The MGCP Call Centric Debug feature enables the filtering of debug output based on selected criteria and this feature also standardizes the format of the MGCP debug header. By sharing a common header format all MGCP debug information for a single call can be identified and correlated across the various layers in Cisco IOS software. Filtering debug output reduces extraneous information displayed on the console port, making it easier to locate the correct information and reducing the impact to platform performance, while mitigating lost data because of buffer limits.

Guide Contents
Troubleshooting Cisco IOS Voice Overview
Debug Command Output on Cisco IOS Voice Gateways
Filtering Troubleshooting Output
Cisco VoIP Internal Error Codes
Troubleshooting Cisco IOS Voice Telephony
Troubleshooting Cisco IOS Voice Protocols
Troubleshooting Cisco IOS Telephony Applications
Monitoring the Cisco IOS Voice Network
Cause Codes and Debug Values

Contents

Restrictions for MGCP Call Centric Debug

Filtering conditions that are set for other Cisco IOS modules also impact the debug output for the MGCP module.

Information About MGCP Call Centric Debug

To use the MGCP Call Centric Debug feature, you should understand the following concepts:

- Generic Call Filter Module
- MGCP Debug Commands that Support Debug Filtering
- Matching Conditions
- Trace Levels for MGCP Debug Output
- Tips on Collecting Debug Output

MGCP Debug Commands that Support Debug Filtering

- **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**

Note: Debug filtering is not supported for the **debug mgcp nas**, **debug mgcp packets**, or **debug mgcp parser** commands.

See the Cisco IOS Debug Command Reference for more information about MGCP debug commands.

Match Conditions for MGCP Debug Filtering

To filter calls, you must first define a list of conditions on which to match. The attributes associated with a call are compared to the configured list of match conditions. Debug output that matches all or some of the conditions in the list is displayed, depending on whether the match criteria is set to either exact or partial match.

The MGCP Call Centric Debug feature supports filtering based on the following conditions:

- Incoming signaling IPv4 local address
- Incoming signaling IPv4 remote address
- Incoming media IPv4 local address
- Incoming media IPv4 remote address
- Incoming dial peer
- Outgoing signaling IPv4 local address
- Outgoing signaling IPv4 remote address
- Outgoing media IPv4 local address
- Outgoing media IPv4 remote address

See the Creating Match Lists for MGCP Filtering Conditions for information on configuring match conditions for filtering MGCP calls.

Trace Levels for MGCP Debug Output

The MGCP Call Centric Debug feature introduces trace levels for MGCP debug output. Trace levels allow you to control the amount of information that is displayed by debug commands based on the importance of the content. Trace levels are associated with priority levels that categorize MGCP debug output depending on the information it contains. The output for each debug command is categorized within three priority levels: high, medium, and low.

The following trace levels can be selected to indicate the priority of the information that is displayed:

- Critical-Displays only MGCP debug information marked as high priority.
- Moderate-Displays MGCP debug information marked as medium or high priority.
- Verbose-Displays all MGCP debug information. This is the default level.

Note: The **debug mgcp errors** and **debug mgcp packets** commands do not support trace levels. Their debug output is set to the highest priority and is displayed for all trace level values.

You can set the desired trace level for an MGCP debug session by using the **tracelevel** keyword in individual MGCP debug commands or by setting a global trace level using the **debug mgcp tracelevel-default** command.

Note: Setting the trace level for an endpoint using the **mgcp debug endpoint** command is independent of the global trace level. The endpoint level takes precedence over the global level. For example, the **debug mgcp event tracelevel moderate** command used with the **debug mgcp endpoint aaln/S2/SU0/0 event tracelevel verbose** command sets the trace level to verbose for that specific endpoint while all of the other endpoints have event debugs set at a moderate level. If the global debug is disabled, the per-endpoint debug remains enabled and vice versa.

Tips on Collecting Debug Output

Logging debug output to the console has disadvantages such as being slower and dropping data more easily than logging to a buffer. Collecting debug information by logging output to a buffer instead of the console reduces the impact to gateway performance and decreases the incidence of dropped data.

To log debug output to a buffer instead of the console, use the **no logging console** and **logging buffered** commands. These commands can only be used, however, if there is enough memory available on the gateway to create a large enough buffer to collect the debug output. To display debug output that was collected and sent to the configured buffer, use the **show logging** command.

Logging debug output to the console may also consume an excessive amount of CPU resources if the **logging console guaranteed** command is enabled, which is the default setting. It is recommended that you disable this functionality by using the **no logging console guaranteed** command when sending debug output to the console.

You may also want to use the **service timestamps debug** and **service timestamps log** commands to control how the timestamps are displayed in the debug output.

How to Enable MGCP Call Centric Debug

This section contains the following procedures:

- Modifying the Debug Header Format for MGCP Debug Output (optional)
- Creating Match Lists for MGCP Filtering Conditions (required)
- Enabling MGCP Debug Filtering Using Match Lists (required)
- Verifying the MGCP Debug Filtering Configuration (optional)
- Enabling MGCP Debug Trace Levels (optional)

Modifying the Debug Header Format for MGCP Debug Output

Perform this procedure to modify the standardized header format for MGCP debug output. Debug output is correlated based on this unique header which is common to all debugs belonging to the same call.

SUMMARY STEPS

1. **enable**
2. **configure** terminal
3. **voice call debug** {full-guid | short-header}
4. **mgcp debug-header**
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
1.	**enable** Example: `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
2.	**configure** terminal Example: `Router# configure terminal`	Enters global configuration mode.
3.	**voice call debug** {full-guid | short-header} Example: `Router(config)# voice call debug full-guid`	(Optional) Specifies the full GUID or short header for debug output. • full-guid-Displays the GUID in a 16-byte header. **Note**: Using the **no voice call debug full-guid** command displays the short 6-byte header. • short-header-Displays the CallEntry ID in the header without displaying the GUID or module-specific parameters. This is the default. **Note**: For more information, see the Debug Header Format.
4.	**mgcp debug-header** Example:	(Optional) Enables the MGCP module-dependent information in the debug header.

	`Router(config)# mgcp debug-header`	**Note**: This command is enabled by default. This step is included to illustrate how to enable the command if it was previously disabled.
5.	**exit** Example: `Router(config)# exit`	Exits to privileged EXEC mode.

Creating Match Lists for MGCP Filtering Conditions

Perform this procedure to define match conditions that are used for filtering MGCP calls.

SUMMARY STEPS

1. **enable**
2. **configure** terminal
3. **call filter match-list'** ***number*** *voice'*
4. **incoming signaling** {local | remote} **ipv4** ip-address
5. **incoming media** {local | remote} **ipv4** ip-address
6. **incoming dialpeer** tag
7. **outgoing signaling** {local | remote} **ipv4** ip-address
8. **outgoing media** {local | remote} **ipv4** ip-address
9. **end**

DETAILED STEPS

	Command or Action	Purpose
1.	**enable** Example: `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
2.	**configure** terminal Example: `Router# configure terminal`	Enters global configuration mode.
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 for call filtering (Step 4 to Step 8) must be

		configured.
4.	**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-MGCP call agent. • ip-address-IP address of the local voice gateway or remote call agent.
5.	**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 for the voice gateway receiving the media stream. • local-Local voice gateway. • remote-Remote voice gateway. • ip-address-IP address of the local or remote voice gateway.
6.	**incoming dialpeer** tag Example: `Router(conf-call-filter-mlist)# incoming dialpeer 14`	(Optional) Specifies the incoming telephony dial peer to be filtered. • tag-Digits that define a specific dial peer. Range is 1 to 2147483647. **Note**: Telephony dial peers are configured using the **dial-peer voice** command.
7.	**outgoing signaling** {local | remote} **ipv4** ip-address Example: `Router(conf-call-filter-mlist)# outgoing signaling local ipv4 192.168.10.255`	(Optional) Specifies the outgoing signaling IPv4 address for the gatekeeper managing the signaling. • local-Local voice gateway. • remote-MGCP call agent. • ip-address-IP address of the local gateway or remote call agent.
8.	**outgoing media** {local | remote} **ipv4** ip-address Example:	(Optional) Specifies the outgoing media IPv4 address for the voice gateway receiving the media stream.

	`Router(conf-call-filter-mlist)# outgoing media local ipv4 192.168.10.255`	• local-Local voice gateway. • remote-Remote voice gateway. • ip-address-IP address of the local or remote voice gateway.
9.	**end** Example: `Router(conf-call-filter-mlist)# end`	Exits to privileged EXEC mode.

Enabling MGCP Debug Filtering Using Match Lists

Perform this procedure to enable the match conditions for filtering MGCP debug output.

Prerequisites

The filtering conditions for the debug output must be set as described in the Creating Match Lists for MGCP Filtering Conditions.

Restrictions

- The **debug mgcp nas**, **debug mgcp packets**, and **debug mgcp parser** commands do not support debug filtering.
- Debug output that is outside the context of a call, for example, RSIP, audit, and some endpoint database information does not support filtering.

SUMMARY STEPS

1. **enable**
2. **debug condition match-list** number {exact-match | partial-match}
3. **debug mgcp** {all | endpoint | endptdb | errors | events | gcfm | media | src | state | voipcac}

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** number {exact-match | partial-match} Example: `Router# debug condition match-list 1 exact-match`	Enables the filter match list for the set conditions. • *number*-Numeric label that uniquely identifies the match list. Range is 1 to 16. This number 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. **Note**: This command impacts all enabled debug commands that support call filtering.

3.	**debug mgcp** {all | endpoint | endptdb | errors | events | gcfm | inout | media | src | state | voipcac}
	
	Example:
	`Router# debug mgcp errors`
	`Router# debug mgcp events`
	`Router# debug mgcp endpoint aaln/s2/su0/1/1/10`
	`Router# debug mgcp media`
		Enables the appropriate MGCP debug commands.
		
		• See the Cisco IOS Debug Command Reference for detailed descriptions of these debug commands.
		
		Note: When enabling MGCP debug commands, you can also set a trace level to further filter output based on the importance of the information. For information, see the Enabling MGCP Debug Trace Levels.

Verifying the MGCP Debug Filtering Configuration

To verify debug filtering conditions, use the following commands:

- **show debug**-Displays the debugs that are enabled.
- **show call filter components**-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**-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.

See the Cisco IOS Debug Command Reference for more information about these commands.

Enabling MGCP Debug Trace Levels

Perform this procedure to enable trace levels for restricting MGCP debug output based on the priority of the information.

Restrictions

Trace levels are not supported for MGCP errors or packets debugging because all of the output from these commands is set to high priority.

SUMMARY STEPS

1. **enable**
2. **debug mgcp tracelevel-default** {critical | moderate | verbose}
3. **debug mgcp endpoint** endpoint-name {{all | events | media} [tracelevel {critical | moderate | verbose}] | {errors | packets}}
4. **debug mgcp** {all | endptdb | events | gcfm | inout | media | nas | parser | src | state | voipcac} [tracelevel {critical | moderate | verbose}]

DETAILED STEPS

Command or Action	Purpose	
1.	**enable** Example: `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
2.	**debug mgcp tracelevel-default** {critical | moderate | verbose} Example: `Router# debug mgcp tracelevel-default critical`	(Optional) Enables the trace level globally for all MGCP debug commands and endpoints. • critical-Only high priority debug information is displayed. • moderate-Medium and high priority debug information is displayed. • verbose-All debug information is displayed. This is the default trace level.
3.		

	debug mgcp endpoint endpoint-name {{all | events | media} [tracelevel {critical | moderate | verbose}] | {errors | packets}} Example: `Router# debug mgcp endpoint aaln/s2/su0/1/1/10`	(Optional) Enables the trace level for a specific endpoint for events or media debug commands. • endpoint-name-Name of the MGCP endpoint for which to enable debugging. Must be a fully specified and supported endpoint.

4.	**debug mgcp** {all | endptdb | events | gcfm | inout | media | nas | parser | src | state | voipcac} [tracelevel {critical | moderate | verbose}]
	
	Example:
	
	`Router# debug mgcp events tracelevel critical`
	`Router# debug mgcp state tracelevel moderate`
	`Router# debug mgcp media moderate`
		(Optional) Enables the trace level for a specific MGCP debug command.

Configuration Examples for MGCP Call Centric Debug

This section contains the following examples:

- Match-List Configuration for MGCP Debug Filtering: Example
- Enabling MGCP Debug Filtering: Example

Match-List Configuration for MGCP Debug Filtering: Example

The following example shows a configuration with a match list defined to filter MGCP debug output.

```
Router# show running-config
Building configuration...
Current configuration : 2068 bytes
!
version 12.4
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
service internal
!
hostname Router
!
boot-start-marker
boot system flash:Router.ios.bin
boot-end-marker
!
logging buffered 10000000 debugging
enable secret 5 $1$abcd
enable password sample
!
no aaa new-model
!
resource policy
!
no network-clock-participate slot 1
```

```
no network-clock-participate slot 2
ip cef
!
!
!
no ip domain lookup
ip host callagenthost 192.168.1.200
voice-card 1
 no dspfarm
!
voice-card 2
 dspfarm
!
!
!
!
!
!
!
controller T1 1/0
 framing esf
 clock source internal
 linecode b8zs
 ds0-group 0 timeslots 1-24 type none service mgcp
!
controller T1 1/1
 shutdown
 framing esf
 clock source internal
 linecode b8zs
 ds0-group 0 timeslots 1-24 type none service mgcp
!
!
!
!
interface FastEthernet0/0
 ip address 192.168.1.79 255.255.255.0
 no ip mroute-cache
 speed auto
 half-duplex
 no cdp enable
!
interface FastEthernet0/1
 no ip address
 no ip mroute-cache
 shutdown
 duplex auto
 speed auto
 no cdp enable
!
!
ip http server
!
snmp-server community public RO
snmp-server enable traps tty
!
!
!
control-plane
!
!
!
call filter match-list 1 voice
 incoming media local ipv4 192.168.1.12
```

```
 outgoing media local ipv4 192.168.1.11
!
voice-port 1/0:0
!
voice-port 1/1:0
!
voice-port 2/0/0
!
voice-port 2/0/1
!
voice-port 2/0/2
!
voice-port 2/0/3
!
voice-port 2/1/0
!
voice-port 2/1/1
!
voice-port 2/1/2
!
voice-port 2/1/3
!
!
mgcp
mgcp call-agent callagenthost 7979 service-type mgcp version 1.0
mgcp package-capability mf-package
mgcp package-capability rtp-package
mgcp package-capability script-package
mgcp sdp simple
!
mgcp profile default
!
!
!
dial-peer voice 211 pots
 service mgcpapp
 port 2/1/1
!
dial-peer voice 213 pots
 service mgcpapp
 port 2/1/3
!
dial-peer voice 210 pots
 service mgcpapp
 port 2/1/0
!
dial-peer voice 200 pots
 service mgcpapp
 port 2/0/0
!
dial-peer voice 212 pots
 service mgcpapp
 port 2/1/2
!
!
line con 0
 exec-timeout 0 0
line aux 0
line vty 0 4
 password temp
 login
!
!
end
```

Enabling MGCP Debug Filtering: Example

The following example shows how to enable filtering and trace levels for MGCP debug output.

```
Router# debug condition match-list 1 exact-match
Router# debug mgcp tracelevel-default critical
Router# debug mgcp errors
Media Gateway Control Protocol errors debugging for all endpoints is on
Router# debug mgcp media
Media Gateway Control Protocol media events debugging for all endpoints is on, trace-level
Critical
Router# debug mgcp state tracelevel verbose
Media Gateway Control Protocol state transition debugging for all endpoints is on,
trace-level Verbose
Router# debug mgcp endpoint S1/ds1-0/1 events tracelevel moderate
Media Gateway Control Protocol events debugging for endpoint s1/ds1-0/1 is on, trace-level
Moderate
Router# show debug
MGCP:
  Media Gateway Control Protocol media events debugging is on, trace level Critical
  Media Gateway Control Protocol errors debugging is on
  Media Gateway Control Protocol state transition debugging is on, trace level Verbose
MGCP: Event debugging for endpoint S1/DS1-0/1 is on, tracelevel is Moderate
Router# show call filter match-list
****************************************
call filter match-list 1 voice
****************************************
  incoming media local ipv4  192.168.1.12
  outgoing media local ipv4  192.168.1.11
debug condition match-list is set to EXACT_MATCH
```