Cisco的MediaSense常见问题

Contents

Introduction

本文描述Cisco媒介感觉服务器的常见问题。

Prerequisites

Requirements

Cisco 建议您了解以下主题：

- 思科 MediaSense
- Cisco Unified Communications管理器(CUCM)

Components Used

本文档中的信息基于以下软件和硬件版本：

Cisco MediaSense 10.5

1.如何关联不同的呼叫情形的参考呼叫ID，在统一的通信管理器电话分叉下？

在Cisco MediaSense中，每次呼叫的元数据只提供xRefCi (参考呼叫ID)和设备ref (扩展名)分叉的设备和远端的设备(可以是会议网桥或其他电话)。

xRefCi参数是统一的通信一特定的媒体流的管理器的标识。他们总是不对应1:1与记录的跟踪。

1.1. MediaSense搜索和作用称为关联功能

MediaSense生成在暂挂/恢复或者转移的情况下被记录，使困难识别在呼叫的所有记录会话的呼叫的多个会话。为了能关联在单个呼叫的这些记录会话， MediaSense介绍作为呼叫关联被叫做的一个新功能。通过此功能，组合与普通的xRefci值的所有严格相关的呼叫。MediaSense 10.5支持构件在网桥记录的呼叫关联功能。

1.2.代理程序暂挂/恢复方案

1. 代理程序A (extn 1000)和呼叫人C (extn 2000)在通话状态互相呼叫和。
2. 代理程序A呼叫暂挂。
3. 代理程序A恢复呼叫。

有此方案的两次记录的会话：

- 与sessionId = S1的会话与这两跟踪，为了在代理程序前的时间/segment呼叫暂挂：
  trackNumber = 0与参与者A (deviceRef = 1000， xRefCi = aaaa)
  trackNumber = 1与参与者B (deviceRef = 2000年， xRefCi = cccc)
- 与sessionId = S2的会话与这两跟踪，为了时间/分段，在代理程序恢复呼叫后。
  trackNumber = 0与参与者A (deviceRef = 1000， xRefCi = aaaa)
  trackNumber = 1与参与者B (deviceRef = 2000年， xRefCi = cccc)

当代理程序呼叫暂挂时， MediaSense不记录呼叫的分段。

1.3.用户暂挂/恢复方案

1. 代理程序A (extn 1000)和呼叫人C (extn 2000)在通话状态alled和。
2. 用户C呼叫暂挂。

3. 用户C恢复呼叫。

整个呼叫在此方案的一次会话上被记录：

- 与sessionId = S1的会话与这两跟踪：
  trackNumber = 0与参与者A (deviceRef = 1000， xRefCi = aaaa)
  trackNumber = 1与参与者B (deviceRef = 2000年， xRefCi = cccc)

在此方案中，当用户呼叫暂挂时， MediaSense也记录呼叫的分段。

1.4.代理程序转移到另一个代理程序方案

1. 呼叫人C (extn 2000)呼叫代理程序A (extn 1000)
2. 代理程序A (extn 1000)与代理程序B (3000)咨询
3. 代理程序A (extn 1000)完成转移。
4. 代理程序B (extn 3000)挂断。

使用统一的通信管理器9.x和前，这是结果：

1.Caller C (extn 2000)呼叫代理程序A (extn 1000)

S1被启动的会话，跟踪0是A (extn 1000)，跟踪1是C (extn 2000)。

对另一个代理程序B (extn 3000)的2.Agent A (extn 1000)转接呼叫。A和B设备为分叉设置。

3.Caller C (extn 2000)听到Music on Hold (MoH)。
4.Agent A (extn 1000)与B (extn 3000)谈。

- S1被结束的会话
- S2被启动的会话，跟踪0是A (extn 1000)，跟踪1是B (extn 3000)
- S3被启动的会话，跟踪0是B (extn 3000)，跟踪1是A (extn 1000)

考虑事项：

- 会话S1末端由于代理程序A电话暂挂呼叫人C。
- 因为两个电话为分叉，被配置会话S2和S3存在。
- 参与者和xRefCi两个参加者的在S2和S3是相同的，但是反向位置从彼此。
- 因为咨询认为一次独立呼叫， S1的xRefCi值在S2或S3没有被反射。

5.Agent A (extn 1000)完成转移。

6.C (extn 2000)谈与B (extn 3000)。

7.A (extn 1000)断开了。

S2被结束的会话。

会话S3更新的和跟踪0是B (extn 3000)，并且跟踪1是C (extn 2000)。

考虑事项：

- 一次远端的转移触发现有的录制会话的更新。
- 远端的参与者更改到那S1。
- S3新的远端的xRefCi匹配S1远端的xRefCi。

- 代理程序B (extn 3000)挂断。
- C (extn 2000)和B (extn 3000)是断开的。
- S3被结束的会话

  Note:一次远端的转移导致现有的会话的更新。分叉的电话在跟踪0在跟踪1更改保持唯一的参与者，但是参加者对新的当事人。

在统一的通信管理器10.0的情况下及以后，这是结果：

1. 呼叫人C (extn 2000)呼叫代理程序A (extn 1000)。

C (extn 2000)与代理程序A (extn 1000)谈。会话S1 -开始-跟踪0是A (extn 1000)，跟踪1是C (2000)

2. 代理程序A (extn 1000)参见代理程序B (extn 3000)。

3. C (extn 2000)听到MoH。A (extn 1000)与B (extn 3000)谈。

- 会话S1 -结束
- 会话S2 -开始-跟踪0是A (extn 1000)，跟踪1是B (extn 3000)
- 会话S3 -开始-跟踪0是B (extn 3000)，跟踪1是A (extn 1000)

考虑事项：

- 因为代理程序A电话暂挂呼叫人C，会话S1结束。
- 因为两个电话为分叉，被配置会话S2和S3两个存在。
- 参与者以及xRefCi两个参加者的在S2和S3相同，但是在被倒转的位置从彼此。
- 因为咨询认为一次独立呼叫， S1 xRefCi值在S2或S3没有被反射

4. 代理程序A (extn 1000)完成转移。

5. C (extn 2000)与B (extn 3000)谈。
6.A (extn 1000)断开了。

- 会话S3 -结束
- 会话S2 -结束
- 会话S4 -开始-跟踪0是B (extn 3000)，跟踪1是C (extn 2000)

考虑事项：

- 一次远端的转移触发一录制会话结尾和另一录制会话开始。
- 虽然个新会话开始，其xRefCi值将匹配上次会话。
- S4远端的xRefCi匹配S1远端的xRefCi，并且S4近端的xRefCi匹配S3近端的xRefCi。

7. 代理程序B (extn 3000)挂断。

8. C (extn 2000)和B (extn 3000)断开了。

- 会话S4 -结束

  Note:一次远端的转移导致一录制会话结尾和另一录制会话开始。

1.5. 与另一个代理程序方案的代理程序会议

1. 呼叫人C (extn 2000)呼叫代理程序A (extn 1000)。
2. 代理程序A (extn 1000)参见代理程序B (extn 3000)。
3. 代理程序A (extn 1000)完成会议。
4. 代理程序A (extn 1000)从会议丢弃。
5. 代理程序B (extn 3000)挂断。

在统一的通信管理器9.x的情况下和前，这是结果：

1. 呼叫人C (extn 2000)呼叫代理程序A (extn 1000)。

2. C (extn 2000)与A (extn 1000)谈。
S1被启动的会话-跟踪0是A (extn 1000)，跟踪1是C (extn 2000)。

3. 代理程序A (extn 1000)参见代理程序B (extn 3000)。

4. C (extn 2000)听到MoH A (extn 1000)谈话对B (extn 3000)

- S1被结束的会话
- S2被启动的会话-跟踪0是A (extn 1000)，跟踪1是B (extn 3000)
- S3被启动的会话-跟踪0是B (extn 3000)，跟踪1是A (extn 1052

考虑事项：

- 因为代理程序A电话暂挂呼叫人C，会话S1结束。
- 因为两个电话为分叉，被配置会话S2和S3存在。
- 参与者以及xRefCi两个参加者的在S2和S3相同，但是在被倒转的位置从彼此。
- 因为咨询认为一次独立呼叫， S1 xRefCi值在S2或S3没有被反射。

5. 代理程序A (extn 1000)完成会议。

6. C (extn 2000)谈与A (extn 1000)和B (extn 3000)。

- S2被结束的会话
- S3更新的会话-跟踪0是B (ext 3000)，跟踪1是会议网桥
- S4被启动的会话-跟踪0是A (extn 1000)，跟踪1是会议网桥

考虑事项：
一次远端的转移触发现有的录制会话的更新。
会议的完成是被实施的：
在咨询期间：

- 咨询的电话暂挂一次主要的呼叫和激活参见呼叫。
- 参见的电话只有一个激活的呼叫(咨询呼叫)。

当会议完成(被连接的所有当事人)：

- 咨询的电话的参见呼叫终止。
- 咨询的电话的主要的呼叫获得一次远端的转移到会议网桥。
- 参见的电话获得一次远端的转移到会议网桥。

结果：

- S2终止，因为表示咨询的电话的参见呼叫，也终止。
- S4开始;它表示A的主要的呼叫继续和远端的转移，但是原始S1不可以是更新的，因为以前是被终止的由于及早暂挂。

- S3更新的获得，因为远端的B's从A调用到会议网桥。
- S4近端的xRefCi值将匹配S1近端的xRefCi值。

7. 代理程序A (extn 1000)从会议丢弃。

8. A (extn 1000)断开了。 C (extn 2000)谈与S3更新的B (extn 3000)会话-跟踪0是B (extn 3000)，跟踪1是C (extn 2000)。

考虑事项：

- 一次会议的降级到一次正常两大政党呼叫里实现作为两个剩余的电话的远端彼此调用
- 一次远端的转移触发现有的录制会话的更新。
- S3和S1将有配比的近端的xRefCi值。注意仅一次会话依然是活动，因为呼叫人C没有被启用的分叉。

9. 代理程序B (extn 3000)挂断。

10. C (extn 2000)和B (extn 3000)断开了。
S4被结束的会话。

考虑事项：

- 一次远端的转移导致现有的会话的更新。分叉的电话在跟踪0在跟踪1更改保持唯一的参与者，但是参加者对新的当事人。
- 会议用所有电话转移创建到会议网桥。所以，会议操作正如一套转移。现有的会话在分叉的电话保持唯一的参与者在跟踪0的那些会话，但是参加者中在跟踪1更改更新对会议网桥。
- 一旦第三方从会议下降，彼此调用当事人。这再更新现有的会话，分叉的电话在跟踪0在跟踪1更改保持唯一的参与者，但是参加者对另一个当事人。
- 如果第四个当事人被添加到会议网桥，没有在元数据的征兆，除非第四个当事人也有其被启用的自己分叉。

在统一的通信管理器10.x的情况下和以后，这是结果：

1. 呼叫人C (extn 2000)呼叫代理程序A (extn 1000)。

2. C (extn 2000)谈与A (extn 1000)会话S1 -开始-跟踪0是A (extn 1000)，跟踪1是C (extn 2000)。

3. 代理程序A (extn 1000)参见代理程序B (extn 3000)。

4. C (extn 2000)听到MoH A (extn 1000)谈与B (extn 3000)。

- 会话S1 -结束
- 会话S2 -开始-跟踪0是A (extn 1000)，跟踪1是B (extn 3000)
- 会话S3 -开始-跟踪0是B (extn 3000)，跟踪1是A (extn 1000)

考虑事项：

- 因为代理程序A的电话暂挂呼叫人C，会话S1结束。
- 因为两个电话为分叉，被配置会话S2和S3存在。
- 参与者以及xRefCi两个参加者的在S2和S3相同，但是在被倒转的位置从彼此。
- 因为咨询认为一次独立呼叫， S1 xRefCi值在S2或S3没有被反射

5. 代理程序A (extn 1000)完成会议。

6. C (extn 2000)谈与A (extn 1000)和B (extn 3000)

- 会话S2 -结束
- 会话S3 -结束
- 会话S4 -开始-跟踪0是A (extn 1000)，跟踪1是会议网桥
- 会话S5 -开始-跟踪0是B (extn 3000)，跟踪1是会议网桥

考虑事项：

一次远端的转移触发一录制会话结尾和另一个记录开始。会议的完成是列出的被实施这里：

- 在咨询期间： 咨询的电话暂挂一次主要的呼叫和激活参见呼叫参见的电话只有一个激活的呼叫(咨询呼叫)
- 当会议完成(被连接的所有当事人) ：
  - 咨询的电话的参见呼叫终止
  - 咨询的电话的主要的呼叫获得一次远端的转移到会议网桥
  - 参见的电话获得一次远端的转移到会议网桥
- 结果： 因为代理程序A和代理程序B有被启用的分叉两次新的会话被创建S4近端的xRefCi值和S1近端的xRefCi重视匹配S5近端的xRefCi值和S3近端的xRefCi重视匹配S4和S5的远端的xRefCi值不配比，即使两个被连接到同一会议网桥

7. 代理程序A (extn 1000)从会议丢弃

8. A (extn 1000)断开了。 C (extn 2000)谈与B (extn 3000)

- 会话S4 -结束
- 会话S5 -结束
- 会话S6 -开始-跟踪0是B (extn 3000)，跟踪1是C (extn 2000)

考虑事项：

- 一次会议的降级到一次正常两大政党呼叫里实现作为两个剩余的电话的远端彼此调用
- 一次远端的转移触发一录制会话结尾和另一录制会话开始
- S6和S5将有配比的近端的xRefCi值。注意仅一次会话依然是活动，因为呼叫人C没有被启用的分叉

9. 代理程序B (extn 3000)挂断

10. C (extn 2000)和B (extn 3000)断开了

- 会话S6 -结束

考虑事项：

- 一次远端的转移导致一次会话结尾和别的开始
- 会议用所有电话转移创建到会议网桥。所以，会议操作正如一套转移。结束现有的会话，并且新的会话被创建在分叉的电话和会议网桥之间
- 一旦第三方从会议下降，彼此调用当事人。这结束包括会议网桥并且启动在两个剩余的终端之间的新的会话的会话
- 如果第四个当事人被添加到会议网桥，没有在元数据的征兆，除非第四个当事人也有其被启用的自己分叉

2.如何关联不同的呼叫会话的参考呼叫ID，在统一的边界网元分叉下？

当统一的边界网元分叉，非常少量情况导致一次呼叫被分裂成多次记录会话。暂挂/恢复，转移，并且会议操作在许多情况下不启动新的记录会话。在新的会话被创建的少量案件，有共用值， CCID (呼叫相关性ID)。此值对在呼叫的所有会话是普通。CCID是Cisco GUID的十进制表，是由Cisco语音路由器生成的一个唯一呼叫键。收到一次呼叫的第一个路由器生成此键，并且通过它在线路下到所有随后的设备包括Cisco MediaSense。

统一的边界网元不创造xRefCi值，但是用统一的通信管理器电话分叉的呼叫创建相似性， Cisco MediaSense也综合一个对每次统一的边界网元呼叫的xRefCi值。这些在跟踪级别的元数据能被看到，与CCID一起，出现于会话级别。

这些情况原因统一将被分裂的边界网元记录成多个会话：

2.1.MID呼叫编码更改

如果转移、会议、会议丢弃，或者其他操作造成当事人重新协商他们的编码， Cisco MediaSense结束当前录音会话并且开始新的。两次会话共享同样CCID和同一个对xRefCi值。

2.2. 咨询传输

咨询传输是从一个代理程序的一次转移到另一个，两个代理程序彼此谈，当原始呼叫人暂挂中时等待。呼叫的咨询段在某个方面与整体呼叫有关，并且配置统一的通信管理器这样是可能的请参见呼叫穿过多维数据集。然而，统一的边界网元和Cisco MediaSense不知道这些呼叫是相关的，并且他们创建新的CCID和一个新的对此会话的xRefCi值。

这些呼叫可以互相产生关联与比较参与者deviceRef和时间戳字段。考虑此方案：

1. 呼叫人C (extn 2000)呼叫代理程序A (extn 1000) (sessionId = S1， CCID = C1)
2. 代理程序A与代理程序B (extn 3000) (sessionId = S2， CCID = C2)咨询
3. 代理程序A丢包和呼叫人C与代理程序B (sessionId = S1、CCID = C1)谈

在此方案的红旗在第2.步。在该周期，代理程序A (deviceRef 1000)立即是一个参加者在两次记录的会话：

- 会话= S1/CCID = C1和
- 会话= S2/CCID = C2

所以， S1与S2和C1有关与C2有关。

2.3. 参见呼叫检测

首先，我们需要清楚的定义参见呼叫：

由一个当前参加者在现有的会话做对终端是外部会话，并且在该会话排除其他参加者的任何附属呼叫。

在理论上，此方案可能包括代理程序暂挂呼叫人用他的上司检查luch中断，甚至代理程序呼叫人暂挂从他的妻子收到呼叫，但是我们暂时忽略那些可能性。

是可能的为了客户端应用能由Cisco MediaSense事件流的跟踪发现一次咨询呼叫在实时。 如果客户端观察会话开始的事件包含一特定deviceRef，与另一个会话开始的事件包含同一deviceRef没有干预的对话端事件，能认为，在两个会话开始的事件和CCIDs找到的sessionIds是关联的。

历史上，客户端能检查其中任一与Cisco MediaSense API参见产生关联与一次特定主要的呼叫的呼叫。假设客户端认识在CCID <C1>的该代理程序A使用的extn 1000。此指令查找任何关联参见呼叫：

Step1。通过发出getSessionByCCID(<C1>)检索主要的呼叫的会话元数据。

Step2.提取sessionStartDate (呼叫它<Ta>)和sessionDuration。

Step3.通过添加sessionDuration计算sessionEndDate (呼叫它<Tb>)到<Ta>。

Step4.运行此API请求：

https://Mediasense IP address:8443/ora/queryService/query/getSessionsByDeviceRef?value=1000&minSessionStartDate=<Ta>&maxSessionStartDate=<Tb>

此查询能返回超过一次会话。 如果它，则所有可以假设与同一个呼叫产生关联。

2.4.参见从广泛参与者的呼叫检测

被提及的程序参见呼叫检测部分，查找全部参见由收到最初的电话的设备做的呼叫。然而，那里若参见由呼叫是随后调用的设备做的呼叫？

考虑此程序：

1. 呼叫人呼叫Agent1
2. Agent1与代理程序2咨询，然后丢包
3. 呼叫人与代理程序2讲话
4. 代理程序2与代理程序3咨询，然后丢包
5. 呼叫人与代理程序3讲话

此程序不捉住在代理程序2和代理程序3.之间的咨询呼叫。

因为这是一次统一的边界网元呼叫，我们能利用事实所有呼叫人和其中每一个的连接代理程序之间在同样录制会话包括和事实介入的所有代理程序在同一次会话列出作为参加者曾经。 因此，从主要的会话元数据，我们能收集是包含的所有deviceRefs的列表。 要查找那些会话，我们能做一系列的呼叫到getSessionsByDeviceRef，与每个请求一deviceRef一起指定主要的会话的时间范围。

或者，进程可以简化与单个**getSessions**请求例如此：

```
{
    "requestParameters": [
        {
            "fieldName": "deviceRef",
            "fieldConditions": [
                {
                    "fieldOperator": "equals",
                    "fieldValues": [
                        "1000"
                    ],
                    "fieldConnector": "OR"
                },
                {
                    "fieldOperator": "equals",
```

```
                    "fieldValues": [
                        "2000"
                    ],
                    "fieldConnector": "OR"
                },
                {
                    "fieldOperator": "equals",
                    "fieldValues": [
                        "3000"
                    ],
                    "fieldConnector": "OR"
                },
                {
                    "fieldOperator": "equals",
                    "fieldValues": [
                        "4000"
                    ]
                }
            ],
            "paramConnector": "AND"
        },
        {
            "fieldName": "sessionStartDate",
            "fieldConditions": [
                {
                    "fieldOperator": "between",
                    "fieldValues": [
                        <Ta>, // session start time
                        <Tb>// session end time
                    ]
                }
            ]
        }
    ]
}
```

此查询回与原始主要的呼叫和所有产生关联的所有咨询电话其转移。

此程序太宽广地实际上转换网。 如果例如，在deviceRef 4000的代理程序执行并且终止了偶然开始，*在*<Ta>后的一次完全独立的呼叫，并且，*在*他被添加了到正在考虑中前的呼叫，此程序包括独立请呼叫在集。 此问题可以虽则解决，与有用的资料在主要的会话的元数据。 每个参与者的信息包括他加入会话和他的使用权的期限的时间偏移。 客户端代码可能使用信息删除无关的会话从接受得上述的列表。 或者，它可能公式化正确地构筑时间在期间每个代理程序在主要的呼叫的一系列的直接getSessions或getSessionByDeviceRef查询。 我们离开作为练习到读者。

2.5.摘要

在前面的部分，我们提交了与一特定Cisco MediaSense录制会话产生关联的所有会话检索的precedures。 然而，我们也看到一次特定呼叫可能分开成超过一次会话，和一旦MID呼叫编码更改。

如何检索与*所有*会话(以及咨询)产生关联的所有记录被连接到呼叫人的交互作用？

答案是延伸此多个detectiion的指令参见呼叫。首先，我们收集共享正在考虑中主要的会话CCID的所有会话。然后，我们创建我们的参与者名单从*所有的*那些会话记录。 其次，我们计算时间范围作为最早的会话的sessionStartDate通过最新的会话的结尾。最后，我们能执行显示的getSessions查询。

和前面，我们能最终获得许多记录捕获，因此我们可能完成后加工步骤从其列表删除那些无关的会

话。

3.如何连结呼叫在Cisco MediaSense中与他们的在其他解决方案组件的外观？

3.1. 标识相关表

这两张表—一个人统一的通信管理器的呼叫，并且一个人统一的边界网元的呼叫。每列表示解决方案组件或协议，与第一列代表Cisco MediaSense。每行表示标识的一种特定类型。

要读表，从表示数据项您知道，然后查找对列水平地表示解决方案组件您要查找呼叫的信元请开始。 在该信元的条目由什么名字指示确切同样数据项在目标组件被认识。 如果目标组件有一个空白的信元在该行，则该数据项没有为该组件所知。 您能寻找您能交叉垂直到另一行该信元不是空白的在目标组件的列的干预的列。

例如，与一次统一的通信管理器呼叫，假设，您认识GED-188 CallReferenceID，并且您在Cisco MediaSense中要查找呼叫。从GED-188列检查左，您看到没有在MediaSense列的值，因此您不能映射直接地到它。

然而，有您能在行间曲折前行的列：统一的通信管理器CDR。客户端能通过搜索选择适当的统一的通信管理器CDR记录IncomingProtocolCallRef匹配GED-188 CallReferenceID的一个。记录包含值在Cisco MediaSense中呼叫destLegCallIdentifier，因此是相同的象MediaSense NearEnd xRefCi，并且可能使用查找对应的记录。

统一的通信管理器CDR记录没有被写，直到一些时间，在呼叫的结尾终止，然而，因此后可能历史上只使用此方法。

有另一条路径。向下从GED-188 CallReferenceID查找。它结果您能也使用AlertingDevice和AnsweringDevice匹配deviceRef字段在MediaSense。此方法在实时也运作。

Call Correlation for Calls Forked by a Unified CM IP Phone

MediaSense	Ingress Gateway or CUBE	AAA RADIUS CDR	UCM CDR	TAPI/JTAPI field	UCCE Database	UCCE Script	CTIOS	GED-188
(1)	Cisco-GUID		IncomingProtocolCallRef	CiscoConnection.UniqueID	TCD.CallGUID	Call.CallGUID		CallReferenceID
NearEnd xRefCi			destLegCallIdentifier	Terminal.ConnectionID				
FarEnd xRefCi			origLegCallIdentifier	Terminal.ConnectionID				
			global_CallID_callManagerId + global_CallID_callId (A.K.A. UCM GCID)	CiscoCall.CallID	TCD.PeripheralCallKey			
deviceId							Agent.AgentInstrument	
deviceRef					TCD.InstrumentPortNumber (2)		Agent.AgentExtension or Agent.Extension	AlertingDevice or AnsweringDevice

Call Correlation for Calls Forked by CUBE

MediaSense	Ingress Gateway or CUBE	AAA RADIUS CDR	UCM CDR	TAPI/JTAPI field	UCCE Database	UCCE Script	CTIOS	GED-188
CCID (3)	Cisco-GUID	Cisco-GUID	IncomingProtocolCallRef		TCD.CallGUID	Call.CallGUID		CallReferenceID
deviceRef	Called or calling party extn				TCD.InstrumentPortNumber (2)		Agent.AgentExtension or Agent.Extension	AlertingDevice or AnsweringDevice

考虑事项：

1. 在统一的通信管理器呼叫记录， Cisco MediaSense从UCM实际上接受一Cisco GUID，但是不是由其他解决方案设备捕获的同样一个。因此MediaSense甚而不存储此值。
2. 对于代理对代理呼叫， TCD.InstrumentPortNumber是目的地代理扩展。呼叫的代理扩展可以在TCD.ANI找到。
3. CCID是Cisco GUID以十进制形式，是4连字符被分离的套10位十进制数字。这些可以被转换成与每10位十进制数字的转换的六角形的表对一个八位数字的六角形的编号的，并且去除连字符。那里Cisco GUID用于UCCE，以其六角形的形式。

4.如何确定哪跟踪有主叫方，并且哪跟踪有被叫方？

4.1.对于呼叫由多维数据集分叉了

对于多维数据集呼叫，跟踪0总是映射对锚点段媒体流。锚点段是配置记录配置文件的媒体的拨号点。 第二个跟踪映射非锚点段。

如果有记录配置文件的媒体启用在*入站*dialpeer，则锚点段成为段。换句话说，主叫方出现于跟踪0，并且被叫方出现于跟踪1。

如果有记录配置文件的媒体启用在*outbound* dialpeer，则锚点段成为外段。 在那种情况下主叫方出现于跟踪1，并且被叫方出现于跟踪0。

4.2.对于呼叫由统一的CM电话分叉了

对于分叉，在简单的呼叫情形的统一的CM您在元数据能使用xRefCi字段确定哪个当事人在哪些媒体请跟踪。数字上更小的xRefCi通常是指主叫方的跟踪。被叫方的跟踪数字上更大(通常由一个，但是它可能更多根据一个合理加载的系统)。然而，这些xRefCi值最终包裹到零。因此，如果发现一值是大数字，并且其他是一个小的编号，您假设他们的位置被倒转。

在更加复杂的方案中，此算法总是不运作。如果附加服务被调用，例如转移和会议，并且UC管理器簇包括超过一个节点，则xRefCi值顺序地不一定被创造，并且您不能假设，他们的顺序有所有意味着。直接方式确定一个特定的对的预定的顺序xRefCi值是否可以委托将查看xRefCi值的第一个字节。此字节表示该特定的标识被创建的UC管理器节点ID。如果两xRefCi值的第一个字节是相同的，则他们预定是正确的。如果他们是不同的，则预定也许不是正确的。

对于这些案件，确定呼叫方向的唯一方法在实时是获取从其他来源的信息，例如JTAPI事件输入。一旦呼叫结束，并且几分钟流逝了，您能总是确定呼叫方向和检查UC管理器的CDR数据为呼叫。特别地，在CDR记录的origLegCallIdentifier字段总是表示呼叫人。

5.什么是CLOSED_ERROR的会话状态的可能的原因？

CLOSED_ERROR的会话状态的可能的原因包括：

1. 呼叫控制服务器从开放的媒体(记录)服务器接受了一个错误反应或Close请求。
2. 呼叫控制服务器发现了一个SIP信号错误，例如缺少ACK。
3. 会话顺利地结束了，但是所有跟踪有零的大小。

当会话在激活状态时，是正常的没有在元数据的期限，因为期限不知道，直到会话是闭合的。

对于在CLOSED_ERROR状态的会话，如果会话或跟踪期限字段不是存在事件或getSessions数据

，然后此跟踪的媒体不是可用的。

6. 被修剪的和被删除的会话有何区别？

考虑这两次查询：

6.1. 使用getAllPrunedSessions查询

此查询返回一套会话，会话状态被删除的所有：

```
https://Mediaserver IP
address:8443/ora/queryService/query/getAllPrunedSessions?minSessionStartDate=1301788800000&maxSe
ssionStartDate=1312329599000
```

6.2.使用getSessions查询

此查询不返回会话：

```
https://MediaServer IP address:8443/ora/queryService/query/getSessions
{
            "requestParameters":
            [{
                        "fieldName" : "sessionState",
                        "fieldConditions":
                        [{
                                    "fieldOperator" : "equals",
                                    "fieldValues" : [ "DELETED" ]
                        }],
                        "paramConnector" : "AND"
            },
            {
                        "fieldName" : "sessionStartDate",
                        "fieldConditions":
                        [{
                                    "fieldOperator" : "between",
                                    "fieldValues" : [ "1301788800000", "1312329599000" ]
                        }]
            }]
}
```

6.3.为什么工作情况区别在被修剪的和被删除的会话上？

工作情况区别是故意的。 参考此在MediaSense文档的部分：

- API参数描述：getAllPrunedSessions API说明：

请使用此API搜索所有被修剪的记录…被修剪的术语是指记录由Cisco MediaSense系统删除。使用deleteSessions API，如果明确地删除了任何记录，则这些被删除的记录没有考虑作为被修剪的记录。

- 在部分积极的存储管理下的MediaSense SRND ：

当会话被修剪时，与这些会话产生关联的元数据在数据库保持，在这些会话被标记作为'修剪'以后。此元数据不采取很多存储空间与记录比较，但是采取若干空间，并且应该周期地去除。要帮助在此活动，客户端可能周期地发行一个API要求被修剪的会话，或者客户端可能决定接收会话被修剪的事件和明确地删除客户端不再需要的那些事件。
要澄清，两次查询是完全不同的。实际上，第二次查询(lwhich搜索状态被删除)的所有会话总是返回空集。正常每日查询过滤会话以被删除的状态，即使那是什么被请求。唯一的例外是**getAllPrunedSessions**。此例外打算协助解决应用程序查找被修剪的会话，以便应用程序能请求这些会话被删除。

一旦使用在您从**getAllPrunedSessions**获得被修剪的会话的列表的**deleteSessions** API，这些会话不再出现于getAllPrunedSessions的结果。这样会话从元数据立即完全地被去除。

另一个方式查看此是被修剪的会话不是事和被删除的会话一样：

1. 被修剪的会话为删除被指示了由在MediaSense系统的一种算法。人在决策未涉及修剪这些会话。因此，即使这些会话被迁移向被删除的状态，这些会话从元数据实际上没有被去除。需要人(或应用程序)干预。由于这些会话在被删除的状态，这些会话不是可视的对多数查询。然而，这些会话是可视的对getAllPrunedSessions查询API。并且，如果任何mp4文件为这些会话生成了，这些mp4文件继续是存在磁盘和继续可以下载，直到被修剪的会话实际上被删除。
2. 被删除的会话通过明确地呼叫deleteSessions API指示。此标记可以完成到已经被修剪的会话或到未被删除的会话。一旦会话由deleteSessions API删除了，此会话不再是可视的对所有查询。这包括getAllPrunedSessions API。这些被删除的会话从元数据立即被去除，以便磁盘空间可以恢复。

7.如何配置媒体分叉的一个TDM网关？

当您有呼叫流和您要记录那些呼叫的一个PSTN网关通过。这些呼叫是TDM对SIP呼叫。然而，媒体分叉只是可用的在SIP对SIP呼叫。

这些呼叫可以被记录。这些呼叫能处理通过路由器每第二次。配置指导和其他详细资料可以在此白皮书找到。

8.如何捕获实际目的地电话，当曾经Hunt Group时？

当您使用分叉从多维数据集时的媒体， MediaSense元数据通常包含被叫方的扩展名。然而，如果数字称为是通信管理器Hunt Group导频号，默认情况下然后元数据只包含该导频号。它不包含实际上应答呼叫电话的扩展名。

有能更改此的通信管理器设置。在搜索/试验配置页，请查找部分题为**被连接的当事人转换**。必须打开设置**显示行组成员DN作为被连接的当事人**。

此功能可用在通信管理器9.0(1)及以后。

9. 统一的通信管理器基于网络记录为什么是建议使用作为首选分叉机制？

使用统一的通信管理器基于网络记录(NBR)，您能使用网关记录呼叫。不管设备、位置或者地理，NBR允许统一的通信管理器路由记录呼叫。使用NBR，呼叫记录媒体可以来源从IP电话或从被连接到在SIP Trunk的统一的通信管理器的网关。统一的通信管理器动态地选择根据呼叫流和呼叫参与者的正确的媒体来源。

NBR提供一个自动退路构件在网桥(围嘴)，当集成服务路由器(ISR)时是未提供的，因为没有需要分开的记录的配置。这是有用的，在用户要包括代理程序代理程序参见在记录策略处的呼叫，因为统一的边界网元不能记录参见呼叫，因此围嘴需要分开被启用。

使用xRefci， NBR和围嘴呼叫可以关联，从统一的通信管理器JTAPI是可得到。CISCO-GUID不是需要的，意味着没有需要CTI服务器和CTIOS连接。尽管有单个相关性标识，在组件间的相关性更加严格，并且可以执行用一个统一方式对立于呼叫流。

当NBR，直接拨号以及拨号程序被起动的出局访问可以关联与他们的在其他解决方案组件的外观。

使用NBR， TDM网关记录自动地使用没有路由器的容量的已分解。目前， TDM网关记录没有用MediaSense 10.5支持。

10.节点很多时间为什么花费升级到MediaSense 10.5 ？

节点能耗费几小时升级取决于拿着记录的数量和大小。对于MediaSense 10.5，当您升级与非常大数据集时的一个节点，需要大约每1百万个记录的90另外的分钟。

11.什么是俄国时间区域更改的影响对MediaSense搜索和作用应用程序？

MediaSense的用户搜索，并且作用应用程序受影响，如果他们位于任何被影响的时间区域或，如果他们选择查寻标准的一个被影响的时间区域。协调与MediaSense的第三方合作伙伴产品类似受影响，直到他们更新他们的各自时间区域表。

解决方法是选择匹配从GMT的正确的偏移量的时间区域，即使城市不再是正确的。

12.什么是MediaSense支持的语言？

这是MediaSense支持的语言：

- 阿拉伯
- 丹麦语
- 荷兰语
- 英语(美国)
- 芬兰语
- 法语
- 德语
- 意大利语

- 日语
- 韩文
- 挪威
- 波兰语
- 葡萄牙(巴西)
- 俄语
- 简体中文
- 西班牙语
- 瑞典语
- 繁体中文
- 土耳其语

13.如何监控MediaSense系统性能？

要监控MediaSense系统性能，请分析这些关键性能指示器(KPIs)的值用RTMT工具或Cisco最初协作保证工具。

关于RTMT工具或Cisco头等协作保证工具的更多信息，请参考Cisco MediaSense用户指南的统一的RTMT管理和Cisco头等协作保证管理部分。

KPIs和他们的门限值

关键绩效指标(KPI)	RTMT计数器	被建议的门限值
呼叫成功率	MediaSense呼叫控制Service>记录数会话不出错误	99.99%
	MediaSense呼叫控制Service>记录数与错误的会话	呼叫成功率=记录数会话不出 * 100
API API的回应平均时间	Cisco MediaSense API Service>平均值查询响应时间	60秒
记录起始平均值设置延迟	Cisco MediaSense呼叫控制Service>平均值设置延迟	3秒
CPU平均值利用率	处理器> % CPU时间	90%
内存平均利用率	使用的内存> %Mem	70%
RTP/UDP信息包丢弃	下降的网络接口> Rx > eth0	0
	网络接口> Rx错误> eth0	0

14.如何配置浏览器运行浏览器球员在MediaSense ？

凭浏览器，请执行这些步骤运行浏览器球员：

Internet Explorer 9	Internet Explorer 11	Mozilla Firefox
1. 在**MediaSense搜索和作用**，请点击**play图标**录制会话。	前提：在新安装MediaSense的情况下11.0，请保证使用各自完全合格的域名(FQDN)， MediaSense节点被添加到簇。 在对MediaSense 11.0的升级的情况下，请保证以前被添加使用主机名-的MediaSense节点，应该由各自FQDN当前显示。检查在**MediaSense服务器配置**窗口(**Cisco MediaSense管理>System的**	1. 添加一个MediaSense节点的证书端口的mp4url 8446在Mo Firefox的委托的权限的。

MediaSense服务器列表>
MediaSense服务器配置)。
执行以下步骤：
1. 设置*集hostnameformediaurl* CLI作为“真”做MediaSense准备mediaurls的mp4url和其余使用仅FQDN。

```
admin:set useHostNameForMediaURL
admin:set useHostNameForMediaURL true
```

2. 重新启动配置服务激活属性。

```
admin:utils service restart Cisco
MediaSense Configuration Service
```

Note:如果服务未适当地重新启动，再请执行同一个命令。

3. 在配置服务重新启动，签字并且签到对**MediaSense搜索和作用后**。

限制：使用IP，万一MediaSense节点以前被添加了，然后节点继续由IP仅显示在升级以后给MediaSense 11.0。浏览器球员在Internet Explorer 11不工作，不考虑*hostnameformediaurl* CLI value命令的。在这种情况下，建议不应该设置CLI命令的*hostnameformediaurl*作为“真”。
执行以下步骤添加MediaSense自签证书到Windows委托权限。
1. 打开**MediaSense搜索和作用**。
安全证书弹出式窗口出现。
2.点击**继续**。
MediaSense搜索和play窗口出现。
3.在地址栏，请点击**验证错误**图标。
4.点击**视图证书**。
认证弹出式窗口出现。
5.点击**安装证书**。
认证导入向导出现。
6.**其次**点击。
7.在**证书存储**窗口，请选择**地方在以下存储**单选按钮的**所有证书**并且点击**访问**。
挑选证书存储对话框出现。
8.检查**Show physical存储**复选框并且选择**可信的根Certificationhorities**文件夹。
9.点击OK键和**其次**。
10.点击**完成**完成认证导入。
警告的安全弹出式窗口看上去确认认证的安装。
11.**是**点击。
下列信息出现。

```
The import was successful.
```

2. **是**点击委托认证。
Note:验证提供的自签证书是被瞄准的MediaSense节点通过验证在认证的技术详细资料的FQDN。

3. 点击**play图标**与所选的记录相应。
浏览器球员播放所选的录制会话。

2. 要添加自签证书，请点击录话的**下载**图标并且选择**mp4**。此**连接是不信任的**弹出式窗口。

2. 要添加自签证书，请点击录话的**下载**图标并且选择**mp4**。此**连接是不信任的**弹出式窗口。**Note:**验证提供的自签证书是准的MediaSense节点通过验证的技术详细资料的FQDN。

12. 单击 **Ok**。
13.点击OK键在**认证**弹出式窗口的。
14.Close和打开浏览器。
15.开放**MediaSense搜索和作用**。
安全证书issxvi仍然仍然存在。
16.去**工具> Internet选项 >Advanced**，不选定**关于认证地址不匹配**检查boxviiunder安全的**警告**。
17.单击**应用**和“确定”。
18.重新启动浏览器和开放**MediaSense搜索和作用**。

保证MediaSense服务器通过在浏览器的FQDN是可取得。否则，请连接对C:\Windows\System32\drivers\etc，打开在Notepad的主机文件并且添加MediaSense服务器和其FQDN的IP地址在文件的底部。浏览器球员开始工作在Internet Explorer 11。

Note:如果记录是存在簇的一个不同的MediaSense节点，提示您添加该MediaSense节点认证在委托的权限的。

3. 点击**我了解风险**链路。
4. 点击**添加例外**。
添加安全例外弹出式窗口出现
5. 点击**确认安全例外**。
8446端口特定MS节点的自签证
加到浏览器的委托的权限。