CMS スペース上のゲストおよびホスト アクセスの設定とトラブルシューティング

内容

概要

このドキュメントでは、API コマンドを使用して Cisco Meeting Server（CMS）のスペース上のゲストおよびホスト アクセスをセットアップする方法について説明します。

前提条件

要件

次の項目に関する知識があることが推奨されます。

- スペースがセットアップされ、そのスペースにコールを行うことができる Cisco Meeting Server（CMS）
- APIクライアント（Poster、Postmanなど）または
- CMS API ガイド

使用するコンポーネント

このドキュメントの情報は CMS バージョン 2.1 に基づいています。

このドキュメントの情報は、特定のラボ環境にあるデバイスに基づいて作成されたものです。本稼働中のネットワークでは、各コマンドによって起こる可能性がある影響を十分確認してください。

背景説明

このドキュメントでは、次のタイプのシナリオを示します。

- ゲストとホストの参加者によって、異なる URI または call-ID が使用される。
- ゲストとホストの参加者によって同じ URI が使用される（PIN またはパスコード エントリ（どちらも空ではない）に基づいて差別化が行われる）
- ゲストとホストの参加者によって同じ URI が使用される（PIN またはパスコード エントリ（空と空以外の混在）に基づいて差別化が行われる）
- ホストユーザはスペースのメンバーであり、webRTCログインを介して許可されます。ゲストユーザはcallIDを使用して会議に参加します。ゲストユーザとホストの参加者が同じURIとcallIDを使用し、ゲストユーザには空または空でないPIN/パスコードを使用する

設定

CMSのゲストとホストの参加者間の差別化には、次の4つの例で説明します。主に、スペースに参加する参加者のコール中の動作を決定する異なる**callLegProfilesに基づいて**います。

まず、ゲストとホストの参加者に異なる**URI**（またはコールID）を使用する方法について説明し、その後、同じ**URIで異なるパスコード（またはタイムアウト）を使用して追加され、ゲストとホストの参加者を区別します。**ゲストユーザのタイムアウトまたは空のPINエントリの3番目の方法は、リリースノートのセクション2.4に示すように、CMS 2.1の新機能として導入さ[れまし]た。4つ目の方法では、割り当てられた所有者/メンバーを持つスペースでゲストおよびホストアクセスを設定し、スペースのメンバー（所有者）をスペースのホストにする方法について説明します。

1)異なるURIを使用した設定

これは、CMS 2.1 リリースの前までの基本設定であり、異なる call-ID を使用する方法と同じです。同じスペースでゲスト/ホストアクセスを差別化するには、次の手順を実行する必要があります。

1. ゲスト **callLegProfile（needsActivation = true**）を作成します。
2. ホスト **callLegProfile（needsActivation = false**）を作成します。
3. ゲスト **callLegProfile を既存のスペースまたは新しいスペースに割り当てます（デフォルトのアクセス方式）。**
4. 同じスペースに対して異なる **URI（および call-ID）で新しい accessMethod を作成し、これにホスト callLegProfile を割り当てます。**

ステップ 1： ゲスト **callLegProfile（needsActivation = true**）を作成します。

callLegProfileはコール中の動作を決定します。デフォルトでは、ゲストのコール中の動作をスペースに割り当てるため、後で同じスペースに別のアクセス方法を割り当て、ホストが参加できるようになります。

注：また、これをテナント レベル（**/api/v1/tenants/<tenant-ID>**）またはシステム レベル（**/api/v1/system/profiles**）で割り当てて、たとえばこれをすべてのスペースに（またはテナント単位で）割り当てることができます。ただし、ここではこれはこのスペース上に示されます。callLegProfileの最も具体的な割り当てが通話中の動作に**対して考慮**してください。

needsActivation パラメータは、ゲスト/ホストの動作に関して最も重要です。これは、このパラメータを true に設定すると、参加者は 1 人以上の**フル/アクティベータ（ホスト）参加者が参加するまで、音声およびビデオを受信または提供できないためです。callLegProfile**のその他のパラメータは、APIガイドのセクション8.4.3に記載されていますす。このセクションで示すパラメータが、この設定にも関連しています（要件に応じて）。

- presentationContributionAllowed
- rxAudioMute
- rxVideoMute
- deactivationMode（deactive |切断 | remainActivated)およびdeactivationModeTime [最後のアクティベータがコールを終了するときに実行するアクション]

guest callLegProfileを作成するには、優先するパラメータを指定して/api/v1/callLegProfilesにPOST要求を行い、needsActivationパラメータをtrueに設定し、その後に次の例を実行します。

```
needsActivation>true


      needsActivation>


      < deactivationMode>deactivate deactivationMode>
```

太字で示されている**callLegProfile-IDをメモします**。これは、（デフォルトの）ゲストアクセスのステップ3のスペースに適用する必要があります。

ステップ 2： ホスト **callLegProfile（needsActivation = false**）を作成します。

同様に、ホストのコール中の動作用のホスト **callLegProfile を作成します。**前述と同じパラメータが適用されますが、パラメータはユーザ独自のプリファレンスと要件に従って選択できます。ここでの主要な要素は、**needsActivation パラメータを false に設定して、ホスト ロールを付与することです。**

これは、優先するパラメータを持つ/api/v1/callLegProfiles上のPOST要求で作成します。Activationパラメータを**falseに設定する必要があります**。その後、次の例で**す。**

```
needsActivation>

   false


      needsActivation>
```

太字で示されている**callLegProfile-IDをメモします。**これは、ホストのアクセスに対してステップ4のスペース**accessMethod**に適用する必要があります。

ステップ 3： ゲスト **callLegProfile を既存のスペースまたは新しいスペースに割り当てます（デフォルトの** accessMethod ）。

既存のスペース（ **/api/v1/coSpaces/<coSpace-ID>** ）**に対して PUT コマンド**を実行してスペースを調整するか、またはステップ 1 でこのスペースでのコール中の動作として作成したゲスト **callLegProfile パラメータを使用して、/api/v1/coSpaces に対して POST コマンドを実行して新しいスペースを作成します。**また、必要に応じて、API ガイドのセクション 6.2 に従って、このスペースに対して **URI、passcode、call-ID** パラメータを設定することもできます。

このスペース（ **/api/v1/coSpaces/<coSpace-ID>** ）**に対して GET** 要求を実行して、ゲスト **callLegProfile が、URI および call-ID 値とともにこのスペースに関連付けられていることを確認します。**この例でステップ1で作成したゲスト**callLegProfile**の出力例は、**URI**値**guest.space**と**call-ID** 628821815（パスコード設定なし）を含んだものです。

```
uricallId callLegProfile>

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

      callLegProfile>
```

太字で示されているスペースIDをメモしてください。これは、ステップ4でその特定のスペースに**accessMethodを作成する**ために使用する必要があります。

ステップ 4： 同じスペースに対して異なる **URI（および call-ID ）で新しい** accessMethod を作成し、これにホスト **callLegProfile を割り当てます。**

スペースへのアクセス方法を、現在デフォルトのゲストアクセス方法とは異なる方法で作成する必要があります。これを行うには、スペース自体のaccessMethodを/api/v1/coSpaces/<coSpace-ID>/accessMethodsでPOSTコマンドを使用して指定します。coSpace-IDは手順3 (7cc797c9-c0a8-47b) 519-8dc5a01f1ade)で、ステップ2のホストcallLegProfileと、異なるURIおよび**call-ID**(RFC 2318)フィールドが適用されます。

そのspace accessMethodに対するGET要求(/api/v1/coSpaces/<coSpace-ID>/accessMethods/<accessMethod-ID>)の後に、異なる**URI** (host.space)とcall-IDを確認できなければなりません(888)は、スペースのデフォルトの**accessMethod**ではなく、ステップ2で設定した特別に関連付けられたホスト**callLegProfile**です。

```
uricallIdpasscode callLegProfile>

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

      callLegProfile>
```

確認

ここで、同じ会議にダイヤルすることができます。

- ゲストとして
  - guest.space URI（その後にコール一致ルールで設定されているドメインが続く）にダイヤル

  - IVR または WebRTC 参加を介して call-ID 値 628821815 を入力（パスコードなし）

- ホストとして
  - host.space URI（その後にコール一致ルールで設定されているドメインが続く）にダイヤル

  - IVR または WebRTC 参加を介して call-ID 値 888 を入力（パスコードなし）

スペースに参加しているのがゲストのみである場合、それらはすべて ロビー ルームに配置され、ホストの参加を待機します。ホストが参加すると、すべてのゲストとホストが会議に参加します。スペースに参加しているホストがまだいくつかのゲストに存在しない場合は、**deactivationMode**パラメータのゲストcallLegProfileに対するdeactivateの設定に従って、ロビー画面に戻ります。

2)同じURIを使用し、空でないゲストとホストPIN/パスコードを使用した設定

この設定は前の例の設定と同様で、CMS 2.1リリース以前のリリースでも使用可能です。この設定では、ゲストとホストの両方が空ではない PIN またはパスコードを入力することで、両者が同じ **URI** にダイヤルするときにそれに基づいて差別化ができるようにする必要があります。

設定手順は、前の設定例とよく似ています。

1. ゲスト **callLegProfile（needsActivation = true**）を作成します。
2. ホスト **callLegProfile（needsActivation = false**）を作成します。
3. ゲスト **callLegProfile を既存のスペースまたは新しいスペースに割り当て、ゲスト パスコード（PIN）を指定します（デフォルトのアクセス方式）。**
4. 同じスペースに対して同じ **URI（異なる call-ID）で新しい** accessMethod を作成し、これにホスト **callLegProfile（ホスト パスコード（PIN）を含む）を割り当てます。**

ステップ 1： ゲスト **callLegProfile（needsActivation = true**）を作成します。

前の例1と同じ設定で、同じゲスト**callLegProfile(d4bfe12d-68cd-41c0-a671-48395ee170ab)**も使用できます。

ステップ 2： ホスト **callLegProfile（needsActivation = false**）を作成します。

前の例1と同じ設定で、同じホスト**callLegProfile(7306d2c1-bc15-4dbf-ab4a-1cbdaabd1912)**も使用できます。

ステップ 3： ゲスト **callLegProfile を既存のスペースまたは新しいスペースに割り当て、ゲスト**

パスコード（PIN）を指定します（デフォルトの accessMethod）。

同様に、既存のスペースに対してPUT操作(/api/v1/coSpaces/<cospace-ID>)またはPOST操作を実行して、URI、パスコード、call-IDなどの目的のパラメータを含む新しいスペース(/api/v1/coSpacesSpaces)をIDなどのコールををにを追加しますAPIガイドのセクション6.2に従って割り当てたLegProfile（ステップ1から）。

このスペースでGET要求を実行する場合は、guestpin.spaceの**URI、call-ID**189、以前に作成したゲストコールレッグプロファイル、**789のパスコードが表示される同様の出力が表示されます。**

```
uricallId callLegProfile>

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

       callLegProfile><

   passcode>789

       passcode>
```

太字で示されているスペースIDをメモしてください。これは、ステップ4でその特定のスペースに**accessMethodを作成する**ために使用する必要があります。

ステップ 4： このスペースに対して同じ **URI（異なる call-ID）で新しい accessMethod** を作成し、これにホスト **callLegProfile（ホスト パスコード（PIN）を含む）を割り当てます。**

最初の設定例のように、このスペースに対してホスト用の異なるアクセス方式を作成します（ゲスト callLegProfile はデフォルト参加オプションとしてこのスペースに割り当てられています）。そのためには、**/api/v1/coSpaces/<coSpace-ID>/accessMethods に対する POST コマンドを使用します。ここで coSpace-ID 値は、前述の例で強調表示されているスペース 22d9f4ca-8b88-4d11-bba9-e2a2f7428c46 です。**このPOSTコマンドでは、URI（guestpin.space、元のURIと同じ）、call-ID(889)、ステップ2で定義したホスト**callLegProfile**、パスコード(**1234)などの異なるパラメータをパラメータを指定できます..**

そのaccessMethodでGET要求を実行する場合は、guestpin.spaceの同じURI、**call-ID**889、ホストの**callLegProfile参**照、ホストの**PIN**1234:

```
uricallId passcode>1234

       passcode><
```

```
callLegProfile>

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

    callLegProfile>
```

確認

ここで、同じ会議にダイヤルすることができます。

- ゲストとして
  - guestpin.space **URI**（コールマッチングルールで設定されているドメインに続く）にダイヤルし、**PIN 789を入力**する

  - **PIN 789 での IVR または WebRTC 参加を介して call-ID 値 189 を入力**

- ホストとして
  - guestpin.space **URI**（コールマッチングルールで設定されているドメインに続く）にダイヤルし、**PIN 1234を入力する。**

  - **PIN 1234 での IVR または WebRTC 参加を介して call-ID 値 889 を入力**

スペースに参加しているのがゲストのみである場合、それらはすべて**ロビー** ルームに配置され、ホストの参加を待機します。ホストが参加すると、すべてのゲストとホストが会議に参加します。スペースに参加しているホストがまだいくつかのゲストに存在しない場合は、**deactivationMode**パラメータのゲストcallLegProfileに対するdeactivateの設定に従って、**ロビー画面に戻ります。**

3)空のゲストPINと空でないホストPINが混在する同じURIを使用した設定

この設定は、新しく追加された **callProfileセクションの passcodeMode および passcodeTimeout の API コマンドのため、CMS バージョン 2.1 以降のみで使用可能です。**この設定では、ゲストは空の PIN（# の入力またはタイムアウト）で参加でき、ホストはスペースにアクセスしてアクティベートするための PIN を持ちます。**callProfile は、SIP（Lync を含む）コールのコール中のエクスペリエンスを管理し、したがって** CMA クライアント（シック クライアントと WebRTC の両方）には適用されません。

設定手順は例 2 に似ていますが、ここでは callProfile を追加します。

1. ゲスト **callLegProfile（needsActivation = true**）を作成します。
2. ホスト **callLegProfile（needsActivation = false**）を作成します。
3. 目的の **passcodeMode と passcodeTimeout 設定を使用して callProfile を作成します。**
4. ゲスト **callLegProfile とステップ 3 の callProfile を既存のスペースまたは新しいスペースに割り当て、ゲスト パスコード（PIN）を指定します（デフォルトのアクセス方式）。**
5. 同じスペースに対して同じ **URI（異なる call-ID）で新しい accessMethod** を作成し、これにホスト **callLegProfile（ホスト パスコード（PIN）を含む）を割り当てます。**

設定は設定例1および2と完全に同じであるため、これらの設定への参照があります。実際、テストでは、例2と同じスペースが使用されましたが、callProfileとともに追加されました。

ステップ 1： ゲスト **callLegProfile (needsActivation = true)** を作成します。

前の例1と同じ設定で、同じゲストcallLegProfile(**d4bfe12d-68cd-41c0-a671-48395ee170ab**)も使用できます。

ステップ 2： ホスト **callLegProfile (needsActivation = false)** を作成します。

前の例1と同じ設定で、同じホストcallLegProfile(**7306d2c1-bc15-4dbf-ab4a-1cbdaabd1912**)も使用できます。

ステップ 3： 目的の **passcodeMode と passcodeTimeout 設定を使用して callProfile を作成します。**

SIP (Lync を含む) コールのコール中のエクスペリエンスを決定する callProfile を作成することができます。録音、ストリーミング、参加者の最大数の制限などの複数の設定が可能ですが、ここでは、CMS 2.1 から追加されたパスコード処理に関連する新しい API について説明します。その他のパラメータは、API ガイドのセクション 8.2 で確認できます。

パスコードの動作は、次の 2 つのパラメータによって決定されます。

- *passcodeMode*
  - **required**：IVRは、ユーザが空のPIN (ゲストの場合) を入力するまで永続的に待機します

  - **timeout**：IVRは、参加者がPINを入力するまで*passcodeTimeout*秒数を待機し、その時間内にエントリが入力されなかった場合は、空(#)のPINが入力されたと見なされます

- ***passcodeTimeout :*** **passcodeMode を timeout に設定した場合にのみ設定する必要があり、パスコードを空として解釈するまでの時間を制御します。**

callProfile を作成するには、必要な **passcodeMode および passcodeTimeout パラメータを指定して、/api/v1/callProfiles に対して POST コマンド (既存のプロファイルを変更する場合は /api/v1/callProfiles/<callProfile-ID> に対して PUT コマンド) を実行します。GET コマンドをその特定の callProfile** に対して実行すると、次のように、モードを timeout として、タイムアウト値を 5 秒として設定した結果を取得できます。

```
passcodeMode>timeout

      passcodeMode><

   passcodeTimeout>5

      passcodeTimeout>
```

太字で示されている**callProfile-ID**をメモします。これは、ステップ4でこのコール中の動作を行う

ためにスペースに割り当てるために使用する必要があるためです。

ステップ 4： ゲスト callLegProfile とステップ 3 の callProfile を既存のスペースまたは新しいスペースに割り当て、ゲスト パスコード（PIN）を指定します（デフォルトのアクセス方式）。

前と同様に、既存のスペース（/api/v1/coSpaces/<cospace-ID>）に対して PUT 操作を実行するか、または POST 操作を実行して、URI、call-ID などの必要なパラメータ、およびゲスト callLegProfile（ステップ 1 から）を使用して新しいスペース（/api/v1/coSpaces）を作成できます。 前の例との違いは、ステップ 3 からの callProfile、およびパスコードが割り当てられないことです。

そのスペースでGET要求を実行する場合は、次の例のような出力が表示される必要があります。guestpin.spaceのURI、call-ID189、以前に作成したゲストコールプロファイルLeg、およびProfile

```
uricallIdcallLegProfile>

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

       callLegProfile><

   callProfile>

   4b0eff60-e4aa-4303-8646-a7e800a4eac6

       callProfile>
```

太字で示されているスペースIDをメモしてください。これは、ステップ5で特定のスペースにaccessMethodを作成するために使用する必要があります。

ステップ 5： 同じスペースに対して同じ URI（異なる call-ID）で新しい accessMethod を作成し、これにホスト callLegProfile（ホスト パスコード（PIN）を含む）を割り当てます。

最初の設定例のように、このスペースに対してホスト用の異なるアクセス方式（ゲスト callLegProfile はデフォルト参加オプションとしてこのスペースに割り当てられています）。そのためには、/api/v1/coSpaces/<coSpace-ID>/accessMethods に対する POST コマンドを使用します。ここで coSpace-ID 値は、前のステップで強調表示されているスペース 22d9f4ca-8b88-4d11-bba9-e2a2f7428c46 に対する値で置き換えられます。このPOSTコマンドでは、URI(元のURIと同じguestpin.space)、call-ID(889)、host callLegProfile(この場合は1234)などの異なるパラメータを指定できます。

そのaccessMethodでGET要求を実行する場合は、guestpin.spaceの同じURI、call-ID889、ホストのcallLegProfile参照、ホストのPIN1234:

```
uricallIdpasscode>1234
```

```
    passcode><

callLegProfile>

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

    callLegProfile>
```

確認

ここで、同じ会議にダイヤルすることができます。

- ゲストとして
  - guestpin.space **URI**（コールマッチングルールで設定されているドメインに続く）にダイヤルし、**PINとして#を入力するか、**5秒後にタイムアウトします
  - IVR または WebRTC 参加を介して **call-ID 値 189 を入力**
- ホストとして
  - guestpin.space **URI**（コールマッチングルールで設定されているドメインの後に続く）にダイヤルし、**PIN 1234を入**力
  - **PIN** 1234 での IVR または WebRTC 参加を介して call-ID 値 889 を入力

4)ホストユーザがスペースのメンバーであり、webRTCログインを介して許可され、ゲストユーザがcallIDで会議に参加します。ゲストユーザとホストの参加者が同じURIとcallIDを使用し、ゲストユーザには空または空でないPIN//パスコードを使用する

次の手順を実行して、スペースのメンバーと非メンバーの同じスペースでゲスト/ホストアクセスを区別する必要があります。

1. ゲスト callLegProfile（**needsActivation = true）を作成します。**
2. ホスト callLegProfile（**needsActivation = false）を作成します。**
3. ゲスト callLegProfile を既存のスペースまたは新しいスペースに割り当てます（デフォルトのアクセス方式）。
4. 同じスペースに同**じURI(およびcall-ID**)で新しいaccessMethodを作成し、ホストcallLegProfileを割り当**てます**
5. ユーザのsownerJIDを同じスペースに割り当てます。（割り当てられていない場合）
6. メンバーユーザーとしてownerIDを同じスペースに追加し、そのメンバーユーザーにhostcallLegProfileを割り当てます

ステップ 1： ゲスト callLegProfile（needsActivation = true）を作成します。

この例では、前の例1と同じ設定を使用し、ゲストcallLegProfile(**bfe7d07f-c7cb-4e90-a46e-4811bbaf6978**)を使用します。

太字で示されている**callLegProfile-ID**をメモしてください。これは、ステップ3でゲストアクセス用にスペースに適用する必要があります。

ステップ 2： ホスト callLegProfile（needsActivation = false）を作成します。

この例では、前の例1とホストcallLegProfile(**0e76e943-6d90-43df-9f23-7f1985a74639**)と同じ設定を使用しています。

太字で示されている**callLegProfile-IDをメモします**。これは、ステップ4のホストアクセス用のスペース**accessMethod**と、ステップ6のcoSpaceメンバに適用する必要があります。

ステップ 3： ゲスト callLegProfile を既存のスペースまたは新しいスペースに割り当てます（デフォルトの accessMethod）。

既存のスペースでPUTコマンド(**/api/v1/coSpaces/<coSpace-ID>**)を実行してスペースを調整するか、**/api/v1/coSpacesでPOSTコマンドを実行し**て、ステップ1で作成したゲストコールLeg Profileパラメータをでそのスペースに対してまた、APIガイドのセクション6.2に従って、そのスペースの**URIとcall-IDのパラメータを必要に応じて設定すること**[もできま](#)す。

そのスペース(**/api/v1/coSpaces/<coSpace-ID>**)でGET要求を実行し、ゲスト**callLegProfileが関連付けられていることを確認し、URIとcall-IDの値を確認します**。この例でステップ1で作成したゲストcallLegProfileの出力例は、**global**値と**call-ID**値1234（パスコード設定なし）、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>
```

太字で示されているスペースIDをメモしてください。これは、ステップ4でその特定のスペースに**accessMethodを作成する**ために使用する必要があります。

ステップ4：同じ**URI**(および**call-ID**)を持つスペースに新しい**accessMethod**を作成し、ホストcallLegProfileを割り当てます。

スペースへのアクセス方法を、現在デフォルトのゲストアクセス方法とは異なる方法で作成する必要があります。これを行うには、スペース自体のaccessMethodをPOSTコマンドで指定します。**/api/v1/coSpaces/<coSpace-ID>/accessMethodsで、coSpace-IDを太字で示した手順3(96d28acb-86668c6-47) d-b81a-a37ffb0adafc)で、ステップ2のホストcallLegProfileと同じURI**および**call-ID**フィールドが適用されます。callIDで接続するホストに空でないパスコードを追加できます（webRTC経由でユーザとしてログインする必要はありません）。

そのspace accessMethodに対するGET要求(/api/v1/coSpaces/<coSpace-ID>/accessMethods/<accessMethod-ID>)の後に、同じ**URI（グローバル）とcall-ID**()が表示されます1234)、および特別に関連付けられたホスト**callLegProfile**(ステップ2**で設定し、ホストパスコー**

ド(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>
```

ステップ5：スペースにユーザのownerJididを割り当てます。（割り当てられていない場合）。**ownerJid** (user1@evacanoalone.net)をスペースに指定して、**ownerJIDをスペースに追加します。このスペースには、/api/v1/coSpaces/<coSpace-ID>のaPUTコマンドを使用します**

そのスペースに対するGET要求の後、ownerIdand ownerJidare assigned to the space:

```
<?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>
```

ownerID (**1d942281-413e-4a2a-b776-91a674c3a5a9**)をメモします。

ステップ6:ownerID(1d942281-413e-4a2a-b776-91a674c3a5a9)をステップ5からスペースに追加し、そのメンバユーザにhostcallLegProfileを割り当て当て当てます。これを行うには、スペース自体(specifyingcoSpaceID)を指定すると、スペース自体(specifyingcoSpaceID)を指定します(**/api/v1/coSpaces/<coSpaceID>/coSpace Users)。** coSpaceUsersは、APIガイドのセクション6.4.2に記載されています。このガイドでは、このセットアップで次の項目を関連させることができます。

```
<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>

メンバーのユーザーがaGETcommand (/api/v1/coSpaces/<coSpaceID>/coSpaceUsers?)によってスペースに追加されたことを確認します

```
<?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>
```

userIDをメモします（異なるフォーム所有者IDの場合はステップ5）。 callLegProfileがcoSpaceUserに割り当てられていることが、GETrequestによってcoSpaceIDとuserID (/api/v1/coSpaces/<coSpaceID>/coSpaceUsers/ <userID>)

```
<?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>
```

確認

ここで、同じ会議にダイヤルすることができます。

- ゲストとして
  - URIにダイヤルする（その後にコールマッチングルールで設定したドメイン）

  - IVR または WebRTC 参加を介して call-ID 値 1234 を入力（パスコードなし）

- ホストとして
  webRTCを介してユーザ(このシナリオではuser1@evacanoalone.netを使用して「ホスト」callLegProfileが割り当てられたスペースのメンバー)としてログインし、スペース（「グローバル

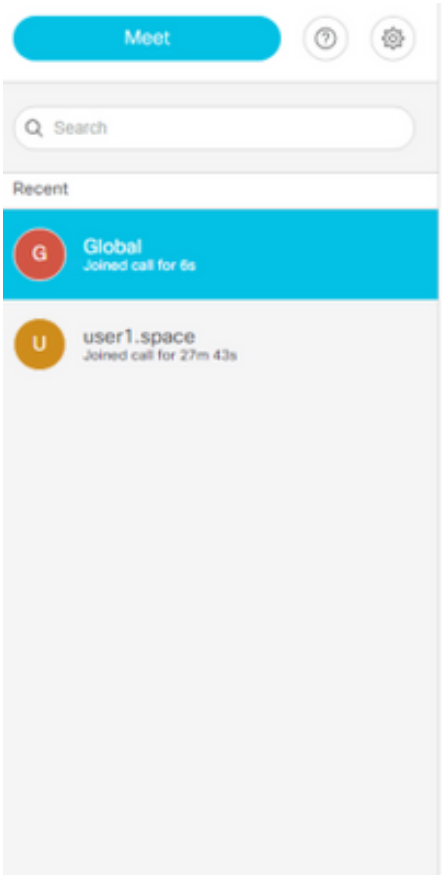
」URI）に参加します。

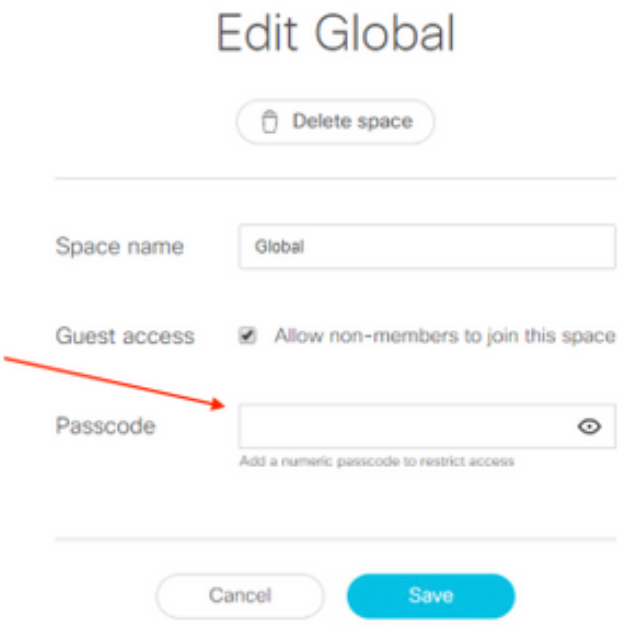
- 「グローバル」URI（続いて、コールマッチングルールで設定されているドメイン）とパスコード12345にダイヤルします。

- IVRまたはWebRTC参加を介してcall-ID値1234を入力（ホストパスコード12345を使用）

スペースに参加しているのがゲストのみである場合、それらはすべてロビー ルームに配置され、ホストの参加を待機します。ホストが参加すると、すべてのゲストとホストが会議に参加します。スペースに参加しているホストがまだいくつかのゲストに存在しない場合は、**deactivationMode**パラメータのゲストcallLegProfileに対するdeactivateの設定に従って、ロビー画面に戻ります。

ホスト（所有者/メンバー）は、WebRTCアプリでゲストのパスワードを直接設定（編集/削除）するか、スペースの非メンバー（ゲスト）アクセスを完全に無効にすることができます。





トラブルシュート

ここでは、設定のトラブルシューティングに使用できる情報を示します。

CMS のログには、いつゲストとして参加したか、またはいつ最初のホストが参加したかが簡潔に示されますが、最もよい方法は、callProfile およびゲストとホストの callLegProfile 定義に対する GET 要求を使用して、対応する accessMethod（またはデフォルトのアクセス方式）またはより高いレベル（グローバル レベルまたはテナント レベル）でのそれらの割り当てを確認することです。

次の構造に従って、すべての情報を取得できます。

1. **/api/v1/callProfiles** に対する GET（これを **passcodeMode** とともに使用している場合）
   >/api/v1/callProfiles/<callProfile-ID>で**GET**を使用して、**目的のcallProfile-IDを詳細に確認します**
2. **/api/v1/callLegProfiles に対する GET**
   >/api/v1/callProfiles/<callProfile-ID>で**GET**を使用して、ゲストおよびホストの目的のcallLegProfile-IDを詳細に確認します
3. **/api/v1/coSpaces に対する GET**
   >/api/v1/coSpaces/<coSpace-ID>の**GET**を使用して、**必要なスペースIDを詳細に確認します**
   >目的のcallProfile-ID（ステップ1）とゲストcallLegProfile（ステップ2）がこのスペースに関連付けられているかどうかを確認します
   （これが存在しない場合は、テナント（**/api/v1/tenants/<tenant-ID>**）やグローバル（**/api/v1/system/profiles**）レベルのような特定性のより低い要素を確認）
4. **/api/v1/coSpaces/<coSpace-ID>/accessMethods に対する GET**
   >/api/v1/coSpaces/<coSpace-ID>/accessMethods/<accessMethod-ID>の**GET**を使用して、ホストcallLegProfileが割り当てられているかどうかを詳細に確認**します**

この例に示すCMSログギングでは、最初に2人のゲスト参加者(2000@steven.labから*38*に、1060@steven.labから*39*にコール)が参加し、ゲストがguestpin.space@acano.steven.labスペースにタイムアウトしてホストが参加します。このスニペットでは、ゲストについて（**非アクティブにするために**）実行する必要があること、およびホスト（stejanss.movi@steven.lab）がスペースに参加したときのこれらのコール変更に対する動作（**非アクティブ化の中止**）を確認できます。 同様に、ホストがスペースに存在しなくなったと同時に（**非アクティブにするために**）ゲストがロビーに再び移動したときに、同じログをもう一度確認できます。

```
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
```

関連情報

- テクニカル サポートとドキュメント – Cisco Systems
- CMSドキュメント