4 10.10.10.1
```

### Running Configuration

```
performance-measurement
  protocol twamp-light
    measurement delay
      querier
        allow-responder
          address ipv4 10.10.10.1
        !
      !
    !
  !
End
```

## Verification

The following example shows output from the IP address of a responder, which is configured on a querier node for delay measurement.

```
Router#show performance-measurement allowed-responder summary
Wed Mar 29 19:38:06.381 UTC
```

---

```
0/RP1/CPU0
```

---

```
Allowed-responder IPv4 prefix                : 2
  10.10.10.1/32 [Auto]
  3.3.3.3/32
Allowed-responder IPv6 prefix                : 1
  fc00:0:1::1/128 [Auto] [Pending Add]
Querier CPU UDP port status:
  TWAMP-Light Default Unauthenticated querier port : N/A
  Opened IPv4 port                                : 27643
  IPv4 Port Update Time                          : Mar 29 2023 18:43:49.080
  Opened IPv6 port                                : 28274
  IPv6 Port Update Time                          : Mar 24 2023 20:58:46.150
```

# Synthetic Loss Measurement

Table 13: Feature History Table

Feature Name	Release	Description
Synthetic Loss Measurement	Release 24.1.1	<p>You can now proactively monitor and address potential network issues before they impact users by measuring key parameters everywhere, packet loss, and jitter. Using this information, you can plan network capacity optimally and ensure quality of service. Such proactive action is possible because this feature reports synthetic Two-Way Active Measurement Protocol (TWAMP) test packets deployed in delay-profile or delay measurement sessions.</p> <p>It also enables you to set the upper and lower limits and notifies when the synthetic packet loss metric is out of the set limit.</p> <p>The feature introduces these changes:</p> <p><b>CLI:</b></p> <ul style="list-style-type: none"> <li>The optional <b>anomaly-loss</b> and <b>anomaly-check</b> keywords are introduced in the <a href="#">performance-measurement delay-profile</a> command.</li> <li><a href="#">show performance-measurement history</a></li> </ul> <p><b>YANG Data Model</b></p> <ul style="list-style-type: none"> <li>New XPath for <code>Cisco-IOS-XR-um-performance-measurement-cfg</code></li> <li>New operations for <code>Cisco-IOS-XR-perf-meas-oper.yang</code> (see <a href="#">GitHub</a>, <a href="#">YANG Data Models Navigator</a>)</li> </ul>

Synthetic packets are data packets that are artificially generated for testing or managing a network, as opposed to being generated by normal network data traffic. These packets are unrelated to regular user activities or data. Instead, they are intentionally designed to serve monitoring, testing, and diagnostic purposes.

Synthetic TWAMP test packets that are deployed in the delay profile are used to simulate network traffic, test network performance, and evaluate network infrastructure. They are useful for assessing the quality of service (QoS) of a network, identifying network vulnerabilities, and troubleshooting network issues. These packets enable network administrators to identify potential problems before they impact the overall network performance and reliability. *For more information on TWAMP, refer to the IP Service Level Agreements chapter in the System Monitoring Configuration Guide for Cisco 8000 Series Routers.*

Overall, synthetic packets play a critical role in network management, providing network engineers with a powerful tool for optimizing network performance, identifying and resolving issues.




---

**Note** This is an inbuilt feature of delay measurement. To get the synthetic packet loss information for delay-measurement sessions, you must only configure the delay sessions. No additional configuration is required for Synthetic Loss Measurement. However, an optional **anomaly-loss** command is introduced.

---

Synthetic packets simulate various types of traffic that can be sent with the normal traffic to measure loss or latency, throughput, or packet loss. This is often done in network performance monitoring and testing, where the goal is to understand how the network or a specific network device performs under typical conditions.

This feature enables you to measure and monitor the Synthetic packets lost and help make informed decisions on the following:

- **Network Performance Monitoring:** Synthetic packets can be used to simulate different types of network traffic in order to measure and test the performance of network devices or the network itself.
- **Troubleshooting:** They can be used to diagnose problems in a network by sending them through different parts of the system to see where they may fail or be slowed down.
- **Security:** They can also be used in security applications, such as penetration testing, where synthetic packets are sent to a system to find vulnerabilities.

## Configure Synthetic Loss Measurement

The following example enables Synthetic Loss Measurement for an SR Policy.




---

**Note** A delay-measurement session is good enough to get the synthetic packet loss automatically, without any extra configuration. The configure shown here is for anomaly loss, which is optional.

---

```
Router(config)#performance-measurement
Router(config-perf-meas)#delay-profile sr-policy default
Router(config-pm-dm-srpolicy)#advertisement
Router(config-pm-dm-srpolicy-adv)#anomaly-loss
Router(config-pm-dm-srpolicy-adv-anom-loss)#upper-bound 30 lower-bound 20
Router(config-pm-dm-srpolicy-adv-anom-loss)#commit
```




---

**Note** Similarly, you can configure Synthetic Loss Measurement for **endpoint default**, **interface default** or **name** (named profile).

---

### Running Configuration

The running configuration for this feature is as shown:

```
performance-measurement
  delay-profile sr-policy default
  advertisement
```

```

    anomaly-loss
      upper-bound 30 lower-bound 20
    !
  !
!
!
!

```

## Verification

Use the show commands to verify the running configuration as shown:

```

Router# show performance-measurement interfaces delay detail
Interface Name: GigabitEthernet0/2/0/0 (ifh: 0x1000020)
...
  Last advertisement:
  ...
  Packets sent: 40, received: 40, lost: 0
  ...
  Next advertisement:
  ...
  Packets sent: 10, received: 10, lost: 0
  ...
  Current computation:
  ...
  Packets sent: 4, received: 4, lost: 0

```

```

Router# show performance-measurement history probe-computation interfaces
Interface Name: GigabitEthernet0/2/0/0 (ifh: 0x1000020)
Delay-Measurement history (uSec):
  Probe Start Timestamp      Pkt(TX/RX)   Average      Min      Max
  Aug 01 2023 08:04:15.230    10/10        704         651     779

```

## Endpoint

Use these show commands to verify the Endpoint configuration. The example include endpoint delay details and advertisement history. You can also verify the aggregation history.

```

Router# show performance-measurement endpoint delay detail
Endpoint name: IPv4-192.168.0.4-vrf-default
...
  Last advertisement:
  ...
  Packets sent: 40, received: 40, lost: 0
...
  Next advertisement:
  ...
  Packets sent: 30, received: 30, lost: 0
  ...
  Current computation:
  ...
  Packets sent: 6, received: 6, lost: 0
  ...

```

```

Router# show performance-measurement history advertisement endpoint
Endpoint name: IPv4-192.168.0.4-vrf-default
...
  Delay-Measurement history (uSec):

```

Advertisement Timestamp	Pkt (TX/RX)	Average	Min	Max	Reason
Aug 01 2023 08:31:18.835	40/40	3948	3127	15503	PER-MAX

## RSVP-TE

Use these show commands to verify the RSVP-TE configuration. The example include rsvp-te delay details and the aggregation history. You can also verify the advertisement history.

```
Router# show performance-measurement rsvp-te detail
```

```
Tunnel name: tunnel-te1 (ifh: 0xd0)
...
  Last advertisement:
  ...
  Packets sent: 40, received:40, lost: 0
  ...
  Next advertisement:
  ...
  Packets sent: 10, received:10, lost: 0
  ...
  Last computation:
  Packets sent: 10, received:10, lost: 0
  ...
  Current computation:
  ...
  Packets sent: 6, received: 6, lost: 0
  ...
  ...
```

```
Router# show performance-measurement history aggregation rsvp-te
```

```
...
Delay-Measurement history (uSec):
  Aggregation Timestamp      Pkt (TX/RX)  Average      Min      Max
  Aug 01 2023 08:37:23.702   40/40        3372         3172    4109
  ...
```

## SR-Policy

Use these show commands to verify the SR-Policy configuration. The example include delay details and advertisement history. You can also verify the aggregation history.

```
Router# show performance-measurement sr-policy delay detail
```

```
SR Policy name: srte_c_10_ep_192.168.0.4
...  Last advertisement:
  ...
  Packets sent: 88, received: 88, lost: 0
  ...
  Next advertisement:
  ...
  Packets sent: 132, received: 132, lost: 0
  ...
  Last computation:
  Packets sent: 4, received: 4, lost: 0
  ...
  Current computation:
  ...
  Packets sent: 4, received: 4, lost: 0
  ...
  Segment-List                : R4
  ...
  Last advertisement:
  ...
  Packets sent: 88, received: 88, lost: 0
```

```

...
Next advertisement:
...
Packets sent: 132, received: 132, lost: 0
...
Last computation:
Packets sent: 4, received: 4, lost: 0
...
Current computation:
Packets sent: 4, received: 4, lost: 0
...

```

**Router# show performance-measurement history advertisement sr-policy**

```

...
Delay-Measurement history (uSec):
  Advertisement Timestamp   Pkt(TX/RX)   Average   Min   Max   Reason
  Aug 01 2023 10:05:14.072   24/24       3408     3408  3408  ACCEL-MAX

```

### Packet loss advertisement

Use this show command to verify multiple configurations with a single show command:

**Router# show performance-measurement history advertisement sr-policy/endpoint**

```

...
Delay-Measurement history (uSec):
  Advertisement Timestamp   Pkt(TX/RX)   Average   Min   Max   Reason
  Sep 14 2023 11:35:38.180   10/9         3595     3411  3696  NEW-SESSION
  Sep 14 2023 11:34:38.178   10/0         0         0       0     SESSION-ERROR
  Sep 14 2023 11:34:08.177   10/7         3733     3733  3733  ANOM-PKT-LOSS
  Sep 14 2023 11:34:02.177   8/7         3733     3733  3733  PKT-LOSS
  Sep 14 2023 11:33:38.176   10/10        3823     3617  4627  FIRST
...

```

**Router# show performance-measurement sessions**

```

...
Last advertisement:
  Advertised at: Sep 14 2023 08:47:16.540 (145.777 seconds ago)
  Advertised reason: Periodic timer, max delay threshold crossed
  Advertised delays (uSec): avg: 5373, min: 3992, max: 26212, variance: 0
  Packets sent: 30, received: 30, lost: 0
  Min-Max A flag set: False
  Loss A flag set: False
...
Last advertisement:
  Advertised at: Sep 14 2023 08:52:39.603 (12.522 seconds ago)
  Advertised reason: PM session anomaly due to packet loss
  Advertised delays (uSec): avg: 5373, min: 3992, max: 26212, variance: 0
  Packets sent: 30, received: 30, lost: 0
  Min-Max A flag set: False
  Loss A flag set: True
...
Last advertisement:
  Advertised at: Sep 15 2023 11:42:41.594 (47.660 seconds ago)
  Advertised reason: Performance measurement session error
...

```



**Note** No metrics or A bit is shown when there is a session error.

**Router# show logging**

```
RP/0/0/CPU0:Sep 14 11:33:38.176 PDT: perf_meas[1001]:
%ROUTING-PERF_MEAS-5-PM_DELAY_EXCEEDS_THRESHOLD : Reason: First advertisement, PM delay
metric avg: 3823, min: 3617, max: 4627, var: 138, min-max anomaly flag: Not set, pkt-loss
anomaly flag: Not set, exceeded threshold on SR Policy N:srte_c_10_ep_192.168.0.4, L:2
RP/0/0/CPU0:Sep 14 11:34:08.177 PDT: perf_meas[1001]:
%ROUTING-PERF_MEAS-5-PM_DELAY_EXCEEDS_THRESHOLD : Reason: PM session anomaly due to packet
loss, PM delay metric avg: 0, min: 0, max: 0, var: 0, min-max anomaly flag: Not set,
pkt-loss anomaly flag: Set, exceeded threshold on SR Policy N:srte_c_10_ep_192.168.0.4, L:2

RP/0/0/CPU0:Sep 14 11:35:38.180 PDT: perf_meas[1001]:
%ROUTING-PERF_MEAS-5-PM_DELAY_EXCEEDS_THRESHOLD : Reason: PM session anomaly due to packet
loss, PM delay metric avg: 3595, min: 3411, max: 3696, var: 58, min-max anomaly flag: Not
set, pkt-loss anomaly flag: Not set, exceeded threshold on SR Policy
N:srte_c_10_ep_192.168.0.4, L:2
```

=====

In this scenario, the router was running normally and then advertised a packet loss anomaly, the flag is set; later the router resumed to normal, the flag is unset, and it advertised a packet loss anomaly again.