Cisco UBE Serviceability for Event Logging and Debug Classification

The Cisco Unified Border Element (Cisco UBE) Serviceability for Event Logging and Debug Classification feature helps support, test, and development engineers to troubleshoot during high-density call volumes without significantly impacting performance. This feature introduces a new mechanism for tracing the calls and issues, and generating and collecting needed information, on Cisco UBE via Event Logging.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions

- Traces captured and not written to files will be lost during HA switchover (but they are captured in core).

- Enabling serviceability will write the content to file. As file read/write operation is slow, there will be an impact on performance.
- The dump folder must be reconfigured if cube_et_folder_map.info and et_fold_size files are deleted from flash.
- The dump folder commands must be reconfigured if folder permissions are modified.

Information About Cisco UBE Serviceability for Event Logging and Debug Classification

Serviceability

In a Cisco Unified Border Element (Cisco UBE) system, serviceability refers to the ability of technical support and engineering personnel to troubleshoot issues and restore the service to customers in a high call-volume systems. Cisco UBE includes the following:

- Enhacements to the existing debug logging mechanisms to allow SIP-INFO-DEBUG to be sub-categorized based on importance level (Verbose, Info, Notify, and Critical) and the feature set.
- Cisco UBE Event Trace Manager, which supports tracing for Voice over IP (VoIP) networks.

Event Tracing

Cisco Unified Border Element (Cisco UBE) event tracing enables support, test, and development engineers to debug specific issues related to Cisco UBE. For example, they can use it to identify the root cause of issues that occur in the past. Event -tracing allows various VoIP/SIP events related to the SIP signaling layer of the VoIP call to be traced as they occur. Event tracing provides flexibility to configure the mechanism to a specific customer topology and deployment, including the ability to filter the traces based on call-parameters and time.

Note The event tracing mechanism allows event-trace messages to be written in raw (binary) or encoded (pretty) format.

Debug Message Categories

The Cisco Unified Border Elelment (Cisco UBE) debug categorization mechanism enhances the existing debug framework by adding more filters to control the verbosity. These categories apply to the existing INFO debugs. The messages are subcategorized to control the amount of information logged when info logging in enabled. Therefore, INFO debugs comprise of the following subcategories based on their importance:

- Critical—These errors are feature specific.
- Notification—These errors provide information on important milestones reached.
- Information—These errors provide details to help an engineer understand the workflow.

- Verbose—These errors provide detailed information on all of the above.

The debug messages can also be subcategorized based on a selected feature set (such as SIP profile, fax, audio, or video).

Note Only one level can be selected. By default Verbose level is enabled. The amount of information provided by the debug messages grows in the increasing order of their listing. For example, Notification provides additional information to that provided by the previous category (Critical) and so on.

Dump File and Folder Management

- Event trace generates multiple files for a single call (5 files per call leg for binary dump and 2 files per call leg for text or pretty dump). A CUBE processing calls for a long time can dump a lot of files into a single folder making it difficult to manage the files. Dumping of event traces to folders results in efficient management of event trace files.
- The following storage can be used for creating event trace files.
    - File system: Flash, hard-disk, USB
    - Network storage: FTP, TFTP
- The user must create a directory in the storage before configuring the dump-file.

Note Folder management applies to device storage and external storage where folders can be created and deleted as per configuration. For FTP and TFTP, all files are dumped into a single folder.

New Events and CCSIP Formatting

The following new events are captured by event traces:

- Out-of-band digits.
- Various error conditions (such as codec mismatch, DNS failure are captured to show more details about a call failure).
- Source of call disconnection (disconnect cause code information is captured). Example— *Feb 25 07:21:09.782: sip_misc : CUBE_ET: TYPE = MISC : Call Disconnect: Initiated at: 0x2600674, Originated at:0x2600675, Cause Code = 22*. This example shows the sample event trace captured for call disconnect, which shows cause code and the hexadecimal line number which initiated call disconnect. This is useful for identifying the trigger for call disconnect.

CCSIP Formatting Details—You can export CCSIP formatting details into an XML format based on the release. This can be used to write a decoder to read the content from the binary event trace files.

High Availability Support

- No data is check pointed between the active and standby device.
- Active and standby creates dump files independently.
- You must create the dump-folder manually on both the active and standby devices before configuring the dump folder.
- In a high availability setup, first configure the event trace on the standby device.

How to Configure Cisco UBE Serviceability for Event Logging and Debug Classification

How to Configure Event Tracing

Controlling Cisco UBE Serviceability Event Tracing

Perform this task to disable, clear, and re-enable event traces, export CCSIP formatting details, and to allow the event traces to be stored permanently to secondary or network storage.

SUMMARY STEPS

1. **enable**
2. **monitor event-trace voip ccsip all dump** [**pretty**]
3. **monitor event-trace voip ccsip all disable**
4. **monitor event-trace voip ccsip all clear**
5. **monitor event-trace voip ccsip all enable**
6. **monitor event-trace voip ccsip export format-xml**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**monitor event-trace voip ccsip all dump** [**pretty**]	Writes the event trace results in ASCII format to the file configured with the global configuration **monitor event-trace voip ccsip dump-file** command. If you do not specify the **pretty** keyword, the trace messages are saved in binary format.

	Command or Action	Purpose
	Example: `Device# monitor event-trace voip ccsip all dump pretty`	
Step 3	**monitor event-trace voip ccsip all disable** **Example:** `Device# monitor event-trace voip ccsip all disable`	Stops all API, Finite State Machine (FSM), Communicating Nested FSM (CNFSM), message and miscellaneous event tracing.
Step 4	**monitor event-trace voip ccsip all clear** **Example:** `Device# monitor event-trace voip ccsip all clear`	Clear the traces for active calls captured so far.
Step 5	**monitor event-trace voip ccsip all enable** **Example:** `Device# monitor event-trace voip ccsip all enable`	If event-tracing is disabled, this command reenables event tracing for API, FSM, CNFSM, message and miscellaneous events that are configured through global configuration mode. This command does not re-enable global or history event tracing.
Step 6	**monitor event-trace voip ccsip export format-xml** **Example:** `Device# monitor event-trace voip ccsip export format-xml`	Exports CCSIP formatting details into an XML format based on the release. This format can be used to write a decoder to read the content from the binary event trace files. Use this command to get release-specific XML format. This command dumps the XML into the configured event trace folder.

Configuring Cisco UBE Serviceability Event Tracing

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **monitor event-trace voip ccsip** *trace-type* [**size** *number*]
4. **monitor event-trace voip ccsip dump** *dump-type*
5. **monitor event-trace voip ccsip dump-file** *file-name*
6. **monitor event-trace voip ccsip limit connections** *max-connections*
7. **monitor event-trace voip ccsip limit memory** *size*
8. **monitor event-trace voip ccsip stacktrace** *number*
9. **monitor event-trace voip ccsip dump-folder size** *size*
10. **monitor event-trace voip ccsip max-dump-limit** *size_in_MB*
11. **monitor event-trace voip ccsip dump all periodic**
12. **monitor event-trace voip ccsip dump marked**
13. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**monitor event-trace voip ccsip** *trace-type* [**size** *number*] **Example:** `Device(config)# monitor event-trace voip ccsip api size 50`	Enables event tracing for various Voice Over IP (VoIP) CCSIP API events. Event tracing for other events, such as Finite State Machine (FSM), Communicating Nested FSM (CNFSM), miscellaneous, message, and global events can be enabled in a similar way.
Step 4	**monitor event-trace voip ccsip dump** *dump-type* **Example:** `Device(config)# monitor event-trace voip ccsip dump marked`	(Optional) Specifies the automatic dump policy for VoIP CCSIP events. Available options are **marked**, **all**, or **none** (default).

	Command or Action	Purpose

Step 5	**monitor event-trace voip ccsip dump-file** *file-name*
	
	Example:
	
	`Device(config)# monitor event-trace voip ccsip dump-file slot0:ccsip-dump-file`
	`OR`
	`Device(config)#monitor event-trace voip ccsip dump-file ftp://username:password@server_ip//path/ccsip-dump-file`
	
	`OR`
	`Device(config)#monitor event-trace voip ccsip dump-file tftp://server_ip//path/ccsip-dump-file.txt`
		(Optional) Specifies the file where event trace messages are written from memory to permanent storage. You can also configure the folder for the dump-file using the **monitor event-trace voip ccsip dump-file** *folder_path* command. The folder path can be flash:*folder_name*
		
		or bootflash:*folder_name*, usb0:*folder_name*, or harddisk:*folder_name*.
		
		• If there is a failure in configuring the dump file, then a syslog is generated as follows: *%SIP-5-EVENT_TRACE_PATH_ERR: Event Trace Dump PATH "tftp://223.255.254.254/eventtrace" not accesible. Verify credentials, directory path and network connectivity.*

Step 6	**monitor event-trace voip ccsip limit connections** *max-connections* **Example:** `Device(config)# monitor event-trace voip ccsip limit connections 500`	(Optional) Limits the resources used by the event tracing mechanism based on the number of connections or call legs. The default limit is 1000 connections.
Step 7	**monitor event-trace voip ccsip limit memory** *size* **Example:** `Device(config)# monitor event-trace voip ccsip limit memory 50`	(Optional) Limits the resources used by the event tracing mechanism to 50 MBytes.
Step 8	**monitor event-trace voip ccsip stacktrace** *number* **Example:** `Device(config)# monitor event-trace voip ccsip stacktrace 9`	(Optional) Enables the stack trace at tracepoints and specifies the depth of the stack trace stored.

Step 9	**monitor event-trace voip ccsip dump-folder size** *size*
	
	Example:
	
	`Device(config)# monitor event-trace voip ccsip dump-folder size 20`
	`OR`
	`Device(config)# monitor event-trace voip ccsip dump-folder time 500`
		(Optional) Configures the dump-folder size (or time) based on the storage available for the rotation of files. You can also configure the dump-folder time using the **monitor event-trace voip ccsip dump-folder time***time* command.
		
		• Sub-folders are created by CUBE under the user-created folder based on timestamp. When one of the above conditions (size or time) is met, folder rotation happens creating a new folder, and CUBE continues to dump the new event trace files to the new folder.

	Command or Action	Purpose
		• Each folder contains a file et_fold_size, which captures the size of the folder that is used for folder rotation.
Step 10	**monitor event-trace voip ccsip max-dump-limit** *size_in_MB* **Example:** `Device(config)# monitor event-trace voip ccsip max-dump-limit 1000`	(Optional) Sets the limit of the maximum size of event traces. • This command creates a file in flash with the name et_fold_size, which is used to store the current size used by event traces. • Once the event trace size reaches the configured maximum dump limit, then the event trace folder is automatically purged. The folder created first (oldest) is purged first. All files under a folder are deleted in a purge. • Select the maximum-dump-limit based on the space available in storage.
Step 11	**monitor event-trace voip ccsip dump all periodic** **Example:** `Device(config)# monitor event-trace voip ccsip dump all periodic`	(Optional) Dumps the buffer content to file when the buffer is full. • When the buffer is full, the content is written or appended to same file so that all event traces for a particular leg is available in a single file. • Periodic dump is applicable for all and marked dumping. • Periodic dump can be configured for binary or pretty format.
Step 12	**monitor event-trace voip ccsip dump marked** **Example:** `Device(config)# monitor event-trace voip ccsip dump marked`	(Optional) Configures dumping of marked traces. • There are certain conditions, which handle fatal and unexpected events; the call will be disconnected abnormally in such cases. Trace-marks are added to capture these details in event trace. These trace-marks are pre-defined in CUBE and you can configure to dump these traces.
Step 13	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode.

Monitoring Cisco UBE Serviceability Event Tracing

Perform this task to monitor Cisco Unified Border Element (Cisco UBE) serviceability for event tracing and logging parameters. Depending on your requirements, you can view the event traces of the Cisco UBE based on several parameters. The commands can be entered in any order.

SUMMARY STEPS

1. **enable**
2. **show monitor event-trace voip ccsip** *trace-type* **filter called-num** *filter-value* **all**
3. **show monitor event-trace voip ccsip** *trace-type* **all**
4. **show monitor event-trace voip ccsip summary**
5. **show monitor event-trace voip history all**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode.
Step 2	**show monitor event-trace voip ccsip** *trace-type* **filter called-num** *filter-value* **all** **Example:** `Device# show monitor event-trace voip ccsip api filter called-num 88888 all`	Displays the captured event traces for API events for in-progress calls made to the specified number.
Step 3	**show monitor event-trace voip ccsip** *trace-type* **all** **Example:** `Device# show monitor event-trace voip ccsip fsm all`	Displays the captured event traces for Finite State Machine (FSM) and Communicating Nested FSM (CNFSM) events.
Step 4	**show monitor event-trace voip ccsip summary** **Example:** `Device# show monitor event-trace voip ccsip summary`	Displays a summary of all captured event traces.
Step 5	**show monitor event-trace voip history all** **Example:** `Device# show monitor event-trace voip ccsip history all`	Displays the captured traces for completed calls.

Configuring Cisco UBE Serviceability Debug Classification

Perform this task to classify debug messages to support Cisco Unified Border Element (Cisco UBE) serviceability features, and to display Cisco UBE debug category code information.

SUMMARY STEPS

1. **enable**
2. **debug ccsip info**
3. **debug ccsip feature** *feature-name feature-name feature-name feature-name feature-name*
4. **debug ccsip level critical**
5. **show cube debug category codes**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**debug ccsip info** **Example:** `Device# debug ccsip info`	Enables CCSIP INFO debugging.
Step 3	**debug ccsip feature** *feature-name feature-name feature-name feature-name feature-name* **Example:** `Device# debug ccsip feature audio cac dtmf fax registration`	Enables filtering of CCSIP INFO debugs based on various features. Debugs for specified and enabled features are printed.
Step 4	**debug ccsip level critical** **Example:** `Device# debug ccsip level critical`	Enables CCSIP critical level debugging messages.
Step 5	**show cube debug category codes** **Example:** `Device# show cube debug category codes`	Displays Cisco Unified Border Element debug category code information.

Monitoring Active Calls

Perform this task to monitor and display information on the total number of active calls in the system.

SUMMARY STEPS

1. **show call active total-calls**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**show call active total-calls**
	
	Example:
	`Device# show call active total-calls`
	`Total Number of Active Calls : 110`
		Displays the total number of active calls in the system.

Configuration Examples for Cisco UBE Serviceability for Event Logging and Debug Classification

Example: Controlling Cisco UBE Serviceability Event Tracing

The following example shows how to allow the event traces to be stored permanently to secondary storage and how to control event trace logging:

```
Device> enable
Device# monitor event-trace voip ccsip all dump pretty
Device# monitor event-trace voip ccsip all disable
Device# monitor event-trace voip ccsip all clear
Device# monitor event-trace voip ccsip all enable
```

Example: Configuring Cisco UBE Serviceability Event Tracing

The following example shows how to configure event tracing in the system:

```
Device> enable
Device# configure terminal
Device(config)# monitor event-trace voip ccsip api size 50
Device(config)# monitor event-trace voip ccsip fsm size 100
Device(config)# monitor event-trace voip ccsip global size 100
Device(config)# monitor event-trace voip ccsip misc size 50
Device(config)# monitor event-trace voip ccsip msg size 50
Device(config)# monitor event-trace voip ccsip dump marked
Device(config)# monitor event-trace voip ccsip dump-file slot0:ccsip-dump-file
Device(config)# monitor event-trace voip ccsip limit connections 1000
Device(config)# monitor event-trace voip ccsip stacktrace 9
Device(config)# exit
```

Example: Monitoring Cisco UBE Serviceability Event Tracing

The following example shows how to monitor event tracing in the system:

```
Device> enable
Device# show monitor event-trace voip ccsip api filter called-num 88888 all
Device# show monitor event-trace voip ccsip fsm all
Device# show monitor event-trace voip ccsip summary
Device# show monitor event-trace voip ccsip history all
```

Example: Configuring Cisco UBE Serviceability Debug Classification

The following example shows how to configure debug messages for Cisco Unified Border Element (Cisco UBE) serviceability features:

```
Device> enable
Device# debug ccsip info
SIP Call info tracing is enabled
Device# debug ccsip feature audio cac dtmf fax registration
audio debugging for ccsip info is enabled (active)
fax debugging for ccsip info is enabled (active)
dtmf debugging for ccsip info is enabled (active)
cac debugging for ccsip info is enabled (active)
registration debugging for ccsip info is enabled (active)
Device# debug ccsip level critical
critical mode tracing for ccsip info is enabled (active)
Device# show cube debug category codes

|------------------------------------------------
| show cube debug category codes values.
|------------------------------------------------
| Indx |  Debug Name           |   Value
|------------------------------------------------
|  01  |  SDP Debugs           |   1
|  02  |  Audio Debugs         |   2
|  03  |  Video Debugs         |   4
|  04  |  Fax Debugs           |   8
|  05  |  SRTP Debugs          |   16
|  06  |  DTMF Debugs          |   32
|  07  |  SIP Profiles Debugs  |   64
|  08  |  SDP Passthrough Deb  |   128
|  09  |  Transcoder Debugs    |   256
|  10  |  SIP Transport Debugs |   512
|  11  |  Parse Debugs         |   1024
|  12  |  Config Debugs        |   2048
|  13  |  Control Debugs       |   4096
|  14  |  Miscellaneous Debugs |   8192
|  15  |  Supp Service Debugs  |   16384
|  16  |  Misc  Features Debugs|   32768
|  17  |  SIP Line-side Debugs |   65536
|  18  |  CAC Debugs           |   131072
|  19  |  Registration Debugs  |   262144
|------------------------------------------------
```

Example: Monitoring Active Calls

The following example shows how to view all active calls in the system:

```
Device> enable
Device# show call active total-calls
Total Number of Active Calls : 110
```

Additional References for Cisco UBE Serviceability for Event Logging and Debug Classification

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
Voice commands	• Cisco IOS Voice Command Reference - A through C • Cisco IOS Voice Command Reference - D through I • Cisco IOS Voice Command Reference - K through R • Cisco IOS Voice Command Reference - S Commands • Cisco IOS Voice Command Reference - T through Z Commands

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for Cisco UBE Serviceability for Event Logging and Debug Classification

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 1: Feature Information for Cisco UBE Serviceability for Event Logging and Debug Classification

Feature Name	Releases	Feature Information
Cisco UBE Serviceability for Event Logging and Debug Classification	15.3(3)M Cisco IOS Release XE 3.10S	The Cisco Unified Border Element (Cisco UBE) Serviceability for Event Logging and Debug Classification feature helps support, test, and development engineers to troubleshoot during high-density call volumes without significantly impacting performance. This feature introduces a new mechanism for tracing the calls and issues, and generating and collecting needed information, on Cisco UBE via Event Logging. The following commands were introduced or modified: **debug ccsip feature**, **debug ccsip level**, **monitor event-trace voip ccsip**, **monitor event-trace voip ccsip (EXEC)**, **monitor event-trace voip ccsip dump-file**, **monitor event-trace voip ccsip dump**, **monitor event-trace voip ccsip limit**, **monitor event-trace voip ccsip stacktrace**, **show call active total-calls**, **show cube debug category codes**, and **show monitor event-trace voip ccsip (EXEC)**.
Cisco UBE Serviceability Enhancements	15.4(2)T Cisco IOS XE Release 3.12S	The event trace functionality was enhanced with the following: • Dump file and folder management • New events • New dump policy • New trace-mark points for auto-dumping • CCSIP formatting details