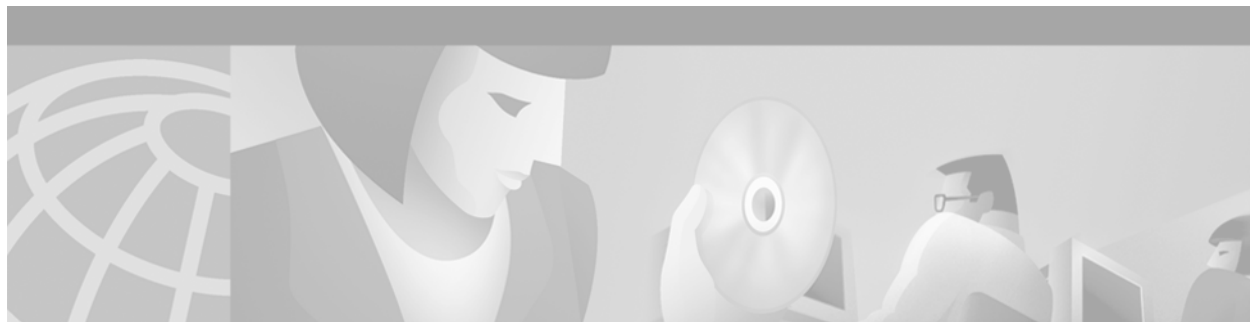




Conditionally Triggered Debugging

When the Conditionally Triggered Debugging feature is enabled, the router generates debugging messages for packets entering or leaving the router on a specified interface; the router will not generate debugging output for packets entering or leaving through a different interface. You can specify the interfaces explicitly. For example, you may only want to see debugging messages for one interface or subinterface. You can also turn on debugging for all interfaces that meet specified conditions. This feature is useful on dial access servers, which have a large number of ports.

Normally, the router will generate debugging messages for every interface, resulting in a large number of message that consume system resources and can make it difficult to find the specific information you need. By limiting the number of debugging messages, you can receive messages related to only the ports you want to troubleshoot.

The Conditionally Triggered Debugging feature controls the output from the following protocol-specific **debug** commands:

- **debug aaa** {**accounting** | **authorization** | **authentication**}
- **debug dialer** {**events** | **packets**}
- **debug isdn** {**q921** | **q931**}
- **debug modem** {**oob** | **trace**}
- **debug ppp** {**all** | **authentication** | **chap** | **error** | **negotiation** | **multilink events** | **packet**}

Although this feature limits the output of the listed commands, it does not automatically enable the generation of debugging output from these commands. Debugging messages are generated only when the protocol-specific **debug** command is enabled. The **debug** command output is controlled through two processes:

- The protocol-specific **debug** commands specify which protocols are being debugged. For example, the **debug dialer events** command generates debugging output related to dialer events.
- The **debug condition** commands limit these debugging messages to those related to a particular interface. For example, the **debug condition username cisco** command generates debugging output only for interfaces with packets that specify a username of cisco.

To configure Conditionally Triggered Debugging, perform the tasks described in the following sections:

- [Enabling Protocol-Specific debug Commands](#)
- [Enabling Conditional Debugging Commands](#)
- [Specifying Multiple Conditions](#)

Enabling Protocol-Specific debug Commands

To generate any debugging output, the protocol-specific **debug** command for the desired output must be enabled. Use the **show debugging** command to determine which types of debugging are enabled. Use the following commands in privileged EXEC mode to enable or disable the desired protocol-specific **debug** commands as needed:

Command	Purpose
show debugging	Determines which types of debugging are enabled.
debug protocol	Enables the desired debugging commands.
no debug protocol	Disables the debugging commands that are not desired.

If you want to have no output, disable all the protocol-specific **debug** commands.

Enabling Conditional Debugging Commands

If no **debug condition** commands are enabled, all debugging output, regardless of the interface, will be displayed for the enabled protocol-specific **debug** commands.

The first **debug condition** command you enter enables conditional debugging. The router will only display messages for interfaces that meet one of the specified conditions. If multiple conditions are specified, the interface must meet at least one of the conditions in order for messages to be displayed.

To enable messages for interfaces specified explicitly or for interfaces that meet certain conditions, perform the tasks described in the following sections:

- [Displaying Messages for One Interface](#)
- [Displaying Messages for Multiple Interfaces](#)
- [Limiting Messages Based on Conditions](#)

Displaying Messages for One Interface

To disable debugging messages for all interfaces except one, use the following command in privileged EXEC mode:

Command	Purpose
debug condition interface <i>interface</i>	Disables debugging messages for all interfaces except one.

If you enter the **debug condition interface** command, the debugging output will be turned off for all interfaces except the specified interface. To reenable debugging output for all interfaces, use the **no debug interface** command.

Displaying Messages for Multiple Interfaces

To enable debugging messages for multiple interfaces, use the following commands in privileged EXEC mode:

Command	Purpose
debug condition interface <i>interface</i>	Disables debugging messages for all interfaces except one.
debug condition interface <i>interface</i>	Enables debugging messages for additional interfaces. Repeat this task until debugging messages are enabled for all desired interfaces.

If you specify more than one interface by entering this command multiple times, debugging output will be displayed for all of the specified interfaces. To turn off debugging on a particular interface, use the **no debug interface** command. If you use the **no debug interface all** command or remove the last **debug interface** command, debugging output will be reenabled for all interfaces.

Limiting Messages Based on Conditions

The router can monitor interfaces to learn if any packets contain the specified value for one of the following conditions:

- Username
- Calling party number
- Called party number

If you enter a condition, such as calling number, debug output will be stopped for all interfaces. The router will then monitor every interface to learn if a packet with the specified calling party number is sent or received on any interfaces. If the condition is met on an interface or subinterface, **debug** command output will be displayed for that interface. The debugging output for an interface is “triggered” when the condition has been met. The debugging output continues to be disabled for the other interfaces. If at some later time the condition is met for another interface, then the debug output will become enabled for that interface as well.

Once debugging output has been triggered on an interface, the output will continue until the interface goes down. However, the session for that interface might change, resulting in a new username, called party number, or calling party number. Use the **no debug interface** command to reset the debug trigger mechanism for a particular interface. The debugging output for that interface will be disabled until the interface meets one of the specified conditions.

To limit debugging messages based on a specified condition, use the following command in privileged EXEC mode:

Command	Purpose
debug condition { username <i>username</i> called <i>dial-string</i> caller <i>dial-string</i> }	Enables conditional debugging. The router will display only messages for interfaces that meet this condition.

To reenable the debugging output for all interfaces, use the **no debug condition all** command.

Specifying Multiple Conditions

To limit debugging messages based on more than one condition, use the following commands in privileged EXEC mode as needed:

Command	Purpose
debug condition { username <i>username</i> called <i>dial-string</i> caller <i>dial-string</i> }	Enables conditional debugging and specifies the first condition.
debug condition { username <i>username</i> called <i>dial-string</i> caller <i>dial-string</i> }	Specifies the second condition. Repeat this task until all conditions are specified.

If you enter multiple **debug condition** commands, debugging output will be generated if an interface meets at least one of the conditions. If you use the **no debug condition** command to remove one of the conditions, using interfaces that meet only that condition will no longer produce debugging output. However, interfaces that meet a condition other than the removed condition will continue to generate output. Only if no active conditions are met for an interface will the output for that interface be disabled.

Conditionally Triggered Debugging Configuration Examples

In this example, four conditions have been set by the following commands:

- **debug condition interface serial 0**
- **debug condition interface serial 1**
- **debug condition interface virtual-template 1**
- **debug condition username cisco**

The first three conditions have been met by one interface. The fourth condition has not yet been met.

```
Router# show debug condition
```

```
Condition 1: interface Se0 (1 flags triggered)
           Flags: Se0
Condition 2: interface Se1 (1 flags triggered)
           Flags: Se1
Condition 3: interface Vt1 (1 flags triggered)
           Flags: Vt1
Condition 4: username cisco (0 flags triggered)
```

When any **debug condition** command is entered, debugging messages for conditional debugging are enabled. The following debugging messages show conditions being met on different interfaces as serial interface 0 and serial interface 1 come up. For example, the second line of output indicates that serial interface 0 meets the username cisco condition.

```
*Mar  1 00:04:41.647: %LINK-3-UPDOWN: Interface Serial0, changed state to up
*Mar  1 00:04:41.715: Se0 Debug: Condition 4, username cisco triggered, count 2
*Mar  1 00:04:42.963: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0, changed
state to up
*Mar  1 00:04:43.271: Vi1 Debug: Condition 3, interface Vt1 triggered, count 1
*Mar  1 00:04:43.271: %LINK-3-UPDOWN: Interface Virtual-Access1, changed state to up
*Mar  1 00:04:43.279: Vi1 Debug: Condition 4, username cisco triggered, count 2
*Mar  1 00:04:43.283: Vi1 Debug: Condition 1, interface Se0 triggered, count 3
```

```
*Mar 1 00:04:44.039: %IP-4-DUPADDR: Duplicate address 172.27.32.114 on Ethernet 0,
sourced by 00e0.1e3e.2d41
*Mar 1 00:04:44.283: %LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access1,
changed state to up
*Mar 1 00:04:54.667: %LINK-3-UPDOWN: Interface Serial1, changed state to up
*Mar 1 00:04:54.731: Se1 Debug: Condition 4, username cisco triggered, count 2
*Mar 1 00:04:54.735: Vi1 Debug: Condition 2, interface Se1 triggered, count 4
*Mar 1 00:04:55.735: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1, changed
state to up
```

After a period of time, the **show debug condition** command displays the revised list of conditions:

```
Router# show debug condition
```

```
Condition 1: interface Se0 (2 flags triggered)
Flags: Se0 Vi1
Condition 2: interface Se1 (2 flags triggered)
Flags: Se1 Vi1
Condition 3: interface Vt1 (2 flags triggered)
Flags: Vt1 Vi1
Condition 4: username cisco (3 flags triggered)
Flags: Se0 Vi1 Se1
```

Next, serial interface 1 and serial interface 0 go down. When an interface goes down, conditions for that interface are cleared.

```
*Mar 1 00:05:51.443: %LINK-3-UPDOWN: Interface Serial1, changed state to down
*Mar 1 00:05:51.471: Se1 Debug: Condition 4, username cisco cleared, count 1
*Mar 1 00:05:51.479: Vi1 Debug: Condition 2, interface Se1 cleared, count 3
*Mar 1 00:05:52.443: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1, changed
state to down
*Mar 1 00:05:56.859: %LINK-3-UPDOWN: Interface Serial0, changed state to down
*Mar 1 00:05:56.887: Se0 Debug: Condition 4, username cisco cleared, count 1
*Mar 1 00:05:56.895: Vi1 Debug: Condition 1, interface Se0 cleared, count 2
*Mar 1 00:05:56.899: Vi1 Debug: Condition 3, interface Vt1 cleared, count 1
*Mar 1 00:05:56.899: Vi1 Debug: Condition 4, username cisco cleared, count 0
*Mar 1 00:05:56.903: %LINK-3-UPDOWN: Interface Virtual-Access1, changed state to down
*Mar 1 00:05:57.907: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0, changed
state to down
*Mar 1 00:05:57.907: %LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access1,
changed state to down
```

The final **show debug condition** output is the same as the output before the interfaces came up:

```
Router# show debug condition
```

```
Condition 1: interface Se0 (1 flags triggered)
Flags: Se0
Condition 2: interface Se1 (1 flags triggered)
Flags: Se1
Condition 3: interface Vt1 (1 flags triggered)
Flags: Vt1
Condition 4: username cisco (0 flags triggered)
```

