Overview

A common reporting requirement in UCCE/PCCE environment is how to report calls released by Agents or Customers.

This report is available with UCCX, while ICM, which is the routing engine of UCCE, does not contain this detail.

The Termination_Call_Detail table contains all the transactions of each call lifecycle (to allow Cradle to Grave reporting), including all individual call legs related to IVR navigation, connections to Agents, Outbound, mid calls.

But when a call is connected to an Agent and then released, ICM does not know who did it. The call was simply released.

Cisco UCM, the IP PBX in the UCCE solution, contains this information but there is a more elegant and 'linked' way to get to this information.

CVP Reporting Database

CVP is involved in the lifecycle of each single call delivered within UCCE, regardless if the call was queued, connected to Agent, Transferred or other.

CVP acts as B2BUA (Back to Back User Agent) which means that it is in between the 2 legs:

the incoming external leg (call signaling from ingress GW or SIP Trunk) and the incoming internal leg (Call signaling path from CVP and VXML GW or CUCM:

Customer phone <---->Ingress GW<---->**CVP CallServer** <----> UCM(Agent device)

You can think to CVP Callserver signaling paths as to you arms and hands: You realize if someone touch your left or right hand.

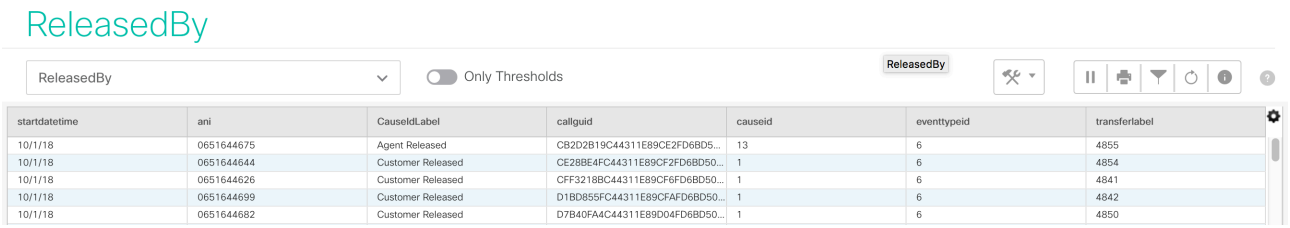
This is why CVP is able to track in its record who released the call.

You can create a CUIC custom report with the following query on CVP reporting database:

SELECT
call.startdatetime,

```
call.ani,
call.callguid,
callevent.transferlabel,
callevent.eventtypeid,
case
  when callevent.causeid = 13 then 'Agent Released'
  when callevent.causeid = 1 then 'Customer Released'
else 'Other' end as CauseIdLabel,
callevent.causeid
 FROM call, callevent
 WHERE call.callguid=callevent.callguid
 AND (callevent.eventtypeid = 6 AND callevent.causeid IN (1, 13))
AND callevent.transferlabel < 5000 -- This is to ignor all VRU labels
```

You get a report like this:



startdatetime	ani	CauseIdLabel	callguid	causeid	eventtypeid	transferlabel
10/1/18	0651644675	Agent Released	CB2D2B19C44311E89CE2FD6BD5...	13	6	4855
10/1/18	0651644644	Customer Released	CE28BE4FC44311E89CF2FD6BD50...	1	6	4854
10/1/18	0651644626	Customer Released	CFF3218BC44311E89CF6FD6BD50...	1	6	4841
10/1/18	0651644699	Customer Released	D1BD855FC44311E89CFAFD6BD50...	1	6	4842
10/1/18	0651644682	Customer Released	D7B40FA4C44311E89D04FD6BD50...	1	6	4850

As you can see the report says who released the call, but the information about the Agent is only available through the extension where Agent was logged on: you don't know who is the Agent.

If your customer is happy with this, that's fine, but if you need to provide also the AgentID or Agent Name and additional CC details (SkillGroup, Service Name), then you need to integrated this report with ICM reporting database on HDS.

How to create a joined query between CVP and ICM

Actually, there is no supported way to do it. Usually you need a Data-mart that gets records from both DBs (this is supported, yes), and then run your queries within the Data-mart. But remember that CUIC only supports MSSQL and Informix Datasources.

A possible alternative is to use a 3rd MSSQL configured with **Linked Server**.

How creating single CUIC Report Definition to run heterogeneous queries against different DBs.

A linked server allows for access to distributed, heterogeneous queries against OLE DB data sources. After a linked server is created, distributed queries can be run against this server, and queries can join tables from more than one data source. If the linked server is defined as an instance of SQL Server, remote stored procedures can be executed (Source Microsoft).

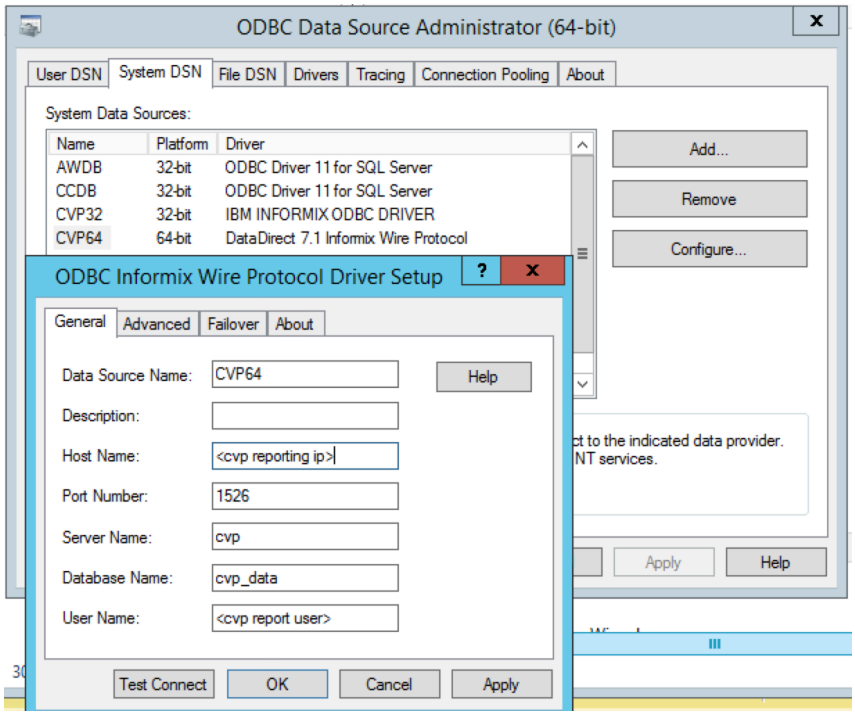
In our example below, just to simplify, we will create a Linked Server in the same MSSQL instance where ICM HDS database is running.

Remember, this is not supported by Cisco since you cannot use ICM server for other purposes. If you need TAC support for other issues, you need to remove all stuff and be able to reproduce your issue in clean environment.

But if you have a dedicated server configured with LinkedServer to CVP reporting database and ICM HDS database (both open for external query), then you are fine.

Installing and Configuring Linked Server

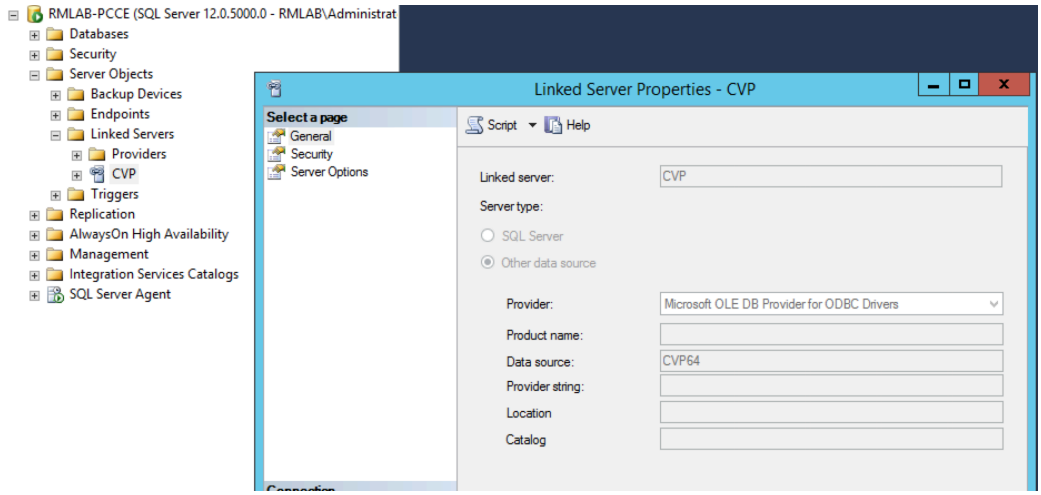
You have to install Informix ODBC driver from IBM website (ensure you run a 64bit version) and one installed, configure it to connect to CVP reporting database. Sample below:



You can also refer to CVP Datasource in CUIC to ensure you get right access details.

Once the ODBC driver is tested and working, you can create the Linked Server.

Run MSSQL Studio, and create a new LinkedServer under Server Objects referring to the previously created System System DSN.



Create now a new Database called **cvp_data** and a View called **callevent** using the following query:

SELECT * FROM OPENQUERY (CVP, 'SELECT * FROM callevent')

This is just to simplify the testing of the Linked Server. In our sample we use StoredProcedure and OPENQUERY with parameters to get better performances.

As you can see, the OPENQUERY is required to run the remote query against the Linked Server.

Remember that the 2nd argument of the OPENQUERY (the sql construct) is executed remotely, therefore you have to refine this query with all the info you have in order to get only the data you need locally and reduce data transfer. The result of the OPENQUERY will be used to join with local ICM tables.

JOINING CVP RECORD WITH ICM RECORDS

From the initial CVP report you see that you can retrieve all records with specific Released By outcome, extension involved, and callguid. This last data is what we use to correlate CVP database to Termination_Call_Detail, which contains all the other data we need.

The callguid is set in the SIP dialog and passed by CVP to ICM in the media.id ECC.

This variable is stored in the Termination_Call_Variable, which is has a RecoveryKey back to the related Termination_Call_Detail record, so that you can correlate them.

As result, you can create a more complex query that, using the callguid of CVP, retrieve the related Termination_Call_Detail record.

The example below refers to a StoredProcedure loaded into the cvp_data Database where we created the Linked Server.

In this Stored Procedure I use a Temp Table to get all relevant data from the Linked Server query, the I refer to this Temp Table to join the ICM tables.

```
USE [cvp_data]

GO

/****** Object:  StoredProcedure [dbo].[ReleasedBy]    Script Date: 10/1/2018 7:34:08 PM ******/

SET ANSI_NULLS ON

GO

SET QUOTED_IDENTIFIER ON

GO

-- =============================================

-- Author:                <Author,,Name>
```

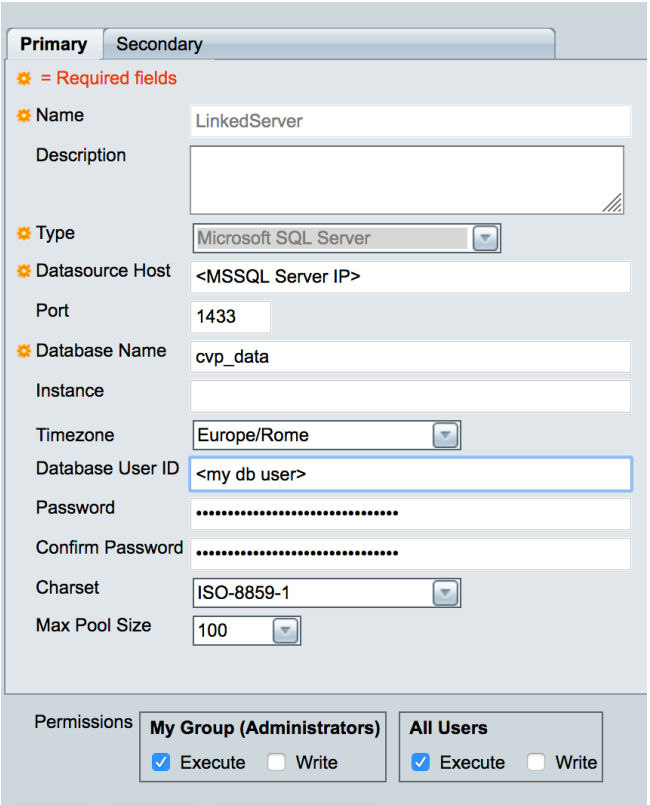
```
-- Create date: <Create Date,,>

-- Description:            <Description,,>

-- =============================================

CREATE PROCEDURE [dbo].[ReleasedBy] ( @mysdate  datetime, @myedate  datetime, @ani varchar(4000), @AgentID int)

AS

BEGIN

                -- SET NOCOUNT ON added to prevent extra result sets from

                -- interfering with SELECT statements.

                SET NOCOUNT ON;

   -- Insert statements for procedure here

SET ARITHABORT OFF SET ANSI_WARNINGS OFF SET NOCOUNT ON

Set ANSI_NULLS ON

Set ANSI_WARNINGS ON

If(OBJECT_ID('tempdb..#MyTempTable') Is Not Null)

Begin

   Drop Table #MyTempTable

End

create table #MyTempTable

(

   callguid Varchar(50),

   eventtypeid int,

   causeid int,

   eventdatetime datetime,

   transferlabel Varchar(50)

);

DECLARE @OPENQUERY nvarchar(4000) ,  @sdate varchar(40), @edate varchar(40)

SET @sdate = (SELECT CONVERT(VARCHAR(19), @mysdate, 120))

SET @edate = (SELECT CONVERT(VARCHAR(19), @myedate, 120))

SET @OPENQUERY = 'SELECT callguid,eventtypeid,causeid,eventdatetime, transferlabel

FROM callevent where (eventdatetime between ''''+@sdate+'''' AND ''''+@edate+'''') AND  eventtypeid = 6 AND transferlabel is not NULL AND
(causeid = 13 or causeid = 1)'
```

```
SET @OPENQUERY = 'SELECT * From openquery(CVP, '''+@OPENQUERY+''')'

SELECT @ani

Insert Into #MyTempTable

exec sp_executesql @OPENQUERY,N'@sdata varchar(40)',N'@edata varchar(40)'

-- select eventdatetime, callguid from #MyTempTable

select   b.DateTime,

        myDateTime = CONVERT(char(19),b.DateTime,120),

        c.eventdatetime,

        b.AgentSkillTargetID,

        b.SkillGroupSkillTargetID,

                        b.ANI,

                        SkillName = e.EnterpriseName,

                        AgentName = d.EnterpriseName,

                        c.callguid,

                        c.transferlabel,

case

  when c.causeid = 13 then 'Agent Released'

  when c.causeid = 1 then 'Customer Released'

else 'Other' end as Outcome

FROM  pcce_awdb.dbo.Skill_Group e, pcce_awdb.dbo.Agent d, pcce_awdb.dbo.Termination_Call_Variable a,
pcce_awdb.dbo.Termination_Call_Detail b,  #MyTempTable c

WHERE a.ECCValue = c.callguid AND

a.TCDRecoveryKey = b.RecoveryKey AND

a.ExpandedCallVariableID = '5002' AND

b.AgentSkillTargetID = d.SkillTargetID AND

b.SkillGroupSkillTargetID = e.SkillTargetID AND

(b.DateTime between @sdate AND @edate) AND

(@ani is null OR (b.ANI = @ani))  AND

(@AgentID is null OR (b.AgentSkillTargetID = @AgentID))

Drop Table #MyTempTable
```

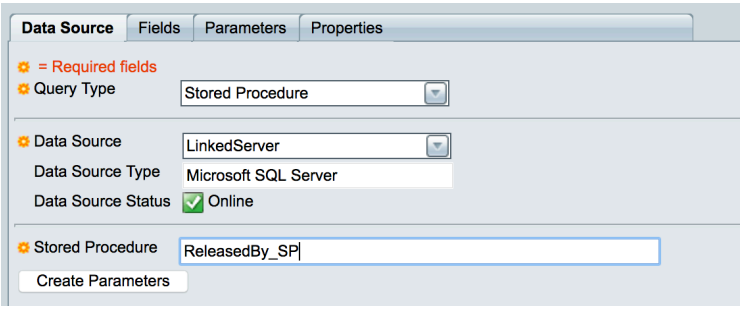
```
END

GO
```

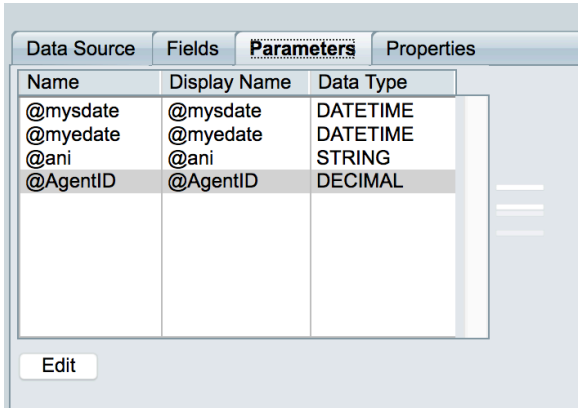
This Stored Procedure accepts 4 parameters: **Start/End datetime**, **AgentID** and **ani**

You can now create a CUIC Report Definition referring to this Stored Procedure, but first of all you have to create a new DataSource in CUIC pointing to your MSSQL where the Linked Server is configured. I our case, we point to the database hosting the Stored Procedure.



You can now create a Report Definition using this DataSource:





You can add Value List to @ani and @AgentID to simplify report usage.