Contents

Voice Call Debug Filtering Overview

Use voice call debug filtering to get selected debugging traces for voice calls. This feature allows you to filter and trace voice call debug messages based on selected filtering criteria, reducing the volume of output for more efficient troubleshooting.

Restrictions for Voice Call Debug Filtering

- End-to-end filtering between gateways is not supported.
- Filtering for CAS, IOS-AAA, IVR Version 1.0, media, and VoiceXML is not supported.
- Matching conditions cannot be set for specific signaling protocols.
- Matching conditions based on current DSP information are not supported.

Information About Voice Call Debug Filtering

Information from using debug commands for voice calls is crucial for troubleshooting, but the volume of raw data can be very large. In order to isolate the most valuable data, use the Voice Call Debug Filtering feature. This feature allows the debug output for the voice call to be filtered according to a variety of criteria, including:

- Calling party number with prefix
- Called party number with prefix
- Carrier IDs
- Dial peers
- Local IP address
- Remote IP address
- Telephony interface or port
- Trunk groups

 **Note**: Call filtering also works on IP-to-IP gateway connections using H.323.
The selected criteria are set on the gateway, and different sets of criteria can be stored.

To better understand the voice call debug filtering on Cisco voice gateways, see the following sections:

- Debug Commands that Support Voice Call Filtering
- Generic Call Filter Module
- Calling and Called Number Strings
- Exact and Partial Matching
- Media and Signaling Streams

Debug Commands that Support Voice Call Filtering

When a call filter is applied, the filtering applies to all of the debugs affected by the call filter. Debug commands that support voice call debug filtering include the following:

- **debug cch323 h225**
- **debug cch323 h245**
- **debug cch323 preauth**
- **debug cch323 session**
- **debug ccsip all**
- **debug ccsip calls**
- **debug ccsip err**
- **debug ccsip events**
- **debug ccsip messages**
- **debug ccsip preauth**
- **debug ccsip states**
- **debug mgcp all**
- **debug mgcp endpoint**

- **debug mgcp endptdb**
- **debug mgcp errors**
- **debug mgcp events**
- **debug mgcp gcfm**
- **debug mgcp inout**
- **debug mgcp media**
- **debug mgcp src**
- **debug mgcp state**
- **debug mgcp voipcac**
- **debug voip aaa**
- **debug voip ccapi error**
- **debug voip ccapi inout**
- **debug voip ipipgw**
- **debug voip ivr all**
- **debug voip ivr applib**
- **debug voip ivr callsetup**
- **debug voip ivr digitcollect**
- **debug voip ivr dynamic**
- **debug voip ivr error**
- **debug voip ivr script**
- **debug voip ivr settlement**
- **debug voip ivr states**
- **debug voip ivr tclcommands**
- **debug voip rawmsg**
- **debug vtsp all**
- **debug vtsp dsp**
- **debug vtsp error**
- **debug vtsp event**
- **debug vtsp port**
- **debug vtsp rtp**
- **debug vtsp send-nse**
- **debug vtsp session**
- **debug vtsp stats**
- **debug vtsp vofr subframe**
- **debug vtsp tone**
- **debug vtsp vofr**

Note: See the Cisco IOS Debug Command Reference for detailed information about these debug commands.

Generic Call Filter Module

The debug commands described in the Debug Commands that Support Voice Call Filtering support the following voice modules within the voice gateway:

- CCAPI
- Dial peers
- H.323
- ISDN
- IVR (Version 2.0 only)
- MGCP
- SIP
- SSAPP
- TGRAM

- Voice AAA
- VTSP

The filtering for these modules is managed by the generic call filter module (GCFM). The filtering conditions are configured in the GCFM, and then the individual modules are informed when a call has to be filtered. The GCFM coordinates between multiple modules to handle filtering conditions.

All modules use the global unique identifier (GUID) to identify an individual call to GCFM. Each call is assigned a GUID and retains the same GUID throughout the entire network and over time. Gateway information and time stamp are embedded in the GUID. GUIDs identify an individual call among the multiple filtered-out calls so that the call can be isolated. For more information about GUIDs and the debug header, see the Debug Command Output on Cisco IOS Voice Gateways.

Activity in the GCFM can be traced using the **debug call filter detail** and **debug call filter inout** commands. See the Cisco IOS Debug Command Referencefor more information about these debug commands.

Calling and Called Number Strings

The string pattern for calling and called numbers can be either a complete telephone number or a partial telephone number with wildcard digits, represented by a period (.) character. Each "." represents a wildcard for an individual digit that the originating voice gateway expects to match. For example, if the calling and called number strings is defined as "555....", then any dialed string beginning with 555, plus at least four additional digits, matches this calling or called number.

Table: Symbols Used in Calling and Called Number Strings shows all of the wildcard symbols that are supported in the calling and called number strings.

Table: Symbols Used in Calling and Called Number Strings

Symbol	Description
.	Indicates a single-digit placeholder. For example, 555.... matches any dialed string beginning with 555, plus at least four additional digits.
[ ]	Indicates a range of digits. A consecutive range is indicated with a hyphen (-); for example, [5-7]. A nonconsecutive range is indicated with a comma (,); for example, [5,8]. Hyphens and commas can be used in combination; for example, [5-7,9]. **Note**: Only single-digit ranges are supported. For example, [98-102] is invalid.
( )	Indicates a pattern; for example, 408(555). It is used in conjunction with the symbol ?, %, or +.
?	Indicates that the preceding digit occurred zero or one time. Enter **ctrl-v** before entering ? from your keyboard.
%	Indicates that the preceding digit occurred zero or more times. This functions the same as the "*" used in regular expression.
+	Indicates that the preceding digit occurred one or more times.
T	Indicates the interdigit timeout. The voice gateway pauses to collect additional dialed digits.

Note: The period (.) is the only wildcard character that is supported for dial strings that are configured using the **answer-address** or **incoming called-number** command.

Table: Number Matching Examples Using Wildcard Symbols shows some examples of how these wildcard symbols are applied to the calling and called number strings and the dial string that results when dial string 4085550199 is matched to the calling or called number. The wildcard symbols follow regular expression rules.

Table: Number Matching Examples Using Wildcard Symbols

Destination Pattern	Dial String Translation	String After Stripping
408555.+	408555, followed by one or more wildcard digits. This pattern implies that the string must contain at least seven digits starting with 408555.	0199
408555.%	408555, followed by zero or more wildcard digits. This pattern implies that the string must contain at least 408555.	0199
408555+	40855, followed by 5 repeated one or more times.	0199
408555%	40855, followed by 5 repeated zero or more times. Any explicitly matching digit before the % symbol is not stripped off.	50199
408555?	40855, followed by 5 repeated zero or one time. Any explicitly matching digit before the ? symbol is not stripped off.	50199
40855[5-7].+	40855, followed by 5, 6, or 7, plus any digit repeated one or more times.	50199
40855[5-7].%	40855, followed by 5, 6, or 7, plus any digit repeated zero or more times.	50199
40855[5-7]+0199	40855, followed by 5, 6, or 7 repeated one or more times, followed by 0199.	50199
408(555)+0199	408, followed by 555, which may repeat one or more times, followed by 0199.	5550199

In addition to wildcard characters, the following symbols can be used in the calling and called number strings:

- Asterisk (*) and pound sign (#)-These symbols on standard touchtone dial pads can be used anywhere in the pattern. They can be used as the leading character (for example, *650), except on the Cisco 3600 series.
- Dollar sign ($)-Disables variable-length matching. It must be used at the end of the dial string.

Exact and Partial Matching

The conditions under each set of call filters are inclusive, so if multiple conditions are specified under a filter, they are all matched. To compare different conditions, create additional filters.

Matching conditions are as follows:

- Exact match-All related debug output is filtered until all conditions in the match list are explicitly met. This is the best choice for most situations because the output is the most concise.
- Partial match-No related debug output is filtered until there is a single explicit match failure. As long as zero or more conditions are met, debug output is not filtered. This choice is useful in debugging call startup problems like digit collection, but is not ideal for many situations because of the large amount of debug output that might be generated before matches explicitly fail.

Media and Signaling Streams

Media streams carry voice, video, fax, and data. Examples of media streams are G.711 or G.723 encoded voice streams or fax data. With the voice call debug filter, the media streams are traced for the voice gateway receiving the media stream. Some traces associated with media streams can be filtered, such as SPI-level traces associated with opening and closing the media channels. However, media RTP/RTCP packet-level traces are not filtered.

Signaling streams include both address signaling and supervisory signaling. Examples of signaling streams include H.323 and SIP protocol streams. With the voice call debug filter, the signaling streams are traced for the gateway or endpoint for the signaling stream.

Configuring the Voice Call Debug Filter

To configure the voice call debug filter, perform the following tasks:

- Configuring Call-Specific Conditions (required)
- Enabling Debug for the Set Filtering Conditions (required)

Configuring Call-Specific Conditions

Configure call-specific conditions to set the attributes that are filtered for voice calls.

SUMMARY STEPS

1. **enable**
2. **configure** terminal
3. **call filter match-list** *number* **voice**
4. **incoming calling-number** *string*
5. **incoming called-number** *string*
6. **incoming secondary-called-number** *string*
7. **incoming port** *string*
8. **incoming signaling** {**local** | **remote**'*}* '**ipv4** *ip_address*
9. **incoming media** {**local** | **remote**'*}'* **ipv4** *ip_address*
10. **incoming dialpeer** *tag*
11. **source carrier-id** *string*
12. **source trunk-group-label** *group-number*
13. **outgoing calling-number** *string*
14. **outgoing called-number** *string*
15. outgoing secondary-called-number *string*
16. **outgoing port** *string*
17. **outgoing signaling** {**local** | **remote**'*}'* **ipv4** *ip_address*
18. **outgoing media** {**local** | **remote**'*}'* **ipv4** *ip_address*
19. **outgoing dialpeer** *tag*
20. **target carrier-id** *string*
21. **target trunk-group-label** *group-number*
22. **end**

DETAILED STEPS

Command or Action	Purpose	
1.	**enable** Example: `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
2.	**configure** terminal	Enters global configuration mode.

	Example: `Router# configure terminal`	
3.	**call filter match-list** *number* **voice** Example: `Router(config)# call filter match-list 1 voice`	Enters call filter match list configuration mode to define the filter conditions. • *number*-Numeric label that uniquely identifies the match list. Range is 1 to 16. **Note**: At least one of the following optional parameters (Step 4 to Step 21) for call filtering must be configured.
4.	**incoming calling-number** *string* Example: `Router(conf-call-filter-mlist)# incoming calling-number 408555`	(Optional) Specifies the incoming calling number to be filtered. • *string*-Numeric string that identifies all or part of the incoming calling number.
5.	**incoming called-number** *string* Example: `Router(conf-call-filter-mlist)# incoming called-number 408555`	(Optional) Specifies the incoming called number to be filtered. • *string*-Numeric string that identifies all or part of the incoming called number.
6.	**incoming secondary-called-number** *string* Example: `Router(conf-call-filter-mlist)# incoming secondary-called-number 408555`	(Optional) Specifies the incoming secondary called number to be filtered. • The secondary called number is the number

		from the second stage in a two-stage scenario. • *string*-Numeric string that identifies all or part of the incoming secondary called number.
7.	**incoming port** *string* Example: `Router(conf-call-filter-mlist)# incoming port 1/0/0`	(Optional) Specifies the incoming port to be filtered. • The telephony interfaces are defined for calls from the PSTN. The string value varies depending on the voice gateway. • *string*-Identifies the incoming port number. This value is platform-specific and varies between platforms.
8.	**incoming signaling** {**local** | **remote**} **ipv4** *ip-address* Example: `Router(conf-call-filter-mlist)# incoming signaling local ipv4 192.168.10.255`	(Optional) Specifies the incoming signaling IPv4 address. • **local**-Local voice gateway • **remote**-Remote IP device • *ip-address*-IP address of the local voice gateway.
9.	**incoming media** {**local** | **remote**} **ipv4** *ip-address* Example: `Router(conf-call-filter-mlist)# incoming media local ipv4 192.168.10.255`	(Optional) Specifies the incoming media IPv4 address. • **local**-Local voice gateway • **remote**-Remote IP device

		• *ip-address*-IP address of the local voice gateway.
10.	**incoming dialpeer** *tag* Example: `Router(conf-call-filter-mlist)# incoming dialpeer 14`	(Optional) Specifies the incoming dial peer to be filtered. • *tag*-Digits that define a specific dial peer. Valid entries are 1 to 2147483647.
11.	**source carrier-id** *string* Example: `Router(conf-call-filter-mlist)# source carrier-id 4321`	(Optional) Specifies the source carrier ID to be filtered. • *string*-Alphanumeric identifier for the carrier ID.
12.	**source trunk-group-label** *group-number* Example: `Router(conf-call-filter-mlist)# source trunk-group-label 20`	(Optional) Specifies the source trunk group to be filtered. • *group-number*-A value from 0 to 23 that identifies the trunk group.
13.	**outgoing calling-number** *string* Example: `Router(conf-call-filter-mlist)# outgoing calling-number 408525`	(Optional) Specifies the outgoing calling number to be filtered. • This number goes out after number translation and expansion are complete. • *string*-Numeric string that identifies all or part of the outgoing calling number.
14.	**outgoing called-number** *string* Example: `Router(conf-call-filter-mlist)# outgoing called-number 408525`	(Optional) Specifies the outgoing called number to be filtered. • This number goes out after number

		translation and expansion are complete. • *string*-Numeric string that identifies all or part of the outgoing called number.
15.	**outgoing secondary-called-number** *string* Example: `Router(conf-call-filter-mlist)# outgoing secondary-called-number 408525`	(Optional) Specifies the outgoing secondary called number to be filtered. • The secondary called number is the number from the second stage in a two-stage scenario. • *string*-Numeric string that identifies all or part of the outgoing secondary called number.
16.	**outgoing port** *string* Example: `Router(conf-call-filter-mlist)# outgoing port 1/0/0`	(Optional) Specifies the outgoing port to be filtered. • The telephony interfaces are defined for calls from PSTN. The string value varies, depending on different voice gateways. • *string*-Identifies the outgoing port number. This value is voice gateway-specific and varies between voice gateways.
17.	**outgoing signaling** {**local** | **remote**} **ipv4** *ip-address*	(Optional) Specifies the outgoing signaling IPv4

	Example: `Router(conf-call-filter-mlist)# outgoing signaling local ipv4 192.168.10.255`	address for the gatekeeper managing the signaling. • **local**-Local voice gateway • **remote**-Remote IP device • *ip-address*-IP address of the local voice gateway.
18.	**outgoing media** {**local** | **remote**} **ipv4** *ip-address* Example: `Router(conf-call-filter-mlist)# outgoing media local ipv4 192.168.10.255`	(Optional) Specifies the outgoing media IPv4 address for the voice gateway receiving the media stream. • **local**-Local voice gateway • **remote**-Remote IP device • *ip-address*-IP address of the local voice gateway.
19.	**outgoing dialpeer** *tag* Example: `Router(conf-call-filter-mlist)# outgoing dialpeer 14`	(Optional) Specifies the outgoing dial peer to be filtered. • *tag*-Digits that define a specific dial peer. Valid entries are 1 to 2147483647.
20.	**target carrier-id** *string* Example: `Router(conf-call-filter-mlist)# target carrier-id 4321`	(Optional) Specifies the target carrier ID to be filtered. • *string*-Alphanumeric identifier for the carrier ID.
21.	**target trunk-group-label** *group-number* Example: `Router(conf-call-filter-mlist)# target trunk-group-label 20`	(Optional) Specifies the target trunk group to be filtered. • *group-number*-A value from 0 to 23 that identifies the trunk group.
22.	**end**	

	Example: `Router(conf-call-filter-mlist)# end`	Exits to privileged EXEC mode.

Troubleshooting Tips

To verify the conditions that you have set, use the **show call filter match-list** command. This command displays the criteria set for the specified match list.

What to Do Next

After the conditions are set for the voice call debug, debug commands can be enabled. Proceed to the Enabling Debug for the Set Filtering Conditions.

Enabling Debug for the Set Filtering Conditions

Use the **debug** command to enable the set conditions to get the filtered output.

Prerequisites

The conditions for the voice call debug filter must be set as described in the Configuring Call-Specific Conditions.

SUMMARY STEPS

1. **enable**
2. **debug condition match-list** *tag* {**exact-match** | **partial-match**}
3. **debug cch323 '***{*'**capacity' | 'h225 '**|' **h245 '| 'preauth' | 'ras '| 'rawmsg' | 'session'***}*'

   or
   debug ccsip '*{*'**all '| 'calls '**|' **err '**|' **events' | 'messages '| 'preauth' |' states'***}*'
   or
   debug isdn q931
   or
   debug voip aaa
   or
   debug voip ccapi '*{*'**error '**|' **inout'***}*'
   or
   debug voip ipipgw
   or
   debug voip ivr '*{*'**all' | 'applib '**|' **callsetup' | 'digitcollect' | 'dynamic '**|' **error' | 'script' | 'settlement '| 'states' | 'tclcommands'***}*'
   or
   debug voip rawmsg
   or
   debug vtsp '*{*'**all '**|' **dsp '| 'error '**|' **event '**|' **port' |' rtp '| 'send-nse '| 'session '**|' **stats '**|' **vofr subframe '**|' **tone '**|' **vofr'***}*'

DETAILED STEPS

Command or Action	Purpose

1.	**enable** Example: `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
2.	**debug condition match-list** ***tag*** **{**exact-match **|** **partial-match**} Example: `Router# debug condition match-list 1 exact-match`	Enables the filter match list for the set conditions. • *tag*-Numeric label that uniquely identifies the match list. Range is 1 to 16. The number for the match list is set using the **call filter match-list** command. • **exact-match**-All related debug output is filtered until all conditions in the match list are explicitly met. This is the best choice for most situations because the output is the most concise. • **partial- match**-No related debug output is filtered until there is a single explicit match failure. As long as zero or more conditions are met, debug output is not filtered. This choice is useful in debugging call startup problems like digit collection, but is not ideal for many situations because a large amount of debug output is generated before matches explicitly fail.
3.	**debug cch323** {**capacity | h225 | h245 | preauth | ras'| rawmsg | session**} or **debug ccsip** {**all | calls | err | events | messages | preauth | states**}	Enables the appropriate voice call debug commands. • See the Cisco IOS Debug Command

	or **debug isdn q931**
	
	or **debug voip aaa**
	
	or **debug voip ccapi** {**error** | **inout**}
	
	or **debug voip ipipgw**
	
	or **debug voip ivr** {**all** | **applib** | **callsetup** | **digitcollect** | **dynamic** | **error** | **script** | **settlement** | **states** | **tclcommands**}
	
	or **debug voip rawmsg**
	
	or **debug vtsp** {**all** | **dsp** | **error** | **event** | **port** | **rtp** | **send-nse** | **session** | **stats** | **vofr subframe** | **tone** | **vofr**}
	
	Example:
	
	`Router# debug cch323 h225`
	`Router# debug ccsip events`
	`Router# debug isdn q931`
	`Router# debug voip aaa`
	`Router# debug voip ccapi inout`
	`Router# debug voip ipipgw`
	`Router# debug voip ivr all`
	`Router# debug voip rawmsg`
	`Router# debug vtsp dsp`
		Reference for detailed descriptions of these debug commands.
		• The debug output commences at this point.

Troubleshooting Tips

To verify debug conditions, use the following commands:

- **show debug**

  This command displays the debugs that are enabled.

- **show call filter components**

  This command displays the components that register internally with the filtering module. This command shows which components are registered with the GCFM, which is the internal module that controls which components are filtered.

- **show call filter match-list**

  This command displays the criteria set for the specified match list. It shows a list of all the match lists, shows which ones are enabled, and shows whether they are enabled for partial or exact matching.

Output Examples for Voice Call Debug Filtering

This section provides configuration examples to match the identified configuration tasks in the previous section:

Exact Match Filtering: Example

When the exact match condition is used for voice call debug filtering, all related debug output is filtered until all conditions in the match list are explicitly met. In the following example, the configuration, enabled debugs, and debug output for a Cisco AS5400 universal gateway are shown.

Dial-Peer Configuration for Exact Match Filtering

```
dial-peer voice 501 pots
 preference 1
 incoming called-number 50200
 destination-pattern 50201
 direct-inward-dial
 port 6/0:D
 prefix 50201
!
dial-peer voice 502 voip
 preference 1
 incoming called-number 50201
 destination-pattern 50200
 session target ipv4:172.16.101.21
 dtmf-relay h245-alphanumeric
 fax-relay ecm disable
 fax rate disable
!
```

Debug Output for Exact Match Filtering

```
 Router# show debug
 The following ISDN debugs are enabled on all DSLs:
 debug isdn error is             ON.
 debug isdn q931 is              ON.   (filter is ON)
 Voice Telephony session debugging is on (filter is ON)
 Voice Telephony dsp debugging is on (filter is ON)
 Voice Telephony error debugging is on (filter is ON)
 voip ccAPI function enter/exit debugging is on (filter is ON)
```

In the following output, the **show call filter match-list** command is used to show which conditions have been set for the specified call filter:

```
Router# show call filter match-list
*****************************************
call filter match-list 9 voice
*****************************************
  incoming calling-number 50200
  incoming called-number 50201
  incoming signal local ipv4 172.16.101.22
  incoming signal remote ipv4  172.16.101.21
  incoming media local ipv4  172.16.101.22
  incoming media remote ipv4  172.16.101.21
  incoming dialpeer 502
  outgoing calling-number 50200
  outgoing called-number 50201
  outgoing port 6/0:D
  outgoing dialpeer 501
 debug condition match-list is set to EXACT_MATCH
*****************************************
call filter match-list 10 voice
*****************************************
  incoming calling-number 50300
  incoming called-number 50301
```

```
  incoming signal local ipv4 172.16.101.22
  incoming signal remote ipv4  172.16.101.21
  incoming media local ipv4  172.16.101.22
  incoming media remote ipv4  172.16.101.21
  incoming dialpeer 504
  outgoing calling-number 50300
  outgoing called-number 50301
  outgoing port 6/1:D
  outgoing dialpeer 503
 debug condition match-list is set to EXACT_MATCH
```

The following debug output contains the exact match for the configured conditions.

```
Feb  6 11:13:30.799: digit_strip:1, pcn:50201, poa:50201
Feb  6 11:13:30.799: pcn:, poa:
Feb  6 11:13:30.799: Final pcn:, poa:, dial_string:50201
Feb  6 11:13:30.803:
//6/CFD853DE8004/VTSP:(6/0:D):-1:0:0/vtsp_gcfm_percall_status_callback: found cdb and
update
Feb  6 11:13:30.803:
//6/CFD853DE8004/VTSP:(6/0:D):-1:0:0/vtsp_update_dsm_stream_mgr_filter_flag: update
dsp_stream_mgr_t  debug flag
Feb  6 11:13:30.803: //5/CFD853DE8004/CCAPI/ccapi_gcfm_percall_status_callback: found
callEntry and update
Feb  6 11:13:30.803: //6/CFD853DE8004/CCAPI/ccapi_gcfm_percall_status_callback: found
callEntry and update
Feb  6 11:13:30.803: //5/CFD853DE8004/SSAPP:502:-1/ssaTraceSct:
cid(5)st(SSA_CS_CALL_SETTING)ev(SSA_EV_CALL_REPORT_DIGITS_DONE)
oldst(SSA_CS_MAPPING)cfid(-1)csize(0)in(1)fDest(1)
Feb  6 11:13:30.803: //5/CFD853DE8004/SSAPP:502:-1/ssaTraceSct:
-cid2(6)st2(SSA_CS_CALL_SETTING)oldst2(SSA_CS_MAPPING)
Feb  6 11:13:30.803: //5/CFD853DE8004/SSAPP:502:-1/ssaDebugPeers: ssaReportDigitsDone
cid(5) peer list: tag(2501) called number (50201)
Feb  6 11:13:30.803: //5/CFD853DE8004/SSAPP:502:-1/ssaReportDigitsDone: callid=5 Reporting
disabled.
Feb  6 11:13:31.007: ISDN Se6/0:23 Q931: callid 0x800B, callref 0x0003,
                guid CFD853DE8004
Feb  6 11:13:31.007: ISDN Se6/0:23 Q931: RX <- CALL_PROC pd = 8  callref = 0x8003
        Channel ID i = 0xA98397
                Exclusive, Channel 23
Feb  6 11:13:31.007: ISDN Se6/0:23 Q931: callid 0x800B, callref 0x0003,
                guid CFD853DE8004
Feb  6 11:13:31.007: ISDN Se6/0:23 Q931: RX <- ALERTING pd = 8  callref = 0x8003
Feb  6 11:13:31.011: //6/CFD853DE8004/VTSP:(6/0:D):22:0:0/vtsp_process_event: vtsp:[6/0:D
(6), S_SETUP_REQUEST, E_TSP_PROCEEDING]
Feb  6 11:13:31.011: //6/CFD853DE8004/VTSP:(6/0:D):22:0:0/act_setup_pend_proceeding: .
Feb  6 11:13:31.011:
//6/CFD853DE8004/DSM:(6/0:D):-1:0:4098/dsp_stream_mgr_reinit_platform_info: .
Feb  6 11:13:31.011: //6/CFD853DE8004/DSM:(6/0:D):-1:0:4098/dsm_open_voice_and_set_params:
.
Feb  6 11:13:31.011: //6/CFD853DE8004/DSM:(6/0:D):-1:0:4098/set_playout_dmgr: playout
default
Feb  6 11:13:31.011:
//6/CFD853DE8004/DSM:(6/0:D):-1:0:4098/dsm_dsp_echo_canceller_control: echo_cancel: 1
Feb  6 11:13:31.011: ISDN Se6/0:23 Q931: callid 0x800B, callref 0x0003,
                guid CFD853DE8004
Feb  6 11:13:31.011: ISDN Se6/0:23 Q931: RX <- CONNECT pd = 8  callref = 0x8003
Feb  6 11:13:31.011: //6/CFD853DE8004/SSAPP:0:-1/ssaTraceSct:
cid(6)st(SSA_CS_CALL_SETTING)ev(SSA_EV_CALL_PROCEEDING)
oldst(SSA_CS_MAPPING)cfid(-1)csize(0)in(0)fDest(0)
Feb  6 11:13:31.011: //6/CFD853DE8004/SSAPP:0:-1/ssaTraceSct:
-cid2(5)st2(SSA_CS_CALL_SETTING)oldst2(SSA_CS_CALL_SETTING)
Feb  6 11:13:31.011: //6/CFD853DE8004/SSAPP:0:-1/ssaCallProc:
```

```
Feb  6 11:13:31.011: //6/CFD853DE8004/SSAPP:0:-1/ssaIgnore: cid(6),
st(SSA_CS_CALL_SETTING),oldst(1), ev(21)
Feb  6 11:13:31.011: //6/CFD853DE8004/VTSP:(6/0:D):22:0:0/vtsp_process_event: vtsp:[6/0:D
(6), S_SETUP_REQ_PROC, E_TSP_ALERT]
Feb  6 11:13:31.011: //6/CFD853DE8004/VTSP:(6/0:D):22:0:0/act_setup_pend_alert: .
Feb  6 11:13:31.011: //6/CFD853DE8004/VTSP:(6/0:D):22:0:0/vtsp_ring_noan_timer_start:
371381
Feb  6 11:13:31.015: ISDN Se6/0:23 Q931: callid 0x800B, callref 0x0003,
               guid CFD853DE8004
Feb  6 11:13:31.015: ISDN Se6/0:23 Q931: TX -> CONNECT_ACK pd = 8  callref = 0x0003
Feb  6 11:13:31.015: //6/CFD853DE8004/SSAPP:0:-1/ssaTraceSct:
cid(6)st(SSA_CS_CALL_SETTING)ev(SSA_EV_CALL_ALERT)
oldst(SSA_CS_CALL_SETTING)cfid(-1)csize(0)in(0)fDest(0)
Feb  6 11:13:31.015: //6/CFD853DE8004/SSAPP:0:-1/ssaTraceSct:
-cid2(5)st2(SSA_CS_CALL_SETTING)oldst2(SSA_CS_CALL_SETTING)
Feb  6 11:13:31.015: //5/CFD853DE8004/SSAPP:502:-1/ssaAlert:
Feb  6 11:13:31.015: //6/CFD853DE8004/DSM:(6/0:D):-1:0:4098/dsm_exec: [Feat SM: S:NONE B
SM: S:S_DSM_INIT E:E_DSM_CC_BRIDGE]
Feb  6 11:13:31.015: //6/CFD853DE8004/DSM:(6/0:D):-1:0:4098/dsm_act_bridge: .
Feb  6 11:13:31.015: //6/CFD853DE8004/VTSP:(6/0:D):22:0:0/vtsp_dsm_bridge_status_cb: .
Feb  6 11:13:31.015: //6/CFD853DE8004/VTSP:(6/0:D):22:0:0/vtsp_dsm_set_fax_feat_param:
Fax relay is ENABLED,  Primary Fax protocol is T38_FAX_RELAY, Fallback Fax protocol is
CISCO_FAX_RELAY
Feb  6 11:13:31.015: //6/CFD853DE8004/VTSP:(6/0:D):22:0:0/vtsp_dsm_peer_event_cb:
E_DSM_CC_CAPS_IND
Feb  6 11:13:31.015: //6/CFD853DE8004/VTSP:(6/0:D):22:0:0/vtsp_process_event: vtsp:[6/0:D
(6), S_SETUP_REQ_PROC, E_TSP_CONNECT]
Feb  6 11:13:31.015: //6/CFD853DE8004/VTSP:(6/0:D):22:0:0/act_setup_pend_connect: .
Feb  6 11:13:31.015: //6/CFD853DE8004/VTSP:(6/0:D):22:0:0/vtsp_ring_noan_timer_stop:
371382
Feb  6 11:13:31.015: //6/CFD853DE8004/DSM:(6/0:D):-1:0:4098/dsp_stream_mgr_play_tone: .
Feb  6 11:13:31.015: //6/CFD853DE8004/DSM:(6/0:D):-1:0:4098/dsm_exec: [Feat SM: S:NONE B
SM: S:S_DSM_BRIDGING E:E_DSM_CC_GEN_TONE]
Feb  6 11:13:31.015: //6/CFD853DE8004/DSM:(6/0:D):-1:0:4098/dsm_act_gen_tone: Tone is not
on, ignoring
Feb  6 11:13:31.015: //6/CFD853DE8004/CCAPI/cc_api_call_connected: setting
callEntry->connected to TRUE
```

Partial Match Filtering: Example

When the partial match condition is used for voice call debug filtering, no related debug output is filtered until there is a single explicit match failure. In the following example, the configuration, enabled debugs, and debug output for a Cisco 3745 modular access router are shown. Because partial match is set, the router displays the ISDN debug messages on all calls and displays only the **debug vtsp event** messages on the specified dial peer, dial peer 1.

Debug Output for Partial Match Filtering

```
Router# show debug
The following ISDN debugs are enabled on all DSLs:
debug isdn error is             ON.
debug isdn q931 is              ON.   (filter is ON)
Voice Telephony event debugging is on (filter is ON)
```

In the following output, the **show call filter match-list** command is used to show which conditions are set for the specified call filter:

```
Router# show call filter match-list 4
  incoming calling-number 10..
  incoming called-number 50..
  incoming dialpeer 1
 debug condition match-list is set to PARTIAL_MATCH
```

The following debug output shows ISDN debug messages on all calls, but displays only the **debug vtsp event** messages on dial peer 1.

```
*Mar  3 16:21:52.024: ISDN Se2/0:23 Q931: callid 0x0000, callref 0x01E6,
                guid 06ED7A20-170A-11CC-81E7-000B465B86B0
*Mar  3 16:21:52.024: ISDN Se2/0:23 Q931: RX <- SETUP pd = 8  callref = 0x01E6
        Bearer Capability i = 0x8090A2
                Standard = CCITT
                Transer Capability = Speech
                Transfer Mode = Circuit
                Transfer Rate = 64 kbit/s
        Channel ID i = 0xE9808381
                Exclusive, Interface 0, Channel 1
        Calling Party Number i = 0x00, 0x80, '1000'
                Plan:Unknown, Type:Unknown
        Called Party Number i = 0x80, '5000'
                Plan:Unknown, Type:Unknown
*Mar  3 16:21:52.028:
//1270/06ED7A20-170A-11CC-81E7-000B465B86B0/VTSP:(2/0:23):0:0:0/vtsp_process_event:
[state:S_SETUP_INDICATED,  event: E_CC_PROCEEDING]
*Mar  3 16:21:52.032: ISDN Se2/0:23 Q931: callid 0x01EA, callref 0x01E6,
                guid 06ED7A20-170A-11CC-81E7-000B465B86B0
*Mar  3 16:21:52.036: ISDN Se2/0:23 Q931: TX -> CALL_PROC pd = 8  callref = 0x81E6
        Channel ID i = 0xA98381
                Exclusive, Channel 1
*Mar  3 16:21:52.080:
//1270/06ED7A20-170A-11CC-81E7-000B465B86B0/VTSP:(2/0:23):0:0:0/vtsp_process_event:
[state:S_PROCEEDING,  event: E_CC_ALERT]
*Mar  3 16:21:52.084: ISDN Se2/0:23 Q931: callid 0x01EA, callref 0x01E6,
                guid 06ED7A20-170A-11CC-81E7-000B465B86B0
*Mar  3 16:21:52.084: ISDN Se2/0:23 Q931: TX -> ALERTING pd = 8  callref = 0x81E6
        Progress Ind i = 0x8088 - In-band info or appropriate now available
*Mar  3 16:21:52.084:
//1270/06ED7A20-170A-11CC-81E7-000B465B86B0/VTSP:(2/0:23):0:0:0/vtsp_process_event:
[state:S_ALERTING,  event: E_CC_CONNECT]
*Mar  3 16:21:52.088: ISDN Se2/0:23 Q931: callid 0x01EA, callref 0x01E6,
                guid 06ED7A20-170A-11CC-81E7-000B465B86B0
*Mar  3 16:21:52.088: ISDN Se2/0:23 Q931: TX -> CONNECT pd = 8  callref = 0x81E6
*Mar  3 16:21:52.392: ISDN Se2/0:23 Q931: callid 0x01EA, callref 0x01E6,
                guid 06ED7A20-170A-11CC-81E7-000B465B86B0
*Mar  3 16:21:52.392: ISDN Se2/0:23 Q931: RX <- CONNECT_ACK pd = 8  callref = 0x01E6
*Mar  3 16:21:57.024: ISDN Se2/0:23 Q931: callid 0x0000, callref 0x01E7,
                guid 09E86AA0-170A-11CC-81E8-000B465B86B0
*Mar  3 16:21:57.024: ISDN Se2/0:23 Q931: RX <- SETUP pd = 8  callref = 0x01E7
        Bearer Capability i = 0x8090A2
                Standard = CCITT
                Transer Capability = Speech
                Transfer Mode = Circuit
                Transfer Rate = 64 kbit/s
        Channel ID i = 0xE9808382
                Exclusive, Interface 0, Channel 2
        Calling Party Number i = 0x00, 0x80, '1001'
                Plan:Unknown, Type:Unknown
        Called Party Number i = 0x80, '5001'
                Plan:Unknown, Type:Unknown
*Mar  3 16:22:02.032: ISDN Se2/0:23 Q931: callid 0x0000, callref 0x01E8,
                guid 0CE49409-170A-11CC-81E9-000B465B86B0
```

```
*Mar  3 16:22:02.032: ISDN Se2/0:23 Q931: RX <- SETUP pd = 8  callref = 0x01E8
        Bearer Capability i = 0x8090A2
                Standard = CCITT
                Transer Capability = Speech
                Transfer Mode = Circuit
                Transfer Rate = 64 kbit/s
        Channel ID i = 0xE9808383
                Exclusive, Interface 0, Channel 3
        Calling Party Number i = 0x00, 0x80, '1002'
                Plan:Unknown, Type:Unknown
        Called Party Number i = 0x80, '5002'
                Plan:Unknown, Type:Unknown
*Mar  3 16:22:07.032: ISDN Se2/0:23 Q931: callid 0x0000, callref 0x01E9,
                guid 0FDF8489-170A-11CC-81EA-000B465B86B0
*Mar  3 16:22:07.032: ISDN Se2/0:23 Q931: RX <- SETUP pd = 8  callref = 0x01E9
        Bearer Capability i = 0x8090A2
                Standard = CCITT
                Transer Capability = Speech
                Transfer Mode = Circuit
                Transfer Rate = 64 kbit/s
        Channel ID i = 0xE9808384
                Exclusive, Interface 0, Channel 4
        Calling Party Number i = 0x00, 0x80, '1003'
                Plan:Unknown, Type:Unknown
        Called Party Number i = 0x80, '5003'
                Plan:Unknown, Type:Unknown
*Mar  3 16:22:12.032: ISDN Se2/0:23 Q931: callid 0x0000, callref 0x01EA,
                guid 12DA7509-170A-11CC-81EB-000B465B86B0
*Mar  3 16:22:12.032: ISDN Se2/0:23 Q931: RX <- SETUP pd = 8  callref = 0x01EA
        Bearer Capability i = 0x8090A2
                Standard = CCITT
                Transer Capability = Speech
                Transfer Mode = Circuit
                Transfer Rate = 64 kbit/s
        Channel ID i = 0xE9808385
                Exclusive, Interface 0, Channel 5
        Calling Party Number i = 0x00, 0x80, '1004'
                Plan:Unknown, Type:Unknown
        Called Party Number i = 0x80, '5004'
                Plan:Unknown, Type:Unknown
*Mar  3 16:22:17.036: ISDN Se2/0:23 Q931: callid 0x0000, callref 0x01EB,
                guid 15D601B1-170A-11CC-81EC-000B465B86B0
*Mar  3 16:22:17.036: ISDN Se2/0:23 Q931: RX <- SETUP pd = 8  callref = 0x01EB
        Bearer Capability i = 0x8090A2
                Standard = CCITT
                Transer Capability = Speech
                Transfer Mode = Circuit
                Transfer Rate = 64 kbit/s
        Channel ID i = 0xE9808386
                Exclusive, Interface 0, Channel 6
        Calling Party Number i = 0x00, 0x80, '1005'
                Plan:Unknown, Type:Unknown
        Called Party Number i = 0x80, '5005'
                Plan:Unknown, Type:Unknown
*Mar  3 16:22:22.040: ISDN Se2/0:23 Q931: callid 0x0000, callref 0x01EC,
                guid 18D18E59-170A-11CC-81ED-000B465B86B0
*Mar  3 16:22:22.040: ISDN Se2/0:23 Q931: RX <- SETUP pd = 8  callref = 0x01EC
        Bearer Capability i = 0x8090A2
                Standard = CCITT
                Transer Capability = Speech
                Transfer Mode = Circuit
                Transfer Rate = 64 kbit/s
        Channel ID i = 0xE9808387
                Exclusive, Interface 0, Channel 7
```

```
        Calling Party Number i = 0x00, 0x80, '1006'
                Plan:Unknown, Type:Unknown
        Called Party Number i = 0x80, '5006'
                Plan:Unknown, Type:Unknown
*Mar  3 16:22:27.040: ISDN Se2/0:23 Q931: callid 0x0000, callref 0x01ED,
                guid 1BCC7ED9-170A-11CC-81EE-000B465B86B0
*Mar  3 16:22:27.040: ISDN Se2/0:23 Q931: RX <- SETUP pd = 8  callref = 0x01ED
        Bearer Capability i = 0x8090A2
                Standard = CCITT
                Transer Capability = Speech
                Transfer Mode = Circuit
                Transfer Rate = 64 kbit/s
        Channel ID i = 0xE9808388
                Exclusive, Interface 0, Channel 8
        Calling Party Number i = 0x00, 0x80, '1007'
                Plan:Unknown, Type:Unknown
        Called Party Number i = 0x80, '5007'
                Plan:Unknown, Type:Unknown
*Mar  3 16:22:32.048: ISDN Se2/0:23 Q931: callid 0x0000, callref 0x01EE,
                guid 1EC8A7A9-170A-11CC-81EF-000B465B86B0
*Mar  3 16:22:32.048: ISDN Se2/0:23 Q931: RX <- SETUP pd = 8  callref = 0x01EE
        Bearer Capability i = 0x8090A2
                Standard = CCITT
                Transer Capability = Speech
                Transfer Mode = Circuit
                Transfer Rate = 64 kbit/s
        Channel ID i = 0xE9808382
                Exclusive, Interface 0, Channel 2
        Calling Party Number i = 0x00, 0x80, '1008'
                Plan:Unknown, Type:Unknown
        Called Party Number i = 0x80, '5008'
                Plan:Unknown, Type:Unknown
*Mar  3 16:22:34.688: ISDN Se2/0:23 Q931: callid 0x01EA, callref 0x01E6,
                guid 06ED7A20-170A-11CC-81E7-000B465B86B0
*Mar  3 16:22:34.688: ISDN Se2/0:23 Q931: RX <- DISCONNECT pd = 8  callref = 0x01E6
        Cause i = 0x8290 - Normal call clearing
*Mar  3 16:22:34.688: ISDN Se2/0:23 Q931: callid 0x01EA, callref 0x01E6,
                guid 06ED7A20-170A-11CC-81E7-000B465B86B0
*Mar  3 16:22:34.688: ISDN Se2/0:23 Q931: TX -> RELEASE pd = 8  callref = 0x81E6
*Mar  3 16:22:34.688:
//1270/06ED7A20-170A-11CC-81E7-000B465B86B0/VTSP:(2/0:23):0:0:0/vtsp_process_event:
[state:S_CONNECT,  event: E_TSP_DISCONNECT_IND]
*Mar  3 16:22:34.688:
//1270/06ED7A20-170A-11CC-81E7-000B465B86B0/VTSP:(2/0:23):0:0:0/vtsp_process_event:
[state:S_CONNECT,  event: E_CC_DISCONNECT]
*Mar  3 16:22:34.692:
//1270/06ED7A20-170A-11CC-81E7-000B465B86B0/VTSP:(2/0:23):0:0:0/vtsp_process_event:
[state:S_WAIT_STATS,  event: E_VTSP_DSM_STATS_COMPLETE]
*Mar  3 16:22:34.692:
//1270/06ED7A20-170A-11CC-81E7-000B465B86B0/VTSP:(2/0:23):0:0:0/vtsp_process_event:
[state:S_WAIT_RELEASE,  event: E_TSP_DISCONNECT_CONF]
*Mar  3 16:22:34.692:
//1270/06ED7A20-170A-11CC-81E7-000B465B86B0/VTSP:(2/0:23):0:0:0/vtsp_process_event:
[state:S_CLOSE_DSPRM,  event: E_VTSP_DSM_CLOSE_COMPLETE]
*Mar  3 16:22:34.812: ISDN Se2/0:23 Q931: callid 0x01EA, callref 0x01E6,
                guid 06ED7A20-170A-11CC-81E7-000B465B86B0
*Mar  3 16:22:34.812: ISDN Se2/0:23 Q931: RX <- RELEASE_COMP pd = 8  callref = 0x01E6
*Mar  3 16:22:37.048: ISDN Se2/0:23 Q931: callid 0x0000, callref 0x01EF,
                guid 21C39829-170A-11CC-81F0-000B465B86B0
*Mar  3 16:22:37.048: ISDN Se2/0:23 Q931: RX <- SETUP pd = 8  callref = 0x01EF
        Bearer Capability i = 0x8090A2
                Standard = CCITT
                Transer Capability = Speech
                Transfer Mode = Circuit
```

```
                Transfer Rate = 64 kbit/s
        Channel ID i = 0xE9808381
                Exclusive, Interface 0, Channel 1
        Calling Party Number i = 0x00, 0x80, '1009'
                Plan:Unknown, Type:Unknown
        Called Party Number i = 0x80, '5009'
                Plan:Unknown, Type:Unknown
*Mar  3 16:22:42.052: ISDN Se2/0:23 Q931: callid 0x0000, callref 0x01F0,
                guid 24BF24D1-170A-11CC-81F1-000B465B86B0
*Mar  3 16:22:42.052: ISDN Se2/0:23 Q931: RX <- SETUP pd = 8  callref = 0x01F0
        Bearer Capability i = 0x8090A2
                Standard = CCITT
                Transer Capability = Speech
                Transfer Mode = Circuit
                Transfer Rate = 64 kbit/s
        Channel ID i = 0xE9808383
                Exclusive, Interface 0, Channel 3
        Calling Party Number i = 0x00, 0x80, '1010'
                Plan:Unknown, Type:Unknown
        Called Party Number i = 0x80, '5010'
                Plan:Unknown, Type:Unknown
*Mar  3 16:22:47.056: ISDN Se2/0:23 Q931: callid 0x0000, callref 0x01F1,
                guid 27BAB179-170A-11CC-81F2-000B465B86B0
*Mar  3 16:22:47.056: ISDN Se2/0:23 Q931: RX <- SETUP pd = 8  callref = 0x01F1
        Bearer Capability i = 0x8090A2
                Standard = CCITT
                Transer Capability = Speech
                Transfer Mode = Circuit
                Transfer Rate = 64 kbit/s
        Channel ID i = 0xE9808384
                Exclusive, Interface 0, Channel 4
        Calling Party Number i = 0x00, 0x80, '1011'
                Plan:Unknown, Type:Unknown
        Called Party Number i = 0x80, '5011'
                Plan:Unknown, Type:Unknown
```