Implementing IP Service Level Agreements

IP Service Level Agreements (IP SLAs) is a portfolio of technology embedded in most devices that run Cisco IOS XR Software, which allows you to analyze IP service levels for IP applications and services, increase productivity, lower operational costs, and reduce the frequency of network outages.

Using IP SLA, service provider customers can measure and provide service level agreements. IP SLA can perform network assessments, verify quality of service (QoS), ease the deployment of new services, and assist administrators with network troubleshooting.

Note For a complete description of the IP SLA commands used in this chapter, refer to the *IP Service Level Agreement Commands on Cisco IOS XR Software* module of *System Management Command Reference for Cisco NCS 6000 Series Routers*.

Feature History for Implementing IP Service Level Agreements

Release	Modification
Release 5.2.3	This feature was introduced.

Prerequisites for Implementing IP Service Level Agreements

Knowledge of general networking protocols and your specific network design is assumed. Familiarity with network management applications is helpful. We do not recommend scheduling all the operations at the same time as this could negatively affect your performance.

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

Restrictions for Implementing IP Service Level Agreements

- The maximum number of IP SLA operations that is supported by Cisco IOS XR Software is 2048.
- The maximum number of IP SLA configurable operations that is supported by Cisco IOS XR Software is 2000.
- The current validated scale numbers for scheduling UDP jitter operations is 100 operations with default frequency.
- We do not recommend scheduling all the operations at the same start time as this may affect the performance. At the same start time, not more than 10 operations per second should be scheduled. We recommend using the `start after` configuration.

Note Setting the frequency to less than 60 seconds will increase the number of packets sent. But this could negatively impact the performance of IP SLA operation when scheduled operations have same start time.

- IP SLA is not HA capable.
- Consider the following guidelines before configuring the frequency, timeout, and threshold commands.
  - For the UDP jitter operation, the following guidelines are recommended:
    - frequency > timeout + 2 seconds + num_packets * packet_interval
    - timeout > rtt_threshold
    - num_packet > loss_threshold

Information About Implementing IP Service Level Agreements

About IP Service Level Agreements Technology

IP SLA uses active traffic monitoring, which generates traffic in a continuous, reliable, and predictable manner to measure network performance. IP SLA sends data across the network to measure performance between multiple network locations or across multiple network paths. It simulates network data and IP services, and collects network performance information in real time. This information is collected:

- Response times
- One-way latency, jitter (interpacket delay variance)
- Packet loss
- Network resource availability

IP SLA originated from the technology previously known as Service Assurance Agent (SAA). IP SLA performs active monitoring by generating and analyzing traffic to measure performance, either between the router or from a router to a remote IP device such as a network application server. Measurement statistics, which are provided by the various IP SLA operations, are used for troubleshooting, problem analysis, and designing network topologies.

For a complete description of the object variables that are referenced by IP SLA, see the text of the CISCO-RTTMON-MIB.my file that is available from the Cisco MIB Locator.

Service Level Agreements

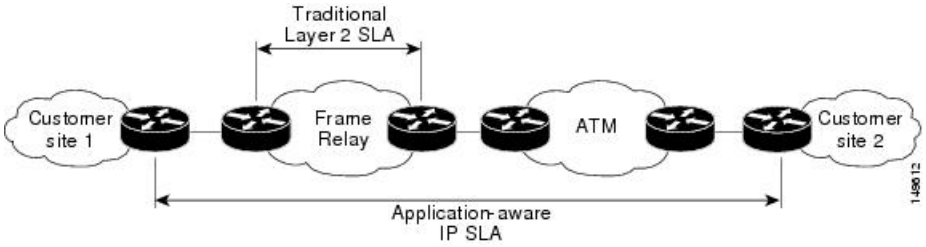
Internet commerce has grown significantly in the past few years as the technology has advanced to provide faster, more reliable access to the Internet. Many companies need online access and conduct most of their business on line and any loss of service can affect the profitability of the company. Internet service providers (ISPs) and even internal IT departments now offer a defined level of service—a service level agreement—to provide their customers with a degree of predictability.

Network administrators are required to support service level agreements that support application solutions. Figure 1: Scope of Traditional Service Level Agreement Versus IP SLA, on page 3 shows how IP SLA has taken the traditional concept of Layer 2 service level agreements and applied a broader scope to support end-to-end performance measurement, including support of applications.

Note

- Provided that the apllication and the IP-SLA processing rates support it, you can specify the flow rate for IP-SLA flow entries to up to 1500.
- To enable high performance for IP-SLA operations, avoid reuse of same source and destination ports for multiple IP SLA operations on the same device, especially when the scale is huge

Figure 1: Scope of Traditional Service Level Agreement Versus IP SLA



This table lists the improvements with IP SLA over a traditional service level agreement.

Table 1: IP SLA Improvements over a Traditional Service Level Agreement

Type of Improvement	Description
End-to-end measurements	The ability to measure performance from one end of the network to the other allows a broader reach and more accurate representation of the end-user experience.

Type of Improvement	Description
Sophistication	Statistics, such as delay, jitter, packet sequence, Layer 3 connectivity, and path and download time, that are divided into bidirectional and round-trip numbers provide more data than just the bandwidth of a Layer 2 link.
Accuracy	Applications that are sensitive to slight changes in network performance require the precision of the submillisecond measurement of IP SLA.
Ease of deployment	Leveraging the existing Cisco devices in a large network makes IP SLA easier to implement than the physical operations that are often required with traditional service level agreements.
Application-aware monitoring	IP SLA can simulate and measure performance statistics generated by applications running over Layer 3 through Layer 7. Traditional service level agreements can measure only Layer 2 performance.
Pervasiveness	IP SLA support exists in Cisco networking devices ranging from low-end to high-end routers and switches. This wide range of deployment gives IP SLA more flexibility over traditional service level agreements.

Benefits of IP Service Level Agreements

This table lists the benefits of implementing IP SLA.

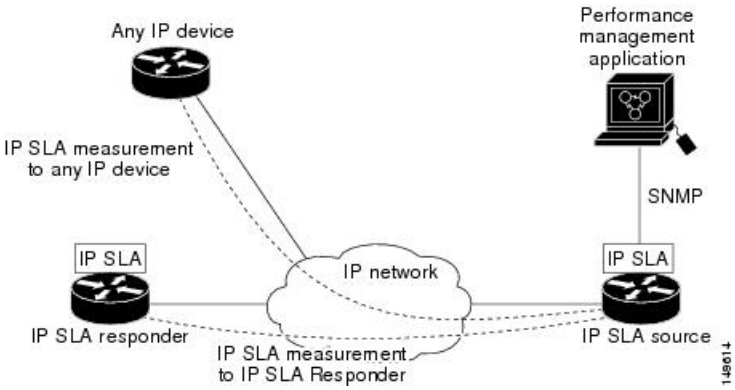
Table 2: List of Benefits for IP SLA

Benefit	Description
IP SLA monitoring	Provides service level agreement monitoring, measurement, and verification.
Network performance monitoring	Measure the jitter, latency, or packet loss in the network. In addition, IP SLA provides continuous, reliable, and predictable measurements along with proactive notification.
IP service network health assessment	Verifies that the existing QoS is sufficient for the new IP services.
Troubleshooting of network operation	Provides consistent, reliable measurement that immediately identifies problems and saves troubleshooting time.

Measuring Network Performance with IP Service Level Agreements

IP SLA uses generated traffic to measure network performance between two networking devices, such as routers. Figure 2: IP SLA Operations, on page 5 shows how IP SLA starts when the IP SLA device sends a generated packet to the destination device. After the destination device receives the packet and if the operation uses an IP SLA component at the receiving end (for example, IP SLA Responder), the reply packet includes information about the delay at the target device. The source device uses this information to improve the accuracy of the measurements. An IP SLA operation is a network measurement to a destination in the network from the source device using a specific protocol, such as User Datagram Protocol (UDP) for the operation.

Figure 2: IP SLA Operations



In responder-based operations, the IP SLA Responder is enabled in the destination device and provides information such as the processing delays of IP SLA packets. The responder-based operation offers the capability of unidirectional measurements. In replies to the IP SLA source device, the responder includes information about processing delays. The IP SLA source device removes the delays in its final performance calculation. Use of the responder is required for the UDP jitter operation.

To implement IP SLA network performance measurement, perform these tasks:

1. Enable the IP SLA Responder, if appropriate.
2. Configure the required IP SLA operation type.
3. Configure any options available for the specified IP SLA operation type.
4. Configure reaction conditions, if required.
5. Schedule the operation to run. Then, let the operation run for a period of time to gather statistics.
6. Display and interpret the results of the operation using Cisco IOS XR Software CLI, XML, or an NMS system with SNMP.

Operation Types for IP Service Level Agreements

IP SLA configures UDP jitter operations. It measures round-trip delay, one-way delay, one-way jitter, two-way jitter, and one-way packet loss.

IP SLA Responder and IP SLA Control Protocol

The IP SLA Responder is a component embedded in the destination Cisco routing device that allows the system to anticipate and respond to IP SLA request packets. The IP SLA Responder provides enhanced accuracy for measurements. The patented IP SLA Control Protocol is used by the IP SLA Responder, providing a mechanism through which the responder is notified on which port it should listen and respond. Only a Cisco IOS XR Software device or other Cisco platforms can be a source for a destination IP SLA Responder.

Figure 2: IP SLA Operations, on page 5 shows where the IP SLA Responder fits relative to the IP network. The IP SLA Responder listens on a specific port for control protocol messages sent by an IP SLA operation. Upon receipt of the control message, the responder enables the UDP port specified in the control message for the specified duration. During this time, the responder accepts the requests and responds to them. The responder

disables the port after it responds to the IP SLA packet or packets, or when the specified time expires. For added security, MD5 authentication for control messages is available.

Note The IP SLA responder needs at least one second to open a socket and program Local Packet Transport Services (LPTS). Therefore, configure the IP SLA timeout to at least 2000 milli seconds.

The IP SLA Responder must be used with the UDP jitter operation. If services that are already provided by the target router are chosen, the IP SLA Responder need not be enabled. For devices that are not Cisco devices, the IP SLA Responder cannot be configured, and the IP SLA can send operational packets only to services native to those devices.

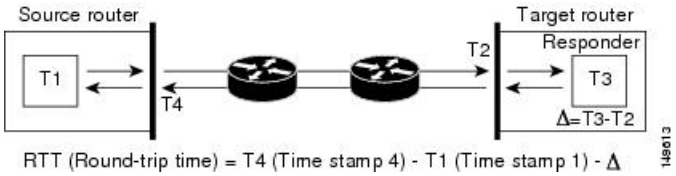
Response Time Computation for IP SLA

T3 is the time the reply packet is sent at the IP SLA Responder node, and T1 is the time the request is sent at the source node. Because of other high-priority processes, routers can take tens of milliseconds to process incoming packets. The delay affects the response times, because the reply to test packets might be sitting in a queue while waiting to be processed. In this situation, the response times would not accurately represent true network delays. IP SLA minimizes these processing delays on the source router and on the target router (if IP SLA Responder is being used) to determine true round-trip times. Some IP SLA probe packets contain delay information that are used in the final computation to make measurements more accurate.

When enabled, the IP SLA Responder allows the target device to take two time stamps, both when the packet arrives on the interface and again just as it is leaving, and accounts for it when calculating the statistics. This time stamping is made with a granularity of submilliseconds.

Figure 2: IP SLA Operations, on page 5 shows how the responder works. Four time stamps are taken to make the calculation for round-trip time. At the target router, with the responder functionality enabled, time stamp 2 (TS2) is subtracted from time stamp 3 (TS3) to produce the time spent processing the test packet as represented by delta. This delta value is then subtracted from the overall round-trip time. Notice that the same principle is applied by IP SLA on the source router on which the incoming time stamp 4 (TS4) is taken in a high-priority path to allow for greater accuracy.

Figure 3: IP SLA Responder Time Stamping



IP SLA Operation Scheduling

After an IP SLA operation is configured, you must schedule the operation to begin capturing statistics and collecting error information. When scheduling an operation, the operation starts immediately or starts at a certain month and day. In addition, an operation can be scheduled to be in pending state, which is used when the operation is a reaction (threshold) operation waiting to be triggered. Normal scheduling of IP SLA operations lets you schedule one operation at a time.

Note Multiple SLA probes with the same configuration (source and port number) must not be scheduled to run simultaneously.

IP SLA—Proactive Threshold Monitoring

This section describes the proactive monitoring capabilities for IP SLA that use thresholds and reaction triggering. IP SLA allows you to monitor, analyze, and verify IP service levels for IP applications and services to increase productivity, lower operational costs, and reduce occurrences of network congestion or outages. IP SLA uses active traffic monitoring to measure network performance.

To perform the tasks that are required to configure proactive threshold monitoring using IP SLA, you must understand these concepts:

IP SLA Reaction Configuration

IP SLA is configured to react to certain measured network conditions. For example, if IP SLA measures too much jitter on a connection, IP SLA can generate a notification to a network management application or trigger another IP SLA operation to gather more data.

IP SLA reaction configuration is performed by using the **ipsla reaction operation** command.

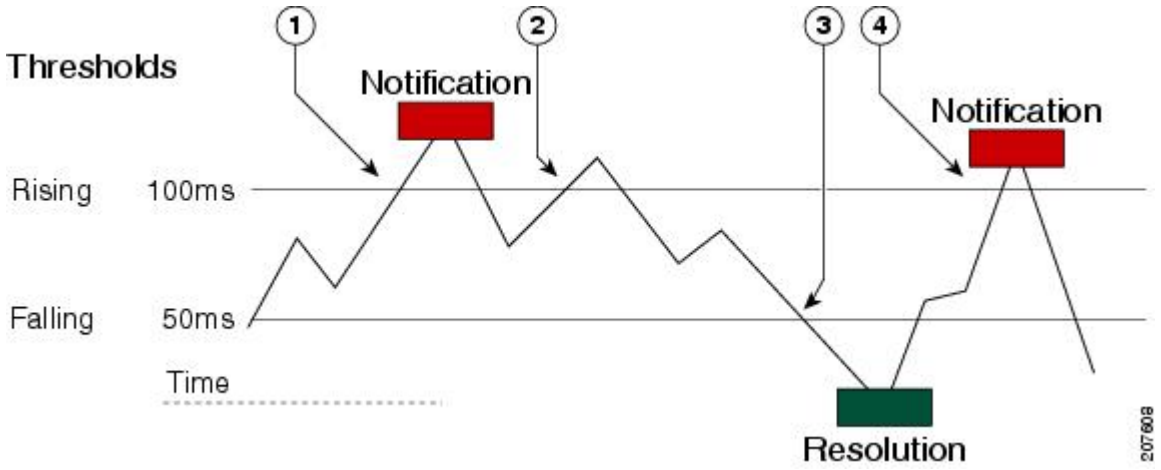
IP SLA Threshold Monitoring and Notifications

IP SLA supports threshold monitoring for performance parameters, such as jitter-average, bidirectional round-trip time, and connectivity. For packet loss and jitter, notifications can be generated for violations in either direction (for example, the source to the destination and the destination to the source) or for round-trip values.

Notifications are not issued for every occurrence of a threshold violation. An event is sent and a notification is issued when the rising threshold is exceeded for the first time. Subsequent threshold-exceeded notifications are issued only after the monitored value falls below the falling threshold before exceeding the rising threshold again.

The following figure illustrates the sequence for a triggered reaction that occurs when the monitored element exceeds the upper threshold.

Figure 4: IP SLAs Triggered Reaction Condition and Notifications for Threshold Exceeded



1	An event is sent and a threshold-exceeded notification is issued when the rising threshold is exceeded for the first time.
2	Consecutive over-rising threshold violations occur without issuing additional notifications.
3	The monitored value goes below the falling threshold.
4	Another threshold-exceeded notification is issued when the rising threshold is exceeded only after the monitored value first fell below the falling threshold.

Similarly, a lower-threshold notification is also issued the first time that the monitored element falls below the falling threshold. Subsequent notifications for lower-threshold violations are issued only after the rising threshold is exceeded before the monitored value falls below the falling threshold again.

How to Implement IP Service Level Agreements

Configuring IP Service Levels Using the UDP Jitter Operation

The IP SLA UDP jitter monitoring operation is designed to diagnose network suitability for real-time traffic applications such as VoIP, Video over IP, or real-time conferencing.

Jitter means interpacket delay variance. When multiple packets are sent consecutively from source to destination—for example, 10 ms apart—and if the network is behaving ideally, the destination can receive them 10 ms apart. But if there are delays in the network (for example, queuing, arriving through alternate routes, and so on), the arrival delay between packets can be greater than or less than 10 ms. Using this example, a positive jitter value indicates that the packets arrived more than 10 ms apart. If the packets arrive 12 ms apart, positive jitter is 2 ms; if the packets arrive 8 ms apart, negative jitter is 2 ms. For delay-sensitive networks like VoIP, positive jitter values are undesirable, and a jitter value of 0 is ideal.

However, the IP SLA UDP jitter operation does more than just monitor jitter. The packets that IP SLA generates carry sending sequence and receiving sequence information for the packets, and sending and receiving time stamps from the source and the operational target. Based on these, UDP jitter operations are capable of measuring the following functions:

- Per-direction jitter (source to destination and destination to source)
- Per-direction packet-loss
- Per-direction delay (one-way delay)
- Round-trip delay (average round-trip time)

As the paths for the sending and receiving of data may be different (asymmetric), the per-direction data allows you to more readily identify where congestion or other problems are occurring in the network.

The UDP jitter operation functions by generating synthetic (simulated) UDP traffic. By default, ten packet-frames (N), each with a payload size of 32 bytes (S) are generated every 20 ms (T), and the operation is repeated every 60 seconds (F). Each of these parameters is user-configurable, so as to best simulate the IP service you are providing, or want to provide.

This section contains these procedures:

Enabling the IP SLA Responder on the Destination Device

The IP SLA Responder must be enabled on the target device, which is the operational target.

By configuring the **ipsla responder** command, you make the IP SLA Responder open a UDP port 1967 and wait for a control request (not for probes). You can open or close a port dynamically through the IP SLA control protocol (through UDP port 1967). In addition, you can configure permanent ports.

Permanent ports are open until the configuration is removed. Agents can send IP SLA probe packets to the permanent port directly without a control request packet because the port can be opened by the configuration.

If you do not use permanent ports, you have to configure only the **ipsla responder** command.

To use a dynamic port, use the **ipsla responder** command, as shown in this example:

```
configure
ipsla responder
```

The dynamic port is opened through the IP SLA control protocol on the responder side when you start an operation on the agent side.

The example is configured as a permanent port on the responder. UDP jitter can use a dynamic port or a permanent port. If you use a permanent port for UDP jitter, there is no check performed for duplicated or out-of-sequence packets. This is because there is no control packet to indicate the start or end of the probe sequence. Therefore, the verification for sequence numbers are skipped when using permanent ports.

SUMMARY STEPS

1. **configure**
2. **ipsla responder**
3. **type udp ipv4 address** *ip-address* **port** *port*
4. Use the **commit** or **end** command.

DETAILED STEPS

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

Step 2	**ipsla responder**
	Example:
	`RP/0/RP0/CPU0:router(config)# ipsla responder`
	`RP/0/RP0/CPU0:router(config-ipsla-resp)#`
		Enables the IP SLA Responder for UDP jitter operations.

Step 3	**type udp ipv4 address** *ip-address* **port** *port* **Example:** `RP/0/RP0/CPU0:router(config-ipsla-resp)# type udp ipv4 address 12.25.26.10 port 10001`	Enables the permanent address and port on the IP SLA Responder.
Step 4	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

What to do next

After enabling the IP SLA Responder, see the Configuring and Scheduling a UDP Jitter Operation on the Source Device, on page 10 section.

Configuring and Scheduling a UDP Jitter Operation on the Source Device

The IP SLA operations function by generating synthetic (simulated) network traffic. A single IP SLA operation (for example, IP SLA operation 10) repeats at a given frequency for the lifetime of the operation.

A single UDP jitter operation consists of N UDP packets, each of size S, sent T milliseconds apart, from a source router to a target router, at a given frequency of F. By default, ten packets (N), each with a payload size of 32 bytes (S), are generated every 20 ms (T), and the operation is repeated every 60 seconds (F). Each of these parameters is user configurable, as shown in Table 3: UDP Jitter Operation Parameters, on page 11.

Table 3: UDP Jitter Operation Parameters

UDP Jitter Operation Parameter	Default	Configured Using
Number of packets (N)	10 packets	• **ipsla operation** command with the *operation-number* argument • **type udp jitter** command • **packet count** command with the *count* argument
Payload size per packet (S)	32 bytes	• **ipsla operation** command with the *operation-number* argument • **type udp jitter** command • **datasize request** command with the *size* argument
Time between packets, in milliseconds (T)	20 ms	• **ipsla operation** command with the *operation-number* argument • **type udp jitter** command • **packet interval** command with the *interval* argument
Elapsed time before the operation repeats, in seconds (F)	60 seconds	• **ipsla operation** command with the *operation-number* argument • **type udp jitter** command • **frequency** command with the *seconds* argument

Note If the **control disable** command is used to disable control packets while configuring IP SLA, the packets sent out from sender do not have sequence numbers. To calculate jitter, sequence number and time stamp values are required. So, jitter is not calculated when you use the **control disable** command.

Prerequisites for Configuring a UDP Jitter Operation on the Source Device

Use of the UDP jitter operation requires that the IP SLA Responder be enabled on the target Cisco device. To enable the IP SLA Responder, perform the task in the Enabling the IP SLA Responder on the Destination Device, on page 9 section.

Configuring and Scheduling a Basic UDP Jitter Operation on the Source Device

You can configure and schedule a UDP jitter operation.

SUMMARY STEPS

1. **configure**
2. **ipsla operation** *operation-number*
3. **type udp jitter**

4. **destination address** *ipv4address*
5. **destination port** *port*
6. **packet count** *count*
7. **packet interval** *interval*
8. **frequency** *seconds*
9. **exit**
10. **ipsla schedule operation** *op-num*
11. **life** { **forever** | *seconds*}
12. **ageout** *seconds*
13. **recurring**
14. **start-time** [*hh:mm:ss* {*day* | *month day*} | **now** | **pending** | **after** *hh:mm:ss*]
15. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters XR Config mode.
Step 2	**ipsla operation** *operation-number* **Example:** `RP/0/RP0/CPU0:router(config)# ipsla operation 432`	Specifies the operation number. The range is from 1 to 2048.
Step 3	**type udp jitter** **Example:** `RP/0/RP0/CPU0:router(config-ipsla-op)# type udp jitter`	Configures the operation as a UDP jitter operation, and configures characteristics for the operation.
Step 4	**destination address** *ipv4address* **Example:** `RP/0/RP0/CPU0:router(config-ipsla-udp-jitter)# destination address 12.25.26.10`	Specifies the IP address of the destination for the UDP jitter operation.
Step 5	**destination port** *port* **Example:** `RP/0/RP0/CPU0:router(config-ipsla-udp-jitter)# destination port 11111`	Specifies the destination port number, in the range from 1 to 65535.

	Command or Action	Purpose
Step 6	**packet count** *count* **Example:** `RP/0/RP0/CPU0:router(config-ipsla-udp-jitter)# packet count 30`	(Optional) Specifies the number of packets to be transmitted during a probe. For UDP jitter operation, the range is 1 to 60000. The default number of packets sent is 10.
Step 7	**packet interval** *interval* **Example:** `RP/0/RP0/CPU0:router(config-ipsla-udp-jitter)# packet interval 30`	(Optional) Specifies the time between packets. The default interval between packets is 20 milliseconds.
Step 8	**frequency** *seconds* **Example:** `RP/0/RP0/CPU0:router(config-ipsla-udp-jitter)# frequency 300`	(Optional) Sets the rate at which a specified IP SLA operation is sent into the network. • (Optional) Use the *seconds* argument to specify the number of seconds between the IP SLA operations. Valid values are in the range from 1 to 12604800 seconds. The default is 60 seconds.

Step 9	**exit**
	Example:
	`RP/0/RP0/CPU0:router(config-ipsla-udp-jitter)# exit`
	`RP/0/RP0/CPU0:router(config-ipsla-op)# exit`
	`RP/0/RP0/CPU0:router(config-ipsla)# exit`
	`RP/0/RP0/CPU0:router(config)#`
		Exits from IP SLA configuration mode and operational mode, and returns the CLI to XR Config mode.

Step 10	**ipsla schedule operation** *op-num*
	Example:
	`RP/0/RP0/CPU0:router(config)# ipsla schedule operation 432`
	`RP/0/RP0/CPU0:router(config-ipsla-sched)#`
		Schedules the start time of the operation. You can configure a basic schedule.

Step 11	**life** { **forever** | *seconds*} **Example:** `RP/0/RP0/CPU0:router(config-ipsla-sched)# life 30`	The **forever** keyword schedules the operation to run indefinitely. The *seconds* argument schedules the lifetime of the operation, in seconds. The default lifetime of an operation is 3600 seconds (one hour).
Step 12	**ageout** *seconds* **Example:** `RP/0/RP0/CPU0:router(config-ipsla-sched)# ageout 3600`	(Optional) Specifies the number of seconds to keep the operation in memory when it is not actively collecting information. The default value of 0 seconds means that the operation never times out.

	Command or Action	Purpose
Step 13	**recurring** **Example:** `RP/0/RP0/CPU0:router(config-ipsla-sched)# recurring`	(Optional) Specifies that the operation starts automatically at the specified time and for the specified duration every day.
Step 14	**start-time** [*hh:mm:ss* {*day* | *month day*} | **now** | **pending** | **after** *hh:mm:ss*] **Example:** `RP/0/RP0/CPU0:router(config-ipsla-sched)# start-time 01:00:00`	Specifies a time for the operation to start. The following keywords are described: • (Optional) Use the **pending** keyword to configure the operation to remain in a pending (unstarted) state. The default is inactive. If the **start-time** command is not specified, no information is collected until the start time is configured or a trigger occurs that performs an immediate start. • (Optional) Use the **now** keyword to indicate that the operation should start immediately. • (Optional) Use the **after** keyword and associated arguments to specify the time after which the operation starts collecting information.
Step 15	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configuring and Scheduling a UDP Jitter Operation with Additional Characteristics

You can configure and schedule a UDP jitter operation.

SUMMARY STEPS

1. **configure**
2. **ipsla operation** *operation-number*
3. **type udp jitter**
4. **vrf** *vrf-name*
5. **destination address** *ipv4address*
6. **destination port** *port*
7. **frequency** *seconds*

8. **statistics** [**hourly** | **interval** *seconds*]
9. **buckets** *hours*
10. **distribution count** *slot*
11. **distribution interval** *interval*
12. **exit**
13. **datasize request** *size*
14. **timeout** *milliseconds*
15. **tos** *number*
16. **exit**
17. **ipsla schedule operation** *op-num*
18. **life** {**forever** | *seconds*}
19. **ageout** *seconds*
20. **recurring**
21. **start-time** [*hh:mm:ss* {*day* | *month day*} | **now** | **pending** | **after** *hh:mm:ss*]
22. Use the **commit** or **end** command.
23. **show ipsla statistics** [*operation-number*]
24. **show ipsla statistics aggregated** [*operation-number*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters XR Config mode.
Step 2	**ipsla operation** *operation-number* **Example:** `RP/0/RP0/CPU0:router(config)# ipsla operation 432`	Specifies the operation number. The range is from 1 to 2048.
Step 3	**type udp jitter** **Example:** `RP/0/RP0/CPU0:router(config-ipsla-op)# type udp jitter`	Configures the operation as a UDP jitter operation, and configures characteristics for the operation.
Step 4	**vrf** *vrf-name* **Example:** `RP/0/RP0/CPU0:router(config-ipsla-udp-jitter)# vrf VPN-A`	(Optional) Enables the monitoring of a VPN (using a nondefault routing table) in a UDP jitter operation. Maximum length is 32 alphanumeric characters.
Step 5	**destination address** *ipv4address* **Example:**	Specifies the IP address of the destination for the proper operation type.

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router(config-ipsla-udp-jitter)# destination address 12.25.26.10`	
Step 6	**destination port** *port* **Example:** `RP/0/RP0/CPU0:router(config-ipsla-udp-jitter)# destination port 11111`	Specifies the destination port number, in the range from 1 to 65535.
Step 7	**frequency** *seconds* **Example:** `RP/0/RP0/CPU0:router(config-ipsla-udp-jitter)# frequency 300`	(Optional) Sets the rate at which a specified IP SLA operation is sent into the network. • (Optional) Use the *seconds* argument to specify the number of seconds between the IP SLA operations. Valid values are in the range from 1 to 12604800 seconds. The default is 60 seconds.

Step 8	**statistics** [**hourly** | **interval** *seconds*]
	Example:
	`RP/0/RP0/CPU0:router(config-ipsla-udp-jitter)# statistics hourly`
	`RP/0/RP0/CPU0:router(config-ipsla-op-stats)#`
		(Optional) Specifies the statistics collection parameters for UDP jitter operation.

Step 9	**buckets** *hours* **Example:** `RP/0/RP0/CPU0:router(config-ipsla-op-stats)# buckets 10`	(Optional) Sets the number of hours in which statistics are maintained for the IP SLA operations. This command is valid only with the **statistics** command with **hourly** keyword. The range is 0 to 25 hours. The default value is 2 hours.
Step 10	**distribution count** *slot* **Example:** `RP/0/RP0/CPU0:router(config-ipsla-op-stats)# distribution count 15`	(Optional) Sets the number of statistic distributions that are kept for each hop during the lifetime of the IP SLA operation. The range is 1 to 20. The default value is 1 distribution.
Step 11	**distribution interval** *interval* **Example:** `RP/0/RP0/CPU0:router(config-ipsla-op-stats)# distribution interval 20`	(Optional) Sets the time interval for each statistical distribution. The range is 1 to 100 ms. The default value is 20 ms.
Step 12	**exit** **Example:** `RP/0/RP0/CPU0:router(config-ipsla-op-stats)# exit`	Exits from IP SLA statistics configuration mode.

	Command or Action	Purpose
Step 13	**datasize request** *size* **Example:** `RP/0/RP0/CPU0:router(config-ipsla-udp-jitter)# datasize request 512`	(Optional) Sets the data size in the payload of the operation's request packets. For UDP jitter, the range is from 28 to 1500 bytes.
Step 14	**timeout** *milliseconds* **Example:** `RP/0/RP0/CPU0:router(config-ipsla-udp-jitter)# timeout 10000`	Sets the time that the specified IP SLA operation waits for a response from its request packet. • (Optional) Use the *milliseconds* argument to specify the number of milliseconds that the operation waits to receive a response.
Step 15	**tos** *number* **Example:** `RP/0/RP0/CPU0:router(config-ipsla-udp-jitter)# tos 255`	Specifies the type of service number.

Step 16	**exit**
	Example:
	`RP/0/RP0/CPU0:router(config-ipsla-udp-jitter)# exit`
	`RP/0/RP0/CPU0:router(config-ipsla-op)# exit`
	`RP/0/RP0/CPU0:router(config-ipsla)# exit`
	`RP/0/RP0/CPU0:router(config)#`
		Exits from IP SLA configuration mode and operational mode, and returns the CLI to XR Config mode.

Step 17	**ipsla schedule operation** *op-num*
	Example:
	`RP/0/RP0/CPU0:router(config)# ipsla schedule operation 432`
	`RP/0/RP0/CPU0:router(config-ipsla-sched)#`
		Schedules the start time of the operation. You can configure a basic schedule.

Step 18	**life** {**forever** | *seconds*} **Example:** `RP/0/RP0/CPU0:router(config-ipsla-sched)# life 30`	The **forever** keyword schedules the operation to run indefinitely. The *seconds* argument schedules the lifetime of the operation, in seconds. The default lifetime of an operation is 3600 seconds (one hour).
Step 19	**ageout** *seconds* **Example:** `RP/0/RP0/CPU0:router(config-ipsla-sched)# ageout 3600`	(Optional) Specifies the number of seconds to keep the operation in memory when it is not actively collecting information. The default value of 0 seconds means that the operation never times out.

	Command or Action	Purpose
Step 20	**recurring** **Example:** `RP/0/RP0/CPU0:router(config-ipsla-sched)# recurring`	(Optional) Specifies that the operation starts automatically at the specified time and for the specified duration every day.
Step 21	**start-time** [*hh:mm:ss* {*day* | *month day*} | **now** | **pending** | **after** *hh:mm:ss*] **Example:** `RP/0/RP0/CPU0:router(config-ipsla-sched)# start-time 01:00:00`	(Optional) Specifies a time for the operation to start. The following keywords are described: • (Optional) Use the **pending** keyword to configure the operation to remain in a pending (unstarted) state. The default is inactive. If the **start-time** command is not specified, no information is collected until the start time is configured or a trigger occurs that performs an immediate start. • (Optional) Use the **now** keyword to indicate that the operation should start immediately. • (Optional) Use the **after** keyword and associated arguments to specify the time after which the operation starts collecting information.
Step 22	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.
Step 23	**show ipsla statistics** [*operation-number*] **Example:** `RP/0/RP0/CPU0:router # show ipsla statistics 432`	Displays the current statistics.
Step 24	**show ipsla statistics aggregated** [*operation-number*] **Example:** `RP/0/RP0/CPU0:router # show ipsla statistics aggregated 432`	Returns the hourly statistics (aggregated data) on the performance of the network. The UDP jitter operation provides the following hourly statistics: • Jitter statistics—Interprets telephony and multimedia conferencing requirements. • Packet loss and packet sequencing statistics—Interprets telephony, multimedia

	Command or Action	Purpose
		conferencing, streaming media, and other low-latency data requirements. • One-way latency and delay statistics—Interprets telephony, multimedia conferencing, and streaming media requirements.

Configuring IP SLA Reactions and Threshold Monitoring

If you want IP SLA to set some threshold and inform you of a threshold violation, the **ipsla reaction operation** command and the **ipsla reaction trigger** command are required. Perform the following procedures to configure IP SLA reactions and threshold monitoring:

Configuring Monitored Elements for IP SLA Reactions

IP SLA reactions are configured to be triggered when a monitored value exceeds or falls below a specified level or a monitored event (for example, timeout or connection-loss) occurs. These monitored values and events are called monitored elements. You can configure the conditions for a reaction to occur in a particular operation.

The types of monitored elements that are available are presented in the following sections:

Configuring Triggers for Connection-Loss Violations

You can configure a reaction if there is a connection-loss for the monitored operation.

SUMMARY STEPS

1. **configure**
2. **ipsla reaction operation** *operation-number*
3. **react** [**connection-loss**]
4. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters XR Config mode.
Step 2	**ipsla reaction operation** *operation-number* **Example:** `RP/0/RP0/CPU0:router(config)# ipsla reaction operation 432`	Configures certain actions that are based on events under the control of the IP SLA agent. The *operation-number* argument is the number of the IP SLA operations for the reactions that are configured. The range is from 1 to 2048.
Step 3	**react** [**connection-loss**]	Specifies an element to be monitored for a reaction.

	Command or Action	Purpose

	Example:
	
	`RP/0/RP0/CPU0:router(config-ipsla-react)# react connection-loss`
	`RP/0/RP0/CPU0:router(config-ipsla-react-cond)#`
		Use the **connection-loss** keyword to specify a reaction that occurs if there is a connection-loss for the monitored operation.

Step 4	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configuring Triggers for Jitter Violations

Jitter values are computed as source-to-destination and destination-to-source values. Events, for example, traps, can be triggered when the jitter value in either direction or both directions rises above a specified threshold or falls below a specified threshold. You can configure jitter-average as a monitored element.

SUMMARY STEPS

1. **configure**
2. **ipsla reaction operation** *operation-number*
3. **react** [**jitter-average** {**dest-to-source** | **source-to-dest**}]
4. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters XR Config mode.
Step 2	**ipsla reaction operation** *operation-number* **Example:** `RP/0/RP0/CPU0:router(config)# ipsla reaction operation 432`	Configures certain actions that are based on events under the control of the IP SLA agent. The *operation-number* argument is the number of the IP SLA operations for the reactions that are configured. The range is from 1 to 2048.
Step 3	**react** [**jitter-average** {**dest-to-source**	**source-to-dest**}] **Example:**	Specifies an element to be monitored for a reaction.

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router(config-ipsla-react)# react jitter-average` `RP/0/RP0/CPU0:router(config-ipsla-react-cond)#`	A reaction occurs if the average round-trip jitter value violates the upper threshold or lower threshold. The following options are listed for the **jitter-average** keyword: • **dest-to-source**—Specifies the jitter average destination to source (DS). • **source-to-dest**—Specifies the jitter average source to destination (SD).
Step 4	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configuring Triggers for Packet Loss Violations

Packet-loss values are computed as source-to-destination and destination-to-source values. Events, for example, traps, can be triggered when the packet-loss values in either direction rise above a specified threshold or fall below a specified threshold. Perform this task to configure packet-loss as a monitored element.

SUMMARY STEPS

1. **configure**
2. **ipsla reaction operation** *operation-number*
3. **react** [**packet-loss** [**dest-to-source** | **source-to-dest**]]
4. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters XR Config mode.
Step 2	**ipsla reaction operation** *operation-number* **Example:** `RP/0/RP0/CPU0:router(config)# ipsla reaction operation 432`	Configures certain actions that are based on events under the control of the IP SLA agent. The *operation-number* argument is the number of the IP SLA operations for the reactions that are configured. The range is from 1 to 2048.

	Command or Action	Purpose

Step 3	**react** [**packet-loss** [**dest-to-source** | **source-to-dest**]]
	Example:
	`RP/0/RP0/CPU0:router(config-ipsla-react)# react packet-loss dest-to-source`
	`RP/0/RP0/CPU0:router(config-ipsla-react-cond)#`
		Specifies an element to be monitored for a reaction.
		The reaction on packet loss value violation is specified. The following options are listed for the **packet-loss** keyword:
		• **dest-to-source**—Specifies the packet loss destination to source (DS) violation.
		• **source-to-dest**—Specifies the packet loss source to destination (SD) violation.

Step 4	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configuring Triggers for Round-Trip Violations

Round-trip time (RTT) is a monitored value of all IP SLA operations. Events, for example, traps, can be triggered when the rtt value rises above a specified threshold or falls below a specified threshold. You can configure rtt as a monitored element.

SUMMARY STEPS

1. **configure**
2. **ipsla reaction operation** *operation-number*
3. **react** [**rtt**]
4. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters XR Config mode.
Step 2	**ipsla reaction operation** *operation-number* **Example:**	Configures certain actions that are based on events under the control of the IP SLA agent. The *operation-number* argument is the number of the IP SLA operations for the reactions that are configured. The range is from 1 to 2048.

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router(config)# ipsla reaction operation 432`	

Step 3	**react** [**rtt**]
	Example:
	`RP/0/RP0/CPU0:router(config-ipsla-react)# react rtt`
	`RP/0/RP0/CPU0:router(config-ipsla-react-cond)#`
		Specifies an element to be monitored for a reaction.
		Use the **rtt** keyword to specify a reaction that occurs if the round-trip value violates the upper threshold or lower threshold.

Step 4	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configuring Triggers for Timeout Violations

You can configure triggers for timeout violations.

SUMMARY STEPS

1. **configure**
2. **ipsla reaction operation** *operation-number*
3. **react** [**timeout**]
4. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters XR Config mode.
Step 2	**ipsla reaction operation** *operation-number* **Example:** `RP/0/RP0/CPU0:router(config)# ipsla reaction operation 432`	Configures certain actions that are based on events under the control of the IP SLA agent. The *operation-number* argument is the number of the IP SLA operations for the reactions that are configured. The range is from 1 to 2048.
Step 3	**react** [**timeout**]	Specifies an element to be monitored for a reaction.

	Command or Action	Purpose

	Example:
	
	`RP/0/RP0/CPU0:router(config-ipsla-react)# react timeout`
	`RP/0/RP0/CPU0:router(config-ipsla-react-cond)#`
		Use the **timeout** keyword to specify a reaction that occurs if there is a timeout for the monitored operation.

Step 4	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configuring Triggers for Verify Error Violations

You can specify a reaction if there is an error verification violation.

SUMMARY STEPS

1. **configure**
2. **ipsla reaction operation** *operation-number*
3. **react** [**verify-error**]
4. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters XR Config mode.
Step 2	**ipsla reaction operation** *operation-number* **Example:** `RP/0/RP0/CPU0:router(config)# ipsla reaction operation 432`	Configures certain actions that are based on events under the control of the IP SLA agent. The *operation-number* argument is the number of the IP SLA operations for the reactions that are configured. The range is from 1 to 2048.
Step 3	**react** [**verify-error**] **Example:** `RP/0/RP0/CPU0:router(config-ipsla-react)# react`	Specifies an element to be monitored for a reaction. Use the **verify-error** keyword to specify a reaction that occurs if there is an error verification violation.

	Command or Action	Purpose
	`verify-error` `RP/0/RP0/CPU0:router(config-ipsla-react-cond)#`	
Step 4	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configuring Threshold Violation Types for IP SLA Reactions

For each monitored element, you can specify:

- Condition to check for the threshold value.
- Pattern of occurrences of the condition that can generate the reaction, such as a threshold type.

For example, you can specify that a reaction can occur for a particular element as soon as you observe the condition of interest by using the **threshold type immediate** command or when you observe the condition for three consecutive times by using the **threshold type consecutive** command.

The type of threshold defines the type of threshold violation (or combination of threshold violations) that triggers an event.

This table lists the threshold violation types.

Table 4: Threshold Violation Types for IP SLA Reactions

Type of Threshold Violation	Description
consecutive	Triggers an event only after a violation occurs a number of times consecutively. For example, the consecutive violation type can be used to configure an action to occur after a timeout occurs five times in a row or when the round-trip time exceeds the upper threshold value five times in a row. For more information, see Generating Events for Consecutive Violations, on page 27.
immediate	Triggers an event immediately when the value for a reaction type (such as response time) exceeds the upper threshold value or falls below the lower threshold value or when a timeout, connection-loss, or verify-error event occurs. For more information, see Generating Events for Each Violation, on page 26.
X of Y	Triggers an event after some number (X) of violations within some other number (Y) of probe operations (X of Y). For more information, see Generating Events for X of Y Violations, on page 28.

Type of Threshold Violation	Description
averaged	Triggers an event when the averaged totals of a value for X number of probe operations exceeds the specified upper-threshold value or falls below the lower-threshold value. For more information, see Generating Events for Averaged Violations, on page 29.

Generating Events for Each Violation

You can generate a trap or trigger another operation each time a specified condition is met.

SUMMARY STEPS

1. **configure**
2. **ipsla reaction operation** *operation-number*
3. **react** [**connection-loss** | **jitter-average** {**dest-to-source** | **source-to-dest**} | **packet-loss** [**dest-to-source** | **source-to-dest**] | **rtt** | **timeout** | **verify-error**]
4. **threshold type immediate**
5. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters XR Config mode.
Step 2	**ipsla reaction operation** *operation-number* **Example:** `RP/0/RP0/CPU0:router(config)# ipsla reaction operation 432`	Configures certain actions that are based on events under the control of the IP SLA agent. The *operation-number* argument is the number of the IP SLA operations for the reactions that are configured. The range is from 1 to 2048.

Step 3	**react** [**connection-loss**	**jitter-average** {**dest-to-source**	**source-to-dest**}	**packet-loss** [**dest-to-source**	**source-to-dest**]	**rtt**	**timeout**	**verify-error**]
								Example:
								`RP/0/RP0/CPU0:router(config-ipsla-react)# react timeout`
								`RP/0/RP0/CPU0:router(config-ipsla-react-cond)#`
									Specifies an element to be monitored for a reaction.
									A reaction is specified if there is a timeout for the monitored operation.

Step 4	**threshold type immediate** **Example:** `RP/0/RP0/CPU0:router(config-ipsla-react-cond)# threshold type immediate`	Takes action immediately upon a threshold violation.

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

Generating Events for Consecutive Violations

You can generate a trap or trigger another operation after a certain number of consecutive violations.

SUMMARY STEPS

1. **configure**
2. **ipsla reaction operation** *operation-number*
3. **react** [**connection-loss** | **jitter-average** {**dest-to-source** | **source-to-dest**} | **packet-loss** [**dest-to-source** | **source-to-dest**] | **rtt** | **timeout** | **verify-error**]
4. **threshold type consecutive** *occurrences*
5. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters XR Config mode.
Step 2	**ipsla reaction operation** *operation-number* **Example:** `RP/0/RP0/CPU0:router(config)# ipsla reaction operation 432`	Configures certain actions that are based on events under the control of the IP SLA agent. The *operation-number* argument is the number of the IP SLA operations for the reactions that are configured. The range is from 1 to 2048.

Step 3	**react** [**connection-loss**	**jitter-average** {**dest-to-source**	**source-to-dest**}	**packet-loss** [**dest-to-source**	**source-to-dest**]	**rtt**	**timeout**	**verify-error**]
								Example:
								`RP/0/RP0/CPU0:router(config-ipsla-react)# react connection-loss`
								`RP/0/RP0/CPU0:router(config-ipsla-react-cond)#`
									Specifies an element to be monitored for a reaction.
									A reaction is specified if there is a connection-loss for the monitored operation.

	Command or Action	Purpose
Step 4	**threshold type consecutive** *occurrences* **Example:** `RP/0/RP0/CPU0:router(config-ipsla-react-cond)# threshold type consecutive 8`	Takes action after a number of consecutive violations. When the reaction condition is set for a consecutive number of occurrences, there is no default value. The number of occurrences is set when specifying the threshold type. The number of consecutive violations is from 1 to 16.
Step 5	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Generating Events for X of Y Violations

You can generate a trap or trigger another operation after some number (X) of violations within some other number (Y) of probe operations (X of Y). The **react** command with the **rtt** keyword is used as an example.

SUMMARY STEPS

1. **configure**
2. **ipsla reaction operation** *operation-number*
3. **react** [**connection-loss** | **jitter-average** {**dest-to-source** | **source-to-dest**} | **packet-loss** [**dest-to-source** | **source-to-dest**] | **rtt** | **timeout** | **verify-error**]
4. **threshold type xofy** *X value Y value*
5. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters XR Config mode.
Step 2	**ipsla reaction operation** *operation-number* **Example:** `RP/0/RP0/CPU0:router(config)# ipsla reaction operation 432`	Configures certain actions that are based on events under the control of the IP SLA agent. The *operation-number* argument is the number of the IP SLA operations for the reactions that are configured. The range is from 1 to 2048.

	Command or Action	Purpose

Step 3	**react** [**connection-loss** | **jitter-average** {**dest-to-source** | **source-to-dest**} | **packet-loss** [**dest-to-source** | **source-to-dest**] | **rtt** | **timeout** | **verify-error**]
	Example:
	`RP/0/RP0/CPU0:router(config-ipsla-react)# react rtt`
	`RP/0/RP0/CPU0:router(config-ipsla-react-cond)#`
		Specifies that a reaction occurs if the round-trip value violates the upper threshold or lower threshold.

Step 4	**threshold type xofy** *X value Y value* **Example:** `RP/0/RP0/CPU0:router(config-ipsla-react-cond)# threshold type xofy 7 7`	When the reaction condition, such as threshold violations, are met for the monitored element after some *x* number of violations within some other *y* number of probe operations (for example, *x* of *y*), the action is performed as defined by the **action** command. The default is 5 for both *x value* and *y value;* for example, **xofy** 5 5. The valid range for each value is from 1 to 16.
Step 5	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Generating Events for Averaged Violations

You can generate a trap or trigger another operation when the averaged totals of X number of probe operations violate a falling threshold or rising threshold.

SUMMARY STEPS

1. **configure**
2. **ipsla reaction operation** *operation-number*
3. **react** [**connection-loss** | **jitter-average** {**dest-to-source** | **source-to-dest**} | **packet-loss** [**dest-to-source** | **source-to-dest**] | **rtt** | **timeout** | **verify-error**]
4. **threshold type average** *number-of-probes*
5. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:**	Enters XR Config mode.

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router# configure`	
Step 2	**ipsla reaction operation** *operation-number* **Example:** `RP/0/RP0/CPU0:router(config)# ipsla reaction operation 432`	Configures certain actions that are based on events under the control of the IP SLA agent. The *operation-number* argument is the number of the IP SLA operations for the reactions that are configured. The range is from 1 to 2048.

Step 3	**react** [**connection-loss** | **jitter-average** {**dest-to-source** | **source-to-dest**} | **packet-loss** [**dest-to-source** | **source-to-dest**] | **rtt** | **timeout** | **verify-error**]
	Example:
	`RP/0/RP0/CPU0:router(config-ipsla-react)# react packet-loss dest-to-source`
	`RP/0/RP0/CPU0:router(config-ipsla-react-cond)#`
		Specifies an element to be monitored for a reaction.
		The reaction on packet loss value violation is specified. The following options are listed for the **packet-loss** keyword:
		• **dest-to-source**—Specifies the packet loss destination to source (DS) violation.
		• **source-to-dest**—Specifies the packet loss source to destination (SD) violation.

Step 4	**threshold type average** *number-of-probes* **Example:** `RP/0/RP0/CPU0:router(config-ipsla-react-cond)# threshold type average 8`	Takes action on average values to violate a threshold.
Step 5	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Specifying Reaction Events

When a reaction condition is detected, you can configure the type of action that occurs by using the **action** command. The following types of actions are configured:

- **logging**—When the **logging** keyword is configured, a message is generated to the console to indicate that a reaction has occurred.
- **trigger**—When the **trigger** keyword is configured, one or more other operations can be started. As a result, you can control which operations can be started with the **ipsla reaction trigger** *op1 op2* command. This command indicates when *op1* generates an action type trigger and operation *op2* can be started.

You can specify reaction events. The **react** command with the **connection-loss** keyword is used as an example.

SUMMARY STEPS

1. **configure**
2. **ipsla reaction operation** *operation-number*
3. **react** [**connection-loss** | **jitter-average** {**dest-to-source** | **source-to-dest**} | **packet-loss** [**dest-to-source** | **source-to-dest**] | **rtt** | **timeout** | **verify-error**]
4. **action** [**logging** | **trigger**]
5. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters XR Config mode.
Step 2	**ipsla reaction operation** *operation-number* **Example:** `RP/0/RP0/CPU0:router(config)# ipsla reaction operation 432`	Configures certain actions that are based on events under the control of the IP SLA agent. The *operation-number* argument is the number of the IP SLA operations for the reactions that are configured. The range is from 1 to 2048.

Step 3	**react** [**connection-loss** | **jitter-average** {**dest-to-source** | **source-to-dest**} | **packet-loss** [**dest-to-source** | **source-to-dest**] | **rtt** | **timeout** | **verify-error**]
	Example:
	`RP/0/RP0/CPU0:router(config-ipsla-react)# react connection-loss`
	`RP/0/RP0/CPU0:router(config-ipsla-react-cond)#`
		Specifies a reaction if there is a connection-loss for the monitored operation.

Step 4	**action** [**logging** | **trigger**] **Example:** `RP/0/RP0/CPU0:router(config-ipsla-react-cond)# action logging`	Specifies what action or combination of actions the operation performs when you configure the **react** command or when threshold events occur. The following action types are described: • **logging**—Sends a logging message when the specified violation type occurs for the monitored element. The IP SLA agent generates a syslog and informs SNMP. Then, it is up to the SNMP agent to generate a trap or not. • **trigger**—Determines that the operational state of one or more operations makes the transition from pending to active when the violation conditions are met. The target operations to be triggered are specified using the **ipsla reaction trigger** command. A target operation continues until its life expires, as specified by lifetime value of the target operation. A triggered

	Command or Action	Purpose
		target operation must finish its life before it can be triggered again.
Step 5	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configuration Examples for Implementing IP Service Level Agreements

This section provides these configuration examples:

Configuring IP Service Level Agreements: Example

The following example shows how to configure and schedule a UDP jitter operation:

```
configure
ipsla
 operation 101
  type udp jitter
   destination address 12.2.0.2
   statistics hourly
    buckets 5
    distribution count 5
    distribution interval 1
   !
   destination port 400
   statistics interval 120
    buckets 5
   !
  !
 !
 schedule operation 101
  start-time now
  life forever
 !
!


show ipsla statistics
Fri Nov 28 16:48:48.286 GMT
```

```
Entry number: 101
    Modification time: 16:39:36.608 GMT Fri Nov 28 2014
    Start time       : 16:39:36.633 GMT Fri Nov 28 2014
    Number of operations attempted: 10
    Number of operations skipped  : 0
    Current seconds left in Life  : Forever
    Operational state of entry    : Active
    Operational frequency(seconds): 60
    Connection loss occurred      : FALSE
    Timeout occurred              : FALSE
    Latest RTT (milliseconds)     : 3
    Latest operation start time   : 16:48:37.653 GMT Fri Nov 28 2014
    Next operation start time     : 16:49:37.653 GMT Fri Nov 28 2014
    Latest operation return code  : OK
    RTT Values:
      RTTAvg  : 3          RTTMin: 3          RTTMax : 4
      NumOfRTT: 10         RTTSum: 33         RTTSum2: 111
    Packet Loss Values:
      PacketLossSD       : 0          PacketLossDS : 0
      PacketOutOfSequence: 0          PacketMIA    : 0
      PacketLateArrival  : 0          PacketSkipped: 0
      Errors             : 0          Busies       : 0
      InvalidTimestamp   : 0
    Jitter Values :
      MinOfPositivesSD: 1          MaxOfPositivesSD: 1
      NumOfPositivesSD: 2          SumOfPositivesSD: 2
      Sum2PositivesSD : 2
      MinOfNegativesSD: 1          MaxOfNegativesSD: 1
      NumOfNegativesSD: 1          SumOfNegativesSD: 1
      Sum2NegativesSD : 1
      MinOfPositivesDS: 1          MaxOfPositivesDS: 1
      NumOfPositivesDS: 1          SumOfPositivesDS: 1
      Sum2PositivesDS : 1
      MinOfNegativesDS: 1          MaxOfNegativesDS: 1
      NumOfNegativesDS: 1          SumOfNegativesDS: 1
      Sum2NegativesDS : 1
      JitterAve: 1         JitterSDAve: 1      JitterDSAve: 1
      Interarrival jitterout: 0              Interarrival jitterin: 0
    One Way Values :
      NumOfOW: 0
      OWMinSD : 0          OWMaxSD: 0          OWSumSD: 0
      OWSum2SD: 0          OWAveSD: 0
      OWMinDS : 0          OWMaxDS: 0          OWSumDS: 0
      OWSum2DS: 0          OWAveDS: 0
```

Configuring IP SLA Reactions and Threshold Monitoring: Example

The following examples show how to configure IP SLA reactions and threshold monitoring.

```
configure
ipsla
 operation 101
  type udp jitter
   destination address 12.2.0.2
   statistics hourly
    buckets 5
    distribution count 5
    distribution interval 1
   exit
   destination port 400
   statistics interval 120
```

```
    buckets 5
   exit
  exit
 exit
 reaction operation 101
  react timeout
   action trigger
   threshold type immediate
  exit
  react rtt
   action logging
   threshold lower-limit 4 upper-limit 5
  exit
 exit
 schedule operation 101
  start-time now
  life forever
 exit
exit
```