

# Table of Contents

|  |   |
|--|---|
| <b><u>Changes in RSVP Debug Commands in Cisco IOS Software Release 12.0(24)S and Later</u></b> ..... | 1 |
| <u>Document ID: 42321</u> .....  | 1 |
| .....  | 1 |
| <u>Introduction</u> .....  | 1 |
| <u>Prerequisites</u> .....   | 1 |
| <u>Requirements</u> .....  | 1 |
| <u>Components Used</u> .....   | 1 |
| <u>Conventions</u> .....   | 2 |
| <u>Differences in Enabling Debugs to Monitor PATH and RESV Messages</u> .....                        | 2 |
| <u>Differences in RSVP debug Command Options</u> .....   | 2 |
| <u>Differences in Enabling Debugs to Monitor PATH and RESV Messages</u> .....                        | 3 |
| <u>Using ACLs to Selectively Debug RSVP Messages</u> .....   | 4 |
| <u>Related Information</u> .....   | 4 |

# Changes in RSVP Debug Commands in Cisco IOS Software Release 12.0(24)S and Later

Document ID: 42321

---

## Introduction

### Prerequisites

- Requirements
- Components Used
- Conventions

### Differences in Enabling Debugs to Monitor PATH and RESV Messages

- Differences in RSVP debug Command Options
- Differences in Enabling Debugs to Monitor PATH and RESV Messages
- Using ACLs to Selectively Debug RSVP Messages

### Related Information

---

## Introduction

The introduction of Cisco IOS® Software Release 12.0(24)S changed Resource Reservation Protocol (RSVP) debugs in both the output and in the command line interface (CLI). This change was part of a project to make the debugs easier to use and to make them more consistent with other router output. This document explains the new debugs, primarily the CLI. Debug output is largely self-explanatory, so is not explored in depth here. The emphasis in this document is on debugs that are useful for Multiprotocol Label Switching Traffic Engineering (MPLS TE), as opposed to classic RSVP.

## Prerequisites

### Requirements

Readers of this document are required to know:

- MPLS TE terms and concepts

### Components Used

The information in this document is based on the software and hardware versions below.

- Gigabit Switch Router (GSR) Series
- Cisco IOS Software Releases 12.0(23)S and 12.0(24)S

The information presented in this document was created from devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If you are working in a live network, ensure that you understand the potential impact of any command before using it.

## Conventions

For more information on document conventions, refer to Cisco Technical Tips Conventions.

## Differences in Enabling Debugs to Monitor PATH and RESV Messages

Changes to the RSVP **debug** commands were introduced in Cisco IOS 12.0(24)S in order to make it easier to use the **debug** commands and to make the **debug** output more consistent with other router output. Below is an example of the commands available in Cisco 12.0(23)S and earlier compared to the **debug** commands available in Cisco IOS 12.0(24)S and later. As you can see, there are significantly more options in Cisco IOS version 12.0(24)S.

### Differences in RSVP debug Command Options

Below are the RSVP **debug** command options when running Cisco IOS version 12.0(23)S or earlier.

```
gsr6# debug ip rsvp ?
<1-199>      Access list
detail       RSVP packet contents
fast-reroute RSVP Fast ReRoute support for LSPs
hello        RSVP Hello Extension Signalling
path         RSVP packet contents (PATH only)
policy       RSVP policy information
resv         RSVP packet contents (RESV only)
timeouts     RSVP Refresh Timeouts only
<cr>
```

Below are the RSVP **debug** command options when running Cisco IOS version 12.0(24)S and later.

```
gsr7#
debug ip rsvp ?
all          RSVP messages for all categories
api          RSVP API events
database     RSVP database debugging
dump-messages Dump RSVP message contents
fast-reroute RSVP Fast Reroute support for LSPs
filter       RSVP filter information
handles      RSVP database handles events
hello        RSVP hello events
messages     RSVP messages (sent/received via IP) debugging
msg-mgr      RSVP Message Manager events
path         RSVP PATH messages
policy       RSVP policy information
proxy        RSVP Proxy API trace
rate-limit   RSVP Rate Limiting Messages
reliable-msg RSVP Reliable Messages events
resv         RSVP RESV messages
routing      RSVP Routing Messages
signaling    RSVP Signalling (PATH and RESV) messages
snmp         RSVP SNMP events
summary-refresh RSVP Srefresh & Bundle Messages events
timer        RSVP timer events
traffic-control RSVP traffic control events
wfq          RSVP WFQ events
<cr>
```

As you can see, there are significantly more options in Cisco IOS version 12.0(24)S.

## Differences in Enabling Debugs to Monitor PATH and RESV Messages

The most common use of RSVP debugs is to dump the contents of PATH and RESV messages. In Cisco IOS 12.0(23)S, this was done with the single **debug ip rsvp [path|resv] {<acl>} {detail}** command.

In Cisco IOS 12.0(24)S, enabling this debug is a two-step process.

1. First, use the **debug ip rsvp filter {<acl>}** command.

```
gsr7# debug ip rsvp filter ?
<1-199> Access list
<cr>
```

The **debug ip rsvp filter 101** command uses access control list (ACL) 101, meaning that only packets that are permitted by ACL 101 will be shown in the debug output. The **debug ip rsvp filter 101** command without any specified ACL removes any configured debug filter, and, therefore, all packets related to the debug will show up in the output.

2. Next, enable the specific debug. To dump PATH and/or RESV messages, use the **deb ip rsvp dump-message** command.

```
gsr7# deb ip rsvp dump ?
hex          Hex dump of packet contents
path        Display contents of Path-related messages
resv       Display contents of Resv-related messages
signaling   Display contents of all signaling messages
<cr>
```

The hex option, shown above, gives a complete hex dump of any messages, very similar to the UNIX **tcpdump** command. The following is an example.

```
2d03h: Incoming Resv:
2d03h:  version:1 flags:0000 cksum:141C ttl:255 reserved:0 length:120
2d03h: 0000 10 02 14 1C FF 00 00 78 00 10 01 07 C0 A8 01 08 .....x.....
2d03h: 0010 00 00 00 07 C0 A8 01 04 00 0C 03 01 C0 A8 0D 08 .....
2d03h: 0020 4D 00 04 02 00 08 05 01 00 00 75 30 00 08 08 01 M.....u0....
2d03h: 0030 00 00 00 12 00 24 09 02 00 00 00 07 05 00 00 06 .....
2d03h: 0040 7F 00 00 05 00 00 00 00 44 7A 00 00 00 00 00 00 .....Dz.....
2d03h: 0050 00 00 00 00 00 00 00 00 00 0C 0A 07 C0 A8 01 04 .....
2d03h: 0060 00 00 02 97 00 08 10 01 00 00 00 00 00 0C 15 01 .....
2d03h: 0070 01 08 C0 A8 0D 08 20 00 .....

```

Without the hex option, data is translated into something more readable as shown below.

```
2d03h: Incoming Resv:
2d03h:  version:1 flags:0000 cksum:141C ttl:255 reserved:0 length:120
2d03h:  SESSION                               type 7 length 16:
2d03h:  Tun Dest:    192.168.1.8 Tun ID: 7 Ext Tun ID: 192.168.1.4
2d03h:  HOP                               type 1 length 12:
2d03h:  Hop Addr: 192.168.13.8 LIH: 0x4D000402
2d03h:  TIME_VALUES                               type 1 length 8 :
2d03h:  Refresh Period (msec): 30000
2d03h:  STYLE                               type 1 length 8 :
2d03h:  Shared-Explicit (SE)
2d03h:  FLOWSPEC                               type 2 length 36:
2d03h:  version = 0 length in words = 7
2d03h:  service id = 5, service length = 6

```

```

2d03h:  tspec parameter id = 127, flags = 0, length = 5
2d03h:  average rate = 0 bytes/sec, burst depth = 1000 bytes
2d03h:  peak rate      = 0 bytes/sec
2d03h:  min unit = 0 bytes, max pkt size = 0 bytes
2d03h:  FILTER_SPEC          type 7 length 12:
2d03h:   Tun Sender: 192.168.1.4, LSP ID: 663
2d03h:  LABEL                type 1 length 8 :
2d03h:   Labels: 0
2d03h:  RECORD_ROUTE          type 1 length 12:
2d03h:   192.168.13.8/32, Flags:0x0 (No Local Protection)

```

The path and resv options show above can be used to limit the output to either the PATH or RESV messages. If you do not use either the path or resv options, then the output will contain both PATH and RESV messages. The **signaling** keyword also covers both the PATH and RESV messages.

Issuing the **debug ip rsvp dump {hex}** command will cover all RSVP messages except for HELLO messages. RSVP HELLOs are covered in their own debug, **debug ip rsvp hello**. They are deliberately left out of the signaling debugs because they are intended to be sent and received on the order of milliseconds, and inadvertently debugging hello messages could overwhelm the router's resources.

## Using ACLs to Selectively Debug RSVP Messages

The ability to use ACLs to selectively debug RSVP messages is available in Cisco IOS release 12.0(23)S and earlier. It has been around for quite some time, but this feature can be confusing, so it is documented here.

### Defining ACL Attributes

You may want to selectively debug RSVP messages for only a specific tunnel. This is done by overloading on UDP extended numbered ACLs. UDP ACLs use the following format: **access-list <100–199> permit udp <src ip> {src port} <dst ip> {dst port}**.

Below are the values for an ACL used for MPLS TE debugging:

- **src port** = The link–state packet (LSP) ID
- **dst port** = The tunnel ID, where the tunnel ID is the tunnel interface number
- **src ip** = The tunnel headend
- **dst ip** = The tunnel tail

The example below will only allow debugs where the session is initiated from 192.168.1.4, towards 192.168.1.8, for Tunnel8 on the headend.

```
access-list 101 permit udp host 192.168.1.4 host 192.168.1.8 eq 8
```

**Note:** This ACL will capture both PATH and RESV messages for the session from 192.168.1.4 to 192.168.1.8, but not any tunnels originating from 1.8 going to 1.4. You can also specify the LSP ID, but that is far less useful, since it changes all the time, and the combination of the head, tail, and tunnel ID is generally enough to limit the output to what you want.

---

## Related Information

- [MPLS Support Page](#)
  - [Technical Support – Cisco Systems](#)
-

