

Set Up and Gather Trace Data in CUE

Document ID: 53207

Introduction

Prerequisites

- Requirements
- Components Used
- Conventions

Trace Overview

- Advanced Integration Module (AIM) versus Network Module (NM)

Configure Traces

Gather Trace Data

- Work with the Trace Buffer
- Stored Trace Log Files
- Trace to an FTP Server

JTAPI Traces

Turn Off Traces

Re-enable Default Traces

Related Information

Introduction

This document provides an overview of the trace capabilities in Cisco Unity Express (CUE). Trace is the debug facility in Cisco Unity Express and is used to troubleshoot a variety of issues. When the trace function is enabled, it can have a negative impact on system performance. Because of this issue, Cisco recommends that you only enable trace at the request of Cisco Technical Support to gather information about specific problems. For systems in the lab or in maintenance windows, the trace function can be used to troubleshoot and understand Cisco Unity Express behavior.

Prerequisites

Requirements

Cisco recommends that you have a basic familiarity with administration and use of Cisco Unity Express through the command-line interface (CLI).

Components Used

This feature requires Cisco Unity Express version 1.0 or later. The integration method (either Cisco CallManager or Cisco CallManager Express) is not important. All sample configurations and screen output are taken from Cisco Unity Express version 1.1.1.

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, make sure that you understand the potential impact of any command.

Conventions

Refer to Cisco Technical Tips Conventions for more information on document conventions.

Trace Overview

People familiar with Cisco IOS® Software are more than likely not familiar with its CLI and powerful debug capability. Cisco Unity Express has tools that are similar in functionality, but have some important differences. In Cisco Unity Express, the **debug** command does not exist. Instead, there is a **trace** command. The trace facility is a diagnostics facility that writes messages within a kernel buffer in memory. This memory space, which can be up to 10 MB in size, is periodically (if configured) written to a local trace file (atrace.log), to a file on an external FTP server, or to both.

Note: The atrace.log file and the trace data logged to the FTP server are not in plain text. The data must be sent to Cisco Technical Support for diagnosis.

You can manually copy each of the files written on the Cisco Unity Express module (atrace.log and messages.log, as well as others) to an external FTP server.

Cisco Unity Express also supports a log facility that is similar to syslog in Cisco IOS Software. These messages are from the Operating System as well as other application sources that are categorized in different levels. These are Info, Warning, Error, and Fatal messages that are written to another file on Cisco Unity Express (messages.log). They can also be written to an external syslog server, as well as to the console of Cisco Unity Express. By default, only Fatal messages are logged on the AIM.

For most general problems, the messages.log file and the trace information for the failure is required.

If instructed by Cisco Technical Support to gather specific traces, you should agree upon the specific traces that need to be enabled and the method for capture. For example, you can use real-time traces, view the trace memory buffer, or capture the trace data on an FTP server.

Advanced Integration Module (AIM) versus Network Module (NM)

Cisco Unity Express has two hardware models, the AIM and the NM. In terms of the trace function, there are some important differences between the two:

AIM	NM
The atrace.log file is disabled by default. Issue the log trace local enable command to begin and the log trace local disable command to stop.	The atrace.log file is enabled by default.
The maximum size of atrace.log is 10 MB.	Tracing to an external server is also supported. The maximum size of atrace.log is 100 MB.
The atrace.log file does <i>not</i> wrap when full.	The atrace.log file wraps when full.

How to enable and view trace data is explained in more detail later in this document. The AIM does not store any trace information in Flash itself by default. Also, the internal storage capacity for trace data, when enabled, is much more limited. This is because the lifespan of the internal compact Flash card on the AIM is related to the number of writes issued to it. Constantly writing traces significantly lowers the lifespan.

Note: Issue the **log trace local disable** command followed by the **log trace local enable** command in configuration mode in order to restart an atrace.log file on an AIM that has reached its maximum size. This removes the original atrace.log file and begins a new one.

For the log facility, there are also important differences:

AIM	NM
Only Fatal messages are logged to the messages.log file by default. Issue the command log console info from configuration mode to see all messages.	All categories of messages are logged to the messages.log file.

Configure Traces



Caution: The configuration of traces on Cisco Unity Express can have a negative impact on system performance, especially when you write to a local log file that is enabled. This can include delays in prompts and dual-tone multifrequency (DTMF) tone response times, as well as quality problems in recorded or played audio. Configure traces with caution.

The trace configuration controls the types of messages that are written to the trace memory buffer. This 10 MB buffer always wraps so that the latest trace information is present. Because systems have different levels of activity, it is impossible to tell what time period this trace buffer covers. However, if configured, the buffer is written to a local atrace.log file or an FTP server.

You can only configure traces from the Cisco Unity Express CLI. Issue the **show trace** command to view the current traces that are enabled.

For example:

```
vnt-3745-44a#service-module service-Engine 4/0 session
Trying 172.18.106.66, 2129 ... Open
vnt-nm-cue#
vnt-nm-cue#show trace
MODULE          ENTITY          SETTING
ccn             Engine          00000001
ccn             LibLdap         00000001
ccn             SubsystemAppl  00000001
ccn             ManagerAppl     00000001
ccn             ManagerChannel 00000001
ccn             SubsystemJtapi 00000001
ccn             SubsystemSip    00000001
ccn             StackSip        00000001
ccn             SubsystemHttp  00000001
ccn             VbrowserCore   00000001
ccn             SubsystemCmt   00000001
ccn             LibMedia        00000001
ccn             ManagerContact 00000001
ccn             StepCall        00000001
ccn             StepMedia       00000001
config-ccn     sip-subsystem   00000001
config-ccn     jtapi-subsystem 00000001
config-ccn     sip-trigger     00000001
config-ccn     jtapi-trigger   00000001
config-ccn     http-trigger    00000001
config-ccn     group           00000001
config-ccn     application     00000001
config-ccn     script          00000001
config-ccn     prompt         00000001
config-ccn     miscellaneous   00000001
voicemail     database        0000008f
voicemail     mailbox         0000003f
voicemail     message         0000002f
```

```
webInterface      initwizard      00000001
vnt-nm-cue#
```

These are the default trace settings for both the NM and the AIM. The AIM does not store the output of these traces anywhere by default. For most general troubleshooting, these trace levels are sufficient. If a problem occurred recently, chances are that there is still some history in the trace memory buffer.

Issue the **trace module entity activity** command to enable traces. For example:

```
vnt-nm-cue#trace ccn subsystemsip debug
```

These are the modules for CUE 1.1.1:

```
vnt-nm-cue#trace ?
BackupRestore Module
all             Every module, entity and activity
ccn             Module
config-ccn     Module
dns            Module
superthread    Module
udppacer       Module
voicemail      Module
webInterface   Module
```

There are many entities under each module. The activity level varies somewhat (usually from module to module). In general, every entity has at least a *debug* (sometimes spelled *DEBUG*) and an *all* activity level. In general, the debug activity level is sufficient.

The **trace module entity activity** command can be issued multiple times until traces for all desired modules and entities are enabled.

It does not matter which traces are set. After a reload, the system reverts to the default trace levels. In order to change these default settings so that they survive a reboot, you must issue the **log trace boot** command.

Gather Trace Data

Once all traces have been configured, the data is written to the memory buffer. Then it can either be displayed real-time as the messages come in, or the buffer can be viewed after the event or test has occurred.

Work with the Trace Buffer

The memory-based trace buffer is one of the first places to examine traces. It can be viewed real-time, so trace messages are displayed as they come in. As an alternative, all or part of the memory buffer can be displayed and examined.

Real-time Traces

Real-time traces are especially useful when you troubleshoot problems in a controlled system (when there are not many simultaneous calls or the system is not yet in production). Because the trace output lines often wrap and the information almost always scrolls by faster than it can be read, log the console output to a text file before you turn on the real-time traces. This allows the information to be viewed later in a text editor. For instance, in Microsoft Hyperterminal, you can choose **Transfer > Capture Text** and then designate a capture file.

The real-time trace function also has the highest performance impact on a system. Use it with caution.

Issue the **show trace buffer tail** command to view trace information real-time. For example:

```
vnt-nm-cue>show trace buffer tail
Press <CTRL-C> to exit...
295 06/22 10:39:55.428 TRAC TIMZ 1 EST EDT 18000
2019 06/22 11:20:15.164 ACCN SIPL 0 receive 1098 from 172.18.106.66:54948
2020 06/22 11:20:15.164 ACCN SIPL 0 not found header for Date
2020 06/22 11:20:15.164 ACCN SIPL 0 not found header for Supported
2020 06/22 11:20:15.164 ACCN SIPL 0 not found header for Min-SE
2020 06/22 11:20:15.165 ACCN SIPL 0 not found header for Cisco-Guid
2020 06/22 11:20:15.165 ACCN SIPL 0 not found header for Remote-Party-ID
2020 06/22 11:20:15.165 ACCN SIPL 0 not found header for Timestamp
2020 06/22 11:20:15.165 ACCN SIPL 0 not found header for Call-Info
2020 06/22 11:20:15.165 ACCN SIPL 0 not found header for Allow-Events
2020 06/22 11:20:15.166 ACCN SIPL 0 -----
INVITE sip:18999@172.18.106.88:5060 SIP/2.0
Via: SIP/2.0/UDP 172.18.106.66:5060;branch=z9hG4bK1678
From: "Caller1" <sip:201@172.18.106.66>;tag=23F5B364-22C9
To: <sip:18999@172.18.106.88>
Date: Tue, 22 Jun 2004 15:20:14 GMT
Call-ID: 7E86EC94-C39611D8-AF50DA50-D3EDBBC9@172.18.106.66
Supported: 100rel,timer
Min-SE: 1800
Cisco-Guid: 2092538615-3281392088-2941114960-3555572681
...
```

This information scrolls similarly to Cisco IOS Software **debug** output. One difference is that you only have to press the **Control-C** key combination to stop it.

Display the Trace Memory Buffer

The trace buffer in memory can be up to 10 MB in size. There are a few command-line parameters to be aware of:

```
vnt-nm-cue>show trace buffer ?
<cr>
containing    Only display events matching a regex pattern
long          Show long format
short         Show short format
tail          Wait for events and print them as they occur
|             Pipe output to another command
```

Most of the time, the only option that should be used is **show trace buffer long**. It is possible to add the paged keyword at the end so that the output is displayed one page at a time. The long format includes expanded text for many error and return codes, while the short format may only include hexadecimal codes. Although it is usually easier to grab everything with the capture function of a terminal program and then use the Find function in a text editor to search for specific things, if you need to search only for specific error conditions, the *containing* keyword is useful. It allows for a regular expression parameter to be used to filter the output to the display.

Note: It is not possible to search for specific calls or port numbers with only the *containing* keyword.

```
vnt-nm-cue>show trace buffer long paged
2029 06/24 17:48:40.479 ACCN SIPL 0 receive 1096 from 172.18.106.66:49255
2030 06/24 17:48:40.480 ACCN SIPL 0 not found header for Date
2030 06/24 17:48:40.480 ACCN SIPL 0 not found header for Supported
2030 06/24 17:48:40.480 ACCN SIPL 0 not found header for Min-SE
2030 06/24 17:48:40.480 ACCN SIPL 0 not found header for Cisco-Guid
2030 06/24 17:48:40.480 ACCN SIPL 0 not found header for Remote-Party-ID
2030 06/24 17:48:40.480 ACCN SIPL 0 not found header for Timestamp
2030 06/24 17:48:40.480 ACCN SIPL 0 not found header for Call-Info
```

```

2030 06/24 17:48:40.480 ACCN SIPL 0 not found header for Allow-Events
2030 06/24 17:48:40.481 ACCN SIPL 0 -----
INVITE sip:18900@172.18.106.88:5060 SIP/2.0
Via: SIP/2.0/UDP 172.18.106.66:5060;branch=z9hG4bK1128
From: "Caller1" <sip:201@172.18.106.66>;tag=2FA6AE58-20E5
To: <sip:18900@172.18.106.88>
Date: Thu, 24 Jun 2004 21:48:40 GMT
Call-ID: 16EEB21C-C55F11D8-BF05DA50-D3EDBBC9@172.18.106.66
Supported: 100rel,timer
Min-SE: 1800
Cisco-Guid: 384701940-3311342040-3204635216-3555572681
User-Agent: Cisco-SIPGateway/IOS-12.x
Allow: INVITE, OPTIONS, BYE, CANCEL, ACK, PRACK, COMET, REFER, SUBSCRIBE,
NOTIFY, INFO, UPDATE, REGISTER
CSeq: 101 INVITE
Max-Forwards: 6

```

Issue the **clear trace** command to clear the trace memory buffer. For most troubleshooting situations, you can set the traces that you want to collect, issue the **clear trace** command to clear the buffer, recreate the action you want to collect the traces for, and then capture the output of the **show trace buffer long** command. This method is the most effective way to collect traces for reproducible problems.

Stored Trace Log Files

In the NM and AIM (when enabled), traces are written to the `atrace.log` file. The **show logs** command displays all available log files:

```

vnt-nm-cue>show logs
dmesg
syslog.log
atrace.log
atrace.log.prev
klog.log
messages.log
messages.log.prev
root_javacore828.1087272313.txt
tomcat_javacore1094.1087272313.txt
workflow_javacore1096.1087272313.txt

```

The important files are `atrace.log` and `messages.log`. The `messages.log` file contains all the system messages (on the AIM, it contains only Fatal and Error messages). Particularly on the AIM, the `messages.log` file is sometimes the only log file that contains any historical information. The `_javacore` files are written when the system is restarted and are not typically as important as the other files (`dmesg`, `syslog.log`, `klog.log`). The `atrace.log.prev` and `messages.log.prev` files can also be important (if present). They are older versions of `atrace.log` and `messages.log`. For example, once an `atrace.log` file fills up, it is copied to `atrace.log.prev` and a new `atrace.log` file is started. The earlier `atrace.log.prev` is replaced and the information is lost.

Each file must be copied to the FTP server individually.

Because the `atrace.log` file can be large (up to 100 MB on the NM and 10 MB on the AIM), you typically want to copy it to an FTP server. The **copy log** command is for this purpose. In this example, the FTP username (`jdoue`) and password (`mypass`) are part of the URL:

```

vnt-nm-cue>copy log atrace.log url ftp://jdoue:mypass@172.18.106.10/cue/atrace.log
  % Total      % Received % Xferd  Average Speed           Time      Curr.
             Dload  Upload Total    Current  Left    Speed
100 1387k    0      0 100 1387k      0  4476k  0:00:00  0:00:00  0:00:00 6104k

```

Note: The `atrace.log` file is not a plain text file. It must be sent to Cisco Technical Support for diagnosis.

It is also possible to view the stored log files from the Cisco Unity Express module itself. However, this is not recommended for the `atrace.log` file because the file is not converted properly in plain text. Here is an example that uses the `messages.log` file:

```
cue-3660-41a#show log name messages.log
#!/bin/cat
19:46:08 logmgr: BEGIN FILE
19:46:08 logmgr: START
<45>Feb 26 19:46:08 localhost syslog-ng[134]: syslog-ng version 1.6.0rc1 starting
<197>Feb 26 19:46:08 localhost syslog-ng: INFO startup.sync syslog-ng arrived
phase online
<197>Feb 26 19:46:10 localhost err_handler: INFO Recovery Recovery startup :CUE
Recovery Script started.
<197>Feb 26 19:46:10 localhost err_handler: INFO Recovery Recovery LDAPVerify
Verifying LDAP integrity
...
```

Note: When you display a log file with the `show log name` command, press the **Control-C** key combination to interrupt the command output. Be aware that it takes a few seconds to return to a prompt after you do so.

Issue the `show trace store` command (or the `show trace store-prev` command, for the `atrace.log.prev` file) for the `atrace.log` file stored on a Cisco Unity Express.

```
vnt-nm-cue>show trace store ?
<cr>
containing Only display events matching a regex pattern
long Show long format
short Show short format
tail Wait for events and print them as they occur
| Pipe output to another command
vnt-nm-cue>show trace store long paged
236 02/26 14:46:24.029 TRAC TIMZ 0 UTC UTC 0
236 02/26 14:46:24.031 TRAC TIMZ 0 UTC UTC 0
885 06/04 13:14:40.811 WFSP MISC 0 WFSysdbLimits::WFSysdbLimits hwModuleType=NM
885 06/04 13:14:40.812 WFSP MISC 0 WFSysdbProp::getProp
885 06/04 13:14:40.812 WFSP MISC 0 keyName = limitsDir
str = /sw/apps/wf/ccnapps/limits
885 06/04 13:14:40.819 WFSP MISC 0 WFSysdbProp::getNodeXml
885 06/04 13:14:40.819 WFSP MISC 0 WFSysdbProp::getProp
885 06/04 13:14:40.820 WFSP MISC 0 keyName = limits
str =
885 06/04 13:14:40.822 WFSP MISC 0 WFSysdbProp::getNodeXml(str, str)
885 06/04 13:14:40.822 WFSP MISC 0 WFSysdbProp::getProp
885 06/04 13:14:40.822 WFSP MISC 0 keyName = app
str =
```

When you display the trace buffer in memory, the long format is important. Issue the `show trace store long` command. This information is from the very beginning of the `atrace.log` file, which can be up to 100 MB large on an NM or 10 MB on the AIM. It is in this situation that the `containing` keyword can occasionally be useful if specific events need to be searched.

Note: If the `atrace.log` file on the AIM has grown to the maximum size, it ceases to log traces to the log file. Issue these commands to restart the logging of traces:

```
VNT-AIM-CUE1>configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
VNT-AIM-CUE1(config)>log trace local disable
VNT-AIM-CUE1(config)>log trace local enable
```

Note: These commands remove the old `atrace.log` file and begin a new one.

Trace to an FTP Server

The best option to trace large amounts of data, especially on the AIM, is to log the information directly to the FTP server. Offline traces also have the least performance impact. This is accomplished from configuration mode.

Note: If the Cisco Unity Express system is an AIM, this command is necessary (the Network Module logs the Information level and greater by default):

```
vnt-nm-cue(config)>log console info
```

Note: This command has been brought down to a second line for spatial reasons.

```
vnt-nm-cue(config)>log trace server url
ftp://172.18.106.10/path/ username jdoe password mypass
```

Note: When you send logs to the FTP server you must also configure **log trace server enable**.

```
vnt-nm-cue(config)>log trace server enable
```

Note: The system generates a file in the designated path on the FTP server. It must have permission to create and modify files in the directory specified, which must exist. The parser extracts the username and password, which appear encrypted in the configuration file itself (**show running-config**).

Note: The trace file logged to the FTP server is not a plain text file. It must be sent to Cisco Technical Support for diagnosis.

JTAPI Traces

JTAPI traces are separate from any other trace facility in Cisco Unity Express. They are only applicable in Cisco CallManager environments. In order to view the current, enabled JTAPI traces, issue a **show ccn trace jtapi** command:

Note: By default, all JTAPI traces disabled.

```
VNT-AIM-CUE1>show ccn trace jtapi
Warning:                                0
Informational:                          0
Jtapi Debugging:                        0
Jtapi Implementation:                  0
CTI Debugging:                          0
CTI Implementation:                    0
Protocol Debugging:                     0
Misc Debugging:                         0
```

Issue these commands to enable all traces:

```
VNT-AIM-CUE1>ccn trace jtapi debug all
You will have to reload the system for your changes to take effect
VNT-AIM-CUE1>ccn trace jtapi informational all
You will have to reload the system for your changes to take effect
VNT-AIM-CUE1>ccn trace jtapi warning all
You will have to reload the system for your changes to take effect
VNT-AIM-CUE1>show ccn trace jtapi
Warning:                                1
Informational:                          1
Jtapi Debugging:                        1
Jtapi Implementation:                  1
```

```
CTI Debugging: 1
CTI Implementation: 1
Protocol Debugging: 1
Misc Debugging: 1
```

Reload the system. Issue the same **ccn trace** commands shown here to disable this at a later time. However, precede each command with the *no* keyword. For example, issue **no ccn trace jtapi debug all**. This is an important step to remember, especially on the AIM. Failure to perform this step affects potential performance, and it reduces the life of the compact Flash card on the AIM.

After the reload, the system begins to write the files CiscoJtapi1.log and CiscoJtapi2.log (when the first one is full).

You can view these logs on Cisco Unity Express if you issue the **show log name CiscoJtapi1.log** command. If you want to copy the log file(s) to an FTP server, and then view the information offline, issue the **copy log CiscoJtapi1.log url ftp://user:passwd@ftpservipaddr/** command.

Turn Off Traces

Traces can be turned off with the **no trace module entity activity** CLI command. When in doubt, you can issue **no trace all** to turn everything off.

You can also leave the trace settings themselves as they are and just disable the writing of the trace file with the **no log trace local enable** command from configuration mode. This is recommended for the AIM, because excessive writing lowers the lifespan of the internal Flash card. Here is an example:

```
vnt-nm-cue>configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
vnt-nm-cue(config)>no log trace local enable
vnt-nm-cue(config)>
```

Issue these commands to disable tracing to an FTP server:

```
vnt-nm-cue>configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
vnt-nm-cue(config)>log trace server disable
vnt-nm-cue(config)>
```

Re-enable Default Traces

When you troubleshoot specific problems, it often makes sense to only enable specific traces. Once completed, it is usually desirable to re-enable the default trace settings. Disable all traces with the **no trace all** command in order to do this. Next, enable the default traces by pasting these commands into the Cisco Unity Express CLI (not configuration mode):

```
trace ccn engine debug
trace ccn libldap debug
trace ccn subsystemappl debug
trace ccn managerappl debug
trace ccn managerchannel debug
trace ccn subsystemjtapi debug
trace ccn subsystemsip debug
trace ccn stacksip debug
trace ccn subsystemhttp debug
trace ccn vbrowsercore debug
trace ccn subsystemcmt debug
trace ccn libmedia debug
trace ccn managercontact debug
```

```
trace ccn stepcall debug
trace ccn stepmedia debug
trace config-ccn sip-subsystem debug
trace config-ccn jtapi-subsystem debug
trace config-ccn sip-trigger debug
trace config-ccn jtapi-trigger debug
trace config-ccn http-trigger debug
trace config-ccn group debug
trace config-ccn application debug
trace config-ccn script debug
trace config-ccn prompt debug
trace config-ccn miscellaneous debug
trace voicemail database query
trace voicemail database results
trace voicemail database transaction
trace voicemail database connection
trace voicemail database execute
trace voicemail mailbox login
trace voicemail mailbox logout
trace voicemail mailbox send
trace voicemail mailbox save
trace voicemail mailbox receive
trace voicemail mailbox delete
trace voicemail message create
trace voicemail message dec
trace voicemail message delete
trace voicemail message get
trace voicemail message inc
trace webinterface initwizard init
```

Related Information

- [Cisco Unity Express Release 1.1 Product Documentation](#)
 - [Voice Technology Support](#)
 - [Voice and IP Communications Product Support](#)
 - [Recommended Reading: Troubleshooting Cisco IP Telephony](#)
 - [Technical Support & Documentation – Cisco Systems](#)
-

[Contacts & Feedback](#) | [Help](#) | [Site Map](#)

© 2008 – 2009 Cisco Systems, Inc. All rights reserved. [Terms & Conditions](#) | [Privacy Statement](#) | [Cookie Policy](#) | [Trademarks of Cisco Systems, Inc.](#)

Updated: Jan 24, 2007

Document ID: 53207
