配置CMS Spaces上的訪客和主機訪問並對其進行故障排除

目錄

簡介

本文說明如何使用API命令在思科會議伺服器(CMS)的空間上設定訪客和主機訪問。

必要條件

需求

思科建議您瞭解以下主題：

- 思科會議伺服器(CMS)，具有空間設定且可向其發出呼叫
- API客戶端（如Poster、Postman）或
- CMS API指南

採用元件

本文檔中的資訊基於CMS版本2.1

本文中的資訊是根據特定實驗室環境內的裝置所建立。如果您的網路運作中，請確保您瞭解任何指令可能造成的影響。

背景資訊

本文檔概述了方案的型別：

- 訪客和主機參與者使用不同的URI或呼叫ID
- 訪客和主機參與者使用相同的URI，其中根據PIN或密碼條目進行區分（兩者均非空）
- 訪客和主機參與者使用相同的URI，其中根據PIN或密碼條目（空/非空混合）進行區分
- 主機使用者是空間的成員，並通過webRTC登入獲得授權，訪客使用者使用callID加入會議。訪客和主機參與者為訪客使用者使用空或非空PIN/密碼使用相同的URI和callID

設定

在CMS中，訪客和主機參與者有四種區別的可能性，將在接下來的4個示例中描述，並且主要基於不同的**callLegProfiles**，這些配置檔案確定在空間上加入的參與者的呼叫中行為。

首先，說明對訪客和主機參與者使用不同**URI**（或呼叫ID）的方法，然後，通過在同一**URI**上使用不同的密碼（或超時）來附加該方法，以區分訪客和主機參與者。第三種訪客使用者超時或空PIN輸入的方法是CMS 2.1上的新功能，如發行說明第2.4節所示。第四種方法解釋如何設定具有已分配所有者/成員的空間上的訪客和主機訪問，並使空間的成員（所有者）成為空間的主機。

1)使用不同URI的配置

這是在CMS 2.1版本之前可用的基本配置，與用於其他呼叫ID的基本配置相同。要獲得相同空間上的訪客/主機訪問差異性，需要執行以下步驟：

1. 建立訪客**callLegProfile**(**needsActivation** = **true**)
2. 建立主機**callLegProfile**(**needsActivation** = **false**)
3. 將訪客**callLegProfile**分配給現有或新空間（預設訪問方法）
4. 在同一空間上使用不同的**URI**（和call-ID）建立新的**accessMethod**，並為其分配主機**callLegProfile**

步驟1.建立訪客**callLegProfile**(**needsActivation** = **true**)。

callLegProfile確定呼叫內行為，並且預設情況下會將訪客呼叫內行為分配給空間，以便以後可以在同一空間上使用不同的訪問方法，以及讓主機能夠加入。

> **附註**：例如，也可以在租戶級別(**/api/v1/tenants/<tenant-ID>)或系統級別(/api/v1/system/profiles**)上分配此項，以便將此項應用於所有空間（或每個租戶），但此處顯示在空間本身上。在通話中行為時，會考慮**callLegProfile**的最具體分配。

needsActivation引數對於訪客/主機行為是此處最重要的引數，因為如果設定為**true**，則在一個或多個**full/activator**（主機）參與者加入之前，參與者無法接收或提供音訊和影片。**callLegProfile**上的其他引數可在API指南的8.4.3部分找到，在該部分下，所顯示的引數也在此設定中相關（取決於您的要求）：

- presentationContributionAllowed
- rxAudioMute
- rxVideoMute
- deactivationMode（停用）|斷開連線 | reallyActivated)和deactivationModeTime [最後一個啟用

器離開呼叫時要執行的操作]

要建立訪客callLegProfile，請對/api/v1/callLegProfiles發出POST請求(首選引數和needsActivation引數設定為true)，以便您隨後可以對**callLegProfile-ID**執行**GET**請求，結果例如：

```
needsActivation>true

       needsActivation>

       < deactivationMode>deactivate deactivationMode>
```

記下以粗體標籤的**callLegProfile-ID**，因為在步驟3中，必須對訪客訪問空間應用此項。

步驟2.建立主機**callLegProfile**(**needsActivation = false**)。

同樣，為主機通話中行為建立主機**callLegProfile**。前面提到的引數同樣適用，不過可根據您自己的偏好和要求選擇引數。這裡的主要元素是將**needsActivation引數**設定為**false**，以便賦予它主機角色。

您可以通過/api/v1/callLegProfiles上的POST請求建立該請求，並且首選引數和needsActivation引數設定為**false**，這樣您就可以隨後對該callLegProfile-ID**執行**GET請求，結果例如：

```
needsActivation>

   false

       needsActivation>
```

記下以粗體標籤的**callLegProfile-ID**，因為必須在步驟4中的**space accessMethod**上為主機訪問應用此命令。

步驟3.將訪客**callLegProfile**分配給現有空間或新空間(預設的**accessMethod**)。

在現有空間(**/api/v1/coSpaces/<coSpace-ID>**)上執行PUT命令以調整該空間，或在**/api/v1/coSpaces**上執行**POST**命令，以使用步驟1中建立的訪客**callLegProfile**引數作為該空間的呼叫內行為，建立新空間。您還可以根據**API指南**第6.2節，為該空間設定**URI**、**passcode**和**call-ID**引數。

對該空間執行GET請求(**/api/v1/coSpaces/<coSpace-ID>**)，以驗證訪客**callLegProfile**是否與其關聯，以及**URI**和**call-ID**值。此範例在步驟1中建立guest **callLegProfile**的範例輸出是具有**URI**值**guest.space**和**call-ID**628821815（沒有密碼集）的輸出：

```
uricallIdcallLegProfile>

   d4bfe12d-68cd-41c0-a671-48395ee170ab
```

```
callLegProfile>
```

記下以粗體標示的空間ID，因為此操作必須用於在步驟4中的該特定空間上建立**accessMethod**。

步驟4.在該空間上使用不同的**URI**(和**call-ID**)建立新的**accessMethod**，並為其分配主機**callLegProfile**。

您想要建立與訪客存取不同的存取空間方式（目前是預設的方式）。這是通過在/api/v1/coSpaces/<coSpace-ID>/accessMethods上使用POST命令在空間本身上指定accessMethod來完成的，其中coSpace-ID是步驟3(7cc797c9-c0a8-47cf-b519-8dc5a01f1ade)中應用了步驟2的主機呼叫LegProfile的以及不同的URI和call-call-ID欄位的call-ID call-ID 的是。

對該空間訪問方法(/api/v1/coSpaces/<coSpace-ID>/accessMethods/<accessMethod-ID>)發出GET請求後，您必須能夠看到類似此型別的輸出，其中可以看到與空間的預設access方法不同的**URI**(host.space)和**call-ID**(888)，以及特別關聯的主機**callLeg配置檔案**，如步驟2中設定：

```
uricallIdpasscodecallLegProfile>

   7306d2c1-bc15-4dbf-ab4a-1cbdaabd1912

      callLegProfile>
```

驗證

現在，您可以撥入同一個會議：

- 作為訪客
  — 撥號到guest.space URI（後接呼叫匹配規則中配置的域）

  — 通過IVR或WebRTC加入628821815入呼叫ID值（無密碼）

- 作為主機
  — 撥號到host.space URI（後跟呼叫匹配規則中配置的域）

  — 通過IVR或WebRTC加入輸入call-ID值888（無密碼）

當只有客人加入空間時，他們都會被安置在大廳中，等待主人加入。一旦主機加入，所有訪客和主機都將加入會議。如果空間中不再有主機加入，但仍存在某些訪客，則根據**CallLegProfile**訪客上的**deactivate** on **deactivationMode**引數配置，這些主機將返回到大廳螢幕，如步驟1所示。

2)使用相同URI但非空訪客和主機PIN/密碼的配置

此配置與上一示例中的配置類似，並且在CMS 2.1發行版之前已可用。它要求訪客和主機輸入非空PIN或密碼，以便在它們撥號到同一個URI時能夠進行區**別**。

設定步驟與先前的設定範例非常相似：

1. 建立訪客**callLegProfile**(**needsActivation = true**)
2. 建立主機**callLegProfile**(**needsActivation = false**)
3. 將訪客**callLegProfile**分配給指定訪客密碼(PIN)（預設訪問方法）的現有空間或新空間
4. 使用相同的**URI**(不同的**call-ID**)在同一空間上建立一個新的**accessMethod**，並將包含主機密碼(PIN)的主機**callLegProfile**分配給它

步驟1.建立訪客**callLegProfile**(**needsActivation = true**)。

與先前的範例1相同的組態，甚至可以使用相同的訪客**callLegProfile**(**d4bfe12d-68cd-41c0-a671-48395ee170ab**)，如圖所示。

步驟2.建立主機**callLegProfile**(**needsActivation = false**)

如前例所示，可以使用與上例1相同的配置，甚至使用相同的主機**callLegProfile**(**7306d2c1-bc15-4dbf-ab4a-1cbdaabd1912**)。

步驟3.將訪客**callLegProfile**分配給指定訪客密碼(PIN)（預設訪問方法）的現有空間或新**空間**。

與以前類似，您可以對現有空間(/api/v1/coSpaces/<cospace-ID>)執行PUT操作，也可以執行POST操作，以建立一個新空間(/api/v1/coSpaces)，該新空間具有所需的URI引數、密碼和呼叫ID以及根據API指南第6.2節分配給它的訪客呼叫LegProfile（從步驟1開始）。

如果在該空間上執行GET請求，您必須能夠看到與此類似的輸出，其中您會看到**URI** guestpin.space、**call-ID** 189、我們以前建立的訪客callLegProfile和**passcode**789:

```
uricallId callLegProfile>

   d4bfe12d-68cd-41c0-a671-48395ee170ab

       callLegProfile><

   passcode>789

       passcode>
```

記下以粗體標示的空間ID，因為此操作必須用於在步驟4中的該特定空間上建立**accessMethod**。

步驟4.使用相同的**URI**(不同**call-ID**)在該空間上建立一個新的**accessMethod**，並將包含主機密碼(PIN)的主機**callLegProfile**分配給該空間。

在此空間上，您還為主機建立不同的訪問方法(因為訪客 **callLegProfile**作為預設連線選項分配給空間本身)，與第一個配置示例中的方法相同。這是使用**/api/v1/coSpaces/<coSpace-ID>/accessMethods**上的**POST**命令完成的，該命令中空間的**coSpace-ID**值為**22d9f4ca-8b88-4d11-bba9-e2a2f7428c46**，如上一步中突出顯示。在此POST命令上，您可以提供不同的引數，例如URI（guestpin.space，與原始引數相同）、call-ID(889)、host **callLegProfile**（如步驟2定義）以及host **passcode**或**PIN**（本例中為1234）。

如果對該accessMethod執行GET請求，您必須能夠看到類似的輸出，顯示相同的URIguestpin.space、**call-ID** 889、host callLegProfile引用和host **PIN**1234:

```
uricallId passcode>1234

      passcode><

   callLegProfile>

   7306d2c1-bc15-4dbf-ab4a-1cbdaabd1912

      callLegProfile>
```

驗證

現在，您可以撥入同一個會議：

- 作為訪客
  - 通過撥打guestpin.space **URI**（後跟您的呼叫匹配規則中配置的域）並輸入**PIN** 789
  - 通過IVR**或WebRTC輸入**call-ID值189與**PIN** 789
- 作為主機
  - 通過撥打guestpin.space **URI**（後跟您的呼叫匹配規則中配置的域）並輸入**PIN** 1234
  - 通過IVR**或WebRTC輸入**呼叫ID值889與**PIN** 1234

當只有客人加入空間時，他們都會被安置在大廳中，等待主人加入。一旦主機加入，所有訪客和主機都將加入會議。如果空間中不再有主機加入，但仍存在某些訪客，則根據**CallLegProfile**訪客上的**deactivate** on **deactivationMode**引數配置，這些主機將返回到大廳螢幕，如步驟1所示。

3)使用具有空訪客PIN和非空主機PIN混合的相同URI的配置

從CMS版本2.1開始，由於在**callProfile**部分新新增了passcodeMode和**passcodeTimeout**等一些API命令，此配置才可用。這允許訪客加入的空PIN（輸入#或超時），而主機擁有訪問空間並啟用空間的PIN。**callProfile**控制SIP（包括Lync）呼叫的通話中體驗，因此不適用於CMA客戶端（胖客戶端和WebRTC）。

配置步驟與示例2中的步驟類似，並新增了callProfile:

1. 建立訪客**callLegProfile**(**needsActivation = true**)
2. 建立主機**callLegProfile**(**needsActivation = false**)
3. 使用所需的**passcodeMode**和**passcodeTimeout**配置建立**callProfile**
4. 將步驟3的訪客**callLegProfile**和**callProfile**分配到指定訪客密碼(PIN)的現有空間或新空間（預設訪問方法）
5. 使用相同的**URI**(不同的**call-ID**)在同一空間上建立一個新的**accessMethod**，並將包含主機密碼(PIN)的主機**callLegProfile**分配給它

由於配置與配置示例1和2完全相同，因此有一些參考說明。事實上，測試使用與示例2相同的空間，但現在已與callProfile一起新增。

步驟1.建立訪客**callLegProfile**(**needsActivation = true**)。

如前一個範例1所示，可以使用相同的組態，甚至可以使用相同的訪客callLegProfile(**d4bfe12d-68cd-41c0-a671-48395ee170ab**)。

步驟2.建立主機**callLegProfile**(**needsActivation = false**)。

如前例所示，可以使用與上例1相同的配置，甚至使用相同的主機callLegProfile(**7306d2c1-bc15-4dbf-ab4a-1cbdaabd1912**)。

步驟3.使用所需的**passcodeMode**和**passcodeTimeout**配置建立**callProfile**。

您可以建立確定SIP（包括Lync）呼叫的通話體驗的callProfile。此處有一些可能的設定，例如允許錄製或串流，或允許最大參加者限制，但目前的焦點是來自CMS 2.1的新API新增內容與密碼處理相關。其他引數可在API指南的第8.2節中找到。

有兩個引數決定此處的密碼行為，即：

- *passcodeMode*
  - ***必需***:ivr會永遠等待使用者輸入PIN或#輸入空的PIN（用於訪客）

-逾時:ivr等待參與者輸入PIN的時間為***passcodeTimeout***秒，如果在該時間內沒有輸入任何內容，則假定已輸入空白(#)PIN

- ***密碼超時：僅當***passcodeMode設定為超時時需要設定，並且控制將密碼解釋為空白密碼之前的時間量

要建立callProfile，請對/api/v1/callProfiles執行POST命令(如果要修改現有配置件，請對/api/v1/callProfiles/<callProfile-ID>執行PUT命令)，並使用**passcodeMode**和**passcodeTimeout**所需的引數。如果對該特定**callProfile**執行**GET**命令，則必須獲得類似的結果，例如，已將模式設定為超時和5秒的超時值：

```
passcodeMode>timeout

      passcodeMode><

   passcodeTimeout>5
```

```
        passcodeTimeout>
```

記下**callProfile-ID**（以粗體標示），因為在步驟4中，必須使用此項為空間分配以具有此通話行為。

步驟4.將步驟3的訪客**callLegProfile**和**callProfile**分配到指定訪客密碼(PIN)（預設訪問方法）的現有空間或新空間。

與以前類似，您可以對現有空間(/api/v1/coSpaces/<cospace-ID>)執行PUT操作，也可以執行POST操作以使用所需的URI和**call-ID**引數(例如來自步驟1的訪客**callLegProfile**)建立新空間(/api/v1/coSpaces)。 與前面示例的不同之處在於步驟3中的**callProfile**以及沒有為其分配密碼這一事實。

如果在該空間上執行GET請求，您必須能夠看到類似以下示例的輸出，其中您看到**guestpin.space的URI、call-ID** 189、以前建立的訪客callLegProfile和**callProfile**（如步驟3中所設定）：

```
uricallId callLegProfile>

   d4bfe12d-68cd-41c0-a671-48395ee170ab

        callLegProfile><

   callProfile>

   4b0eff60-e4aa-4303-8646-a7e800a4eac6

        callProfile>
```

請記下以粗體標出的空間ID，因為在步驟5中，必須使用該空間在該特定空間上建立accessMethod。

步驟5.使用相同的**URI**(不同**call-ID**)在同一空間上建立一個新的**accessMethod**，並將包含主機密碼(PIN)的主機**callLegProfile**分配給它。

在此空間上，您還為主機建立不同的訪問方法(因為訪客 **callLegProfile**作為預設連線選項分配給空間本身)，與第一個配置示例中的方法相同。這是使用**/api/v1/coSpaces/<coSpace-ID>/accessMethods**上的**POST**命令完成的，其中**coSpace-ID**值被替換為空間的值**22d9f4ca-8b88-4d11-bba9-e2a2f7428c46**，在此案例的上一步中突出顯示。在此**POST**命令上，您可以提供不同的引數，例如**URI**（guestpin.space，與原始引數相同）、**call-ID**(889)、步驟2中定義的主機**callLegProfile**，以及主機密碼或PIN（本例中為1234）。

如果對該accessMethod執行GET請求，您必須能夠看到類似的輸出，顯示相同的

URIguestpin.space、**call-ID** 889、host callLegProfile引用和host **PIN**1234:

```
uricallId passcode>1234

        passcode><

    callLegProfile>

    7306d2c1-bc15-4dbf-ab4a-1cbdaabd1912

        callLegProfile>
```

驗證

現在，您可以撥入同一個會議：

- 作為訪客
  — 撥打至guestpin.space **URI**（後接您的呼叫匹配規則中配置的域），輸入#作為**PIN**，或使其在5秒後超時

  — 通過IVR或**WebRTC加入**輸入call-ID值189

- 作為主機
  — 通過撥打guestpin.space **URI**（後接呼叫匹配規則中配置的域）並輸入**PIN** 1234

  — **通**過IVR或WebRTC輸入**call-ID值**889，並使用**PIN** 1234

4)主持人使用者是空間的成員，並通過webRTC登入獲得授權，來賓使用者使用callID加入會議。訪客和主機參與者為訪客使用者使用空或非空PIN/密碼使用相同的URI和callID

需要執行後續步驟，以便針對空間的成員和非成員在同一空間上區分訪客/主機的訪問許可權：

1. 建立訪客**callLegProfile**(**needsActivation = true**)
2. 建立主機**callLegProfile**(**needsActivation = false**)
3. 將訪客**callLegProfile**分配給現有或新空間（預設訪問方法）
4. 在同一個空間上使用同一**URI**（和call-ID）建立新的**accessMethod**，並為其分配主機**callLegProfile**
5. 將使用者的ownerJID分配給相同的空間。（如果未分配）
6. 將the ownerIDas成員使用者新增到同一空間，並將assignhostcallLegProfileto該成員使用者

步驟1.建立訪客callLegProfile(needsActivation = true)。

與先前的示例1和訪客callLegProfile(**bfe7d07f-c7cb-4e90-a46e-4811bbaf6978**)相同的配置在本示例

中使用。

記下以粗體標籤的**callLegProfile-ID**，因為必須在步驟3中的空間上為訪客訪問應用此命令。

步驟2.建立主機callLegProfile(needsActivation = false)

此範例中使用與先前範例1相同的組態和主機callLegProfile(**0e76e943-6d90-43df-9f23-7f1985a74639**)。

記下以粗體標籤的**callLegProfile-ID**，因為此操作必須應用於步驟4中主機訪問的**space accessMethod**以及步驟6中的coSpace成員。

步驟3.將訪客**callLegProfile分**配給現有空間或新空間（預設的accessMethod）。

在現有空間(**/api/v1/coSpaces/<coSpace-ID>**)上執行PUT命令以調整該空間，或在**/api/v1/coSpaces**上執行**POST**命令，以使用步驟1中建立的訪客**callLegProfile**引數作為該空間的呼叫內行為，建立新空間。根據API指南第6.2節，您還可以針對該空間設定**URI**和**call-ID**引數。

對該空間執行GET請求(**/api/v1/coSpaces/<coSpace-ID>**)，以驗證訪客**callLegProfile**是否與其關聯，以及**URI**和**call-ID**值。此示例在步驟1中建立的訪客callLegProfile的示例輸出是URI值**global**和**call-ID**為1234（沒有密碼集）的輸出，**nonMemberAccessset**totrue:

```
<?xml version="1.0" ?>
<coSpace id="96d28acb-86c6-478d-b81a-a37ffb0adafc">
        <name>Global</name>
        <autoGenerated>false</autoGenerated>
        <uri>global</uri>
        <callId>1234</callId>
        <callLegProfile>bfe7d07f-c7cb-4e90-a46e-4811bbaf6978</callLegProfile>
        <nonMemberAccess>true</nonMemberAccess>
        <secret>0w4O2zTTF0WdL4ymF8D0_A</secret>
        <defaultLayout>allEqual</defaultLayout>
</coSpace>
```

記下以粗體標示的空間ID，因為此操作必須用於在步驟4中的該特定空間上建立**accessMethod**。

步驟4.使用相同的**URI**(和**call-ID**)在該空間上建立新的**accessMethod**，並為其分配主**機callLegProfile。**

您想要建立與訪客存取不同的存取空間方式（目前是預設的方式）。這是通過**/api/v1/coSpaces/<coSpace-ID>/accessMethods上的POST命令在空間本身上指定accessMethod來完成**的，此處的**coSpace-ID是步驟3中的粗體標籤值(96d28acb-86c6-478d-b81a-a37ffb0adafc)，在此應用了步驟2的主機callLegProfile以及相同的URI**和**CALL-ID來完成欄位。**您可以為通過callID連線的主機新增非空密碼（無需以使用者身份通過webRTC登入）。

在該空間accessMethod(/api/v1/coSpaces/<coSpace-ID>/accessMethods/<accessMethod-ID>)上發出GET請求後，您必須能夠看到類似此輸出的類型，其中可以看到**第2步**上設定的**URI**（全域性）和**call-ID**(1234)以及專門關聯的主機**callLegProfile**和主機密碼**12345:**

```
<?xml version="1.0" ?>
<accessMethod id="c4ecc16e-945f-4e35-ba03-d9b69107b32c">
        <uri>global</uri>
        <callId>1234</callId>
        <passcode>12345</passcode>
        <callLegProfile>0e76e943-6d90-43df-9f23-7f1985a74639</callLegProfile>
```

```
        <secret>kffO1zTTE0feL4fsdf43w_B </secret>
</accessMethod>
```

步驟5.將使用者的所有者Jid分配給空間。（如果未分配）。 通過aPUTcommand on/api/v1/coSpaces/<coSpace-ID>在空間上通過**specifyingownerJid(**user1@evacanoalone.net)將ownerJID新增到空間

在該空間上發出GET請求後，必須能夠看到已為該空間分配了ownerIdand ownerJid:

```
<?xml version="1.0" ?>
<coSpace id="96d28acb-86c6-478d-b81a-a37ffb0adafc">
        <name>Global</name>
        <autoGenerated>false</autoGenerated>
        <uri>global</uri>
        <callId>1234</callId>
        <callLegProfile>bfe7d07f-c7cb-4e90-a46e-4811bbaf6978</callLegProfile>
        <nonMemberAccess>true</nonMemberAccess>
        <ownerId>1d942281-413e-4a2a-b776-91a674c3a5a9</ownerId>
        <ownerJid>user1@evacanoalone.net</ownerJid>
        <secret>0w4O2zTTF0WdL4ymF8D0_A</secret>
        <numAccessMethods>1</numAccessMethods>
        <defaultLayout>allEqual</defaultLayout>
</coSpace>
```

記下所有者ID(**1d942281-413e-4a2a-b776-91a674c3a5a9**)。

步驟6.將步驟5中的ownerID(1d942281-413e-4a2a-b776-91a674c3a5a9)作為成員使用者新增到空間，並向該成員使用者分配signhostcallLegProfile。這是通過在空間本身(通過POST命令(**/api/v1/coSpaces/<coSpaceID>/coSpaceUsers)指定**userJIdandhost callLegProfile)來實現的。**coSpaceUsers**的其他引數可在API指南6.4.2部分找到，在這些部分下，所顯示的引數也可用於此設定：

<canDestroy>true</canDestroy>

<canAddRemoveMember>true</canAddRemoveMember>

<canChangeName>true</canChangeName>

<canChangeUri>false</canChangeUri>

<canChangeCallId>false</canChangeCallId>

<canChangePasscode>true</canChangePasscode>

<canPostMessage>true</canPostMessage>

<canDeleteAllMessages>false</canDeleteAllMessages>

<canRemoveSelf>false</canRemoveSelf>

驗證成員使用者已通過aGET命令(**/api/v1/coSpaces/<coSpaceID>/coSpaceUsers?**)新增到空間。

```
<?xml version="1.0" ?>
<coSpaceUsers total="1">
<coSpaceUser id="1d942281-413e-4a2a-b776-91a674c3a5a9">
<userId>1d942281-413e-4a2a-b776-91a674c3a5a9</userId>
```

```
<userJid>user1@evacanoalone.net</userJid>
<autoGenerated>false</autoGenerated>
</coSpaceUser>
</coSpaceUsers>
```

記下userID（如果表單的ownerID與步驟5不同）。 驗證aGETrequest 指定的coSpaceIDanduserID(**/api/v1/coSpaces/<coSpaceID>/coSpaceUsers/<userID>)已將**callLegProfile分配給coSpaceUser

```
<?xml version="1.0" ?>
<coSpaceUser id="1d942281-413e-4a2a-b776-91a674c3a5a9">




        <autoGenerated>false</autoGenerated>
        <canDestroy>true</canDestroy>
        <canAddRemoveMember>true</canAddRemoveMember>
        <canChangeName>true</canChangeName>
        <canChangeUri>false</canChangeUri>
        <canChangeCallId>false</canChangeCallId>
        <canChangePasscode>true</canChangePasscode>
        <canPostMessage>true</canPostMessage>
        <canDeleteAllMessages>false</canDeleteAllMessages>
        <canRemoveSelf>false</canRemoveSelf>
        <canChangeNonMemberAccessAllowed>true</canChangeNonMemberAccessAllowed>




        0e76e943-6d90-43df-9f23-7f1985a74639
</coSpaceUser>
```

驗證

現在，您可以撥入同一個會議：

- 作為訪客
  — 通過撥打URI（後接呼叫匹配規則中配置的域）

  — 通過IVR或WebRTC加入輸入call-ID值1234（無密碼）

- 作為主機
  通過webRTC以使用者(分配了「host」callLegProfile的空間成員，在此場景中為user1@evacanoalone.net)身份登入並加入空間(「global」URI)。

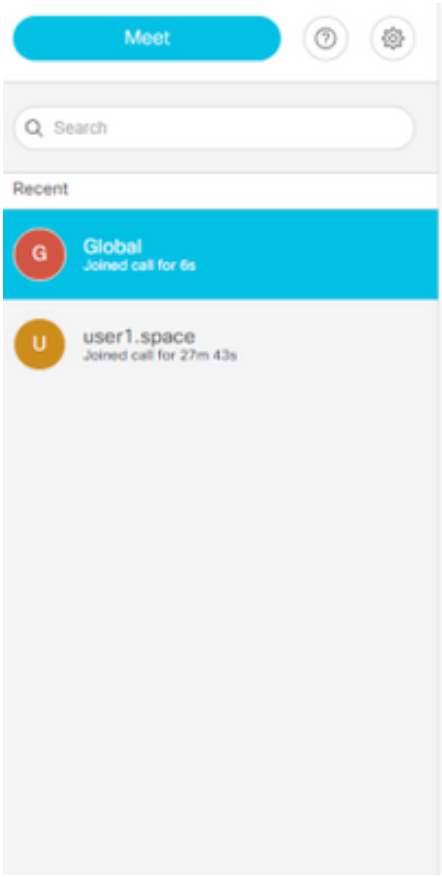
  — 撥打到「全域性」URI（後接在呼叫匹配規則上配置的域）和密碼12345。

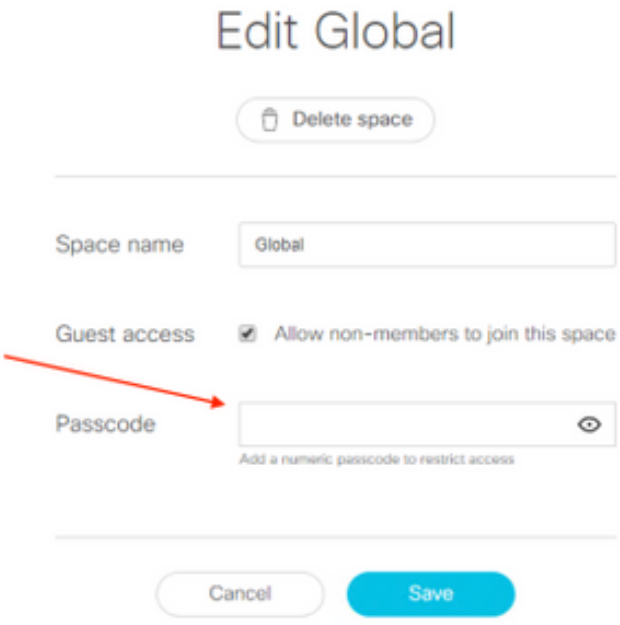
  — 通過IVR或WebRTC加入輸入call-ID值1234(主機密碼為12345)

當只有客人加入空間時，他們都會被安置在大廳中，等待主人加入。一旦主機加入，所有訪客和主機都將加入會議。如果空間中不再有主機加入，但仍存在某些訪客，則根據**CallLegProfile**訪客上的**deactivate** on **deactivationMode**引數配置，這些主機將返回到大廳螢幕，如步驟1所示。

主機（所有者/成員）可以直接在webRTC應用中為訪客設定（編輯/刪除）密碼，或完全禁用非成員

（訪客）訪問空間：





疑難排解

本節提供的資訊可用於對組態進行疑難排解。

當您作為訪客加入或第一台主機加入時，CMS的日誌記錄會短暫顯示我們，但最好使用GET請求呼叫配置檔案以及訪客和主機callLegProfile定義，以及這些定義在各自的accessMethods（或預設訪問方法）或更高級別（全域性級別或租戶級別）上的分配。

您可以按照此結構獲取所有資訊：

1. **/api/v1/callProfiles上的GET**(如果將此與passcodeMode一起**使用**)
   >使用**GET on**/api/v1/callProfiles/<callProfile-ID>詳細驗證所需的callProfile-ID
2. **GET** on**/api/v1/callLegProfiles**
   >使用**GET on /api/v1/callProfiles/<callProfile-ID>**詳細驗證訪客和主機的所需callLegProfile-ID
3. **GET** on **/api/v1/coSpaces**
   >使用**GET on /api/v1/coSpaces/<coSpace-ID>**詳細驗證所需的space-ID
   >檢查所需的callProfile-ID（步驟1）和訪客callLegProfile（步驟2）是否與此空間關聯
   [如果沒有，請檢查不太具體的元素，如租戶(**/api/v1/tenants/<tenant-ID>)或全域性(/api/v1/system/profiles**)級別]
4. **GET** on **/api/v1/coSpaces/<coSpace-ID>/accessMethods**
   >使用**GET** on **/api/v1/coSpaces/<coSpace-ID>/accessMethods/<accessMethod-ID>**詳細驗證所需的accessMethod，以檢查是否分配了主機**callLegProfile**

在本示例中顯示的CMS日誌記錄中，您有前兩個來賓參與者進入(從2000@steven.lab呼叫38，從

1060@steven.lab呼叫*39*)，他們超時到guestpin.space@acano.steven.lab空間，然後主機加入。從代碼片斷中您可以看到，對於訪客，它會通知我們需要對其執行哪些操作（要停用），並且您可以看到當主機(stejanss.movi@steven.lab)在空間上加入時(停止啟用**時)這些呼叫的此行為發生變**化。同樣，一旦空間上不再有主機（將被停用），當訪客再次移動到大廳時，您會再次看到**相同的記**錄。

```
2017-02-21  17:48:54.809  Info      call 38: incoming encrypted SIP call from
"sip:2000@steven.lab" to local URI "sip:guestpin.space@acano.steven.lab" 2017-02-21 17:48:54.822
Info call 38: setting up UDT RTP session for DTLS (combined media and control) 2017-02-21
17:48:54.837 Info call 38: compensating for far end not matching payload types 2017-02-21
17:48:54.847 Info sending prompt response (2) to BFCP message 2017-02-21 17:48:54.847 Info call
38: sending BFCP hello as client following receipt of hello when BFCP not active 2017-02-21
17:48:54.883 Warning call 38: replacing pending BFCP message "PrimitiveHelloAck" with
"PrimitiveHelloAck" 2017-02-21 17:48:54.883 Info call 38: BFCP (client role) now active 2017-02-
21 17:48:59.294 Info call 39: incoming encrypted SIP call from "sip:1060@steven.lab" to local
URI "sip:guestpin.space@acano.steven.lab" 2017-02-21 17:48:59.310 Info call 39: setting up UDT
RTP session for DTLS (combined media and control) 2017-02-21 17:48:59.323 Info call 39:
compensating for far end not matching payload types 2017-02-21 17:48:59.569 Info sending prompt
response (2) to BFCP message 2017-02-21 17:48:59.569 Info call 39: sending BFCP hello as client
following receipt of hello when BFCP not active 2017-02-21 17:48:59.746 Info call 39: BFCP
(client role) now active 2017-02-21 17:49:07.971 Info configuring call e2264fb0-483f-45bc-a4f3-
5a4ce326e72c to be deactivated
2017-02-21  17:49:07.972  Info      participant "2000@steven.lab" joined space 22d9f4ca-8b88-
4d11-bba9-e2a2f7428c46 (Guest/Host PIN)
2017-02-21  17:49:12.463  Info      configuring call b1b5d433-5ab5-49e1-9ae3-3f4f71703d1b to be
deactivated
2017-02-21  17:49:12.463  Info      participant "1060@steven.lab" joined space 22d9f4ca-8b88-
4d11-bba9-e2a2f7428c46 (Guest/Host PIN)
2017-02-21  17:49:12.463  Info      conference "Guest/Host PIN": unencrypted call legs now
present
2017-02-21  17:49:16.872  Info      call 40: incoming encrypted SIP call from
"sip:stejanss.movi@steven.lab" to local URI "sip:guestpin.space@acano.steven.lab" 2017-02-21
17:49:16.885 Info call 40: setting up UDT RTP session for DTLS (combined media and control)
2017-02-21 17:49:24.260 Info call 40: audio prompt play time out 2017-02-21 17:49:26.670 Info
participant "stejanss.movi@steven.lab" joined space 22d9f4ca-8b88-4d11-bba9-e2a2f7428c46
(Guest/Host PIN)
2017-02-21  17:49:26.670  Info      call e2264fb0-483f-45bc-a4f3-5a4ce326e72c ceasing to be
deactivated
2017-02-21  17:49:26.670  Info      call b1b5d433-5ab5-49e1-9ae3-3f4f71703d1b ceasing to be
deactivated
2017-02-21  17:49:30.832  Info      call 40: ending; remote SIP teardown - connected for 0:14
2017-02-21  17:49:30.833  Info      participant "stejanss.movi@steven.lab" left space 22d9f4ca-
8b88-4d11-bba9-e2a2f7428c46 (Guest/Host PIN)
2017-02-21  17:49:30.833  Info      configuring call e2264fb0-483f-45bc-a4f3-5a4ce326e72c to be
deactivated
2017-02-21  17:49:30.833  Info      configuring call b1b5d433-5ab5-49e1-9ae3-3f4f71703d1b to be
deactivated
```

相關資訊

- 技術支援與文件 - Cisco Systems
- CMS文檔