为统一边界要素(CUBE)和时分复用(TDM)网关配置调试集合

目录

简介

本文档介绍在Cisco IOS/IOS-XE语音路由器中收集语音调试的一些最佳实践。

先决条件

本文档中的信息都是基于特定实验室环境中的设备编写的。本文档中使用的所有设备最初均采用原始（默认）配置。如果您的网络处于活动状态，请确保您了解所有命令的潜在影响。

要求

- 集成多业务路由器(ISR)内部的Cisco IOS/IOS-XE基础知识。
- 特权访问，以便在ISR路由器中执行命令。
- 需要具有IP语音(VoIP)协议方面的经验。
- 对于VoIP跟踪，最低需要Cisco IOS-XE 17.4.1或17.3.2。

使用的组件

本文档中使用的组件包括：

- 思科ISR 3925
- 思科ISR 4451
- PuTTY

本文档中的信息都是基于特定实验室环境中的设备编写的。本文档中使用的所有设备最初均采用原始（默认）配置。如果您的网络处于活动状态，请确保您了解所有命令的潜在影响。

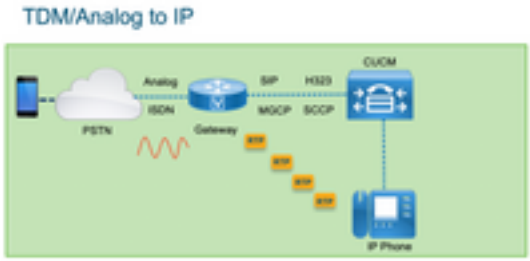
背景

这些平台中的Debug收集过程存在挑战，可能会影响设备的性能。当语音路由器中建立多个活动呼叫时，挑战和风险会增加。在某些情况下，如果未正确收集调试，则会导致CPU使用率过高，从而损害路由器的容量，甚至导致软件崩溃。本文档介绍思科统一边界要素(CUBE)和TDM/模拟网关之间的区别。

TDM语音网关与CUBE

TDM语音网关主要用于将内部电话系统与另一个专用交换机(PBX)或公共交换电话网络(PSTN)互连。 TDM网关中使用的连接类型是T1/E1控制器（ISDN或CAS）和模拟电路，例如FXS和FXO端口。数字信号处理器(DSP)将音频从原始形式转换为RTP数据包。类似地，DSP处理完RTP数据包并在特定电路上发送音频后，RTP数据包会转换为原始音频。这些网关可以在VoIP端与H323、MGCP或SCCP进行互操作，而在TDM端，其ISDN PRI电路或模拟电路作为到PSTN或终端的最常见连接。

如图所示，TDM网关在您的内部VoIP基础设施和模拟或ISDN服务提供商之间提供了一个桥梁。



随着VoIP的引入，客户开始迅速将其传统系统转变为现代VoIP基础设施。在运营商端也发生了同样的情况，他们现在使用连接将本地电话服务与运营商VoIP基础设施互联，并扩展其功能以提供更好的服务。目前最常用的VoIP协议是会话发起协议(SIP)，目前被全球的客户和互联网电话服务提供商(ITSP)广泛使用。

引入CUBE是为了通过将SIP作为主要VoIP协议的ITSP将这些内部VoIP系统与外部世界进行内部连接。CUBE只是一个IP-IP网关，不再需要任何TDM类型的连接，如T1/E1控制器或模拟端口。CUBE与TDM网关在同一平台上运行。

最常用的VoIP协议是SIP（用于呼叫建立和呼叫中断）和RTP（用于媒体传输）。在CUBE中，不需要使用DSP，除非需要转码器。RTP流量从ITSP端到端传输到终端，CUBE充当中间人，地址隐藏是其提供的众多功能之一。

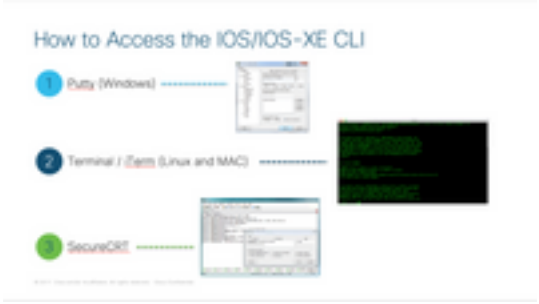
如图所示，CUBE将您的内部VoIP基础设施与SIP ITSP区分开来：

Cisco IOS/IOS-XE语音调试集合

语音功能在不同平台列表上运行，例如ISR、ASR、CAT8Ks等，但是它们使用的是通用软件，即Cisco IOS或Cisco IOS-XE（Cisco IOS和Cisco IOS-XE之间的区别本文未涉及）。 让我们从有关如何访问Cisco IOS路由器的基本信息开始。

如何通过命令行界面(CLI)访问Cisco IOS/IOS-XE路由器

与任何其他基于CLI的设备一样，路由器需要终端监控器才能通过Secure Shell(SSH)或Telnet运行命令。SSH是目前最常用的访问设备协议，因为它提供了到设备的安全加密连接。用于访问路由器CLI的一些常用终端监控器包括：

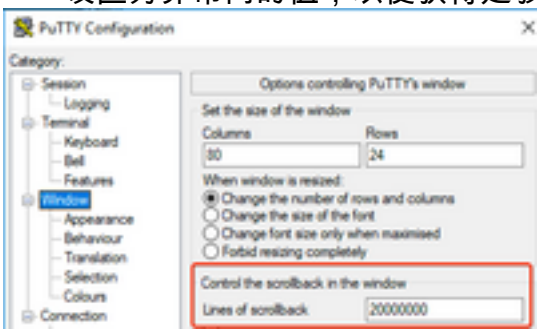


如何设置终端监控器以收集show命令或调试

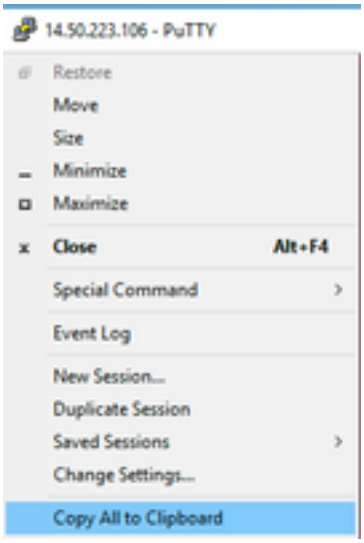
收集CLI的输出有多种方法。建议从路由器的CLI将信息导出到单独的文件中。这样可以更轻松地与外部各方共享信息。

收集设备输出的几种方法如下：

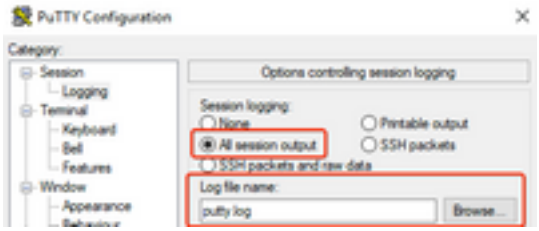
- 转储终端中的所有输出，为此，您需要确保有足够行的回滚，否则回滚会错过输出的第一部分，数据可能不完整。要增加Putty中的回滚行，请导航到Putty配置>窗口>回滚行。通常，该值设置为非常高的值，以便获得足够的回滚输出：



然后，您可以使用**Copy All to Clipboard**选项从终端监控器收集信息，并将输出粘贴到文本文件中：



- 另一个选项是将整个会话输出记录到.txt文件中。使用此选项，所有输入的命令和收集的输出将立即记录到文本文件中。这是一种在会话中记录所有输出的常见做法。要将所有会话输出记录到Putty中的文件，请导航到**Putty Configuration > Session > Logging**，然后选择All Session Output，如下所示：



注意：如果未指定其他名称，则使用默认日志文件名称，单击“浏览”按钮可准确了解文件的保存位置，以便稍后查找。另请确保不要覆盖同一文件路径中的另一个putty.log文件。

从CLI收集基本show命令输出

在进行任何调试收集之前，需要使用show命令从路由器收集基本信息。Show命令收集速度很快，而且大多数情况下不会对路由器的性能产生任何影响。仅使用show命令输出即可立即开始隔离问题。

连接到路由器后，终端长度可以设置为0。这样可以加快收集速度，以便一次显示所有输出，并且避免使用空格键。用于收集路由器详细信息的命令是“show tech”，您也可以收集**show tech voice**命令，该命令可显示特定于路由器中启用的语音功能的数据：

```
Router# terminal length 0
Router# show tech
!or
Router# show tech voice
Router# terminal default length !This cmd restores the terminal length to default
```

从CLI收集调试输出

Cisco IOS/IOS-XE中的调试输出收集有时可能是一个挑战，因为存在路由器崩溃的风险。下面几节将介绍一些最佳实践，以避免出现任何问题。

内存检查

启用任何调试之前，需要确保有足够的内存来将输出存储在缓冲区中。

运行命令**show process memory**找出可以分配多少内存以记录缓冲区中的所有输出：

提示：使用命令**terminal length default**或**terminal length <num_lines>** 返回终端中显示的有限行数。

```
Router# show process memory
Processor Pool Total: 8122836952 Used: 456568400 Free: 7666268552
lsmpi_io Pool Total: 6295128 Used: 6294296 Free: 832
```

在本例中，有7666268552字节(7.6GB)可供路由器使用。此内存由路由器在所有系统进程之间共享，这意味着您不能使用整个可用内存将输出记录到缓冲区中，但您可以根据需要使用大量系统内存。

大多数情况至少需要10MB才能收集足够的调试输出，然后输出才会丢失或被覆盖。在极少数情况下，需要收集的数据量较大；在那些特定情况下，可以在缓冲区中得到50MB到100MB的输出，也可以使用更高的值（只要有可用内存）。

如果可用内存不足，则可能存在内存泄漏问题。如果是这种情况，请架构TAC团队修正导致内存不足的原因。

中央处理器(CPU)检查

CPU受系统中处于活动状态的进程、功能和呼叫数量的影响。系统中激活的功能或呼叫越多，CPU就越繁忙。

一个好的基准是确保路由器的CPU使用率不超过30%，这意味着您可以安全地启用从基本到高级的调试（使用高级调试时始终关注CPU）。 如果路由器CPU使用率约为50%，则可以运行基本调试并仔细监控CPU。如果CPU使用率超过80%，请立即停止调试（如本文稍后所示），并联系TAC寻求帮助。

使用**show process cpu sorted | exclude 0.00命**令检查最后5秒、60秒和5分钟的CPU值以及排名靠前的进程。

```
Router# show processes cpu sorted | exclude 0.00
CPU utilization for five seconds: 1%/0%; one minute: 0%; five minutes: 0%
PID Runtime(ms) Invoked uSecs 5Sec 1Min 5Min TTY Process
211 4852758 228862580 21 0.15% 0.06% 0.07% 0 IPAM Manager
84 3410372 32046994 106 0.07% 0.04% 0.05% 0 IOSD ipc task
202 3856334 114790390 33 0.07% 0.05% 0.05% 0 VRRS Main thread
```

在输出中，路由器没有太多活动，CPU资源不足，可以安全地启用调试。

> **警告**：如果CPU的利用率达到50%或更高，并且顶层进程是语音进程，则要特别注意处于活动状态的顶级CPU进程，否则只能启用基本调试。使用命令持续监控CPU，以确保路由器的整体性能不受影响。

当前活动呼叫检查

每台路由器具有不同的容量阈值。检查路由器中处于活动状态的呼叫数量以确保其未接近最大容量非常重要。Cisco Unified Border Element Version 12 Data Sheet提供了有关每个平台容量的信息以供参考。

使用**show call active total-calls**命令可了解系统中处于活动状态的呼叫数：

```
Router# show call active total-calls
Total Number of Active Calls : 0
```

使用**show call active voice summary**命令可获取有关处于活动状态的特定呼叫类型的更详细信息：

```
Router# show call active voice summary
Telephony call-legs: 0
SIP call-legs: 0
H323 call-legs: 0
Call agent controlled call-legs: 0
SCCP call-legs: 0
STCAPP call-legs: 0
Multicast call-legs: 0
Total call-legs: 0
```

一些常用值包括：

- **电话呼叫段**:TDM网关呼叫，包括模拟呼叫和PRI/ISDN呼叫。
- **SIP呼叫段**:SIP呼叫总数。如果这是一个CUBE路由器，则显示每个呼叫2个呼叫段。将此处显示的总呼叫数除以2可得到准确的数字。
- **H323呼叫段**:H323呼叫总数。
- **SCCP呼叫段**:路由器中使用的CUCM受控媒体资源，如转码器和MTP。

日志记录缓冲区设置

要将路由器配置为在缓冲区中存储调试输出，需要进入configure terminal模式以手动调整CLI中的设置。此配置对路由器没有影响，但如前文部分所示，如果需要回滚配置，则需要在路由器上执行**show tech**或**show running-config**命令。

接下来可以看到配置示例，这是TAC工程师使用的通用基线。此示例分配了10MB的缓冲内存，但可以根据需要增加：

```
# configure terminal
service timestamps debug datetime msec localtime show-timezone year
service timestamps log datetime msec localtime show-timezone year
service sequence-numbers
logging buffered 10000000
no logging console
no logging monitor
logging queue-limit 10000
logging rate-limit 10000
voice iec syslog
```

命令可完成以下任务：

- **服务时间戳调试或日志**:确保本地路由器时间以毫秒的精度写入记录的每条消息。这对于根据时间查找呼叫很有用。当同一毫秒内出现两行时，毫秒时间戳允许您将调试行分组为逻辑相关事件。
- **service sequence-numbers**:在行中写入调试的序列号。当日志转发到系统日志服务器时，这很有用（基本上是必需的）。这对于确定网络中是否丢弃了系统日志服务器的任何调试消息非常有用。序列号是调试中的第一个项目，位于时间戳和实际日志消息之前。请注意，这不同于系统日志服务器可以在其文件中本地写入的时间戳/序列号。
- **日志缓冲器**:告知路由器将调试发送到其本地缓冲内存。缓冲区大小以字节为单位。在配置中，缓冲区大小设置为10MB。
- **no logging console**和**no logging monitor**:控制台或终端监控器中不打印日志消息。如果未配置这些命令，可能会损害路由器性能和调试输出准确性。
- **语音iec系统日志**：启用语音内部错误代码消息以确定断开原因。

配置系统日志设置

有时，问题可能是随机的，并且需要一种持续收集调试直到事件发生的方法。在缓冲区中存储调试时，它会持续收集这些调试。请注意，限制为可分配的内存量，一旦达到该内存量，缓冲区就会转圈并丢弃最早的消息，从而导致隔离问题所需的有价值信息不完整。

使用Syslog，路由器可以将所有调试消息发送到外部服务器，Syslog服务器软件将外部服务器存储在文本文件中。尽管这是收集debug输出的好方法，但它不是日志收集的首选方法。由于服务器拥塞，系统日志服务器倾向于跳过或丢弃已接收输出的行，因为调试输出可能使服务器不堪重负，或者数据包可能由于网络条件而被丢弃。但在某些情况下，系统日志是解决问题的唯一途径。

尽可能使用可靠的传输方法（如TCP）以避免信息丢失，并建议将系统日志服务器连接到路由器所连接的交换机或尽可能靠近路由器。它仍然不能保证所有数据都存储在文件中，但会降低数据丢失的几率。

默认情况下，系统日志服务器使用UDP作为端口514上的传输协议。

```
#configure terminal
service timestamps debug datetime msec localtime show-timezone year
service timestamps log datetime msec localtime show-timezone year
service sequence-numbers

!Optional in case you still want to store debug output in the buffer.
logging buffered 10000000

no logging console
no logging monitor

logging trap debugging

!Replace the 192.168.1.2 with the actual Syslog Server IP Address
logging host 192.168.1.2 transport [tcp|udp] port
```

配置命令后，路由器立即将消息转发到Syslog服务器IP地址。

调试收集

启用调试后，必须先清除缓冲区，然后才能重现问题。这样做是为了确保输出尽可能干净，并避免分析时不需要的任何额外数据。运行clear log命令，这将确保清除缓冲区。如果路由器中有其他呼叫处于活动状态并且启用了调试，输出将立即显示在缓冲区中。

```
Router# clear log
Clear logging buffer [confirm]
Router#
```

重现问题后，立即禁用调试，停止缓冲区中的更多输出。然后收集日志。您可以使用以下命令转储终端中的所有输出：

```
Router# undebug all
Router# terminal length 0
Router# show log
```

有时PuTTY会关闭，因为它无法同时处理所有输出，这是正常的，并不意味着发生了故障，如果发生故障，请重新打开会话并正常继续。在日志记录缓冲区过大或终端监控程序由于需要打印的数据量而崩溃的情况下，请使用**show log命令直接将缓冲区输出复制到外部设备 |重定向**：

```
Router# show log | redirect ftp://username:password@192.168.1.2/debugs.txt
```

该命令将整个缓冲区输出复制到IP地址为192.168.1.2且文件名为debug.txt的ftp中。必须始终指定文件名。可用于导出该数据的其他目标包括：

```
Router# sh log | redirect ?
```

```
bootflash: Uniform Resource Locator
flash: Uniform Resource Locator
ftp: Uniform Resource Locator
harddisk: Uniform Resource Locator
http: Uniform Resource Locator
https: Uniform Resource Locator
nvram: Uniform Resource Locator
tftp: Uniform Resource Locator
```

可以在语音路由器中启用哪些调试？

每个呼叫流程和功能类型(TDM、CUBE或SCCP（媒体资源）)不同，您可以启用特定调试。必须同时启用所有需要的调试。如果一次仅捕获一个调试，则不会产生任何效果，并且会在分析数据时造成更多混乱。

在CLI执行提示级别Router#内启用调试，该级别要求您具有特权执行模式权限。

有基本调试和高级调试。基本调试用于收集SIP、H323或MGCP中的信令信息，其中显示了路由器与其对等设备的会话。

高级调试非常详细，通常用于在基本调试无法显示内部堆栈错误时收集更多信息。这些调试通常占用大量的CPU。

> **提示：**启用调试后，请记住运行clear logging命令。此命令可确保清空缓冲区，以便更清楚地捕获调试。

内部呼叫控制API(CCAPI)调试

在每个Cisco IOS/IOS-XE路由器内都有一个呼叫控制API，负责不同VoIP应用或协议与数据平面组件（如RTP、DSP、语音卡等）之间的通信。为了捕获来自此层的数据，可以使用一个特定的调试：

```
debug voip ccapi inout
```

此调试还有其他选项，但debug voip ccapi inout包括所有基本拨号方案和呼叫建立信息，这些信息通常足以了解此层的状态。

> **提示：debug voip ccapi inout通常对路由器CPU的影响最**小，建议与任何信令调试一起启用，以便提供包含呼叫及其不同状态的信息的完整日志集。

SIP呼叫流

这些调试最常用于SIP呼叫流，并且可以在CUBE和TDM网关内启用，在路由器和CUCM或任何其他SIP服务器/代理之间使用SIP分支。

基本SIP调试

```
debug ccsip messages
debug ccsip error
debug ccsip non-call !Optional, applies for SIP OPTIONS and SIP REGISTER Messages.
```

高级SIP调试

```
debug ccsip all
debug ccsip verbose
debug voice ccapi inout
```

数字(PRI、BRI)呼叫流

以下调试适用于主速率接口(PRI)T1/E1或基本速率接口(BRI):

基本数字调试

```
debug isdn q931
```

高级数字调试

```
debug isdn q921
```

模拟呼叫流

当涉及模拟电路(如外部交换用户(FXS)或外部交换局(FXO)端口时，使用以下调试：

```
debug vpm signal
debug voip vtsp all
```

MGCP呼叫流

当MGCP用作语音网关和CUCM之间的语音协议时，使用这些调试。

基本调试

```
debug mgcp packets
debug mgcp errors
```

CCM-Manager调试

debugs ccm-manager用于跟踪CUCM和语音网关之间的配置下载、MoH和PRI/BRI回传消息。这些调试根据需要使用，并且取决于故障场景。

```
debug ccm-manager backhaul !For PRI and BRI Deployments
debug ccm-manager errors
debug ccm-manager events
debug ccm-manager config-download !Troubleshoot Configuration download issues from CUCM TFTP
debug ccm-mananger music-on-hold !Troubleshoot internal MoH Process
```

高级MGCP调试

```
debug mgcp all
```

H323呼叫流

虽然H323并未得到广泛使用，但仍有一些配置了H323的部署：

基本H323调试

```
debug h225 asn1
debug h245 asn1
debug h225 events
debug h245 events
```

高级H323调试

```
debug cch323 h225
debug cch323 h245
debug cch323 all
```

SCCP媒体资源

这些调试用于解决与媒体终端点(MTP)或注册到Cisco Unified Communications Manager(CUCM)服务器的转码器相关的瘦呼叫控制协议(SCCP)媒体资源问题：

基本SCCP调试

```
debug sccp messages
debug sccp events
debug sccp errors
```

高级SCCP调试

```
debug sccp all
```

VoIP跟踪

随着Cisco IOS-XE 17.4.1和17.3.2的引入，在思科统一边界元素(CUBE)内有一个捕获语音日志的新选项。 这一新功能称为VoIP跟踪。这是一个新的可维护性框架，用于记录SIP信令和事件，无需启用任何调试。

VoIP跟踪默认启用，可根据需要随时禁用。VoIP跟踪仅捕获SIP呼叫的特定信息：

- SIP中继到中继呼叫的SIP消息
- 从SIP层到CUBE中其他层的事件和API调用
- SIP错误
- 呼叫控制（CUBE处理的统一通信呼叫流）
- 有限状态机(FSM)状态和事件
- 拨号对等体匹配
- 分配的RTP端口
- IEC错误与SIP信令关联

限制

- VoIP跟踪不会记录与对话外SIP消息相关的信息： 注册选项订阅/通知信息

- 支持HA中的VoIP跟踪，但以下警告适用： 备用路由器默认启用VoIP跟踪。只有备用进程的适用跟踪才会显示，直到它变为活动状态一旦待机处于活动状态，它将不包含**来自检查点调用的完整跟踪**和仅包含新呼叫**show voip trace <key>**仍然适用于备用路由器，并显示呼叫的覆盖缓冲区和媒体流数据

如何启用VoIP跟踪

如前所述，此功能默认启用。启用此功能的命令为：

```
Router# configuration terminal
Router(config)# voice service voip
Router(conf-voi-serv)# trace
Router(conf-serv-trace)#
```

如何禁用VoIP跟踪

要禁用此功能，命令如下：

```
Router(conf-serv-trace)# no trace
!or
Router(conf-serv-trace)# shutdown
```

警告：禁用VoIP Trace后，将清除所有内存并丢失信息。

跟踪配置模式中可用的命令包括：

```
Router(conf-serv-trace)# ?
default       Set a command to its defaults
 exit          Exit from voice service voip trace mode
 memory-limit  Set limit based on memory used
 no            Negate a command or set its defaults
 shutdown      Shut Voip Trace debugging
```

配置内存限制

内存限制决定VoIP跟踪用于存储数据的内存量。默认情况下，平台中可用内存的10%，但最大可更改为1GB，最小可更改为10MB。动态分配的内存，这意味着此功能仅根据需要使用内存，并且取决于呼叫量。达到可用的最大内存后，它会绕圈并删除旧条目。

当内存限制被修改为大于10%的可用内存时，命令行界面中会显示一条消息：

```
Router(conf-serv-trace)# memory-limit 1000
Warning: Setting memory limit more than 10% of available platform memory (166 MB) will affect
system performance.
```

要设置默认的10%内存使用率，可使用命令**memory-limit platform**:

```
Router(conf-serv-trace)# memory-limit platform
 Reducing the memory-limit clears all VoIP Trace statistics and data.
 If you wish to copy this data first, enter 'no' to cancel,
 otherwise enter 'yes' to proceed. Continue? [no]:
```

警告：当内存限制降低时，所有VoIP跟踪数据都将丢失。在内存减少之前，必须收集数据的备份。

如何显示VoIP跟踪数据

要显示VoIP Trace中的数据，需要使用特定的show命令。数据可以在同一终端会话中显示，也可以通过Syslog发送到现成的系统日志服务器。

注意：从收到呼叫的BYE时起，32秒后转储跟踪。

注意：SIP信令按支路显示，不会像常规调试那样组合。**debug ccsip messages**等常规调试以事件发生的确切顺序显示呼叫的SIP信令。在VoIP跟踪中，每个支路是独立的。要确定正确的顺序，需要使用时间戳。

可用于显示数据的命令包括：

```
Router# show voip trace ?
 all            Display all VoIP Traces
 call-id        Filter traces based on Internal Call Id
 correlator     Filter traces based on FPI Correlator
 cover-buffers  Display the summary of all cover buffers
 session-id     Filter traces based on SIP Session ID
 sip-call-id    Filter traces based on SIP Call Id
 statistics     Display statistics for VoIP Trace
```

show voip trace all

此命令显示缓冲区中可用的所有VoIP跟踪数据。使用此命令会影响路由器的性能。输入命令后，会显示警告消息，警告风险并确认继续：

```
Router# show voip trace all
 Displaying 11858 cover buffers
 This may severely impact system performance.
 Continue? [yes/no] no
```

show voip trace cover-buffers

此命令显示VoIP跟踪下报告的所有呼叫的呼叫详细信息的概述。每个呼叫段都创建了一个封面缓冲区，其中包含记录的呼叫摘要。

```
Router# show voip trace cover-buffers
------------------ Cover Buffer ---------------
Search-key = 8845:3002:659
Timestamp = *Sep 30 01:17:33.615
Buffer-Id = 1
CallID = 659
Peer-CallID = 661
Correlator = 4
Called-Number = 3002
Calling-Number = 8845
SIP CallID = 20857880-1ec12085-13b930-411b300a@10.48.27.65
SIP Session ID = 2b1289c400105000a0002c3ecf872659
```

```
GUID = 208578800000
-------------------------------------------

------------------ Cover Buffer ---------------
Search-key = 8845:3002:661
Timestamp = *Sep 30 01:17:33.634
Buffer-Id = 2
CallID = 661
Peer-CallID = 659
Correlator = 4
Called-Number = 3002
Calling-Number = 8845
SIP CallID = 8D6DEC28-1F111EB-829FD797-1B22F6DB@10.48.55.11
SIP Session ID = 0927767800105000a0005006ab805584
GUID = 208578800000
-------------------------------------------
```

有关每个字段的详细信息，请参阅下表：

字段	描述
搜索键	包含呼叫、被叫号码和呼叫ID的组合
时间戳	覆盖缓冲区的创建时间
缓冲区ID	覆盖缓冲区的缓冲区ID
呼叫ID	到覆盖缓冲区的各个呼叫支路的呼叫ID
对等呼叫ID	对等分支的呼叫ID
相关器	呼叫的FPI相关器
被叫号码	覆盖缓冲区的各个呼叫段的被叫号码
主叫号码	覆盖缓冲区的各个呼叫段的呼叫号码
Sip呼叫ID	覆盖缓冲区的各个呼叫段的SIP call-id
Sip会话ID	覆盖缓冲区的各个呼叫段的SIP会话ID
GUID	封面缓冲区的相应调用的GUID
锚脚	如果各自的呼叫支路是呼叫分流流或媒体代理部署中的锚支路，则锚支路设置为yes
分叉腿	如果各自的呼叫分支是呼叫分支流或媒体代理部署中的锚点，则Forked Leg设置为yes
关联的Call ID	关联的分支腿的呼叫ID

要过滤覆盖缓冲区，可以使用include和section命令：

```
Router# show voip trace cover-buffers | include Search-key | 8845 | 3002
Search-key = 8845:3002:661
!or
Router# show voip trace cover-buffers | section Search-key | 8845 | 3002
Search-key = 8845:3002:661
```

show voip trace call-id

与前面的命令结合使用，show voip trace call-id可用于查找呼叫。确定call-id后，此命令可用于显示有关特定呼叫段的所有信息：

```
Router# show voip trace cover-buffers | include Search-key | 8845 | 3002
Search-key = 8845:3002:661
Router# show voip trace call-id 661
```

show voip trace statistics

此show命令显示有关状态、内存消耗、错误或故障调用、成功调用、最新和最早条目的时间戳等信息的详细输出。

```
Router# show voip trace statistics
VoIP Trace Statistics
Tracing status             : ENABLED at *Sep 12 06:44:02.349
Memory limit configured    : 803209216 bytes
Memory consumed            : 254550928 bytes (31%)
Total call legs dumped     : 2
Oldest trace dumped        : *Sep 12 07:29:21.077 Search-key: 9898:30000:64
Latest trace dumped        : *Sep 12 07:29:21.010 Search-key: 9898:30000:63
Total call legs captured   : 11858
Total call legs available  : 11858
Oldest trace available     : *Sep 12 06:57:23.923, Search-key: 5250001:4720001:11
Latest trace available     : *Sep 13 05:08:25.353, Search-key: 19074502232:30000:13177
Total traces missed        : 0
```

有关每个字段的详细信息，请参阅下表：

字段	描述
跟踪状态	显示跟踪状态，包括启用VoIP跟踪的时间和日期。
配置的内存限制	显示配置的内存限制。这是处理器池内存大小的10%
消耗的内存	显示VoIP跟踪动态消耗的内存量
转储的呼叫段总数	显示转储到日志记录缓冲区的失败呼叫段数。转储呼叫是指与IEC错误相关的呼叫段
已转储最早的跟踪	显示自启用VoIP跟踪以来最旧的失败呼叫的时间戳和搜索密钥
最新跟踪已转储	显示自启用VoIP跟踪以来的最新失败呼叫的时间戳和搜索密钥
捕获的呼叫段总数	显示启用VoIP跟踪后捕获的总支路
可用的呼叫段总数	显示历史记录中可用的呼叫段总数。这可以与捕获的总呼叫段数相同或不同，具体取决存限制。
可用的最早跟踪	显示内存中可用的最旧覆盖缓冲区的时间戳和搜索键
提供最新跟踪	显示内存中可用的最新封面缓冲区的时间戳和搜索键
丢失的跟踪总数	显示由于内存限制而错过的呼叫段数。

其他show命令

字段	使用率	
show voip trace correlator <correlator>	show voip trace correlator 4	过滤并显示从覆盖缓冲区开始的特定呼叫ID的VOIP跟踪
show voip trace session-id <session-id>	show voip trace session-id 87003120822b5dbd8fd80f62d8e57c48	根据SIP会话ID过滤并显示呼叫的VOIP
show voip trace sip-call-id <call-id>	show voip trace sip-call-id 01e60dfa9d8442848336d79e3155a8a1	根据SIP Call-ID过滤和显示VOIP跟踪

关于此翻译
思科采用人工翻译与机器翻译相结合的方式将此文档翻译成不同语言，希望全球的用户都能通过各自的语言得到支持性的内容。

请注意：即使是最好的机器翻译，其准确度也不及专业翻译人员的水平。

Cisco Systems, Inc. 对于翻译的准确性不承担任何责任，并建议您总是参考英文原始文档（已提供链接）。