Collaboration Conferencing

Deploying Certificates Cisco Meeting Server

Practical Guidelines for Projects Implementation

Design your certificates for CMS services
and integrate with Cisco UCM Expressway and TMS

Redouane MEDDANE

Lulu Press, Inc

Deploying Certificates Cisco Meeting Server

Design your certificates for CMS services
and integrate with Cisco UCM Expressway and TMS

Redouane MEDDANE 3xCCNP Collaboration, Security and Enterprise

Practical Guideline for projects implementation

Mastering a topic takes effort
Demystifying and Simplifying
takes a lot of effort
but it is worth it.

Lulu Press, Inc
Morrisville, North Carolina

Deploying Certificates Cisco Meeting Server

Design your certificates for CMS services and integrate with Cisco UCM Expressway and TMS

Published by:
Lulu Press, Inc
Morrisville, North Carolina

ISBN : 978-1-387-71581-7

Description:

Cisco Meeting Server brings premises-based video, audio, and web communication together to provide effective collaboration and to provide an intuitive conferencing solution for users.

The Cisco Meeting Server is very secure, the services and applications running on the node use the TLS protocol for communication. TLS allows Cisco Meeting Server nodes to exchange digital certificates and public keys in order authenticate the other entity, exchange symmetric encryption algorithms and the encrypted key to encrypt data transmitted between the entities.

This book is a series of projects that helps you as a reference to deploy cisco meeting server with the appropriate digital certificates in real environment either in new deployment or an extension for an existing deployment.

Goals and Methods:

The idea behind this book is to provide a dedicated guideline for your real deployments and projects, the goal is to explain with practice how to create and install digital certificates for different deployment types "single combined and Resilient & Scalable deployment"

How to create a single Multi-SAN certificate where the same certificate can be deployed on multiple servers and multiple services. and different certificates where each service requires its own certificate, for WebAdmin, WebBridge, CallBridge and Database services, starting with the appropriate CSR (Certificate Signing Request), how to populate the CN (Common Name) and SAN (Subject Alternative Name) attributes with appropriate values in order to successfully enable cisco CMS Services and to ensure successful SSL Handshake negotiation. Also, how to prepare the certificates for Streamer, recorder, and scheduler services.

In addition, you will learn in depth the call routing logic on Cisco Meeting Server with the main tables -Incoming Rules- Outbound Rules- Forwarding Rules with chart call flow.

Cisco Meeting Server profiles are covered to determine functionality for groups in the system for users, calls, and call legs.

How to integrate the Cisco CMS with different cisco collaborations components such as:

- Cisco Unified Communication Manager with appropriate Dial Plan to allow telepresence systems or Cisco Jabber to join a meeting hosted on Cisco Meeting Server.
- Cisco Expressway Series to provide WebRTC connection for external user in an easy and secure fashion.
- integration with the Cisco Meeting Management and Cisco Telepresence Management Suite to manage and schedule conferences.

The Recorder and Scheduler services are covered with detailed implementation to allow users to record and schedule meeting through the spaces, Streamer integration with Wowza Streaming Engine for live streaming, integration of Cisco CMS as AdHoc conference bridge.

Access Method and coSpaces users are explained to allow customers to create different roles and rights for different users when connecting to a space or a meeting, for example add or remove participants, mute audio or video, or when there are only guests joined to the space, they are all put in a lobby room waiting for the host or the owner to join in, in addition Call Branding feature to customize the Cisco Web App and SIP calls such as customized background picture and sign in logo or wav file for prompts according to the enterprise activity.

Also important collaboration concepts are demystified with Wireshark such as the Accept Replaces Header and Rerouting CSS for Call Bridge Group with atypical call flow, also one of the most complex and important concepts in collaboration which are STUN, TURN and ICE for NAT traversal through atypical call flow, why we started in the past with STUN then we moved to TURN and finally why we added ICE, I gone in depth analysis to demystify each protocol and to understand why we jumped from STUN to TURN.

How to read this book?

This book is a series of hands of labs, and there is no relationship between them, you can start at any lab, from the last, the first or the middle, the purpose is to provide you a granularity to freely switch between topics and go to the one you need in depth lecture and understanding.

About the author

Redouane MEDDANE is 3xCCNP Collaboration, Security and Enterprise certified and he is a published author of some of the most important OSPF Protocol, Security and Collaboration books in the world titled: OSPF Demystified With RFC, Network Security All-in-one, and Dial Plan Call Routing Demystified on CUCM. He is also a blogger at ipdemystify.com and writes articles about OSPF, collaboration and security to demystify the most complex topics.

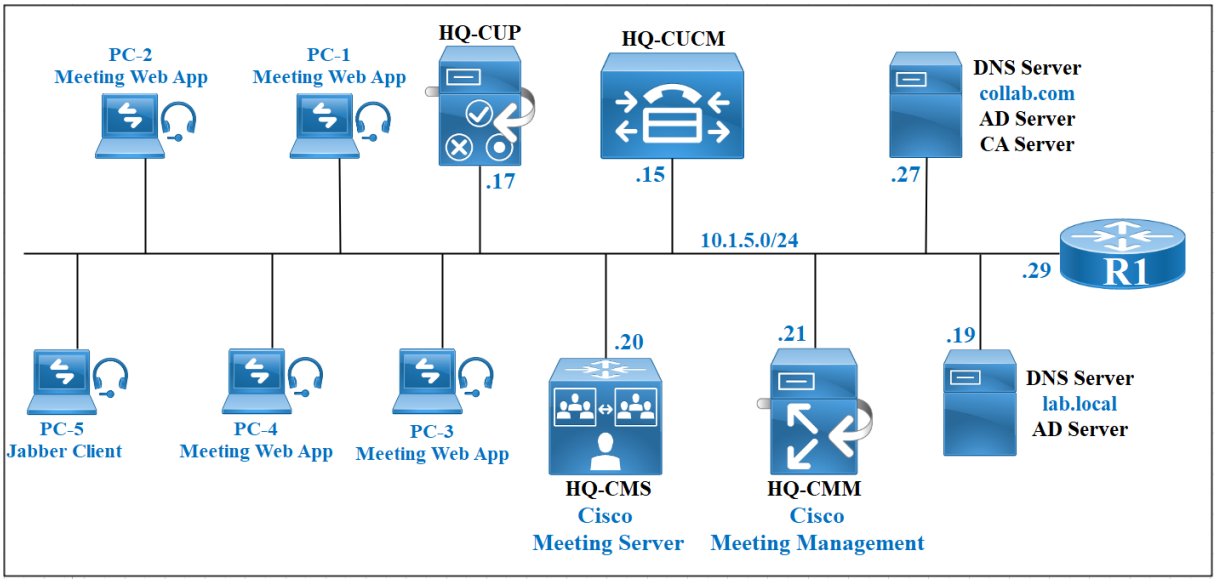
His books are known for their technical depth and accuracy especially the OSPF Demystified With RFC book, which is considered as the best OSPF book in the world and named "One of the best OSPF ebooks of all time" by BookAuthority.

Table of Contents:

- **Project 1:** Single-Combined Deployment Multi-SAN Certificates
- **Project 2:** Scalability and Resilience Deployment Multi-SAN certificates
- **Project 3:** Multiple Certificates with Multiple CA Servers
- **Project 4:** Scalability and Resilience Deployment Certificates Advanced Scenario
- **Project 5:** Streamer Service Certificates with Wowza Media Server
- **Project 6:** Scheduler Service Certificates
- **Project 7:** Access Methods CoSpace Users, Blast Dial and AdHoc
- **Project 8:** Cisco CMS Web Server Hosted Branding

- Appendix A: Cisco Meeting Server Clustering Reference CLI
- Appendix B: Why do we need the Accept Replaces Header
- Appendix C: Cisco TMS Installation
- Appendix D: Cisco TMS Clustering Installation

Project 1: Single-Combined Deployment Multi-SAN Certificates



Network Setting Configuration of Cisco Meeting Server

Log into **HQ-CMS** with default user **admin** and **admin** as password. After you log in, it asks you to change the password.

Change the hostname.

```
acano>hostname hq-cms
```

After this command, the CMS will ask you to reboot CMS in order to activate new hostname. You can reboot the CMS with the **reboot** command.

```
acano>reboot
hq-cms>
```

Configure a static IP address and a gateway. In this command, **a** is the name of a interface.

```
hq-cms>ipv4 a add 10.1.5.20/24 10.1.5.29
```

You need DNS and NTP servers for CMS to work properly. Considering that they are already up and running.

```
hq-cms>ntp server add 10.1.5.29
hq-cms>dns add forwardzone collab.com 10.1.5.27
```

Certificate Preparation for Cisco Meeting Server

Certificate configuration is required for the Call Bridge, Web Bridge and Web Admin services. Certificates should be signed by internal or external certificate authorities.

To generate a Certificate Signing Request (CSR) and private key locally, the following command is used, I give the name **cmscert**.

```
hq-cms> .collab.com,webadmin.collab.com,hq-cms.collab.com,*.lab.local,10.1.5.20
..........
..........................
Created key file cmscert.key and CSR cmscert.csr
CSR file cmscert.csr ready for download via SFTP
hq-cms>
```

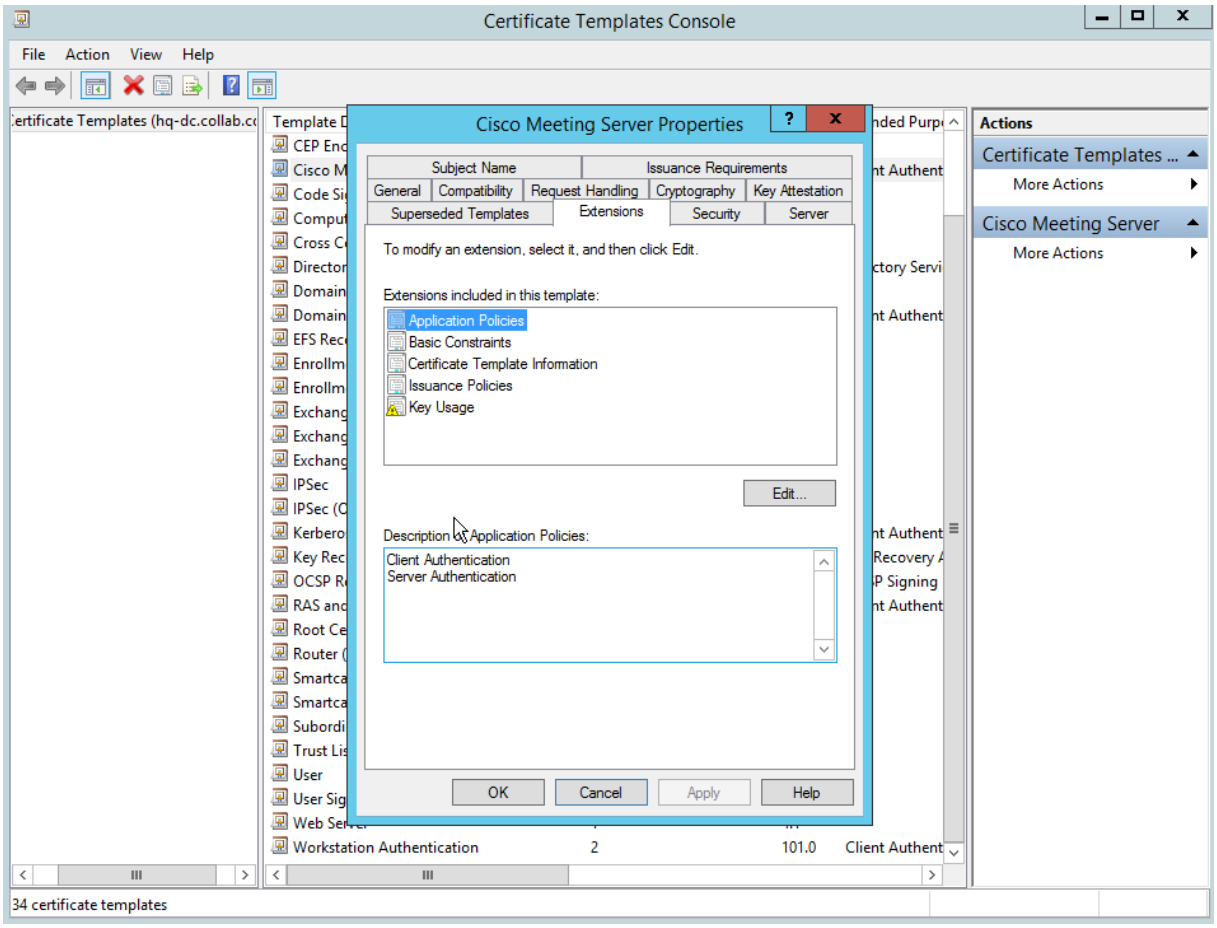
```
hq-cms>pki csr cmscert CN:collab.com OU:CCNP O:Collaboration L:Hydra
ST:Algiers C:AL
subjectAltName:webbridge.collab.com,xmpp.collab.com,callbridge.collab.com,j
oin.collab.com,webadmin.collab.com,hq-
cms.collab.com,*.lab.local,10.1.5.20,10.1.5.42,10.1.5.0/24
```

To retrieve the CSR, login to HQ-CMS using WinSCP.

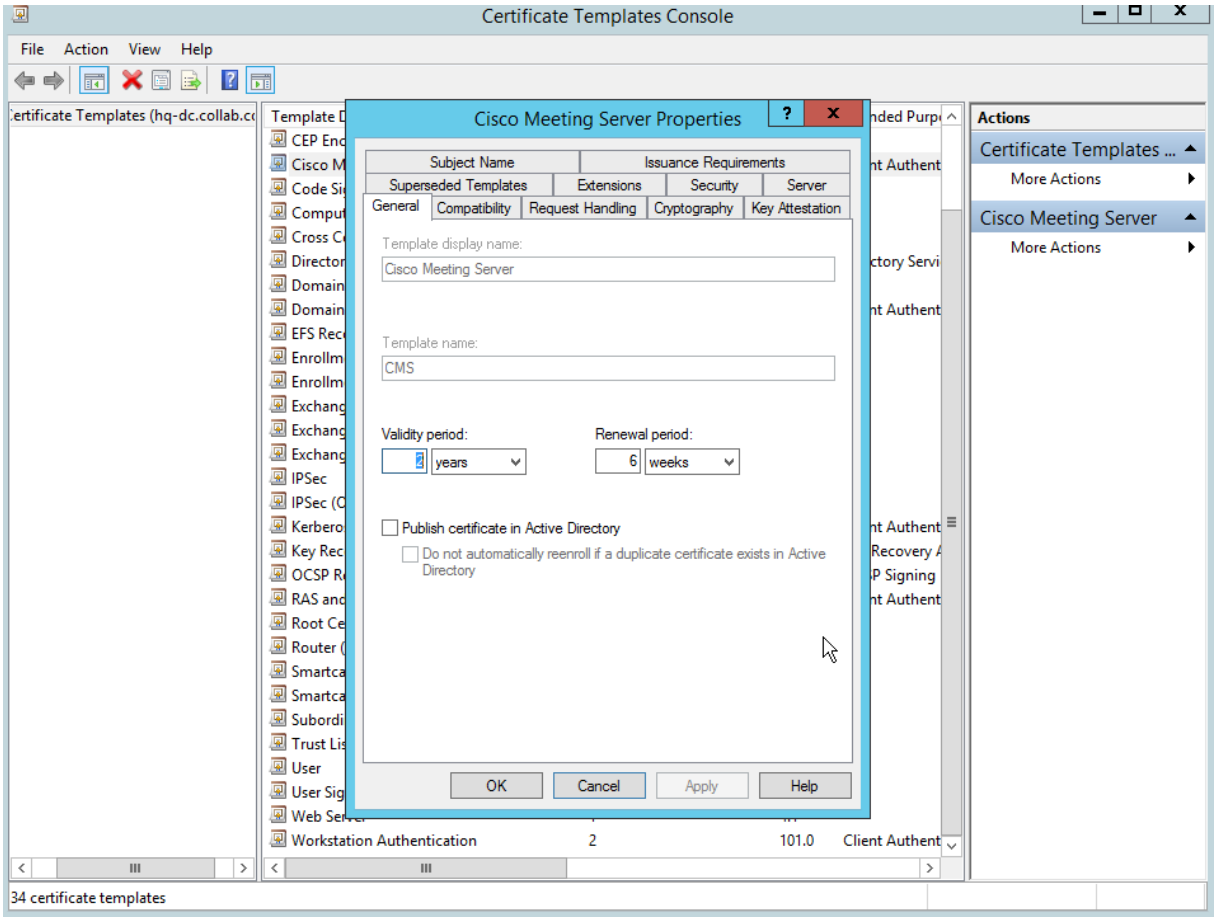
Access the **CA server 10.1.6.27**.

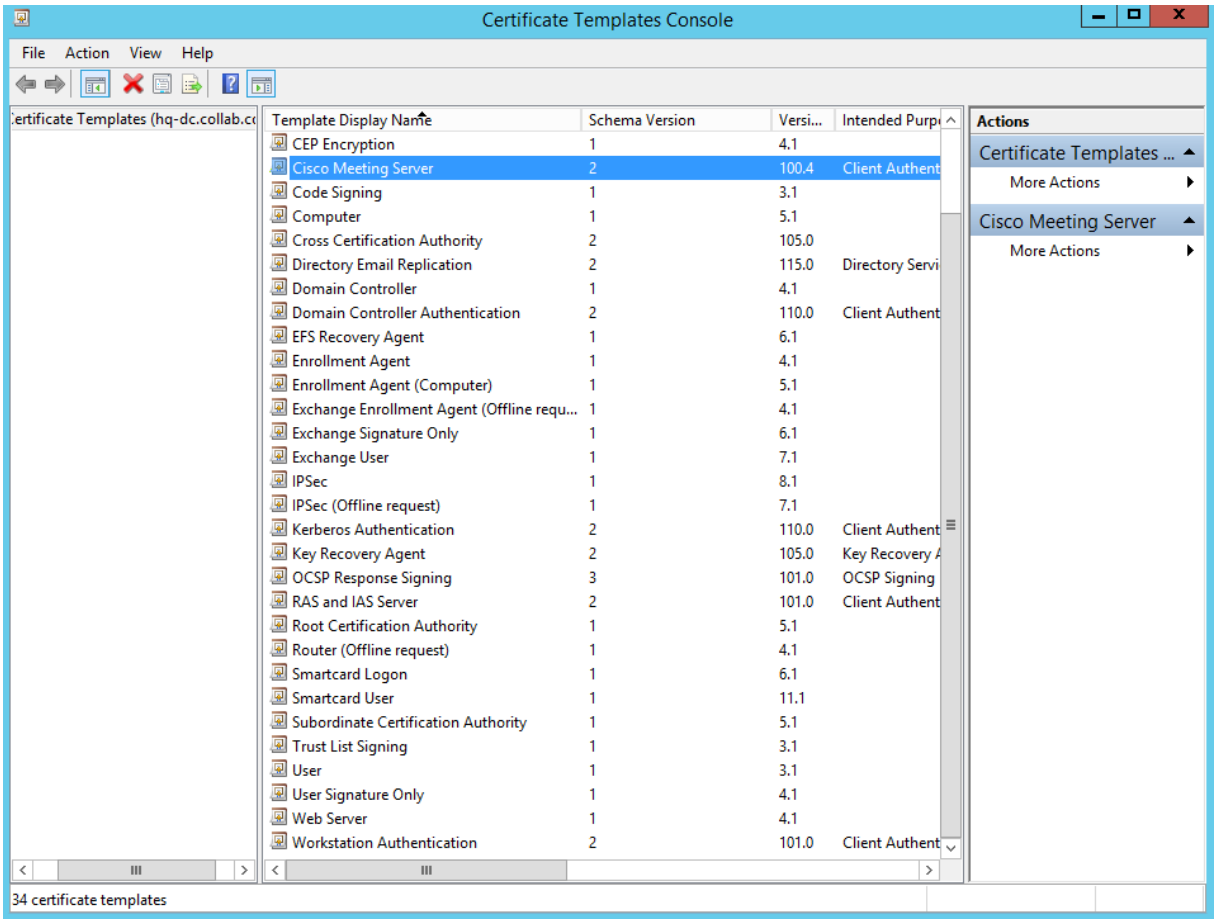
Start the **Certification Authority** console, select **Certificate Template**. Right-click the **Certificate Template** and select **Manage**.

Duplicate the **Web Server** template and configure the duplicate template to allow server and client authentication.

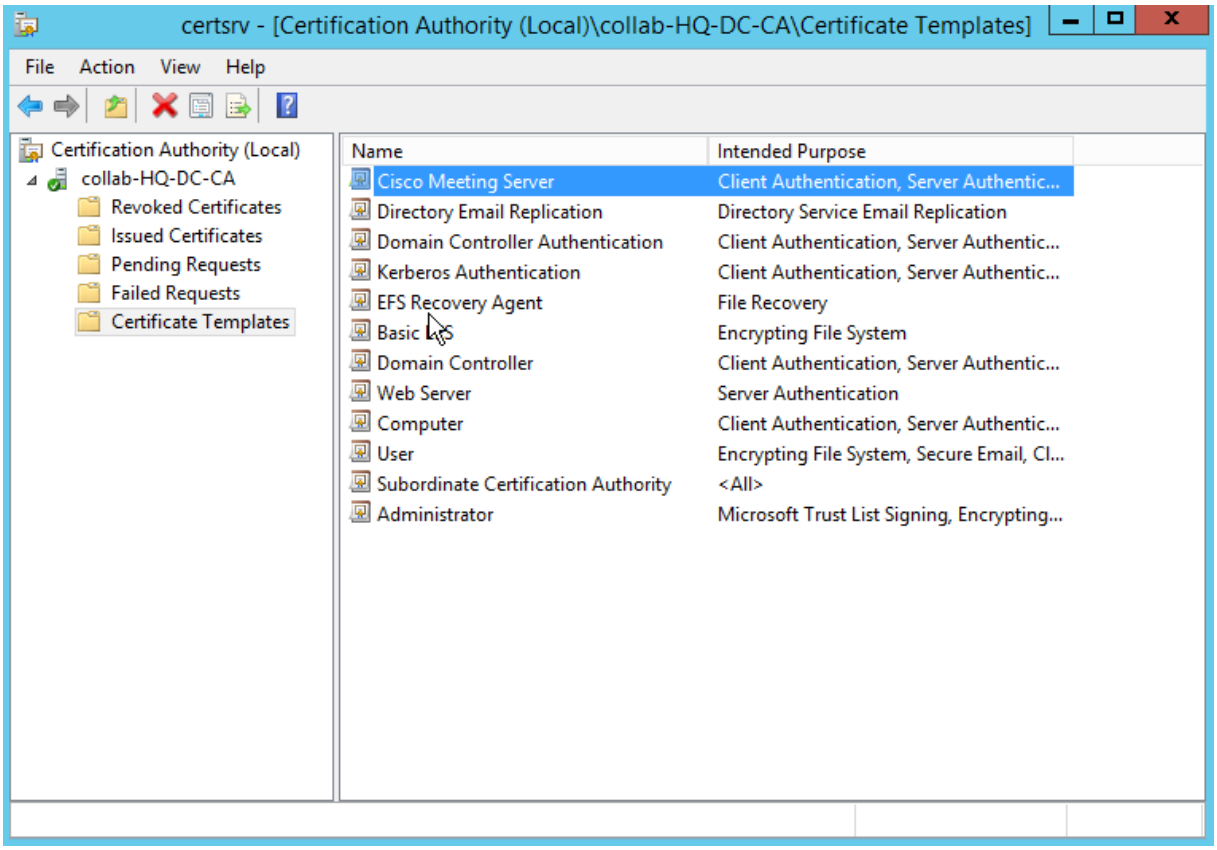


Configure the **Template Name** and **Template display name** of the duplicate template to **CMS** and **Cisco Meeting Server** respectively.


Certificate Templates Console
File Action View Help
Cisco Meeting Server Properties
Subject Name
Issuance Requirements
Superseded Templates
Extensions
Security
Server
General
Compatibility
Request Handling
Cryptography
Key Attestation
Template display name:
Cisco Meeting Server
Template name:
CMS
Validity period:
years
Renewal period:
6
weeks
Publish certificate in Active Directory
Do not automatically reenroll if a duplicate certificate exists in Active Directory
OK
Cancel
Apply
Help
Actions
Certificate Templates ...
More Actions
Cisco Meeting Server
More Actions
Workstation Authentication
2
101.0
Client Authent
34 certificate templates

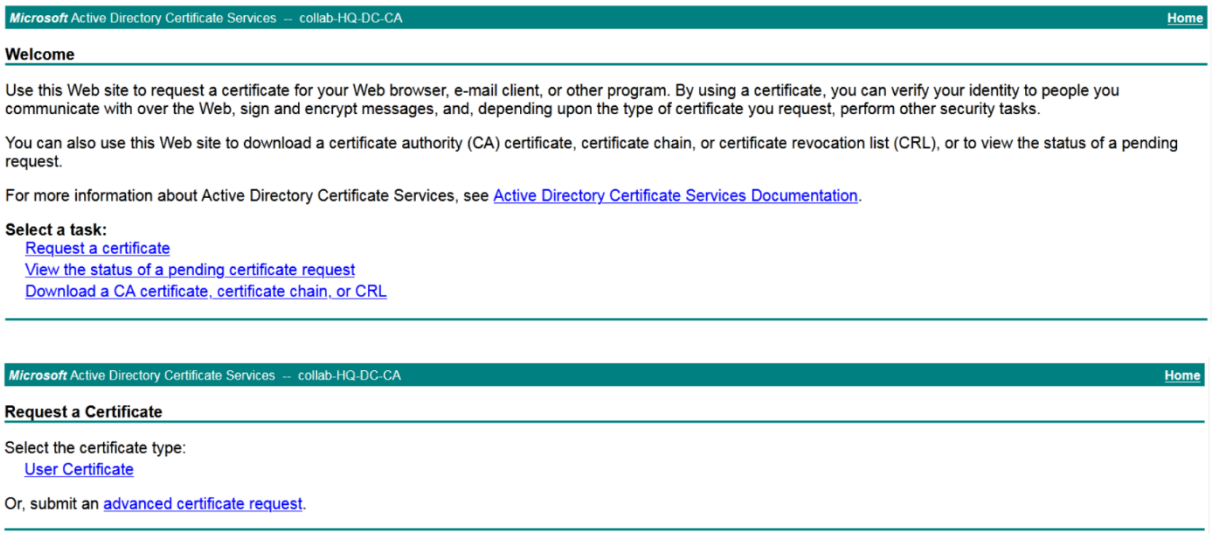


On the Certificate Console, issue a new certificate template named **CMS**.

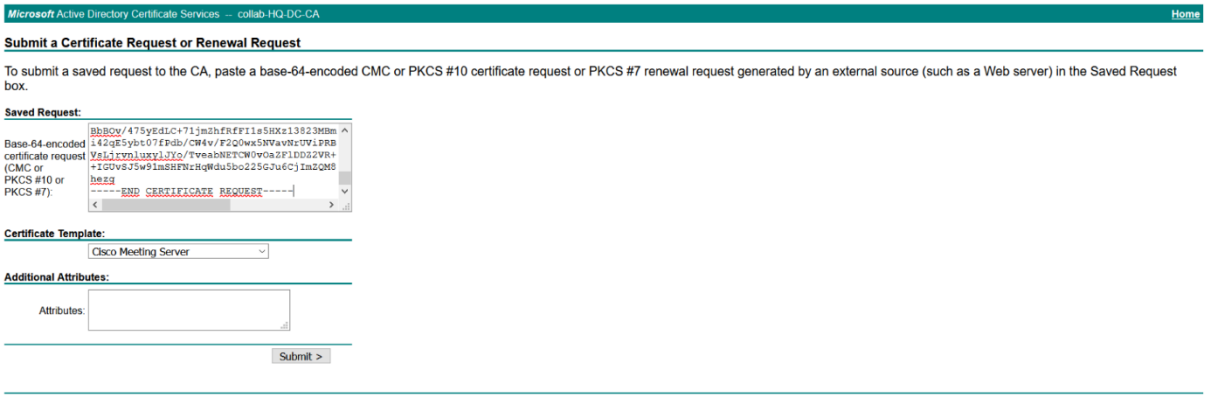


Access the **CA server 10.1.5.27** GUI using the url **http://10.1.5.27/certsrv**.

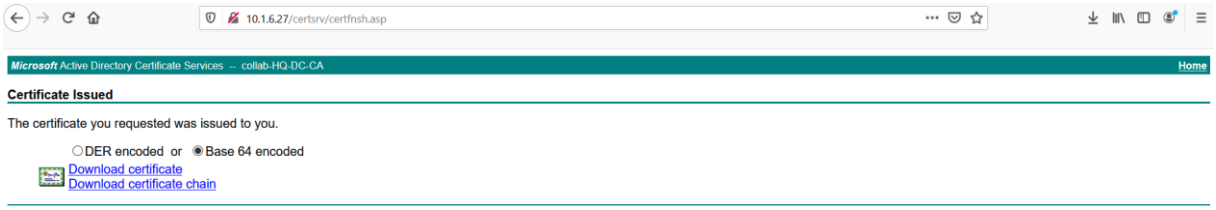
Click **Request a certificate** and the click **advanced request certificate**.



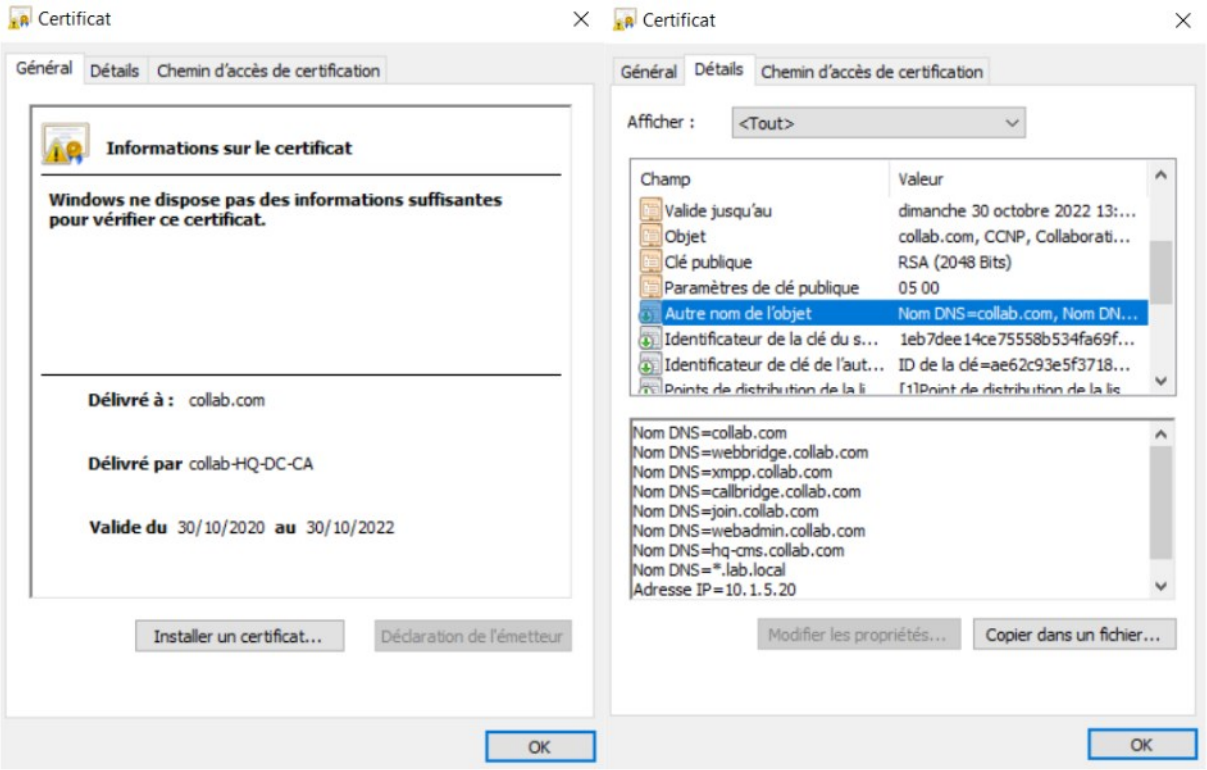
Edit the CSR in notepad and past the content. In the Certificate Template, select **Cisco Meeting Server**.



Select **Base 64 Encoded** and click **Download certificate**.



Below the Certificate named **cmscert** after submitting the CSR to the CA.



A chain certificate is required to trust the **cmscert** certificate when you will enable webadmin, callbridge.

A chain certificate is a single file (with an extension of .pem, .cer or.crt) holding a copy of the Root CA's certificate and all intermediate certificates in the chain.

To create a chain certificate, you need the Root CA or the CA's certificate and a Subordinate CA's certificate with the **Common Name : collab.com**.

To get a Subordinate CA's certificate, we need to generate a CSR.

You can use openssl tool to generate a CSR with **Common Name : collab.com**.

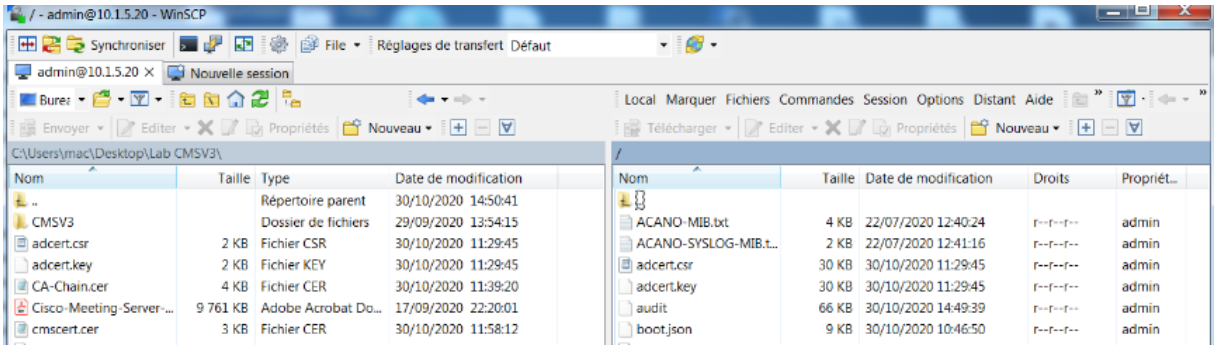
If you did not install openssl, you can generate the CSR on Cisco Meeting Server.

Access the **HQ-CMS** GUI using the url **https://10.1.5.20:445**.

From the CLI, type the following command, the name of the CSR is **adcert** and the **Common Name** is **collab.com**.

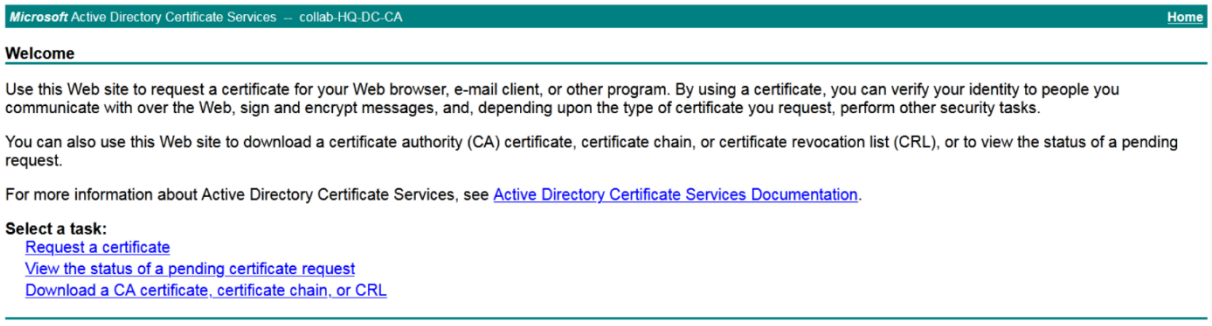
```
hq-cms>pki csr adcert CN:collab.com OU:CCNP O:Collaboration L:Hydra
ST:Algiers C:AL
```

Retrieve the CSR named **adcert** using WinSCP, access **HQ-CMS** using WinSCP, then copy the **adcert** CSR into your PC.

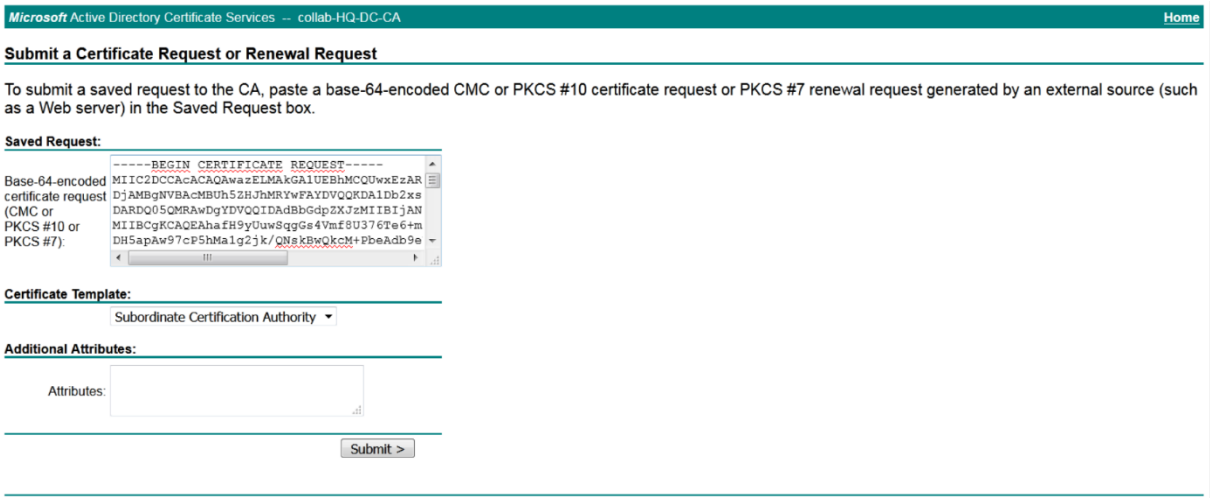


Access the **CA server 10.1.5.27** GUI using the url **http://10.1.5.27/certsrv**.

Click **Request a certificate** and the click **advanced request certificate**.

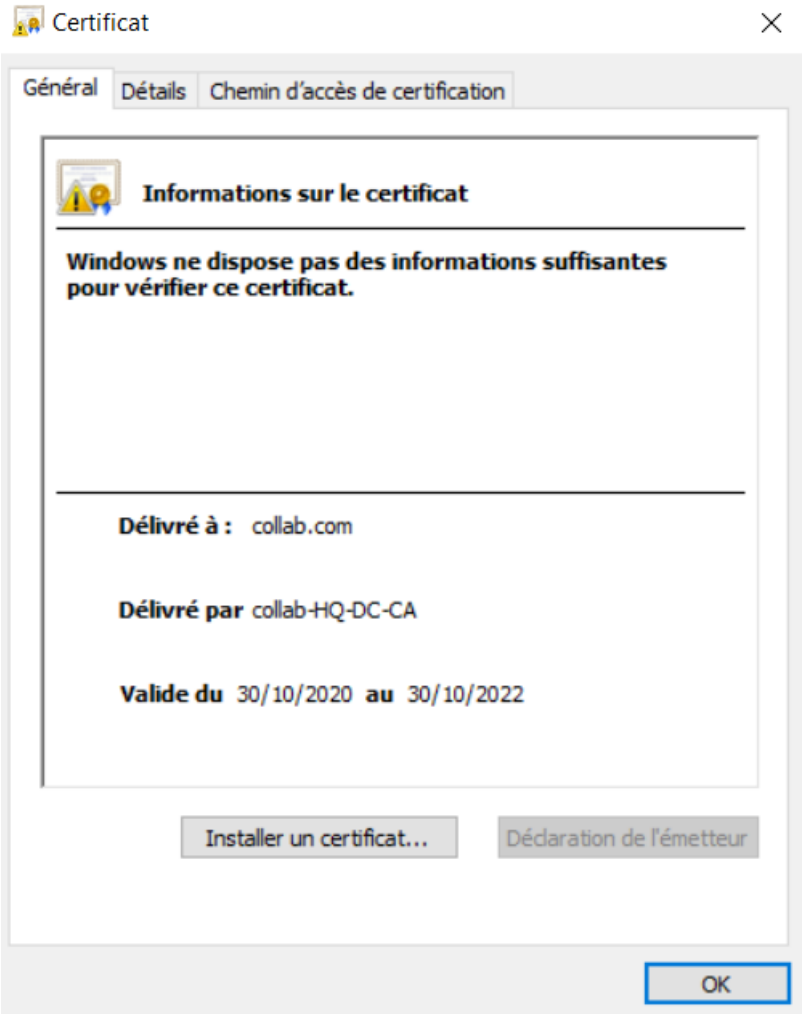


Edit the CSR in notepad and past the content. In the Certificate Template, select **Subordinate Certification Authority**.



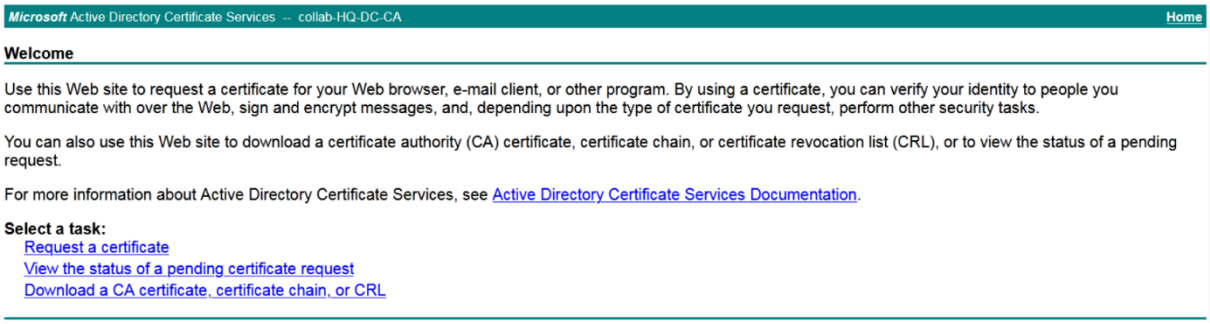
Select **Base 64 Encoded** and click **Download certificate**.

Below the the Certificate named **adcert** after submitting the CSR to the CA.

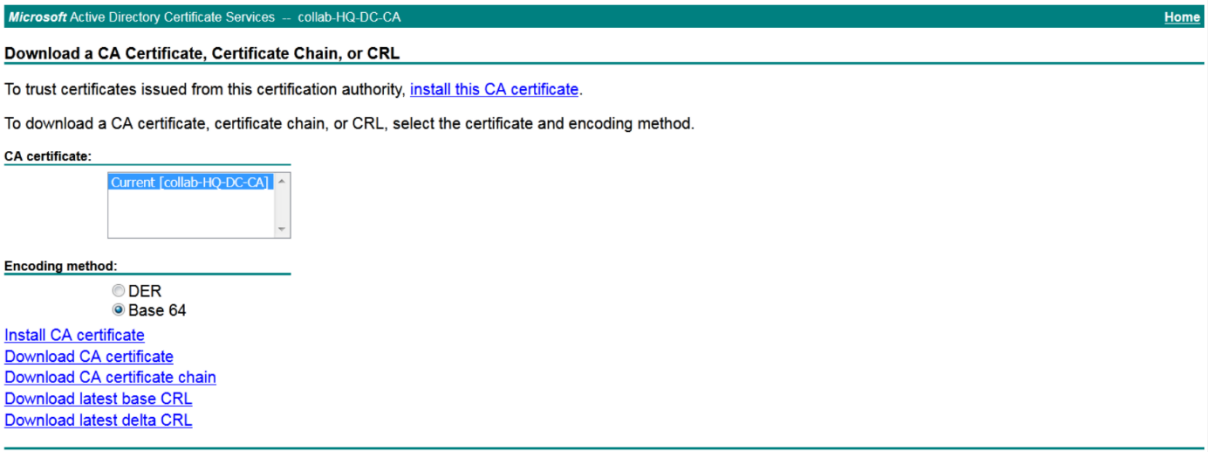


Access the **CA server 10.1.5.27** GUI using the url **http://10.1.5.27/certsrv**.

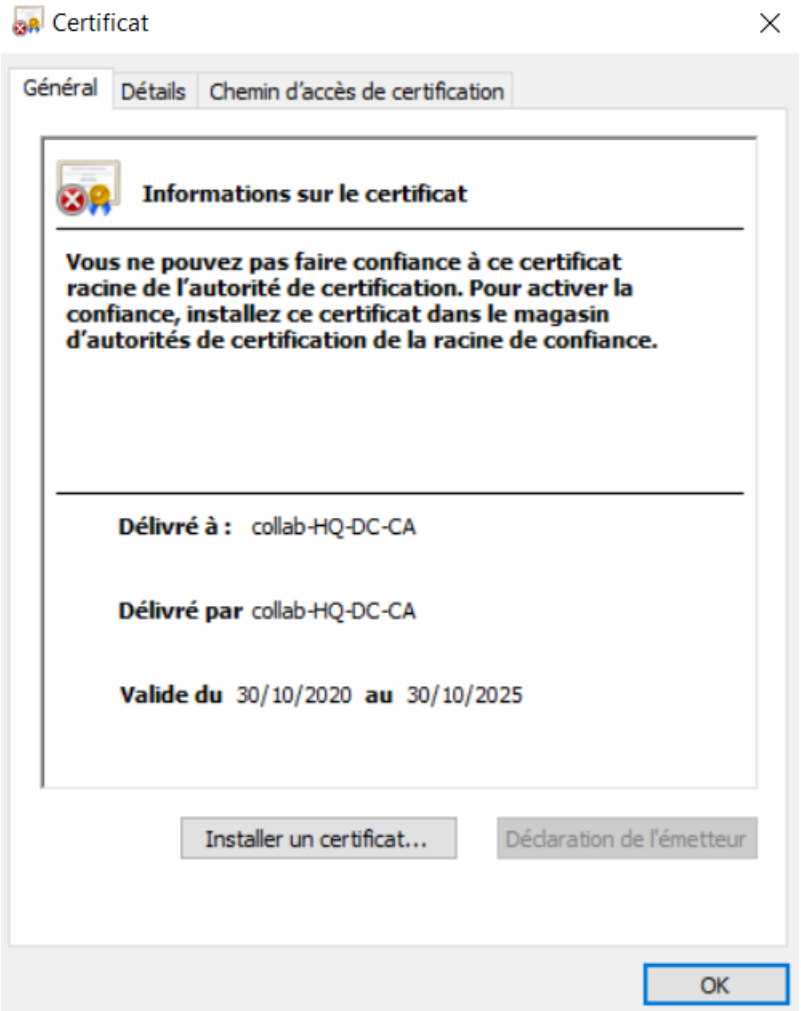
Click **Download a CA certificate, certificate chain, or CRL**.



Select **Base 64**, then click **Download CA certificate**, name it **Root-CA**.



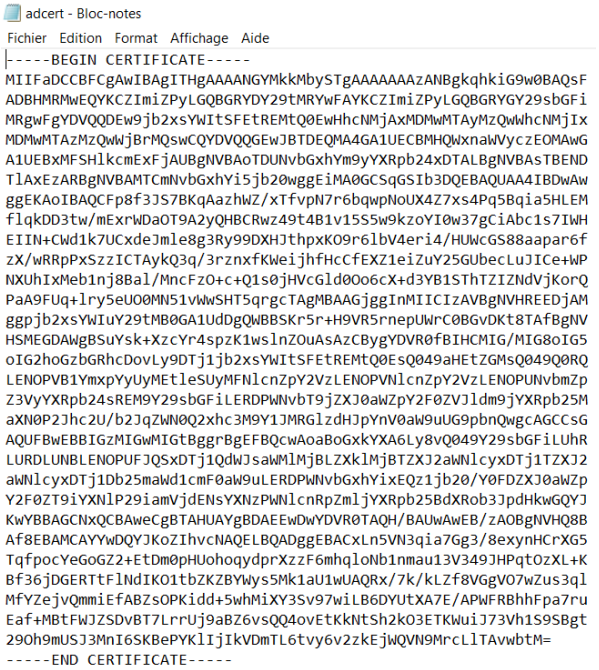
Below the CA's certificate.



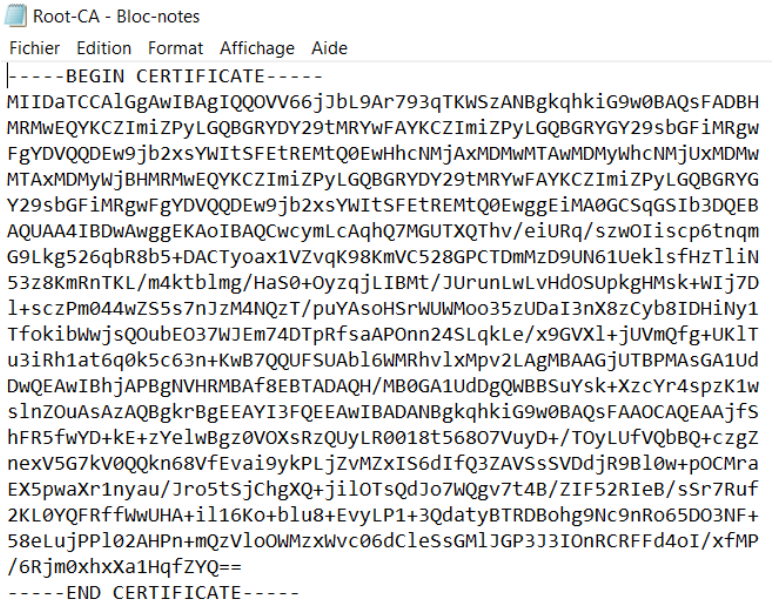
Now the CA's certificate and the Subordinate CA's certificate with the **Common Name : collab.com** are ready, we can create a chain certificate.

To create a chain certificate, use a plain text editor such as notepad. All of the characters including the **-----BEGIN CERTIFICATE-----** and **-----END CERTIFICATE-----** tags need to be inserted into the document. There should be no space between the certificates, for example no spaces or extra lines between **-----END CERTIFICATE----- of certificate 1 and -----BEGIN CERTIFICATE-----** of certificate 2. Certificate 1 will end with **-----END CERTIFICATE----** and the very next line will have **-----BEGIN CERTIFICATE----** for certificate 2. At the end of the file there should be 1 extra line. Save the file with an extension of .pem, .cer, or .crt.

Edit the certificate named **adcert** created previously with nodepad.

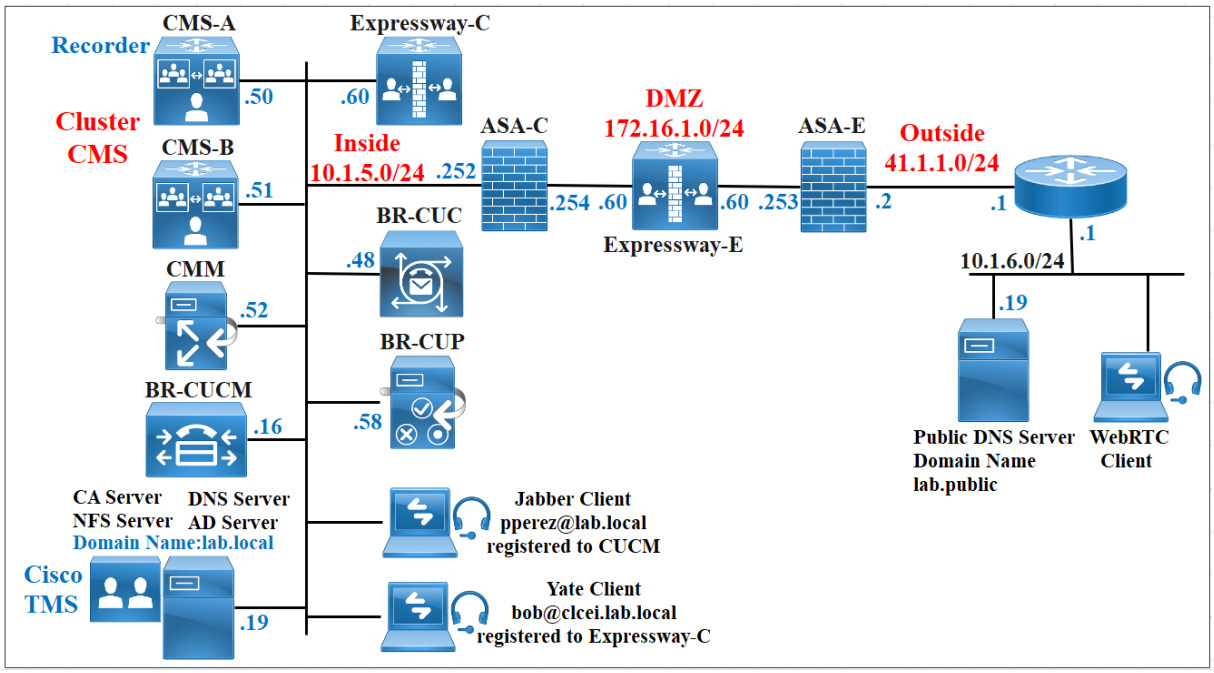


Edit the **Root-CA** certificate with nodepad.



Past the **adcert** certificate first and then past the **Root-CA** certificate at the end, save the file with **.cer** extension. Name it **CA-Chain.cer**.

Project 2: Scalability and Resilience Deployment Multi-SAN certificates



1. Basic configuration of Cisco Meeting Server
2. Configuration of Cisco Meeting Server Cluster
3. Webbridge 3 Configuration
4. CMS Outbound Calls and Incoming Calls
5. Configuration of Cisco Expressway-C expc1
6. Secure Traversal Expressway-C
7. Configuration of Cisco Expressway-E expe1 DUAL NIC
8. Secure Traversal Expressway-E
9. Cisco Expressway Unifed Edge
10. Configure TurnServers using CMS API
11. Active Directory integration with CMS
12. Cisco Unified Communication Manager Dial Plan to CMS
13. Cisco Expressway-C Dial Plan to CMS
14. Configuration of Cisco Meeting Management
15. Test Conference
16. TMS Scheduled Conferencing
17. Test Scheduled Conference
18. Call Bridge Group Load Balancing
19. CMS Recorder
20. Appendix : DNS Records

Basic configuration of Cisco Meeting Server

Log into **CMS-A** with default user **admin** and **admin** as password. After you log in, it asks you to change the password.

Change the hostname.

```
acano>hostname cms-a
```

After this command, the CMS will ask you to reboot CMS in order to activate new hostname. You can reboot the CMS with the **reboot** command.

```
acano>reboot
cms-a>
```

Configure a static IP address and a gateway. In this command, **a** is the name of a interface.

```
cms-a>ipv4 a add 10.1.5.40/24 10.1.5.252
```

You need DNS and NTP servers for CMS to work properly. Considering that they are already up and running.

```
cms-a>ntp server add 10.1.5.29
cms-a>dns add forwardzone lab.local 10.1.5.19
```

Repeat the same steps for **CMS-b**.

Log into **CMS-b** with default user **admin** and **admin** as password. After you log in, it asks you to change the password.

```
acano>hostname cms-b
acano>reboot
cms-b>
cms-b>ipv4 a add 10.1.5.41/24 10.1.5.252
cms-b>ntp server add 10.1.5.29
cms-b>dns add forwardzone lab.local 10.1.5.19
```

Configuration of Cisco Meeting Server Cluster

Database Clustering

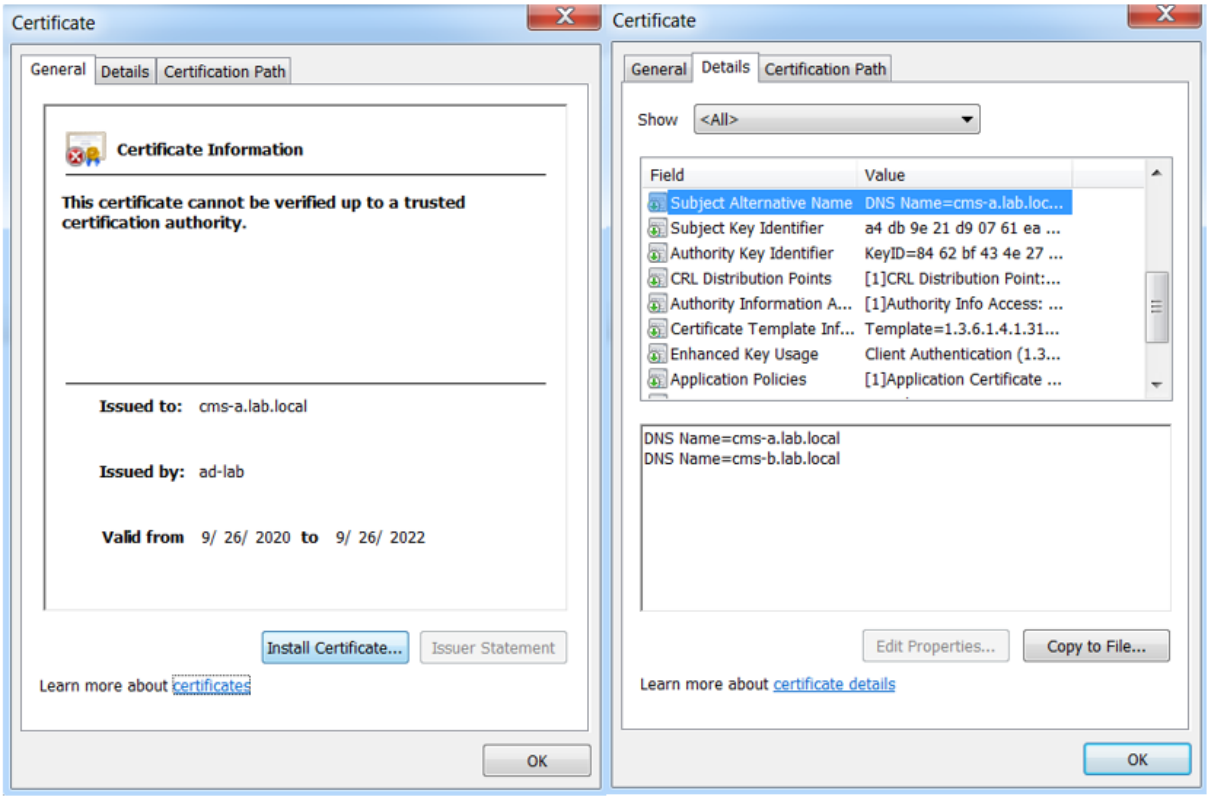
Prepare certificates

For database you need to generate two CSRs with the corresponding private keys, the client and the server.
Use one CMS to generates these two CSRs, once you get the server and client certificates from your CA, copy the two certificates with their private keys to the second CMS-b using WinSCP.

Let's start with cms-a which is the Master Database.

For Server certificate use the following command, give a name for example dbcert, it is important to put the CN to the FQDN of the Master Database cms-a and the SAN to the second cms-b, if you have multiple CMS, add their FQDN in the same line. For example: cms-b.lab.local,cms-c.lab.local,cms-d.lab.local.

```
cms-a>pki csr dbcert CN:cms-a.lab.local OU:CCNP O:Collaboration L:lab
ST:local C:US subjectAltName:cms-b.lab.local
```

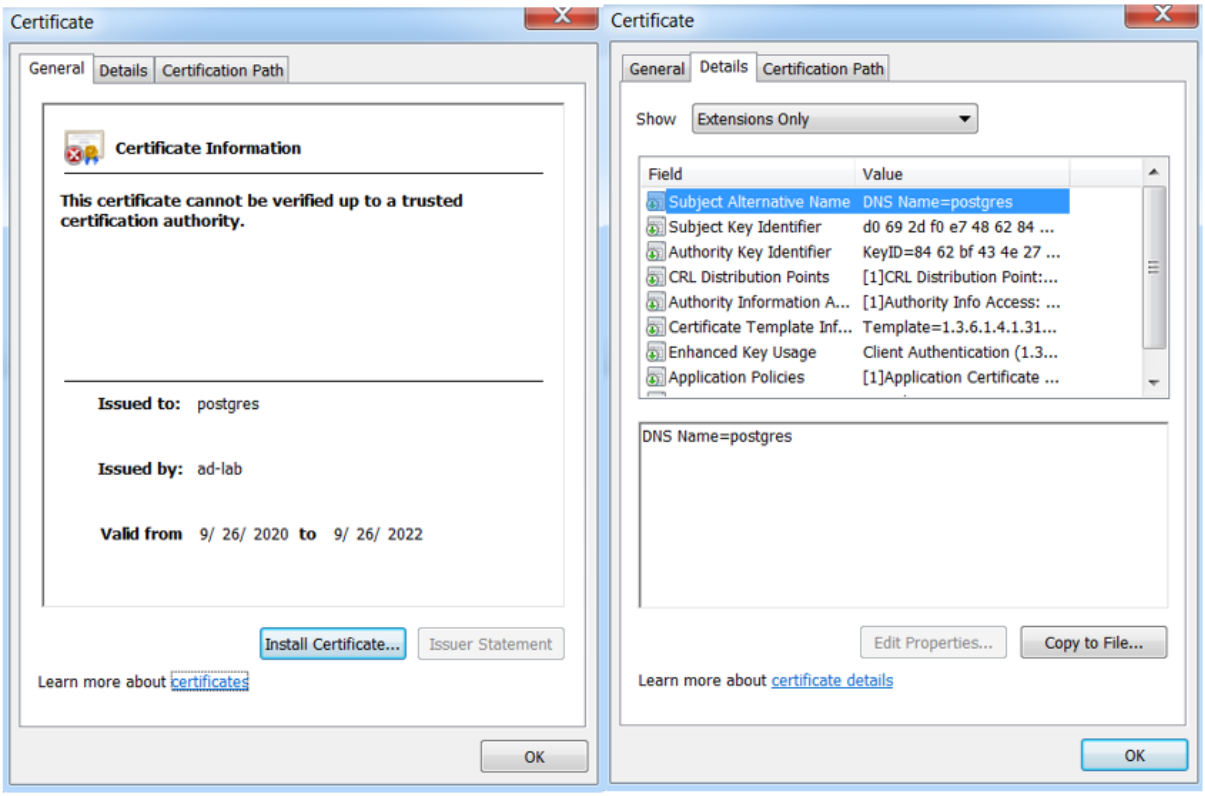


For Client certificate use the following command, give a name for example **dbclt**.

```
cms-a>pki csr dbclt CN:postgres
```

When generating Certificate for client, use the same template as the database certificate (Client and Server authentication).

Below example of the Client Certificate named **dbclt** after submitting the CSR to the CA.



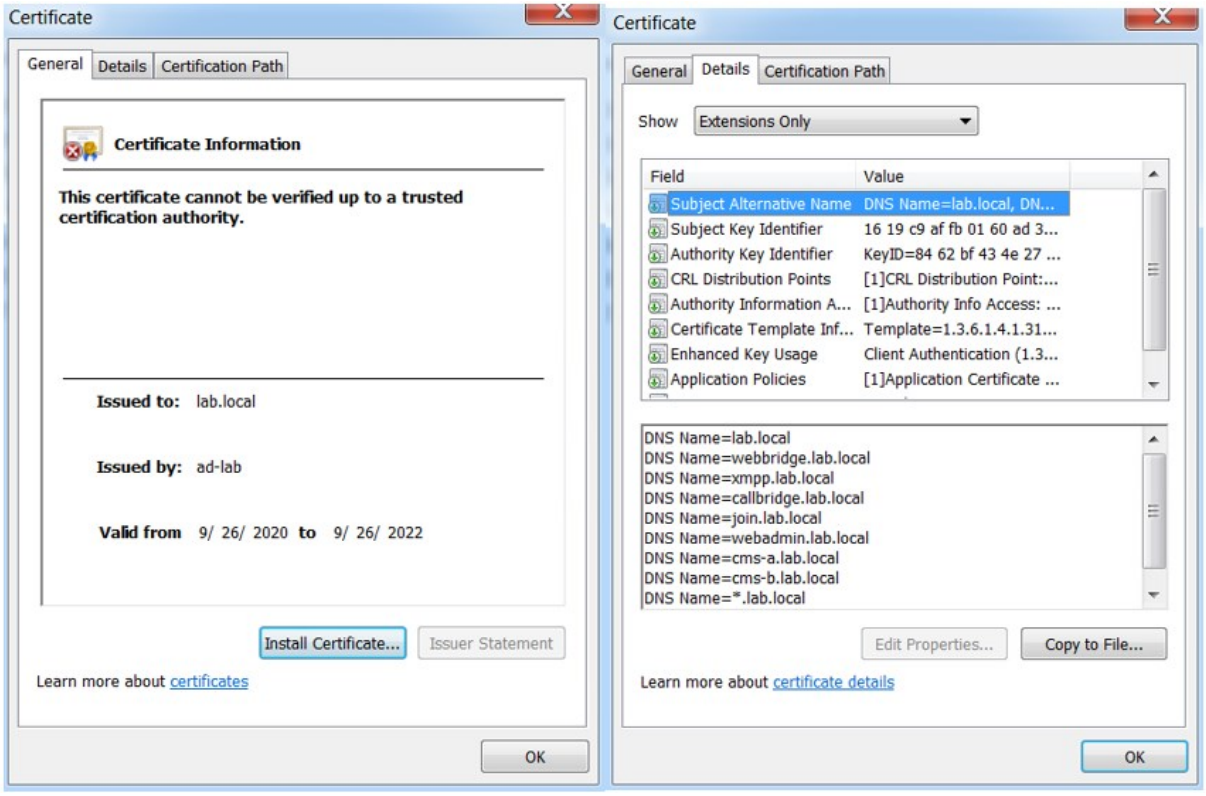
Certificate configuration is required also for the Call Bridge, XMPP, Web Bridge and Web Admin services. Certificates should be signed by internal or external certificate authorities.

In a Single Server deployment the certificate should have the server FQDN as the Common Name (CN), in Cluster deployment, I use the domain name as the Common Name **lab.local** and the certificate must have the Subject Alternate Name (SAN) attributes, for example. XMPP, Join (WebRTC) etc.

To generate a Certificate Signing Request (CSR) and private key locally, the following command is used, I give the name **1cert**.

```
cms-a>pki csr 1cert CN:lab.local OU:CCNP O:Collaboration L:lab ST:local
C:US subjectAltName:webbridge.lab.local,xmpp.lab.local,
callbridge.lab.local,join.lab.local,webadmin.lab.local,cms-a.lab.local,cms-
b.lab.local,*.lab.local
```

Below the Certificate named **1cert** after submitting the CSR to the CA.



A chain certificate is required for Webbridge3 in version 3, it is a single file (with an extension of .pem, .cer or.crt) holding a copy of the Root CA's certificate and all intermediate certificates in the chain.

To create a chain certificate, use a plain text editor such as notepad. All of the characters including the **-----BEGIN CERTIFICATE-----** and **-----END CERTIFICATE-----** tags need to be inserted into the document. There should be no space between the certificates, for example no spaces or extra lines between **-----END CERTIFICATE----- of certificate 1 and ----- BEGIN CERTIFICATE-----** of certificate 2. Certificate 1 will end with **-----END CERTIFICATE----** and the very next line will have **-----BEGIN CERTIFICATE----** for certificate 2. At the end of the file there should be 1 extra line. Save the file with an extension of .pem, .cer, or .crt.

A chain certificate is required to trust the **cmscert** certificate when you will enable webadmin, callbridge.

A chain certificate is a single file (with an extension of .pem, .cer or.crt) holding a copy of the Root CA's certificate and all intermediate certificates in the chain.

To create a chain certificate, you need the Root CA or the CA's certificate and a Subordinate CA's certificate with the **Common Name : lab.local**.

To get a Subordinate CA's certificate, we need to generate a CSR.

You can use openssl tool to generate a CSR with **Common Name : lab.local**.

If you did not install openssl, you can generate the CSR on Cisco Meeting Server.

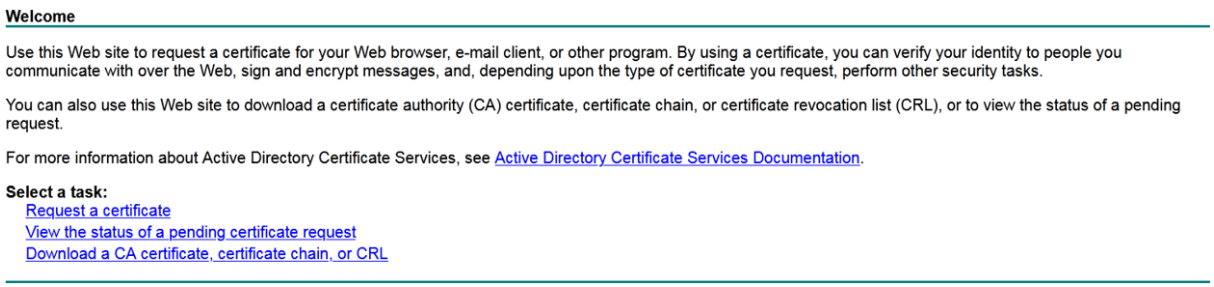
From the CLI, type the following command, the name of the CSR is **adcert** and the **Common Name** is **lab.local**.

```
hq-cms>pki csr adcert CN:lab.local OU:CCNP O:Collaboration L:Hydra
ST:Algiers C:AL
```

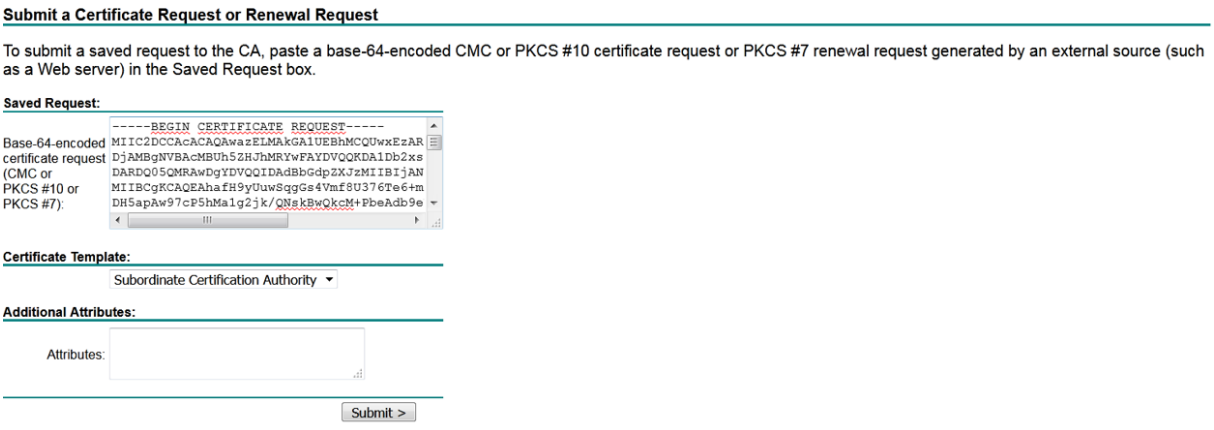
Retrieve the CSR named **adcert** using WinSCP, access **CMS-A** using WinSCP, then copy the **adcert** CSR into your PC.

Access the **CA server 10.1.5.19** GUI using the url **http://10.1.5.19/certsrv**.

Click **Request a certificate** and the click **advanced request certificate.**



Edit the CSR in notepad and past the content. In the Certificate Template, select **Subordinate Certification Authority**.



Select **Base 64 Encoded** and click **Download certificate**.

```
cms-a> pki list
User supplied certificates and keys:
1cert.key
1cert.csr
1cert.cer
Bundle-CA.cer
dbcert.key
dbcert.csr
dbclt.key
dbclt.csr
dbcert.cer
dbclt.cer
fullchain.cer
cms-a>
```

```
cms-b> pki list
User supplied certificates and keys:
1cert.cer
1cert.key
Bundle-CA.cer
dbcert.cer
dbcert.key
dbclt.cer
dbclt.key
fullchain.cer
cms-b>
```

Enabling the Web Admin Service

By default, Web Admin listens on HTTPS port of 443. However, we will enable the Web Bridge for conference users and this service will be available on the default HTTPS port 443. To enable both services to co-exist, we will configure Web Admin to listen on port **445**.
On **CMS-A**, specify the interface and HTTPS port **445** for the web interface.

```
cms-a>webadmin listen a 445
```

For the certificate to be used, specify the certificate **1cert** created in previously with the relevant key.

```
cms-a>webadmin certs 1cert.key 1cert.cer
```

Route HTTP requests to HTTPS.

```
cms-a>webadmin http-redirect enable
```

Finally activate the web admin service.

```
cms-a>webadmin enable
```

Verify that the webadmin service is running is using the **webadmin** command.

```
cms-a> webadmin
Enabled                  : true
TLS listening interface  : a
TLS listening port       : 445
Key file                 : 1cert.key
Certificate file         : 1cert.cer
CA Bundle file           : Bundle-CA.cer
HTTP redirect            : Enabled
STATUS                   : webadmin running
cms-a>
```

Repeat the same steps for **CMS-B**.

```
cms-b>webadmin listen a 445
```

```
cms-b>webadmin certs 1cert.key 1cert.cer
```

Route HTTP requests to HTTPS.

```
cms-b>webadmin http-redirect enable
```

Finally activate the web admin service.

```
cms-b>webadmin enable
```

```
cms-b> webadmin
Enabled                  : true
TLS listening interface  : a
TLS listening port       : 445
Key file                 : 1cert.key
Certificate file         : 1cert.cer
CA Bundle file           : Bundle-CA.cer
HTTP redirect            : Enabled
STATUS                   : webadmin running
cms-b>
```

Cluster Database Configuration

From the **CMS-A** CLI, specify the database client and server certificates created previously and named **dbcert** and **dbclt**. The **Bundle-CA** certificate is added to verify the validity of the client/ server certificates.

```
cms-a>database cluster certs dbcert.key dbcert.cer dbclt.key dbclt.cer
Bundle-CA.cer
```

Specify which interface to use for the database clustering.

```
cms-a>database cluster localnode a
```

Initialize the master database.

```
cms-a>database cluster initialize
```

From the **CMS-B** CLI, specify the database client and server certificates created previously and named **dbcert** and **dbclt**. The **Bundle-CA** certificate is added to verify the validity of the client/ server certificates.

```
cms-b>database cluster certs dbcert.key dbcert.cer dbclt.key dbclt.cer
Bundle-CA.cer
```

Specify the interface to use.

```
cms-b>database cluster localnode a
```

Connect **CMS-B** to the master database **CMS-A**.

```
cms-b>database cluster join cms-a.lab.local
```

On both **CMS-A** and **CMS-B**, Verify the status of the database cluster.

```
cms-a> database cluster status
Status                    : Enabled

Nodes:
    10.1.5.50 (me)        : Connected Primary
    10.1.5.51             : Connected Replica ( In Sync )
Node in use               : 10.1.5.50

Interface                 : a

Certificates
  Server Key              : dbcert.key
  Server Certificate      : dbcert.cer
  Client Key              : dbclt.key
  Client Certificate      : dbclt.cer
  CA Certificate          : Bundle-CA.cer

Last command              : 'database cluster initialize' (Success)

cms-a>
```

```
cms-b> database cluster status
Status                    : Enabled

Nodes:
    10.1.5.50             : Connected Primary
    10.1.5.51 (me)        : Connected Replica ( In Sync )
Node in use               : 10.1.5.50

Interface                 : a

Certificates
  Server Key              : dbcert.key
  Server Certificate      : dbcert.cer
  Client Key              : dbclt.key
  Client Certificate      : dbclt.cer
  CA Certificate          : Bundle-CA.cer

Last command              : 'database cluster join cms-a.lab.local' (Success)

cms-b>
```

Cluster Call Bridges Configuration

Configure callbridge on **CMS-A** to listen on the interface **a**.

```
cms-a>callbridge listen a
```

Specify the certificate **1cert** created in previously with the relevant key.

```
cms-a>callbridge certs 1cert.key 1cert.cer Bundle-CA.cer
```

Restart the callbridge

```
cms-a>callbridge restart
```

Repeat the same steps for **CMS-B.**

```
cms-b>callbridge listen a
```

```
cms-b>callbridge certs 1cert.key 1cert.cer Bundle-CA.cer
```

```
cms-b>callbridge restart
```

Verify the callbridge on both **CMS-A** and **CMS-B**.

```
cms-a> callbridge
Listening interfaces  : a
Preferred interface   : none
Key file              : 1cert.key
Certificate file      : 1cert.cer
Address               : none
CA Bundle file        : Bundle-CA.cer
C2W trusted certs                    : Bundle-CA.cer
Callbridge cluster trusted certs     : none
cms-a>
```

```
cms-b> callbridge
Listening interfaces  : a
Preferred interface   : none
Key file              : 1cert.key
Certificate file      : 1cert.cer
Address               : none
CA Bundle file        : Bundle-CA.cer
C2W trusted certs                    : Bundle-CA.cer
Callbridge cluster trusted certs     : none
cms-b>
```

Webbridge 3 Configuration

From the **CMS-A** CLI, enter the following commands.

```
cms-a>
cms-a> webbridge3 https listen a:443
cms-a> webbridge3 https certs 1cert.key fullchain.cer
cms-a> webbridge3 c2w listen a:9999
cms-a> webbridge3 c2w certs 1cert.key fullchain.cer
cms-a> webbridge3 c2w trust Bundle-CA.cer
cms-a>
cms-a> webbridge3 enable
SUCCESS: HTTPS Key and certificate pair match
SUCCESS: HTTPS full chain of certificates verifies correctly
SUCCESS: C2W Key and certificate pair match
SUCCESS: C2W full chain of certificates verifies correctly
SUCCESS: Webbridge3 enabled
cms-a>
cms-a> callbridge trust c2w Bundle-CA.cer
cms-a>
cms-a> callbridge restart
SUCCESS: listen interface configured
SUCCESS: Key and certificate pair match
SUCCESS: certificate verified against CA bundle
SUCCESS: c2w trust bundle parsed.
cms-a>
```

Repeat the same commands on **CMS-B**.

```
cms-b> webbridge3 https listen a:443
cms-b> webbridge3 https certs 1cert.key fullchain.cer
cms-b> webbridge3 c2w listen a:9999
cms-b> webbridge3 c2w certs 1cert.key fullchain.cer
cms-b> webbridge3 c2w trust Bundle-CA.cer
cms-b> webbridge3 enable
SUCCESS: HTTPS Key and certificate pair match
SUCCESS: HTTPS full chain of certificates verifies correctly
SUCCESS: C2W Key and certificate pair match
SUCCESS: C2W full chain of certificates verifies correctly
SUCCESS: Webbridge3 enabled
cms-b>
cms-b> callbridge trust c2w Bundle-CA.cer
cms-b> callbridge restart
SUCCESS: listen interface configured
SUCCESS: Key and certificate pair match
SUCCESS: certificate verified against CA bundle
SUCCESS: c2w trust bundle parsed.
cms-b>
```

On both **CMS-A** and **CMS-B**, verify the **webbridge3** configuration.

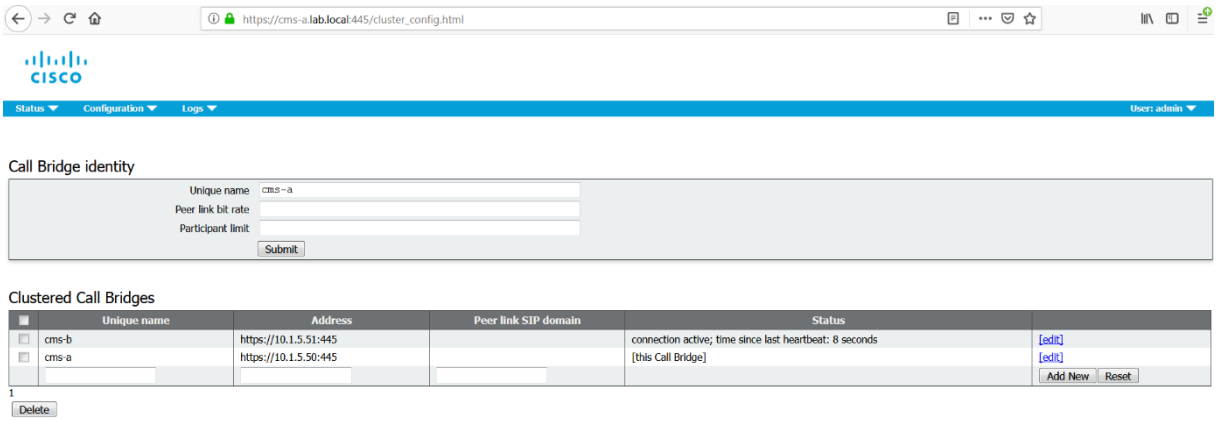
```
cms-a> webbridge3
Enabled                          : true
HTTPS Interface whitelist        : a:443
HTTPS Key file                   : 1cert.key
HTTPS Full chain certificate file : fullchain.cer
HTTP redirect                    : Disabled
C2W Interface whitelist          : a:9999
C2W Key file                     : 1cert.key
C2W Full chain certificate file  : fullchain.cer
C2W Trust bundle                 : Bundle-CA.cer
Beta options                     : none
cms-a>
```

```
cms-b> webbridge3
Enabled                          : true
HTTPS Interface whitelist        : a:443
HTTPS Key file                   : 1cert.key
HTTPS Full chain certificate file : fullchain.cer
HTTP redirect                    : Disabled
C2W Interface whitelist          : a:9999
C2W Key file                     : 1cert.key
C2W Full chain certificate file  : fullchain.cer
C2W Trust bundle                 : Bundle-CA.cer
Beta options                     : none
cms-b>
```

On the web GUI of the **CMS-A** navigate to **Configuration** > **Cluster**.

Under the section **Call Bridge Identity**, enter a **Unique name** for this Callbridge: **cms-a.** Click **Submit**.

Under the section **Clustered Call Bridge**, add each Call bridge unique name to the cluster including this node. In our scenario add **cms-a** in **Unique name** section. Under the **Address** section add **https://10.1.5.50:445**. Add **cms-b** in **Unique name** section. Under the **Address** section add **https://10.1.5.51:445**.



CMS Outbound Calls and Incoming Calls

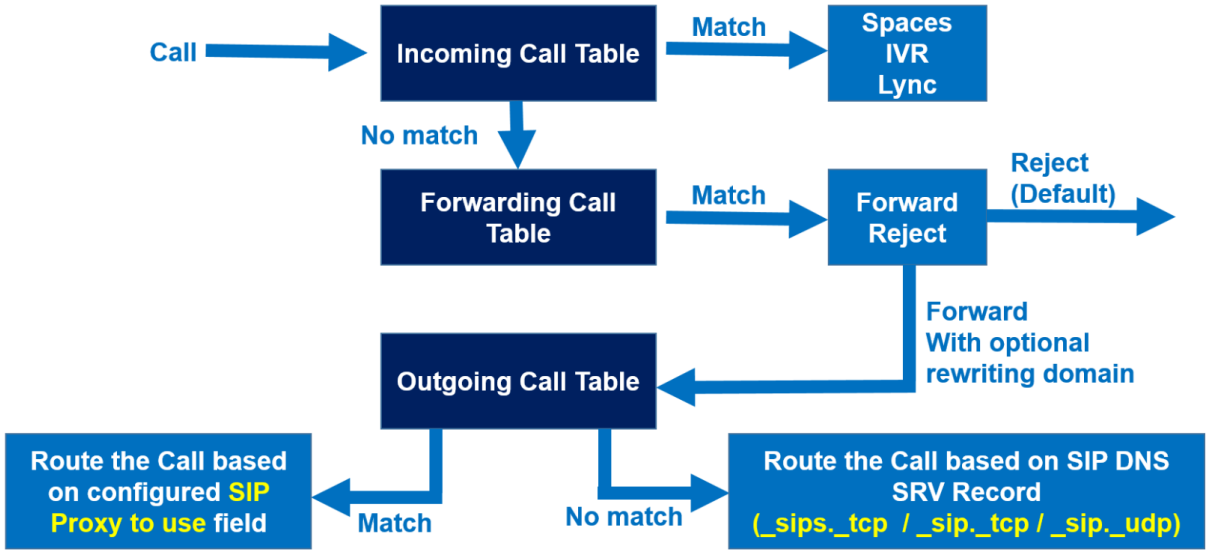
The call routing on CMS involves a three call routing different tables.

Order of call processing

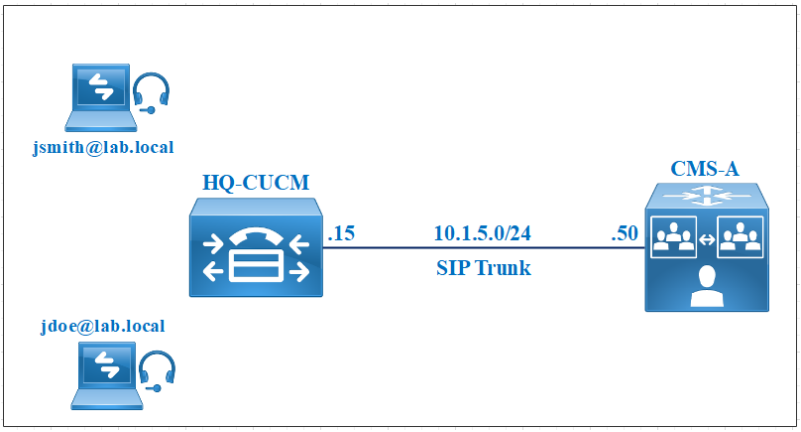
1. Incoming Call Matching Table
2. Incoming Call Forwarding Table
3. Outbound Call Table

The only exception would be for CMS initiated calls (either CMS directly for TelePresence Management Suite (TMS) scheduled outbound calls or CMA client calls out) in which the call forwarding table is bypassed.

Below the chart flow.



Incoming Call Table

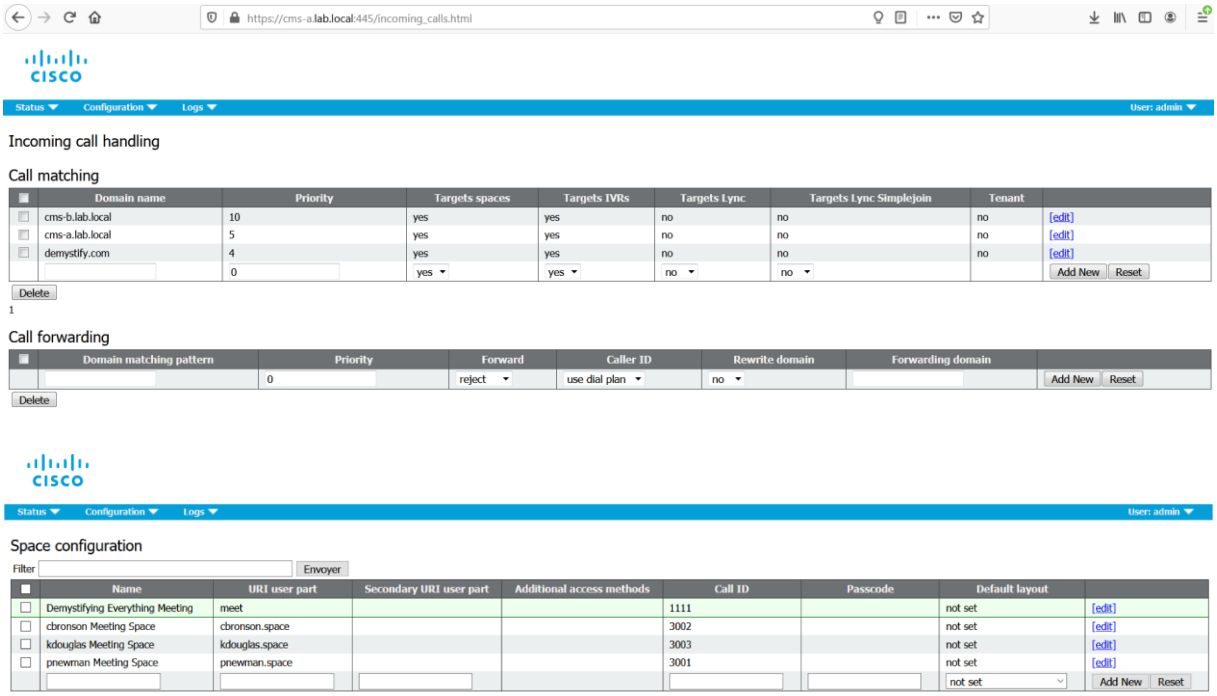


This is the first step in the process in which CMS determines whether the inbound call is destined for the Cisco Meeting Server itself and would need to process on it further or whether it is a call destined for a different system in which CMS is the agent that interworks the call and handles both the media and signaling (f.e. Skype gateway calls to Standard SIP endpoints or vice versa).

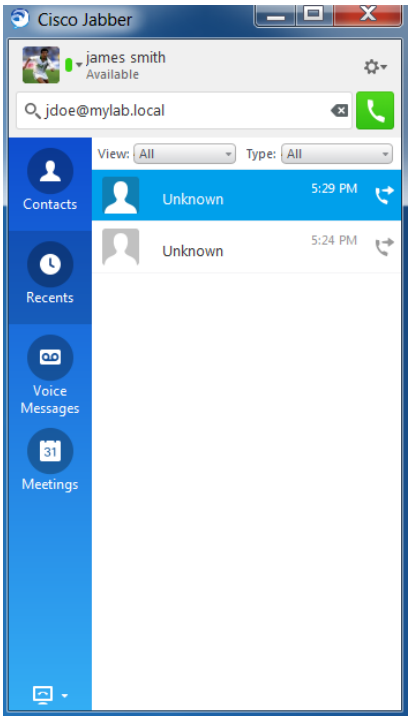
It checks if the domain part of the incoming URI matches the incoming matching table or not. If it does match, it is able to route the call to space, user, IVR or do a Lync conference lookup.

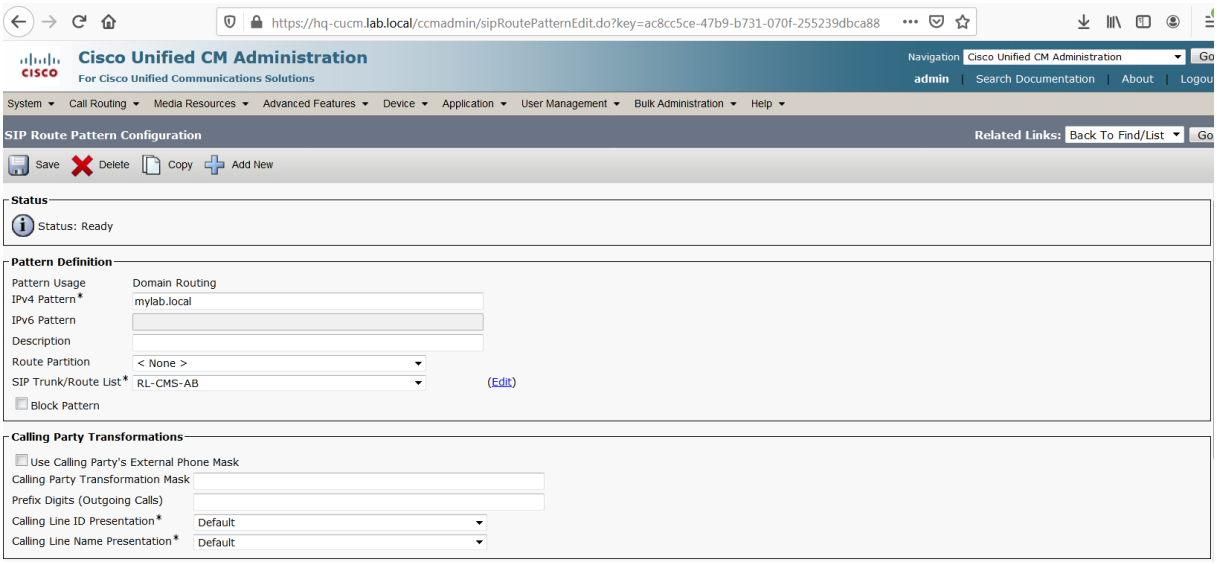
For example. When a user registered to **HQ-CUCM** wants to join this meeting, he dials **meet@demystify.com**, the call is sent to the call control **HQ-CUCM**, a SIP route pattern with domain routing **demystify.com** should be configured on **HQ-CUCM** that points to the SIP Trunk to CMS Cluster (the configuration will be done later), the CMS lookup the host portion **@demystify.com** for a matching in the Incoming Calls Table and a Domain name **demystify.com** exists, then the CMS lookup the User portion **meet@** to find a matching in

the Spaces Table and a space with User Portion URI exists, finally the user joins the meeting named Demystifying Everything Meeting.



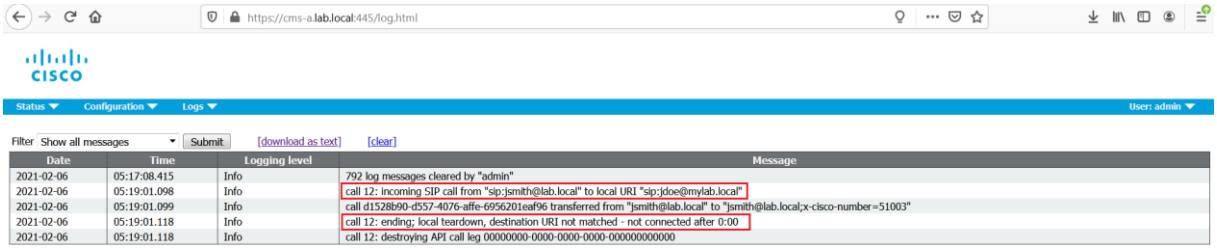
To understand how Cisco Meeting Server processes the calls through the three tables. Let's dial jdoe@mylab.local from jsmith jabber client, the call is sent to the **HQ-CUCM**, the call control **HQ-CUCM** has a SIP Route Pattern with domain routing **mylab.local** that points to **CMS-A.**





In the SIP Trace of **CMS-A**, we can see that since the call did not match any incoming rule with domain name **mylab.local**. The **CMS-A** goes to the Call Forwarding Table.

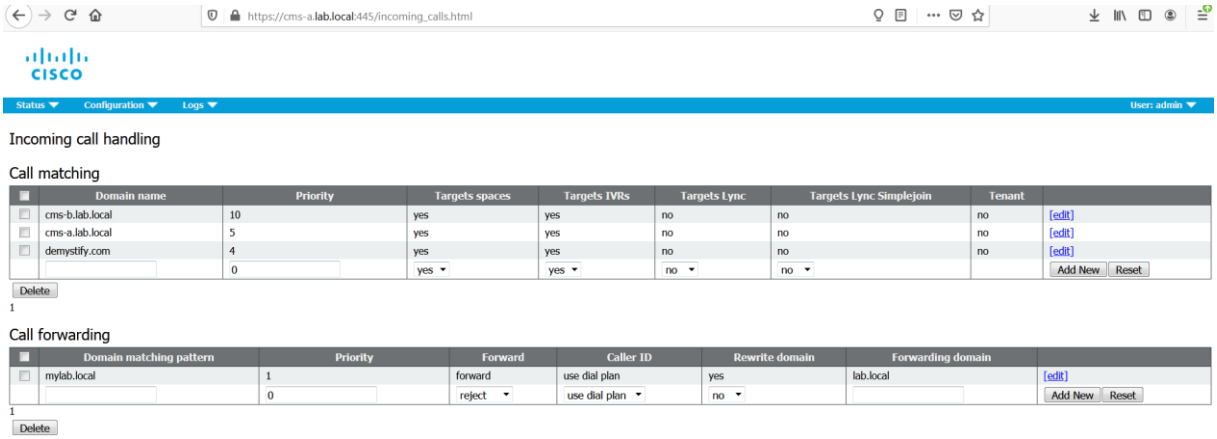
The Forwarding Table does not have any rule or a reject rule, then the event log does not explicitly show this. **CMS-A** just informs us that the call did not match a **Destination URI** in other words any space, user, IVR or Lync meeting.



Incoming Call Forwarding Table

If the call did not match any of the rules on the incoming call matching table, it goes through the second table called the call forwarding table. This is domain based only, you can block calls to certain domains or allow calls to certain domains.

Let's configure a Call Forwarding Rule with the Domain matching pattern **mylab.local**. The **Rewrite Domain** option allows you to rewrite the domain of the inbound call to a different one and changes the To header of the SIP message as well. To ensure that the jdoe@lab.local jabber client will ring, the domain dialed **mylab.local** must be changed to **lab.local**.



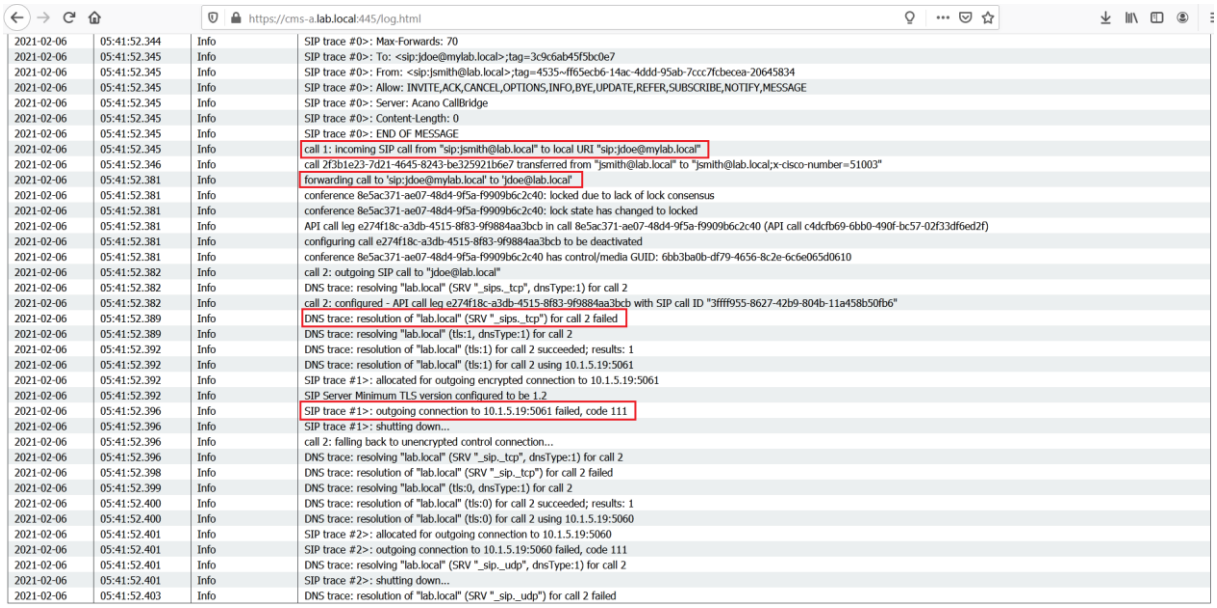
Let's try another call from jsmith jabber client to jdoe@lab.local by dialing jdoe@mylab.com.

The SIP Trace shown that an inbound call with domain **mylab.com** is received from jsmith@lab.local to jdoe@mylab.local but without a match on the incoming call table (either on spaces, users, IVR or Skype conferences), next the **CMS-A** lookup the Call Forwarding Table.

A match is found with the **domain matching pattern mylab.local** to process the call.

In the SIP Trace, the **Forwarding Call** line shows the modification of the domain. It displays the modification of **mylab.local** domain to **lab.local** domain.

But when the **CMS-A** lookup the Outbound Calls Table, there is no match and the outgoing call fails.



Outbound Call Table

This is the last table in the call routing logic of Cisco Meeting Server that makes the call out to different server. In other words CMS will use an outbound rule to send the call to another call control for example Cisco Unified Communication Manager or Cisco Expressway-C.

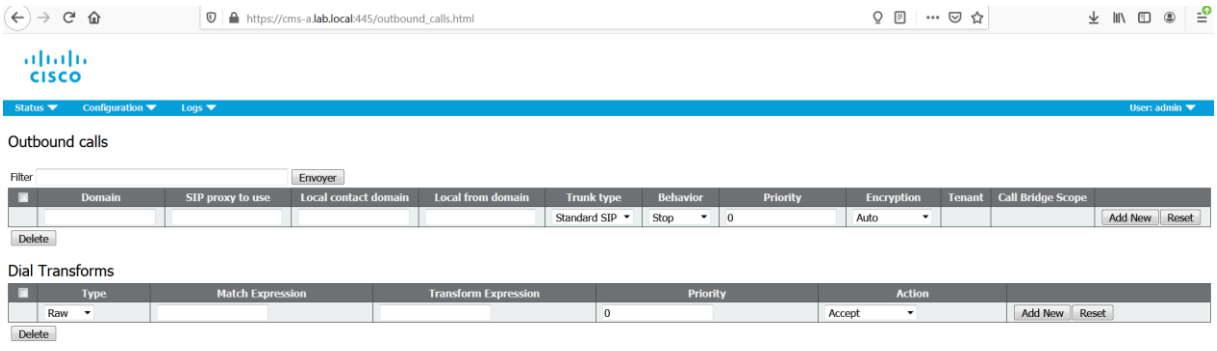
To better understand let's take another scenario:

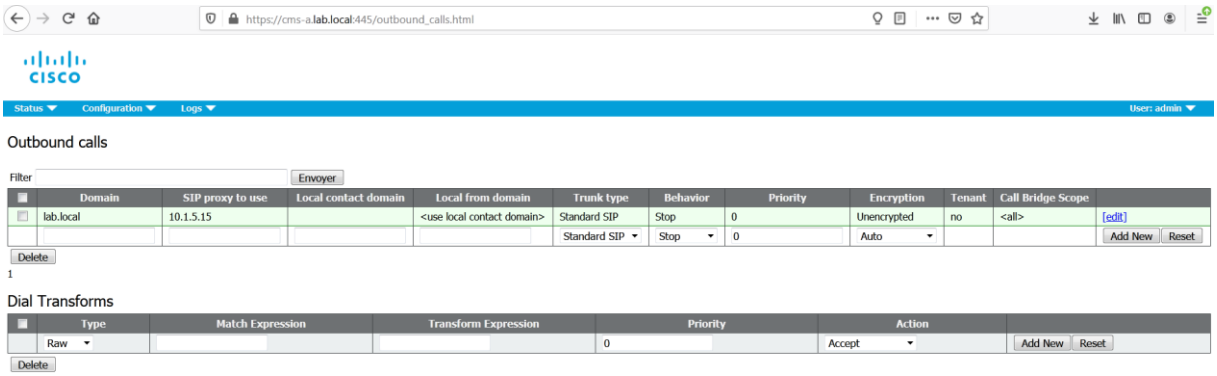
1. User jdoe@lab.local is registered using Cisco Jabber to **HQ-CUCM**.
2. Active Directory is integrated to CMS Cisco Meeting Server and the user jdoe@lab.local is NOT imported into CMS.
3. Another user join a conference using a web browser.
4. An administrator tries to add the user jdoe@lab.local to meeting using the **"Add Participant"** option in Cisco Meeting Management CMM.
5. The CMS will try to find an Outbound Call Rule with Domain **lab.local** and the **SIP Proxy to use** (in this case **10.1.5.15**) to route out the call, an Outbound Call Rule is similar to SIP Route Pattern on CUCM.
6. The **HQ-CUCM** will receive the call and find that the user jdoe@lab.local is registered and the Client Jabber rings.

Therefore to allow the **CMS-A** to route out or to send a call to any domain for example **lab.local**, the Outbound Calls Table is there to accomplish this task after the Incoming Calls Table and Call Forwarding Table checked respectively.

In case you match a rule with the Behavior set to Stop, the call does not go through the rest of the table after that match and the call has failed if that SIP Proxy failed to route the call for example. When that setting would be set to Continue, you could allow for a fallback to a different route or different node in the cluster.

Let's add **Outbound Call Rule**, in the **Domain**, enter **lab.local**, in the **SIP proxy to use**, enter the IP address of **HQ-CUCM 10.1.5.15**, we can use **hq-cucm.lab.local**. The CMS will query the internal DNS Server to resolve the IP address of **HQ-CUCM**.

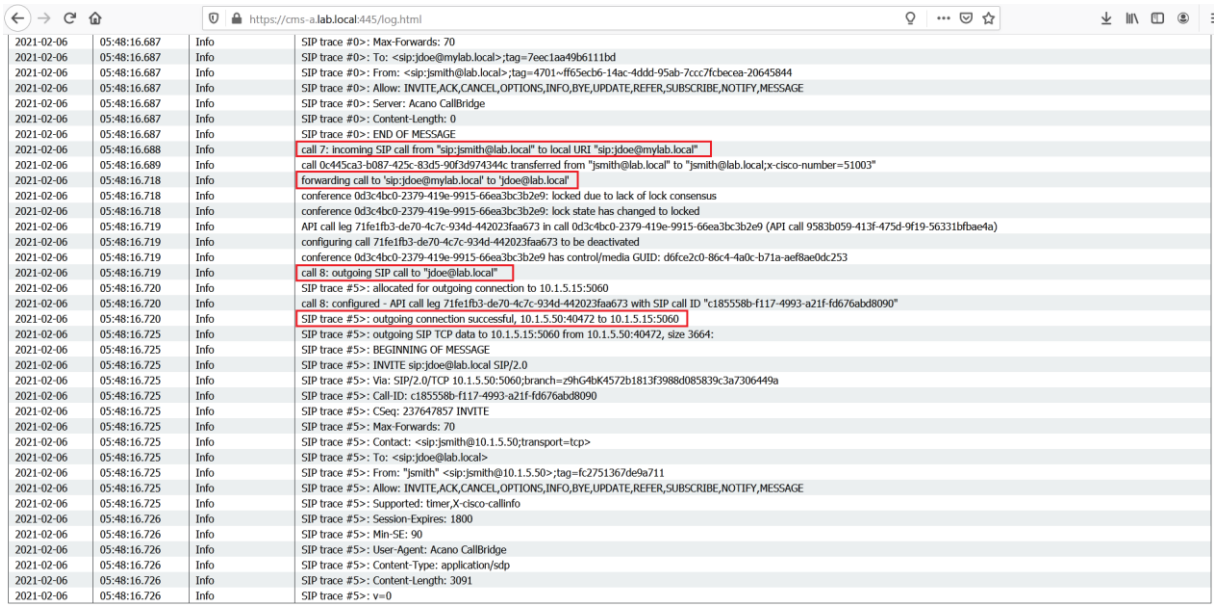




Let's try another call from jsmith jabber client to **jdoe@lab.local** by dialing **jdoe@mylab.com**..

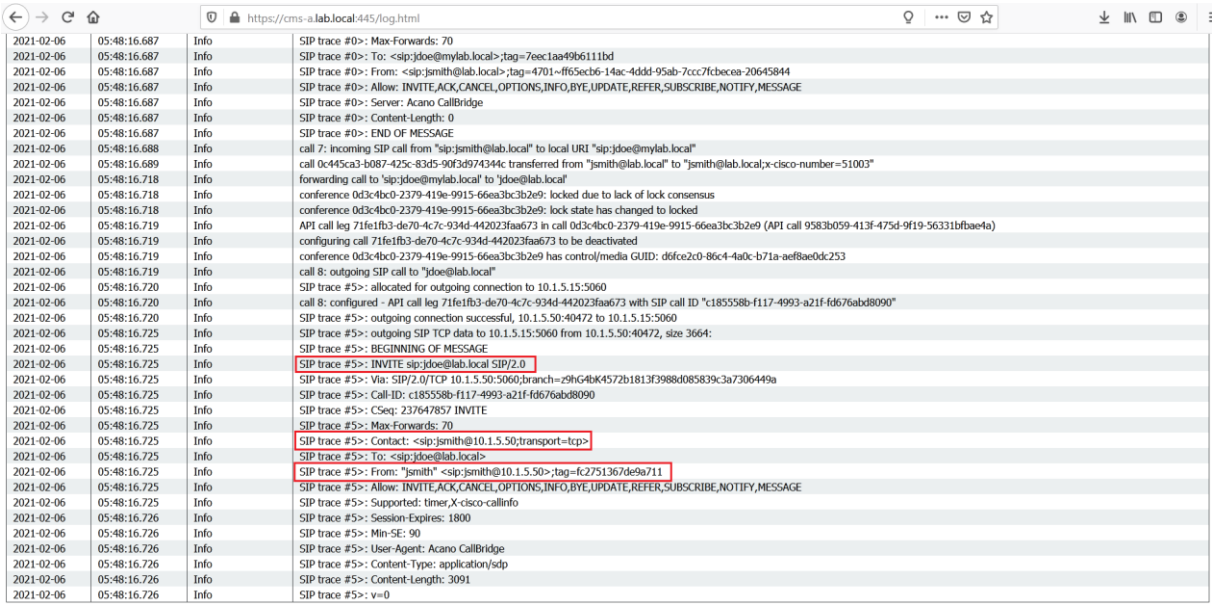
The SIP Trace shown that after a non-matching in the Incoming Calls Table, the **CMS-A** lookup in the Call Forwarding Table and find a match for **mylab.local**, it applies the transformation of the Host part of the URI of the destination **jdoe@mylab.local**, from **mylab.local** to **lab.local**. **Caller ID** of the Call Forwarding Rule is set to use **Dial Plan** option. There are two options here that can be set on the forwarding rules. Either it is set to **pass through** and then no modification is made on domain of the **outbound INVITEs** or it is set to use **dial plan** which allows the system to modify the domain of the inbound call to a different one as per the Outbound Calls rules, in this case the Call Forwarding Rule with **Caller ID** set to **Dial Plan** will modify the destination domain **mylab.local** to **lab.local** in order to match the Outbound Call Rule configured with a domain **lab.local**.

The **outgoing connection successful** line confirms that a match is found in the Outbound Calls table.



As mentioned for Call Forwarding Table, for **Caller ID** field, there are two options here that can be set on the forwarding rules. Either it is set to **pass through** and then no modification is made on the **From header** of the outbound INVITEs or it is set to use dial plan which allows the system to modify the **From header** as per the outbound rules.

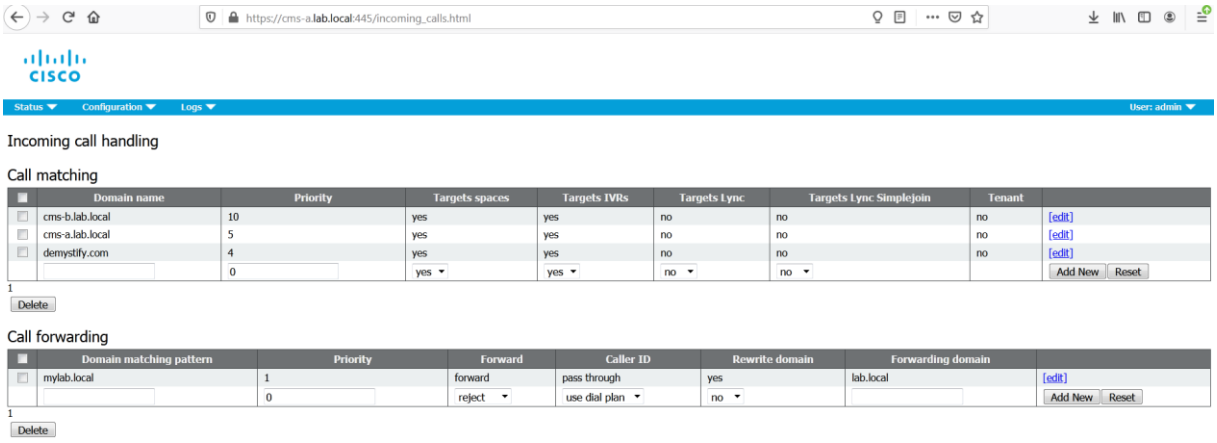
Below there is an outbound dial plan rule to **lab.local** (as after the rewrite domain on the call forwarding table). Notice the From header **jsmith@10.1.5.50** (IP of the CallBridge) with **Caller ID** set to Dial Plan. The **Contact header** **jsmith@10.1.5.50** is always adapted as CMS CallBridge.



https://cms-a.lab.local:445/log.html

2021-02-06	05:48:16.687	Info	SIP trace #0>: Max-Forwards: 70
2021-02-06	05:48:16.687	Info	SIP trace #0>: To: <sip:jdoe@mylab.local>;tag=7eec1aa49b6111bd
2021-02-06	05:48:16.687	Info	SIP trace #0>: From: <sip:jsmith@lab.local>;tag=4701~ff65ecb6-14ac-4ddd-95ab-7ccc7fcbecea-20645844
2021-02-06	05:48:16.687	Info	SIP trace #0>: Allow: INVITE,ACK,CANCEL,OPTIONS,INFO,BYE,UPDATE,REFER,SUBSCRIBE,NOTIFY,MESSAGE
2021-02-06	05:48:16.687	Info	SIP trace #0>: Server: Acano CallBridge
2021-02-06	05:48:16.687	Info	SIP trace #0>: Content-Length: 0
2021-02-06	05:48:16.687	Info	SIP trace #0>: END OF MESSAGE
2021-02-06	05:48:16.688	Info	call 7: incoming SIP call from "sip:jsmith@lab.local" to local URI "sip:jdoe@mylab.local"
2021-02-06	05:48:16.689	Info	call 0c445ca3-b087-425c-83d5-90f3d974344c transferred from "jsmith@lab.local" to "jsmith@lab.local;x-cisco-number=51003"
2021-02-06	05:48:16.718	Info	forwarding call to 'sip:jdoe@mylab.local' to 'jdoe@lab.local'
2021-02-06	05:48:16.718	Info	conference 0d3c4bc0-2379-419e-9915-66ea3bc3b2e9: locked due to lack of lock consensus
2021-02-06	05:48:16.718	Info	conference 0d3c4bc0-2379-419e-9915-66ea3bc3b2e9: lock state has changed to locked
2021-02-06	05:48:16.719	Info	API call leg 71fe1fb3-de70-4c7c-934d-442023faa673 in call 0d3c4bc0-2379-419e-9915-66ea3bc3b2e9 (API call 9583b059-413f-475d-9f19-56331bfbae4a)
2021-02-06	05:48:16.719	Info	configuring call 71fe1fb3-de70-4c7c-934d-442023faa673 to be deactivated
2021-02-06	05:48:16.719	Info	conference 0d3c4bc0-2379-419e-9915-66ea3bc3b2e9 has control/media GUID: d6fce2c0-86c4-4a0c-b71a-aef8ae0dc253
2021-02-06	05:48:16.719	Info	call 8: outgoing SIP call to "jdoe@lab.local"
2021-02-06	05:48:16.720	Info	SIP trace #5>: allocated for outgoing connection to 10.1.5.15:5060
2021-02-06	05:48:16.720	Info	call 8: configured - API call leg 71fe1fb3-de70-4c7c-934d-442023faa673 with SIP call ID "c185558b-f117-4993-a21f-fd676abd8090"
2021-02-06	05:48:16.720	Info	SIP trace #5>: outgoing connection successful, 10.1.5.50:40472 to 10.1.5.15:5060
2021-02-06	05:48:16.725	Info	SIP trace #5>: outgoing SIP TCP data to 10.1.5.15:5060 from 10.1.5.50:40472, size 3664:
2021-02-06	05:48:16.725	Info	SIP trace #5>: BEGINNING OF MESSAGE
2021-02-06	05:48:16.725	Info	SIP trace #5>: INVITE sip:jdoe@lab.local SIP/2.0
2021-02-06	05:48:16.725	Info	SIP trace #5>: Via: SIP/2.0/TCP 10.1.5.50:5060;branch=z9hG4bK4572b1813f3988d085839c3a7306449a
2021-02-06	05:48:16.725	Info	SIP trace #5>: Call-ID: c185558b-f117-4993-a21f-fd676abd8090
2021-02-06	05:48:16.725	Info	SIP trace #5>: CSeq: 237647857 INVITE
2021-02-06	05:48:16.725	Info	SIP trace #5>: Max-Forwards: 70
2021-02-06	05:48:16.725	Info	SIP trace #5>: Contact: <sip:jsmith@10.1.5.50;transport=tcp>
2021-02-06	05:48:16.725	Info	SIP trace #5>: To: <sip:jdoe@lab.local>
2021-02-06	05:48:16.725	Info	SIP trace #5>: From: "jsmith" <sip:jsmith@10.1.5.50>;tag=fc2751367de9a711
2021-02-06	05:48:16.725	Info	SIP trace #5>: Allow: INVITE,ACK,CANCEL,OPTIONS,INFO,BYE,UPDATE,REFER,SUBSCRIBE,NOTIFY,MESSAGE
2021-02-06	05:48:16.725	Info	SIP trace #5>: Supported: timer,X-cisco-callinfo
2021-02-06	05:48:16.726	Info	SIP trace #5>: Session-Expires: 1800
2021-02-06	05:48:16.726	Info	SIP trace #5>: Min-SE: 90
2021-02-06	05:48:16.726	Info	SIP trace #5>: User-Agent: Acano CallBridge
2021-02-06	05:48:16.726	Info	SIP trace #5>: Content-Type: application/sdp
2021-02-06	05:48:16.726	Info	SIP trace #5>: Content-Length: 3091
2021-02-06	05:48:16.726	Info	SIP trace #5>: v=0

Let's change the **Caller ID** in the Call Forwarding Rule to **pass through**. With pass through, there would not be any modification on the **From header** as shown below **jsmith@lab.local**. The **Contact header** is still set to **jsmith@10.1.5.50** as CMS starts a new callLeg and thus must add in a **Contact header** to itself.



https://cms-a.lab.local:445/incoming_calls.html

Status Configuration Logs User: admin

Incoming call handling

Call matching

	Domain name	Priority	Targets spaces	Targets IVRs	Targets Lync	Targets Lync Simplejoin	Tenant	
	cms-b.lab.local	10	yes	yes	no	no	no	[edit]
	cms-a.lab.local	5	yes	yes	no	no	no	[edit]
	demystify.com	4	yes	yes	no	no	no	[edit]
		0	yes	yes	no	no		Add New Reset

1

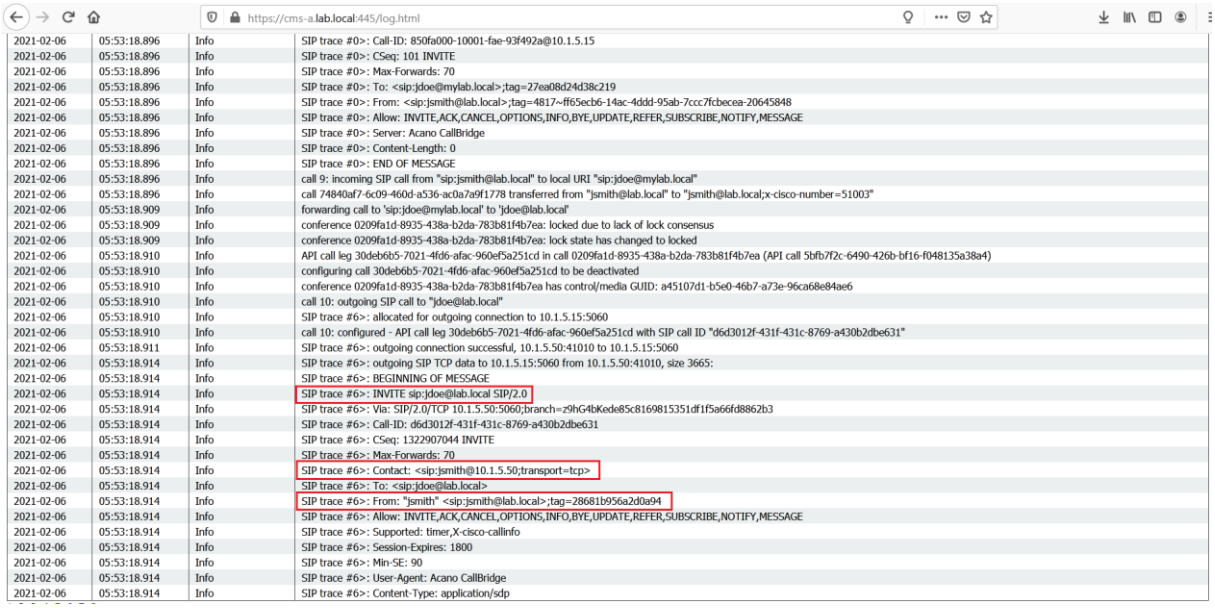
Delete

Call forwarding

	Domain matching pattern	Priority	Forward	Caller ID	Rewrite domain	Forwarding domain	
	mylab.local	1	forward	pass through	yes	lab.local	[edit]
		0	reject	use dial plan	no		Add New Reset

1

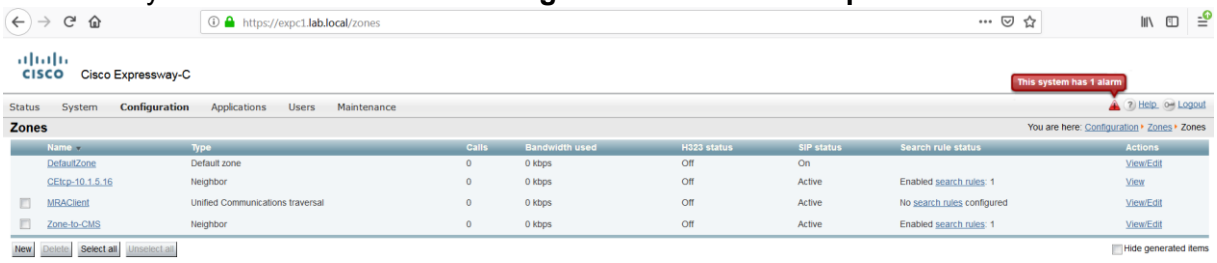
Delete



2021-02-06	05:53:18.896	Info	SIP trace #0>: Call-ID: 850fa000-10001-fae-93f492a@10.1.5.15
2021-02-06	05:53:18.896	Info	SIP trace #0>: CSeq: 101 INVITE
2021-02-06	05:53:18.896	Info	SIP trace #0>: Max-Forwards: 70
2021-02-06	05:53:18.896	Info	SIP trace #0>: To: <sip:jdoe@mylab.local>;tag=27ea08d24d38c219
2021-02-06	05:53:18.896	Info	SIP trace #0>: From: <sip:jsmith@lab.local>;tag=4817~ff65ecb6-14ac-4ddd-95ab-7ccc7fcbecea-20645848
2021-02-06	05:53:18.896	Info	SIP trace #0>: Allow: INVITE,ACK,CANCEL,OPTIONS,INFO,BYE,UPDATE,REFER,SUBSCRIBE,NOTIFY,MESSAGE
2021-02-06	05:53:18.896	Info	SIP trace #0>: Server: Acano CallBridge
2021-02-06	05:53:18.896	Info	SIP trace #0>: Content-Length: 0
2021-02-06	05:53:18.896	Info	SIP trace #0>: END OF MESSAGE
2021-02-06	05:53:18.896	Info	call 9: incoming SIP call from "sip:jsmith@lab.local" to local URI "sip:jdoe@mylab.local"
2021-02-06	05:53:18.896	Info	call 74840af7-6c09-460d-a536-ac0a7a9f1778 transferred from "jsmith@lab.local" to "jsmith@lab.local;x-cisco-number=51003"
2021-02-06	05:53:18.909	Info	forwarding call to 'sip:jdoe@mylab.local' to 'jdoe@lab.local'
2021-02-06	05:53:18.909	Info	conference 0209fa1d-8935-438a-b2da-783b81f4b7ea: locked due to lack of lock consensus
2021-02-06	05:53:18.909	Info	conference 0209fa1d-8935-438a-b2da-783b81f4b7ea: lock state has changed to locked
2021-02-06	05:53:18.910	Info	API call leg 30deb6b5-7021-4fd6-afac-960ef5a251cd in call 0209fa1d-8935-438a-b2da-783b81f4b7ea (API call 5bfb7f2c-6490-426b-bf16-f048135a38a4)
2021-02-06	05:53:18.910	Info	configuring call 30deb6b5-7021-4fd6-afac-960ef5a251cd to be deactivated
2021-02-06	05:53:18.910	Info	conference 0209fa1d-8935-438a-b2da-783b81f4b7ea has control/media GUID: a45107d1-b5e0-46b7-a73e-96ca68e84ae6
2021-02-06	05:53:18.910	Info	call 10: outgoing SIP call to "jdoe@lab.local"
2021-02-06	05:53:18.910	Info	SIP trace #6>: allocated for outgoing connection to 10.1.5.15:5060
2021-02-06	05:53:18.910	Info	call 10: configured - API call leg 30deb6b5-7021-4fd6-afac-960ef5a251cd with SIP call ID "d6d3012f-431f-431c-8769-a430b2dbe631"
2021-02-06	05:53:18.911	Info	SIP trace #6>: outgoing connection successful, 10.1.5.50:41010 to 10.1.5.15:5060
2021-02-06	05:53:18.914	Info	SIP trace #6>: outgoing SIP TCP data to 10.1.5.15:5060 from 10.1.5.50:41010, size 3665:
2021-02-06	05:53:18.914	Info	SIP trace #6>: BEGINNING OF MESSAGE
2021-02-06	05:53:18.914	Info	SIP trace #6>: INVITE sip:jdoe@lab.local SIP/2.0
2021-02-06	05:53:18.914	Info	SIP trace #6>: Via: SIP/2.0/TCP 10.1.5.50:5060;branch=z9hG4bKede85c8169815351df1f5a66fd8862b3
2021-02-06	05:53:18.914	Info	SIP trace #6>: Call-ID: d6d3012f-431f-431c-8769-a430b2dbe631
2021-02-06	05:53:18.914	Info	SIP trace #6>: CSeq: 1322907044 INVITE
2021-02-06	05:53:18.914	Info	SIP trace #6>: Max-Forwards: 70
2021-02-06	05:53:18.914	Info	SIP trace #6>: Contact: <sip:jsmith@10.1.5.50;transport=tcp>
2021-02-06	05:53:18.914	Info	SIP trace #6>: To: <sip:jdoe@lab.local>
2021-02-06	05:53:18.914	Info	SIP trace #6>: From: "jsmith" <sip:jsmith@lab.local>;tag=28681b956a2d0a94
2021-02-06	05:53:18.914	Info	SIP trace #6>: Allow: INVITE,ACK,CANCEL,OPTIONS,INFO,BYE,UPDATE,REFER,SUBSCRIBE,NOTIFY,MESSAGE
2021-02-06	05:53:18.914	Info	SIP trace #6>: Supported: timer,X-cisco-callinfo
2021-02-06	05:53:18.914	Info	SIP trace #6>: Session-Expires: 1800
2021-02-06	05:53:18.914	Info	SIP trace #6>: Min-SE: 90
2021-02-06	05:53:18.914	Info	SIP trace #6>: User-Agent: Acano CallBridge
2021-02-06	05:53:18.914	Info	SIP trace #6>: Content-Type: application/sdp

The **Local From Domain** and **Local Contact Domain** in the Call Outbound rule can be filled in to point to another domain **redouane.com** and **meddane.com** respectively. This then reflects the change on From **jsmith@redouane.com** and Contact header **jsmith@meddane.com** of the outbound SIP INVITE.

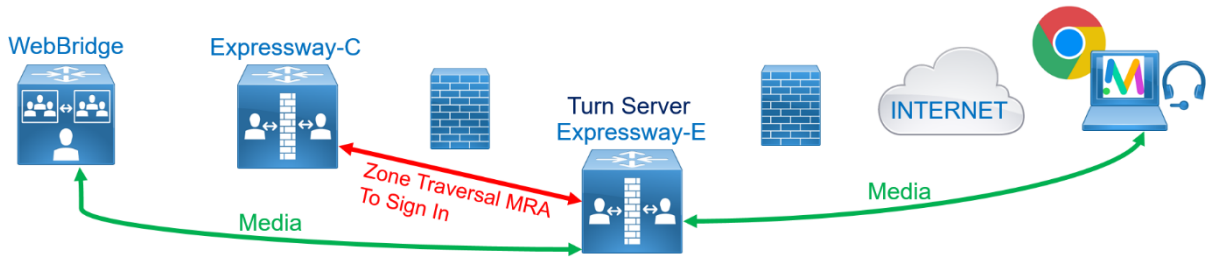
Let's modify the **Caller ID** of the **Incoming Call Rule** to use **dial plan**.



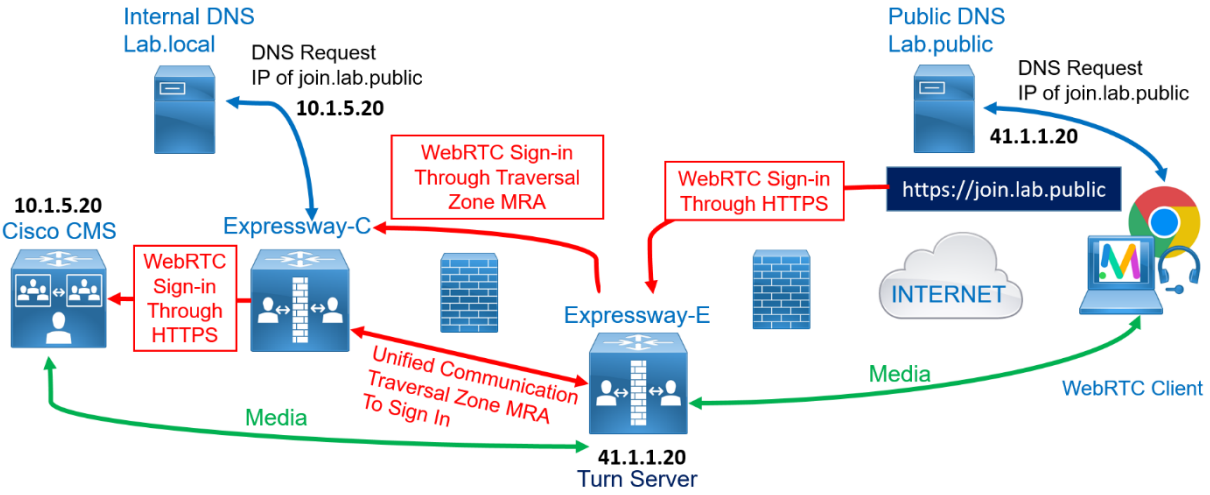
Cisco Expressway Unifed Edge

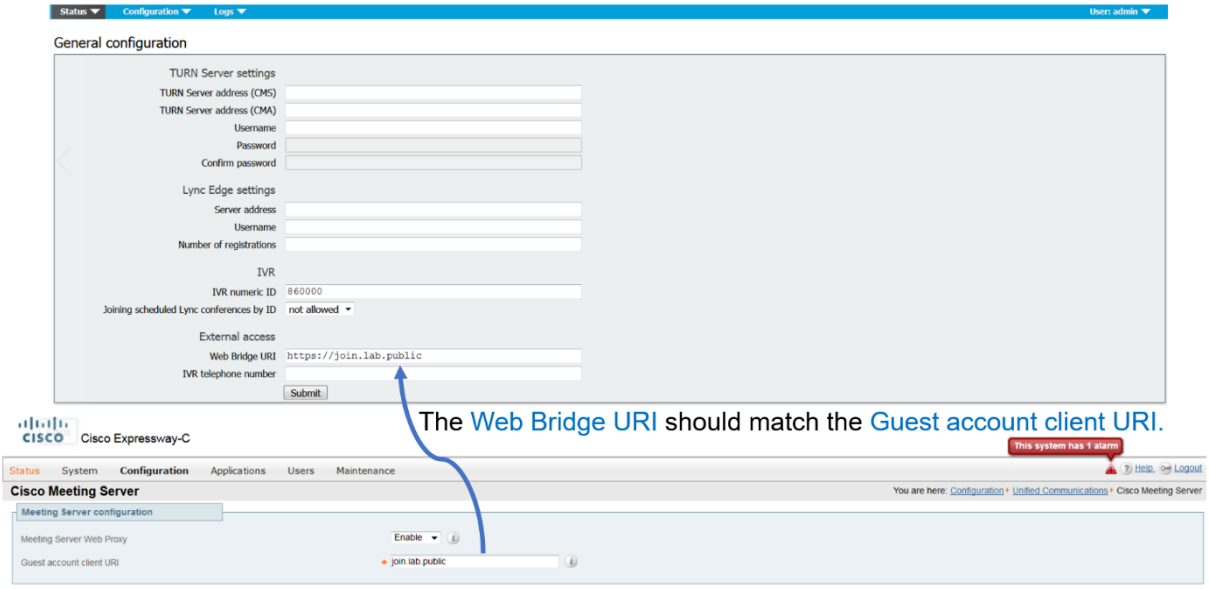
For users or guests to use the Cisco Meeting Server web app when outside the corporate network, access through the firewalls needs to be provided using the Cisco Expressways to proxy the WebRTC traffic from outside to in. The Cisco Expressway-E will proxy the HTTPS traffic back through an SSH tunnels to the Cisco Expressway-C. The Cisco Expressway-C will then route the traffic to an internal Web Bridge. For the Cisco Meeting Server web app to be able to access Web Bridge3.0 the Expressways must be running version X12.6 or later and if the Cisco Meeting Server web app is to use the Cisco Expressway-E TURN server then the TURN option key must be installed

WebRTC proxying can be used on the same Expressway pair that is used for Mobile and Remote Access (MRA), Business to Business (B2B) communication and any Microsoft Integration. The only service that WebRTC proxying cannot coexist with is Jabber Guest. Jabber Guest requires a dedicated pair of Expressways.



In an environment where the Cisco Expressways are proxying the HTTP traffic from outside to in, the external DNS server would route the Web Bridge URL from the browser of the external user to the Cisco Expressway-E. The Cisco Expressway-E would pass the traffic to the Cisco Expressway-C via the SSH tunnel that is established between them. The Cisco Expressway-C would use the internal DNS server to connect to one of the internal Web Bridges and forward the traffic. The Web Bridge would then treat the incoming traffic in the same way it would if the Cisco Meeting Server web app was inside the firewall. It would forward the necessary requests to the Call Bridge for processing such as user authentication, call requests and serve pages and manage the WebRTC connections back to the Cisco Meeting Server web app via the Expressways.



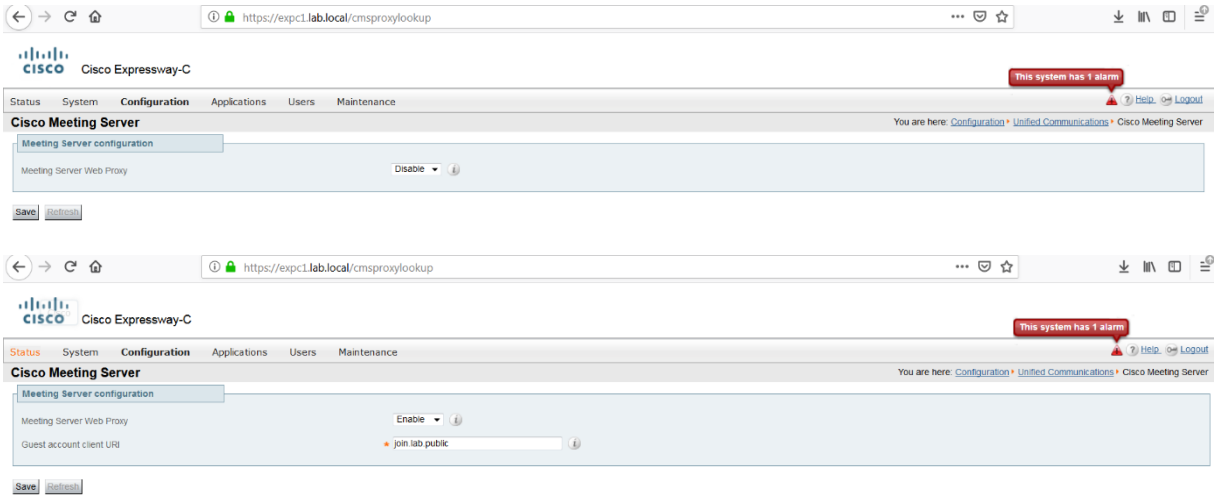


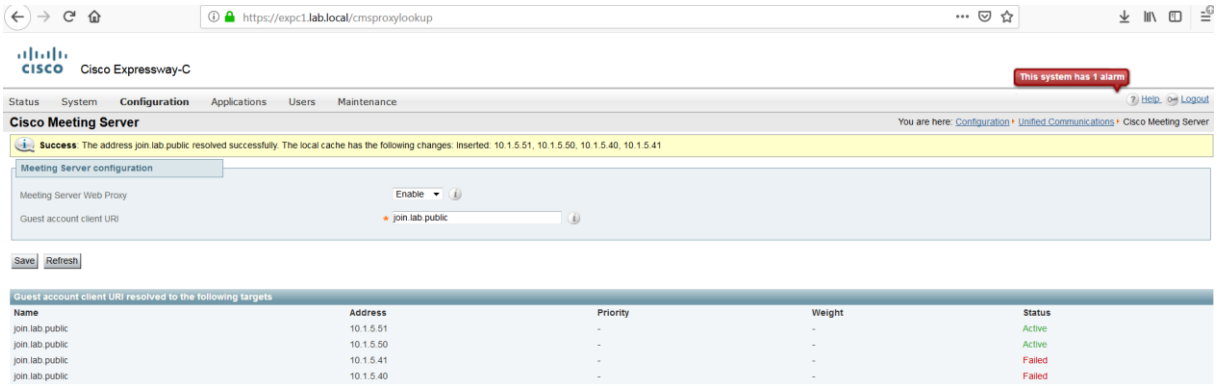
Configuring the Expressway-C will require enabling CMS meeting server web proxy and changing the guest account client URI.

On the web GUI of the Expressway-C expc1, navigate to **Configuration** > **Unified Communications** > **Cisco Meeting Server**.

Change the **Meeting Server Web Proxy** from **Disable** to **Enable**.

Enter **join.lab.public** in **the Guest account client URI**. This **Guest account client URI** should be resolved to the IP addresses of the CMS-1 and CMS-2 (**10.1.5.50** and **10.1.5.51**).

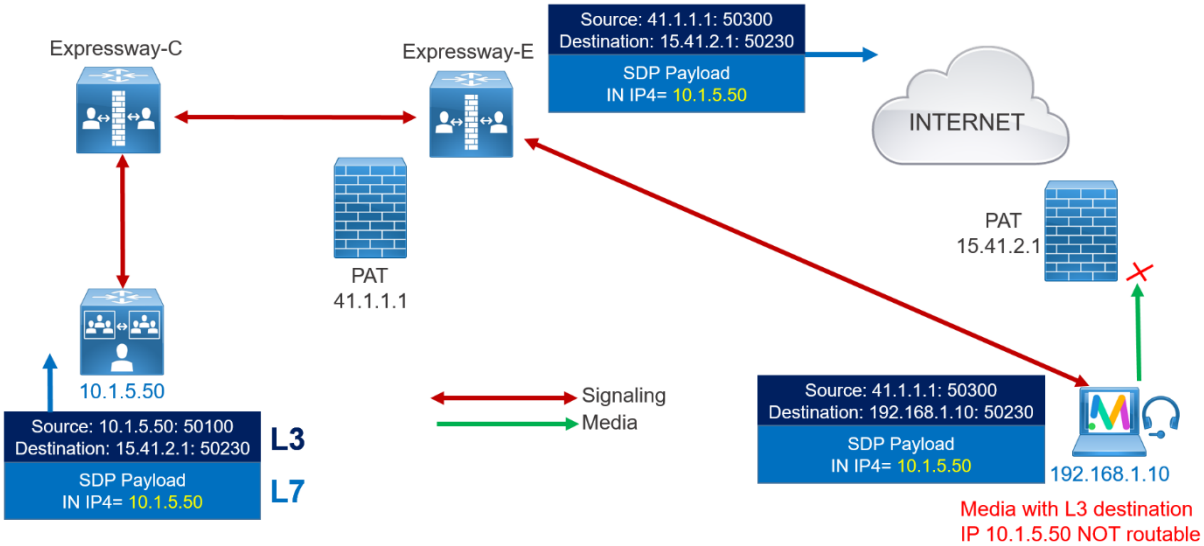




Configure the Expressway-E as the TURN server.
Why we do we need TURN? What is STUN TURN and ICE?

Before looking at the details of the different protocols and standards, it is important to understand why they are needed. ICE, STUN, and TURN are used to establish a MEDIA connection between two devices on different networks separated by firewalls and NAT servers. They do not apply to signaling between devices which means that if you cannot establish a signaling connection between the two devices then ICE, STUN and TURN are irrelevant. There must be a signaling connection established and only then will the devices attempt to establish a media connection.

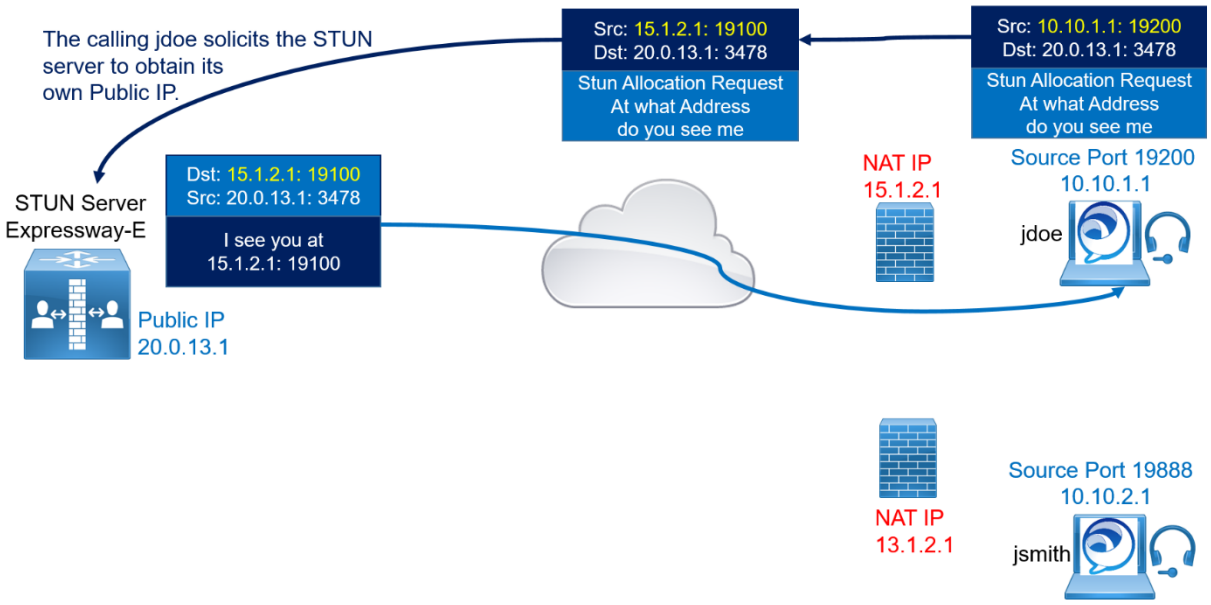
The issues with NAT and collaboration media connections occurs during the signaling. The media IP address and port of each stream is shared inside the SDP messages during call setup. So even when a packet has the source addresses changed by NAT, the media information inside the SDP messages remains the same. When the called device receives the SIP message, it doesn't try streaming the media to the source address on the received SIP packet but to the IP media addresses inside the SDP message which is the internal address of the device and therefore the media connection cannot be established. To overcome this issue ICE, STUN and TURN are used.



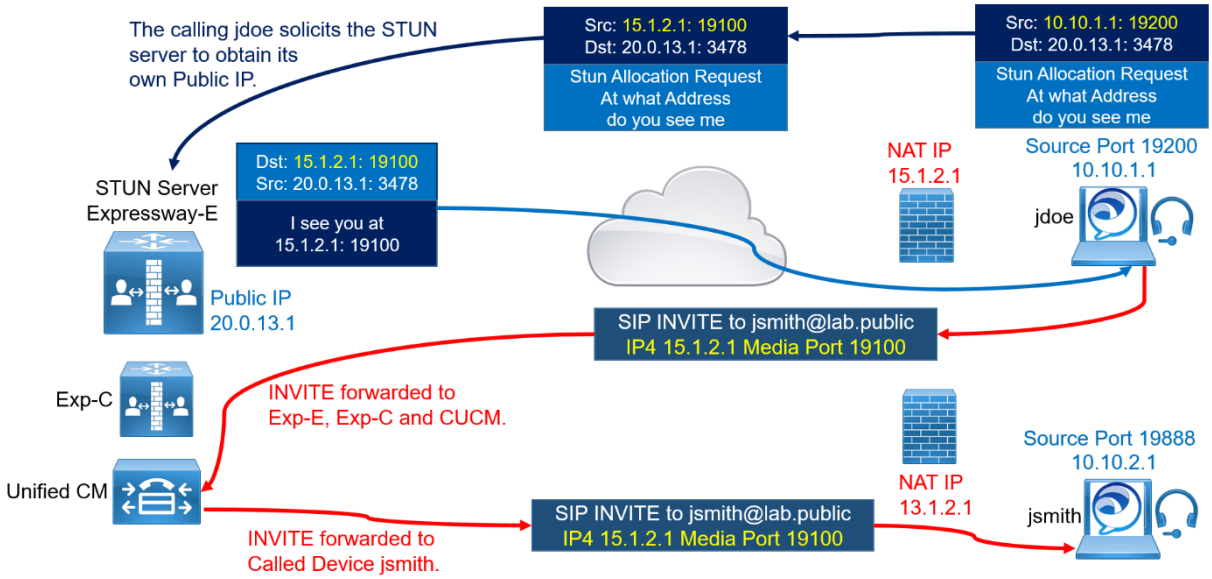
How are they used to traverse media through a Firewall and override the issue caused by Network Address Translation (NAT)? What is the role of each protocol? What are the differences? How the interaction occurs between STUN, TURN and ICE.

Let's start by looking at Session Traversal Utilities for NAT (STUN). The sole purpose of STUN is for a device behind a firewall to discover what its NAT'd address and port are when it sends UDP traffic outside. A STUN server is installed outside the firewall and the device inside sends a STUN request to the STUN server.

Before sending the INVITE, the calling endpoint jdoe starts a session with the STUN server to obtain its own client NATed IP 15.1.2.1 and port 19100.

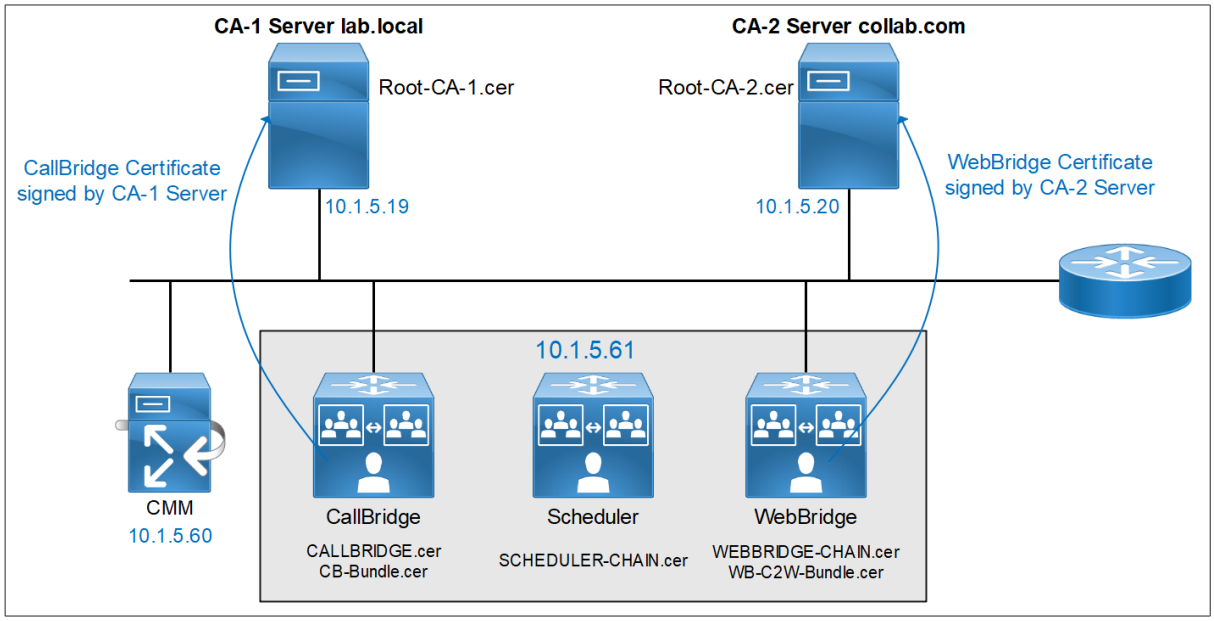


The client NATed IP 15.1.2.1 and port 19100 are sent in the SDP INVITE.
The INVITE is then forwarded by Expressway-E, Expressway-C, and Unified CM toward the called MRA device jsmith.



When the called device jsmith receives the SIP INVITE from jdoe, it starts communicating with the Expressway-E STUN server in order to obtain its own Client NATed IP 13.1.2.1 and port 1988

Project 3: Multiple Certificates with Multiple CA Servers



The CallBridge, Scheduler and WebBridge services are running in the same node.

1. The CallBridge service should use the CA-1 Server to sign the CallBridge certificate called CALLBRIDGE.cer.
2. The WebBridge service should use the CA-2 Server to sign the WebBridge certificate called WEBBRIDGE.cer.
3. The CallBridge must use the subordinate CA generated from CA-1 Server.
4. The WebBridge must use the subordinate CA generated from CA-2 Server.
5. Create a Bundle CA Called CB-Bundle.cer for CallBridge service using the Subordinate CA and Root certificate of CA-1 server.
6. Create a Bundle CA called WB-C2W-Bundle.cer for WebBridge service using the Subordinate CA and Root certificate of CA-2 server.
7. Create a chain certificate called WEBBRIDGE-CHAIN.cer for WebBridge3 using the previous subordinate CA, the Root certificate of CA-2 server and the WebBridge certificate.
8. Make sure that the CallBridge service will trust only the WebBridge certificate chain signed by only the certificate WB-C2W-Bundle.cer.
9. Make sure that the WebBridge service will trust only the CallBridge's certificate signed by only the CB-Bundle.cer certificate.
10. Enable the Scheduler, Since the Scheduler is required to run on a server which also has a colocated Callbridge, it is possible to use the Callbridge certificate and C2W trust cert for the Scheduler service, but a chain certificate is required for Scheduler, therefore bundle the certificates CALLBRIDGE.cer and CB-Bundle.cer used previously for CallBridge to create chain certificate SCHEDULER-CHAIN.cer.

Create a CSR named **CALLBRIDGE** for the CallBridge.

```
cms1>
cms1> pki csr CALLBRIDGE CN:cms1.lab.local subjectAltName:10.1.5.61,callbridge.lab.local
.........................+++++
.................................................+++++
Created key file CALLBRIDGE.key and CSR CALLBRIDGE.csr
CSR file CALLBRIDGE.csr ready for download via SFTP
cms1>
```

Create a CSR named **WEBBRIDGE** for the WebBridge.

```
cms1>
cms1> pki csr WEBBRIDGE CN:webbridge.collab.com subjectAltName:join.collab.com,meet.collab.com
.....................................................................................................+++++
...+++++
Created key file WEBBRIDGE.key and CSR WEBBRIDGE.csr
CSR file WEBBRIDGE.csr ready for download via SFTP
cms1>
```

Generate a CSR named **SuborCB** for a Subordinate CA of the CA Server lab.local.

```
cms1>
cms1> pki csr SuborCB CN:lab.local OU:CCNP O:Collaboration L:NY ST:NY C:US
.................+++++
...+++++
Created key file SuborCB.key and CSR SuborCB.csr
CSR file SuborCB.csr ready for download via SFTP
cms1>
```

Generate a CSR named **SuborWB** for a Subordinate CA of the CA Server collab.com.

```
cms1>
cms1> pki csr SuborWB CN:collab.com OU:CCNP O:Collaboration L:SJ ST:CAL C:US
.........................+++++
.........................................+++++
Created key file SuborWB.key and CSR SuborWB.csr
CSR file SuborWB.csr ready for download via SFTP
cms1>
```

From the **CA Server lab.local**. Submit the CSRs named **SuborCB** and **CALLBRIDGE** to generate the certificates, use the Certificate Template **Subordinate Certificate Authority**, and **Client and Server Authentication CMS** respectively.

Submit a Certificate Request or Renewal Request

To submit a saved request to the CA, paste a base-64-encoded CMC or PKCS #10 box.

Saved Request:

Base-64-encoded certificate request (CMC or PKCS #10 or PKCS #7):

```
2FqEZS51YLqxX0ixuXhFwBxLWWBx2+ZjeS4jzo9d
HvOtRVCMNYLz7zN6106VghprefUmT3knDoAGt1r0
IQMySNV4EF9O530I2gH9GHoikXyQVN3L3cLBcFK0
b1k=
-----END CERTIFICATE REQUEST-----
```

Certificate Template:

Subordinate Certification Authority

Additional Attributes:

Attributes:

Submit >

Submit a Certificate Request or Renewal Request

To submit a saved request to the CA, paste a base-64-encoded CMC or PKCS #10 box.

Saved Request:

Base-64-encoded certificate request (CMC or PKCS #10 or PKCS #7):

```
2FqEZS51YLqxX0ixuXhFwBxLWWBx2+ZjeS4jzo9d
HvOtRVCMNYLz7zN6106VghprefUmT3knDoAGt1r0
IQMySNV4EF9O545I2gH9GHoikXyQVN3L3cLBcFK0
F2e=
-----END CERTIFICATE REQUEST-----
```

Certificate Template:

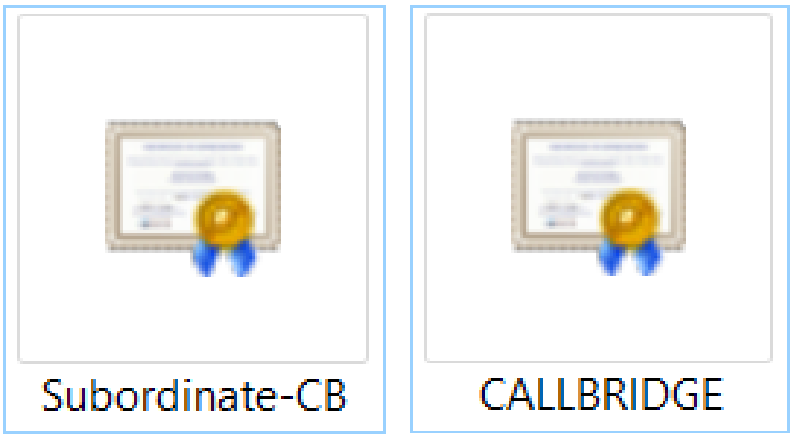
Client and Server Authentication CMS

Additional Attributes:

Attributes:

Submit >

Save the certificates with the name **Subordinate-CB.cer** and **CALLBRIDGE.cer** respectively.



From the **CA Server collab.com**. Submit the CSRs named **SuborWB** and **WEBBRIDGE** to generate the certificates, use the Certificate Template **Subordinate Certificate Authority**, and **Client and Server Authentication CMS** respectively.

Submit a Certificate Request or Renewal Request

To submit a saved request to the CA, paste a base-64-encoded CMC or PKCS #10 box.

Saved Request:

Base-64-encoded certificate request (CMC or PKCS #10 or PKCS #7):

```
2FqEZS51YLqxX0ixuXhFwBxLWWBx2+ZjeS4jzo9d
HvOtRVCMNYLz7zN6106VghprefUmT3knDoAGt1r0
IQMySNV4EF9O530I2gH9GHoikXyQVN3L3cLBcFK0
b1k=
-----END CERTIFICATE REQUEST-----
```

Certificate Template:

Subordinate Certification Authority

Additional Attributes:

Attributes:

Submit >

Submit a Certificate Request or Renewal Request

To submit a saved request to the CA, paste a base-64-encoded CMC or PKCS #10 box.

Saved Request:

Base-64-encoded certificate request (CMC or PKCS #10 or PKCS #7):

```
2FqEZS51YLqxX0ixuXhFwBxLWWBx2+ZjeS4jzo9d
HvOtRVCMNYLz7zN6106VghprefUmT3knDoAGt1r0
IQMySNV4EF9O545I2gH9GHoikXyQVN3L3cLBcFK0
F2e=
-----END CERTIFICATE REQUEST-----
```

Certificate Template:

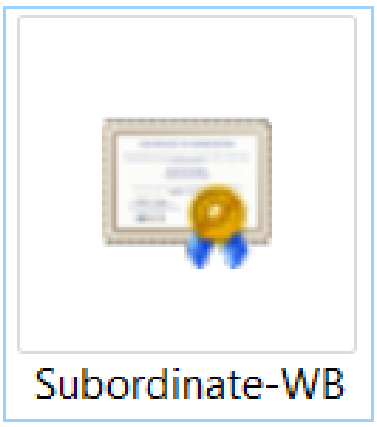
Client and Server Authentication CMS

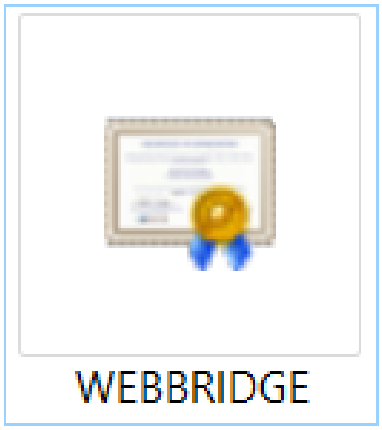
Additional Attributes:

Attributes:

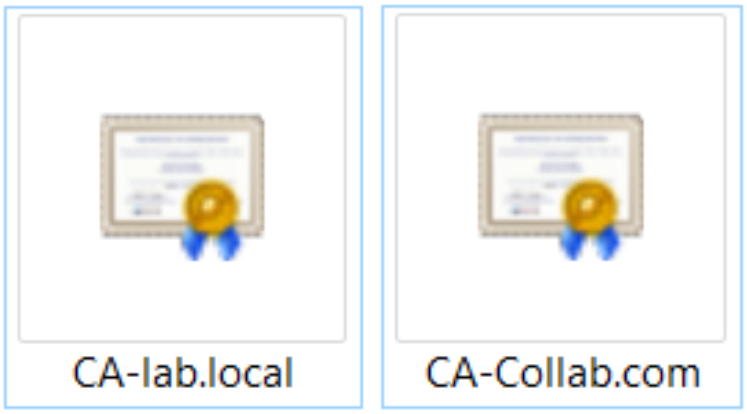
Submit >

Save the certificates with the name **Subordinate-WB.cer** and **WEBBRIDGE.cer** respectively.



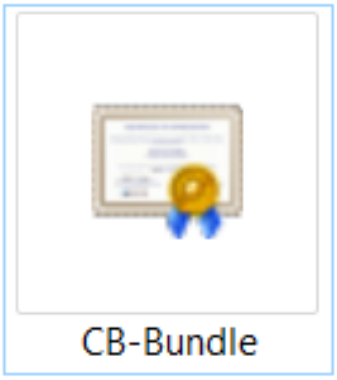


Download the Root Certificates of **lab.local** and **collab.com**.



Create a bundle CA for CallBridge.

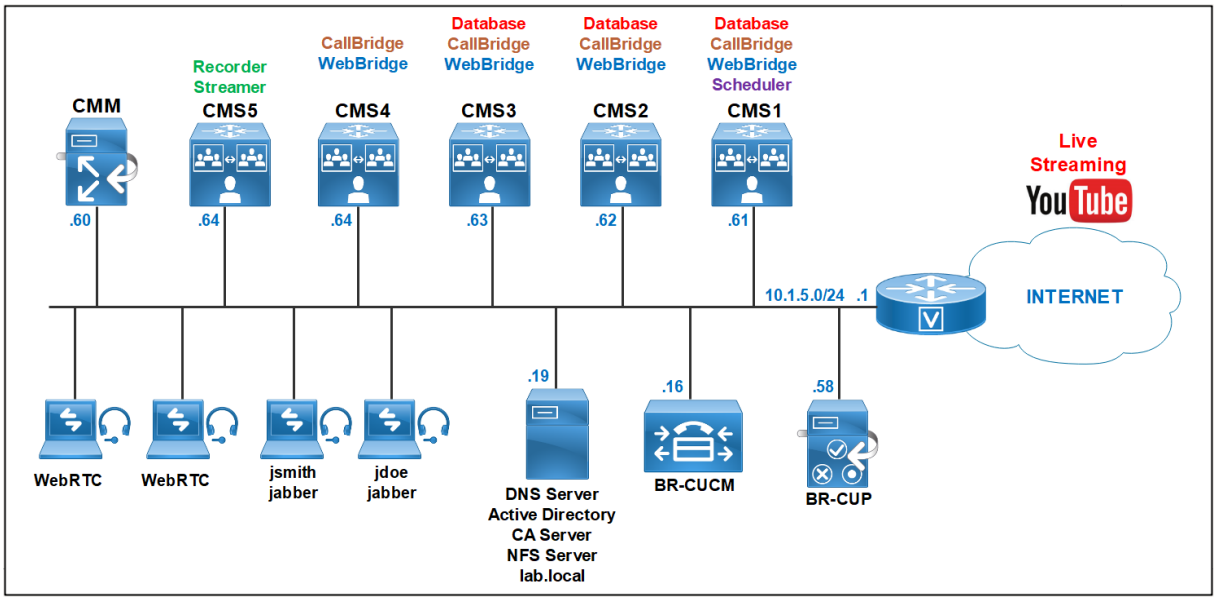
Use a plain text editor such as notepad. Past the **SuborCB.cer** certificate first and then past the **CA-lab.local.cer** certificate at the end, save the file with **.cer** extension. Name it **CB-Bundle.cer**.



Create a bundle CA for WebBridge.

Use a plain text editor such as notepad. Past the **SuborWB.cer** certificate first and then past the **CA-collab.com.cer** certificate at the end, save the file with **.cer** extension. Name it **WB-C2W-Bundle.cer**.

Project 4: Scalability and Resilience Deployment Certificates Advanced Scenario



Requirements:

- CMS1, CMS2 and CMS3 are planned to run a cluster database, CallBridge and WebBridge.
- CMS1 must be configured with the Scheduler service.
- CMS4 must be added with the CallBridge and WebBridge services.
- CMS5 must be configured with the Recorder and Streamer services.

1. Cluster Database Configuration between cms1 cms2 and cms3
2. WebAdmin CallBridge and WebBridge Certificates
3. Enabling the Web Admin Service
4. Enabling the CallBridge service
5. Enabling the WebBridge 3 service
6. CallBridge cluster configuration
7. Active Directory integration
8. Cisco Unified Communication Manager Dial Plan
9. Cisco Meeting Server Dial Plan
10. Integration of new node cms4 with CallBridge and WebBridge without the database
11. Enabling the Recorder and Streamer services on a dedicated server
12. Enabling the Scheduler service

Cluster Database Configuration between cms1 cms2 and cms3

For the server certificate, you can use a multi-SAN certificate containing all the database server FQDN in the SAN attribute reducing thus the certificate management by using a single certificate for all nodes and all services.

For the client certificate, the Common Name (CN) must be set to postgres. When a server database receives a client certificate, they check that the CN field is equal to postgres to validate the authentication.

For database we need to generate two CSRs with the corresponding private keys, the client and the server.
Use one CMS to generates these two CSRs, once you get the server and client certificates from your CA, copy the two certificates with their private keys to all nodes using WinSCP.

For Server certificate use the following command, give a name for example **dbcert**, it is important to put the CN to the FQDN of the Master Database **cms1** and the FQDN of the slaves in the SAN. For example: cms1.lab.local in the Common Name, cms2.lab.local and cms3.lab.local in the SAN.

```
cms1>pki csr dbcert CN:cms1.lab.local OU:CCNP O:Collaboration L:lab
ST:local C:US subjectAltName:cms2.lab.local,cms3.lab.local
```

For Client certificate use the following command, give a name for example **dbclt**.

```
cms1>pki csr dbclt CN:postgres
```

On CMS1.

Configure the database client and server certificates created previously and named **dbcert** and **dbclt**. The **Root-CA** certificate is added to verify the validity of the client/ server certificates. Specify which interface to use for the database clustering and initialize the master database.

```
cms1>database cluster certs dbcert.key dbcert.cer dbclt.key dbclt.cer Root-
CA.cer
cms1>database cluster localnode a
cms1>database cluster initialize
```

On CMS2 and CMS3.

Configure the database client and server certificates created previously and named **dbcert** and **dbclt**. The **Root-CA** certificate is added to verify the validity of the client/ server certificates, specify the interface to use, connect **cms2** and **cms3** to the master database **cms1**.

```
cmsx>database cluster certs dbcert.key dbcert.cer dbclt.key dbclt.cer Root-
CA.cer
cmsx>database cluster localnode a
cmsx>database cluster join cms1.lab.local
```

```
cms2>
cms2> database cluster join cms1.lab.local
WARNING!!!
Are you sure you wish to join this node to an existing database cluster? (Y/n)
 The contents of this node's database will be destroyed!
 The callbridge and web administration will restart at the end of this procedure.
 NOTE: This node is already in a cluster.
 Knowledge of all nodes in the old cluster will be removed
  Server certificate/key validated..
  Client certificate/key validated..
Please wait...
Join started...
cms2>
```

```
cms3>
cms3> database cluster join cms1.lab.local
WARNING!!!
Are you sure you wish to join this node to an existing database cluster? (Y/n)
 The contents of this node's database will be destroyed!
 The callbridge and web administration will restart at the end of this procedure.
  Server certificate/key validated..
  Client certificate/key validated..
Please wait...
Join started...
cms3>
```

On both **cms1**, **cms2** and **cms3**, Verify the status of the database cluster.

The database status on **CMS2** and **CMS3**. The status shown **ERROR Cannot find primary node in cluster**.

```
cms2>
cms2> database cluster status
ERROR                   : Cannot find primary node in cluster
Status                  : Error

Nodes:
    10.1.5.61           : Disconnected
Node in use             : None

Interface               : a

VerifyMode              : verify-ca

Certificates
  Server Key            : dbcert.key
  Server Certificate    : dbcert.cer
  Client Key            : dbclt.key
  Client Certificate    : dbclt.cer
  CA Certificate        : Root-CA.cer

Last command            : 'database cluster join cms1.lab.local' (Failed)

cms2>
```

```
cms3>
cms3> database cluster status
ERROR                       : Cannot find primary node in cluster
Status                      : Error

Nodes:
    10.1.5.61               : Disconnected
Node in use                 : None

Interface                   : a

VerifyMode                  : verify-ca

Certificates
  Server Key                : dbcert.key
  Server Certificate        : dbcert.cer
  Client Key                : dbclt.key
  Client Certificate        : dbclt.cer
  CA Certificate            : Root-CA.cer

Last command                : 'database cluster join cms1.lab.local' (Failed)

cms3>
```

CMS2 and **CMS3** fail to connect to the primary node **CMS1**. So what's wrong here? let's check the logs using the **syslog follow** command on **CMS** 2 and **CMS3**.

The output of **CMS2**'s log indicates the message **CMS1: error could not translate host name "cms1 to address.**

CMS2 tries to resolve the server's name **CMS1** but cannot find the IP address 10.1.5.61.

```
Sep 24 16:35:32.833 user.info cms2 sfpool:  Health check cms1: error (up = 1): could not translate host name "cms1" to address: Tempora
ry failure in name resolution|
Sep 24 16:35:34.775 user.info cms2 sfpool:  Failover Monitor: Unexpected roll call discrepancy; failover saw 0 connected node(s) (), 1
node(s) up (10.1.5.61)
Sep 24 16:35:34.848 user.info cms2 sfpool:  Health check cms1: error (up = 1): could not translate host name "cms1" to address: Tempora
ry failure in name resolution|
Sep 24 16:35:36.851 user.info cms2 sfpool:  Health check cms1: error (up = 1): could not translate host name "cms1" to address: Tempora
ry failure in name resolution|
Sep 24 16:35:37.782 user.info cms2 sfpool:  Failover Monitor: Unexpected roll call discrepancy; failover saw 0 connected node(s) (), 1
node(s) up (10.1.5.61)
Sep 24 16:35:38.839 user.info cms2 sfpool:  Health check cms1: error (up = 1): could not translate host name "cms1" to address: Tempora
ry failure in name resolution|
^Ccms2>
cms2>
```

The output of **CMS3**'s log indicates the message **CMS1: error could not translate host name "cms1 to address.**

CMS3 tries to resolve the server's name **CMS1** but find the IP address 10.1.5.61.

```
Sep 24 16:36:48.034 user.info cms3 sfpool:  Health check cms1: error (up = 1): could not translate host name "cms1" to address: Tempora
ry failure in name resolution|
Sep 24 16:36:48.947 user.info cms3 sfpool:  Failover Monitor: Unexpected roll call discrepancy; failover saw 0 connected node(s) (), 1
node(s) up (10.1.5.61)
Sep 24 16:36:50.020 user.info cms3 sfpool:  Health check cms1: error (up = 1): could not translate host name "cms1" to address: Tempora
ry failure in name resolution|
Sep 24 16:36:51.960 user.info cms3 sfpool:  Failover Monitor: Unexpected roll call discrepancy; failover saw 0 connected node(s) (), 1
node(s) up (10.1.5.61)
Sep 24 16:36:52.035 user.info cms3 sfpool:  Health check cms1: error (up = 1): could not translate host name "cms1" to address: Tempora
ry failure in name resolution|
^Ccms3>
cms3>
```

To solve the issues, we need DNS A record to resolve the name **CMS1** to the IP address **10.1.5.61**. Fortunately, Cisco Meeting server allows to create a DNS RR record to resolve the server's name locally.

On **CMS2** and **CMS3**, add a DNS RR record using the following commands.

```
cms2>
cms2>
cms2> dns add rr "cms1. IN A 10.1.5.61"
cms2>
cms2>
```

```
cms3>
cms3>
cms3> dns add rr "cms1. IN A 10.1.5.61"
cms3>
cms3>
```

Now let's verify the database status on **CMS2** and **CMS3**.

The output show that both are in the **Connected** status to the primary node **CMS1 10.1.5.61**.

But both nodes **CMS2** and **CMS3** fails to connect to each other.

```
cms2>
cms2> database cluster status
Status                    : Enabled

Nodes:
    10.1.5.61             : Connected Primary
    10.1.5.62 (me)        : Connected Replica ( In Sync )
    10.1.5.63             : Disconnected
Node in use               : 10.1.5.61

Interface                 : a

VerifyMode                : verify-ca

Certificates
  Server Key              : dbcert.key
  Server Certificate      : dbcert.cer
  Client Key              : dbclt.key
  Client Certificate      : dbclt.cer
  CA Certificate          : Root-CA.cer

Last command              : 'database cluster join cms1.lab.local' (Success)

cms2>
```

```
cms3>
cms3> database cluster status
Status                    : Enabled

Nodes:
    10.1.5.61             : Connected Primary
    10.1.5.62             : Disconnected
    10.1.5.63 (me)        : Connected Replica ( In Sync )
Node in use               : 10.1.5.61

Interface                 : a

VerifyMode                : verify-ca

Certificates
  Server Key              : dbcert.key
  Server Certificate      : dbcert.cer
  Client Key              : dbclt.key
  Client Certificate      : dbclt.cer
  CA Certificate          : Root-CA.cer

Last command              : 'database cluster join cms1.lab.local' (Success)

cms3>
```

Let's verify the logs on **CMS2** and **CMS3**.

CMS2 cannot find the IP address **10.1.5.63** of the hostname **CMS3**.

```
Sep 24 16:47:44.425 user.info cms2 sfpool:  Health check cms3: error (up = 1): could not translate host name "cms3" to address: Tempora
ry failure in name resolution|
Sep 24 16:47:44.864 user.info cms2 sfpool:  Failover Monitor: Unexpected roll call discrepancy; failover saw 2 connected node(s) (cms1
and cms2), 3 node(s) up (10.1.5.61, 10.1.5.62 and 10.1.5.63)
Sep 24 16:47:46.427 user.info cms2 sfpool:  Health check cms3: error (up = 1): could not translate host name "cms3" to address: Tempora
ry failure in name resolution|
Sep 24 16:47:47.887 user.info cms2 sfpool:  Failover Monitor: Unexpected roll call discrepancy; failover saw 2 connected node(s) (cms1
and cms2), 3 node(s) up (10.1.5.61, 10.1.5.62 and 10.1.5.63)
Sep 24 16:47:48.418 user.info cms2 sfpool:  Health check cms3: error (up = 1): could not translate host name "cms3" to address: Tempora
ry failure in name resolution|
Sep 24 16:47:50.421 user.info cms2 sfpool:  Health check cms3: error (up = 1): could not translate host name "cms3" to address: Tempora
ry failure in name resolution|
Sep 24 16:47:50.910 user.info cms2 sfpool:  Failover Monitor: Unexpected roll call discrepancy; failover saw 2 connected node(s) (cms1
and cms2), 3 node(s) up (10.1.5.61, 10.1.5.62 and 10.1.5.63)
^Ccms2>
cms2>
```

CMS3 cannot find the IP address **10.1.5.62** of the hostname **CMS2**.

```
Sep 24 16:48:28.256 user.info cms3 sfpool:  Failover Monitor: Unexpected roll call discrepancy; failover saw 2 connected node(s) (cms1
and cms3), 3 node(s) up (10.1.5.61, 10.1.5.62 and 10.1.5.63)
Sep 24 16:48:28.396 user.info cms3 sfpool:  Health check cms2: error (up = 1): could not translate host name "cms2" to address: Tempora
ry failure in name resolution|
Sep 24 16:48:30.415 user.info cms3 sfpool:  Health check cms2: error (up = 1): could not translate host name "cms2" to address: Tempora
ry failure in name resolution|
Sep 24 16:48:31.271 user.info cms3 sfpool:  Failover Monitor: Unexpected roll call discrepancy; failover saw 2 connected node(s) (cms1
and cms3), 3 node(s) up (10.1.5.61, 10.1.5.62 and 10.1.5.63)
Sep 24 16:48:32.414 user.info cms3 sfpool:  Health check cms2: error (up = 1): could not translate host name "cms2" to address: Tempora
ry failure in name resolution|
^Ccms3>
cms3>
```

On the primary node **CMS1**, verify the database status, we can see that the primary node fails to connect to **CMS2** and **CMS3**.

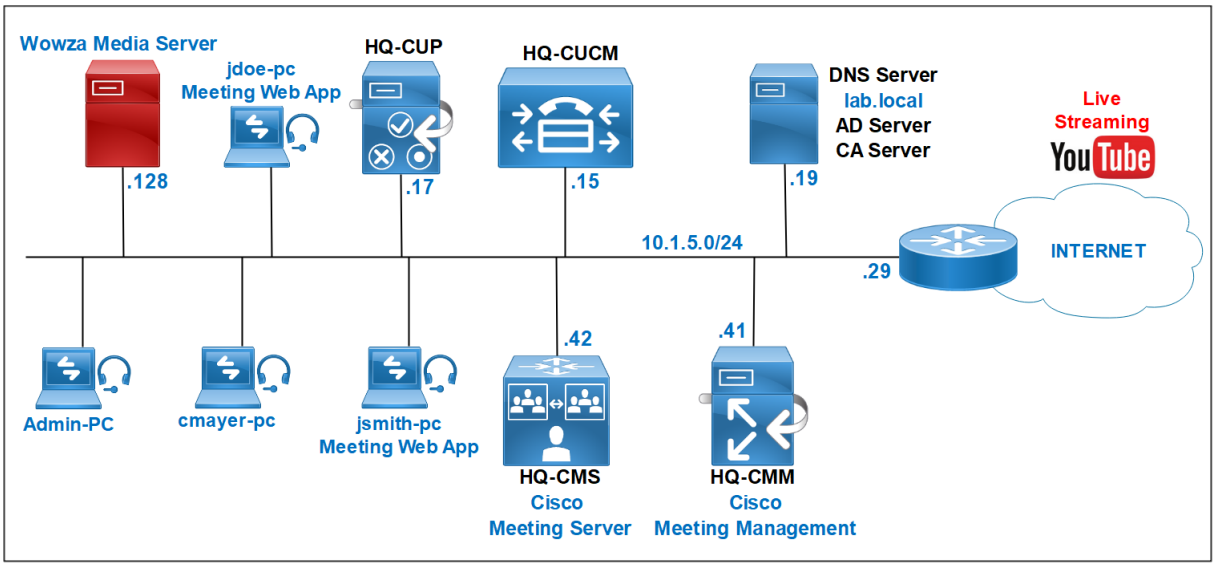
```
cms1>
cms1> database cluster status
Status                    : Enabled

Nodes:
    10.1.5.61 (me)        : Connected Primary
    10.1.5.62             : Disconnected
    10.1.5.63             : Disconnected
Node in use               : 10.1.5.61

Interface                 : a

VerifyMode                : verify-ca

Certificates
  Server Key              : dbcert.key
  Server Certificate      : dbcert.cer
  Client Key              : dbclt.key
  Client Certificate      : dbclt.cer
  CA Certificate          : Root-CA.cer

Last command              : 'database cluster initialize' (Success)

cms1>
```

Let's check the logs on **CMS1**. The same issue is displayed, **CMS1** fails to find the IP addresses of the server's name **CMS2** and **CMS3**.

```
Sep 24 16:46:50.723 user.info cms1 sfpool:  Failover Monitor: Unexpected roll call discrepancy; failover saw 1 connected node(s) (cms1)
, 3 node(s) up (10.1.5.61, 10.1.5.62 and 10.1.5.63)
Sep 24 16:46:51.149 user.info cms1 sfpool:  Health check cms2: error (up = 1): could not translate host name "cms2" to address: Tempora
ry failure in name resolution|
Sep 24 16:46:52.212 user.info cms1 sfpool:  Health check cms3: error (up = 1): could not translate host name "cms3" to address: Tempora
ry failure in name resolution|
Sep 24 16:46:53.167 user.info cms1 sfpool:  Health check cms2: error (up = 1): could not translate host name "cms2" to address: Tempora
ry failure in name resolution|
Sep 24 16:46:53.736 user.info cms1 sfpool:  Failover Monitor: Unexpected roll call discrepancy; failover saw 1 connected node(s) (cms1)
, 3 node(s) up (10.1.5.61, 10.1.5.62 and 10.1.5.63)
Sep 24 16:46:55.195 user.info cms1 sfpool:  Health check cms2: error (up = 1): could not translate host name "cms2" to address: Tempora
ry failure in name resolution|
^Ccms1>
cms1>
```

To solve the issues, we need to add two DNS RR Records on **CMS1** to resolve the server's name of **CMS2** and **CMS3**. Configure the following commands as shown below

Project 5: Streamer Service Certificates with Wowza Media Server



The new streamer component requires to listen to SIP connections, the streamer server must have a valid certificate for TLS connection.

Configure the listening interface of the streamer and the SIP TCP and TLS ports to listen on using the following command.

```
hq-cms> streamer sip listen a 6000 6001
```

Configure the certificates to be used for the SIP streamer. Specify the key file, certificate, and CA trust bundle.

```
hq-cms> streamer sip certs cmscert.key cmscert.cer Chain-CA.cer
```

Optionally select the quality for streaming

```
hq-cms> streamer sip resolution 720p
```

```
hq-cms>
hq-cms> streamer sip listen a 6000 6001
hq-cms> streamer sip certs cmscert.key cmscert.cer Chain-CA.cer
hq-cms> streamer sip resolution 720p
hq-cms> tls sip trust Chain-CA.cer
hq-cms>
```

Enable the streamer. All message must show "SUCCESS" as below displayed below.

```
hq-cms> streamer enable
```

```
hq-cms>
hq-cms> streamer enable
SUCCESS: Key and certificate pair match
SUCCESS: certificate verified against CA bundle
SUCCESS: Streamer enabled
hq-cms>
```

Verify the streamer status.

```
hq-cms>
hq-cms> streamer
Enabled                 : true
SIP interfaces          : tcp a:6000, tls a:6001
SIP key file            : cmscert.key
SIP certificate file    : cmscert.cer
SIP CA Bundle file      : Chain-CA.cer
SIP Resolution          : 720p
SIP traffic trace       : Disabled
Call Limit              : 5
hq-cms>
```

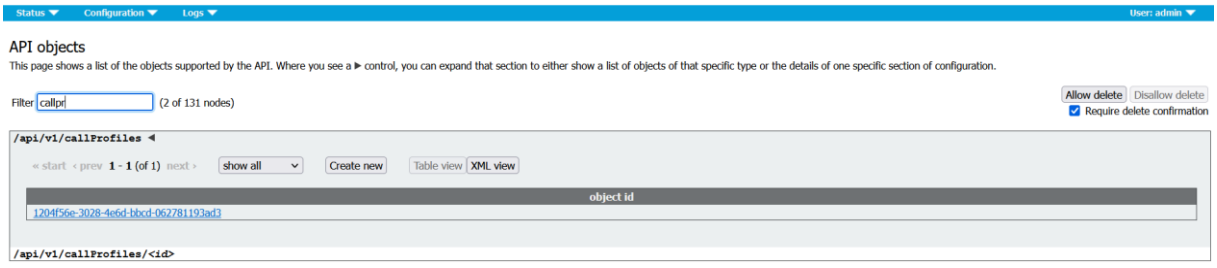
Configuring the Stream URI via the API.

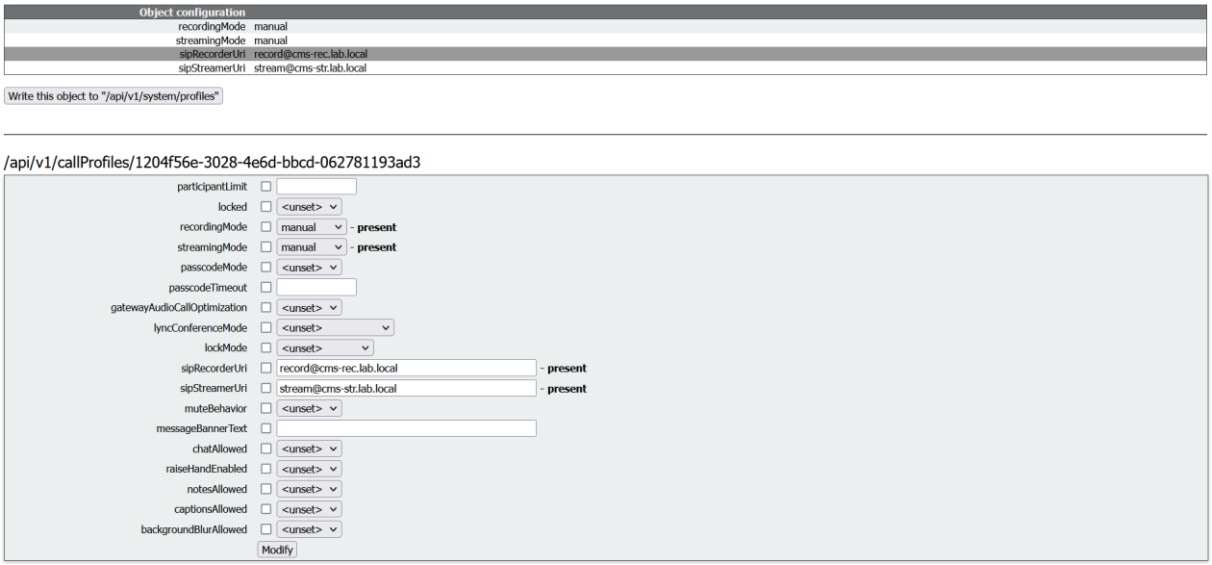
Once the new SIP streamer is enabled, it can be configured and used in the Call Bridge using the sipStreamerUri API parameter specified in the API call profile object.

On the web GUI of the CMS navigate to **Configuration > API**.

In the **Filter** section, type "**callprofile**"
Click the **Create new** button to create a new CallProfile or edit the existing profile and populate the following information.

- **streamingMode:** select manual from the drop menu
- **sipStreamerURI:** stream@cms-str.lab.local

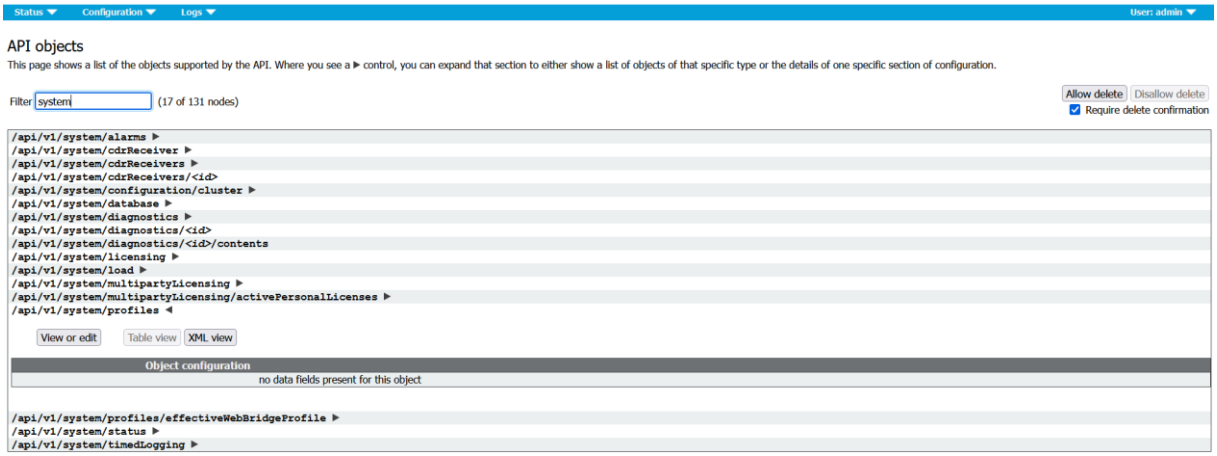


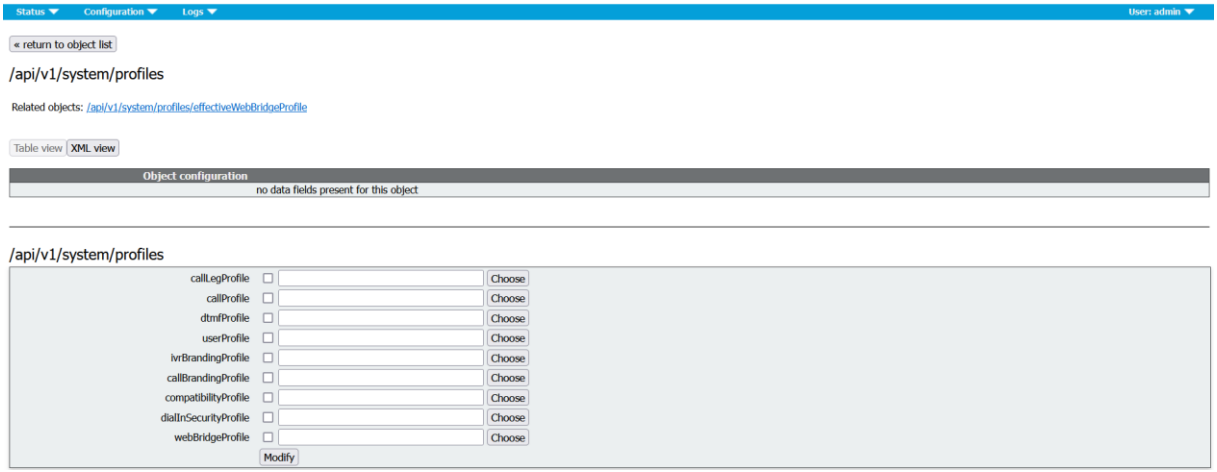


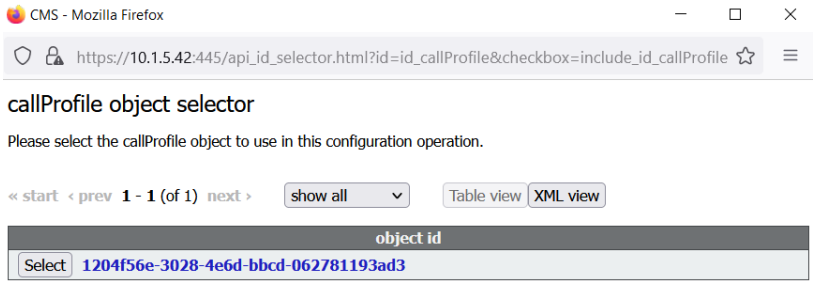
Add the created callProfile above to the /system/profiles. This is a global configuration and the configured "**sipStreamerUri**" will be used for streamer operation.

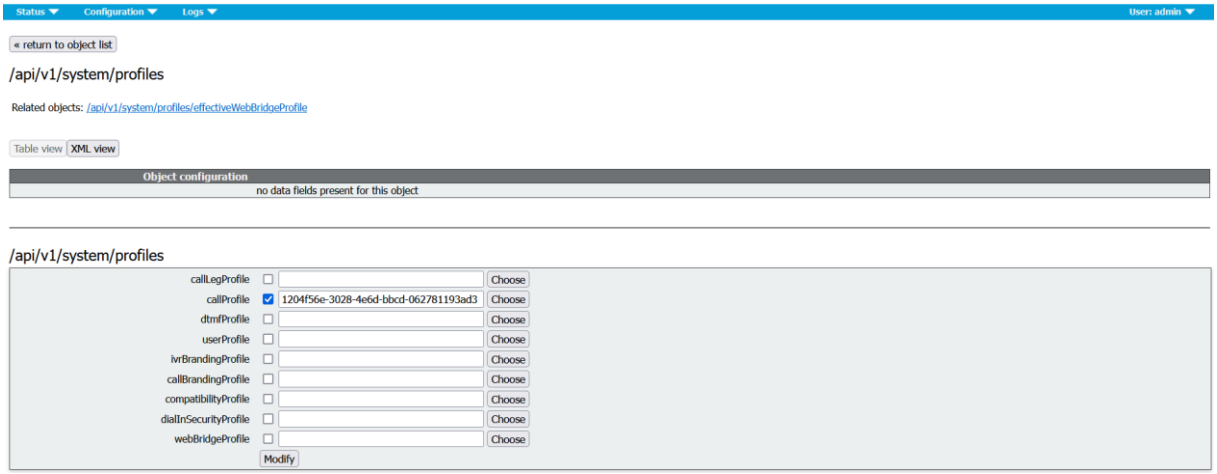
Click return to **object** field.
In the **Filter** section, type “**system**”.
Click the **View or edit** button for system profiles.
Then select Choose **Next** to the field for **callProfile**, and then Select the callProfile previously created

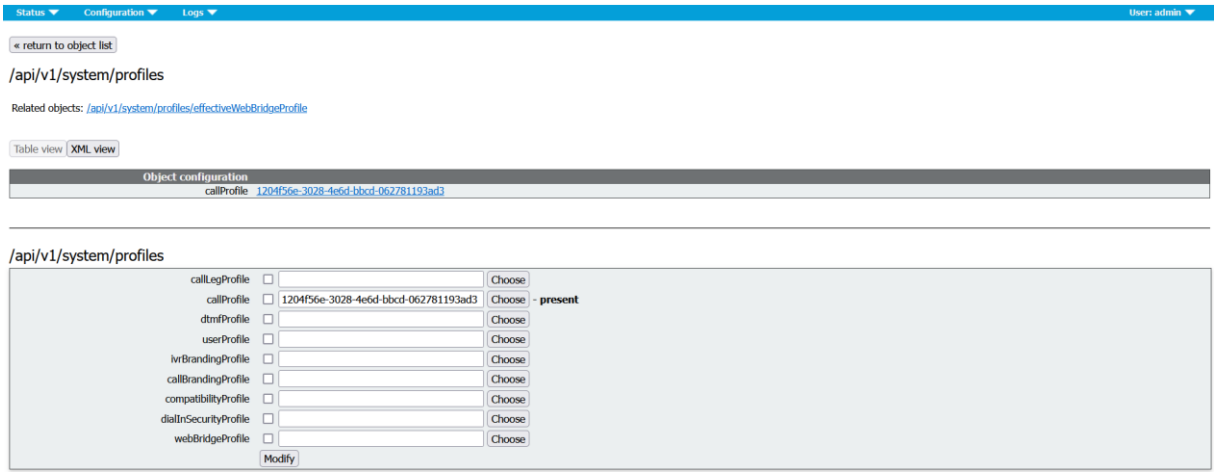
Click **Modify**.

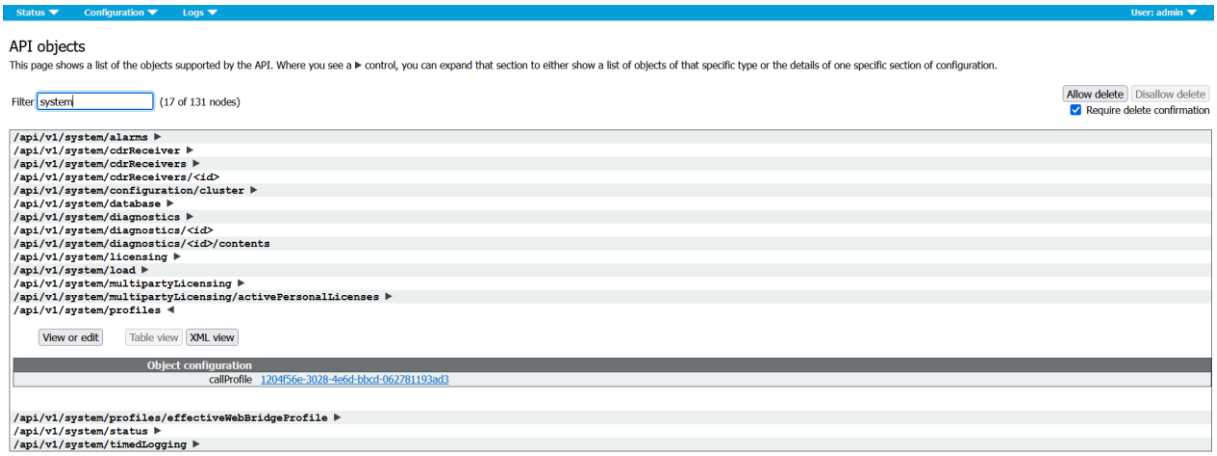



Status
Configuration
Logs
User: admin
« return to object list
/api/v1/system/profiles
Related objects: /api/v1/system/profiles/effectiveWebBridgeProfile
Table view
XML view
Object configuration
no data fields present for this object
/api/v1/system/profiles
callLegProfile
callProfile
dtmfProfile
userProfile
ivrBrandingProfile
callBrandingProfile
compatibilityProfile
dialInSecurityProfile
webBridgeProfile
Choose
Modify


CMS - Mozilla Firefox
https://10.1.5.42:445/api_id_selector.html?id=id_callProfile&checkbox=include_id_callProfile
callProfile object selector
Please select the callProfile object to use in this configuration operation.
« start
‹ prev
1 - 1 (of 1)
next ›
show all
Table view
XML view
object id
Select
1204f56e-3028-4e6d-bbcd-062781193ad3


Status
Configuration
Logs
User: admin
« return to object list
/api/v1/system/profiles
Related objects: /api/v1/system/profiles/effectiveWebBridgeProfile
Table view
XML view
Object configuration
no data fields present for this object
/api/v1/system/profiles
callLegProfile
callProfile
1204f56e-3028-4e6d-bbcd-062781193ad3
dtmfProfile
userProfile
ivrBrandingProfile
callBrandingProfile
compatibilityProfile
dialInSecurityProfile
webBridgeProfile
Choose
Modify


Status
Configuration
Logs
User: admin
« return to object list
/api/v1/system/profiles
Related objects: /api/v1/system/profiles/effectiveWebBridgeProfile
Table view
XML view
Object configuration
callProfile 1204f56e-3028-4e6d-bbcd-062781193ad3
/api/v1/system/profiles
callLegProfile
callProfile
1204f56e-3028-4e6d-bbcd-062781193ad3
- present
dtmfProfile
userProfile
ivrBrandingProfile
callBrandingProfile
compatibilityProfile
dialInSecurityProfile
webBridgeProfile
Choose
Modify

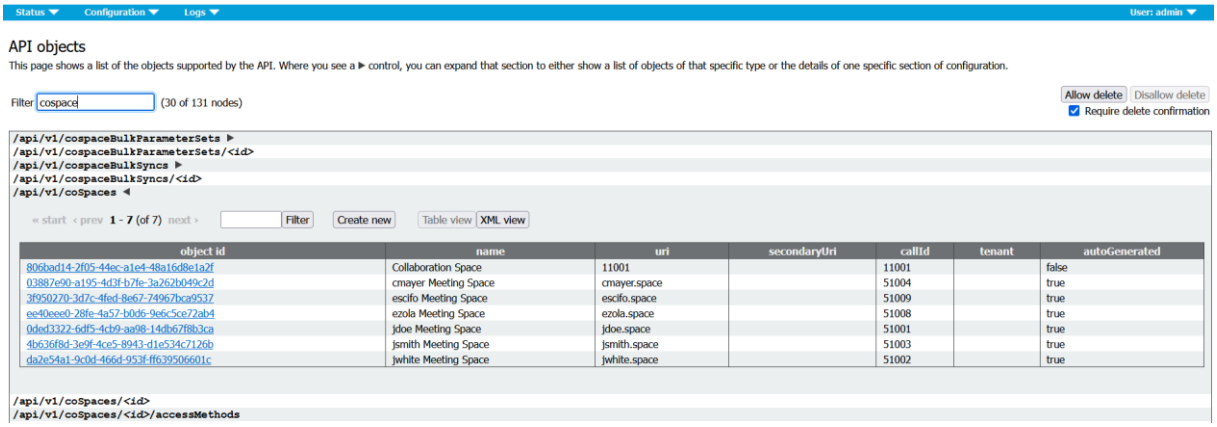


Live Streaming with Twitch

In the **Filter** section, type “**cospace**”.
Click the object ID for jdoe Meeting Space.

Modify the space to add “StreamURL”. The 'streamURL' in the following format:
rtmp://52.223.195.116/app/live_756394105_XlmtBGbdzZyBqUi4rYTmyVCk2oIrYM

Click **Modify**.



/api/v1/coSpaces/0ded3322-6df5-4cb9-aa98-14db67f8b3ca

userProvisionedCoSpace		**GUID (none available)**
name	jdoe Meeting Space	**- present**
uri	jdoe.space	(URI user part) **- present**
secondaryUri		(URI user part)
callId	51001	**- present**
cdrTag		
passcode		
defaultLayout	<unset>	
tenant		Choose
callLegProfile	7129d09a-4caf-42f0-a12a-e96f4b09ce3f	Choose **- present**
callProfile	1204f56e-3028-4e6d-bbcd-062781193ad3	Choose **- present**
callBrandingProfile		Choose
dialInSecurityProfile		Choose
defaultAccessMethod		**GUID (none available)**
requireCallId	<unset>	
secret	K6sBNdYW6sNJRfgaeh9RCw	**- present**
regenerateSecret	<unset>	
nonMemberAccess	true **- present**	
ownerJid	jdoe@lab.local	**- present**
streamUrl	rtmp://52.223.195.116/app/live_756394105_duRbtEd	(URL) **- present**
ownerAdGuid		**GUID (none available)**
meetingScheduler		
panePlacementHighestImportance		
panePlacementSelfPaneMode	<unset>	
panePlacementActiveSpeakerMode	<unset>	

Modify

Status Configuration Logs User: admin

« return to object list

/api/v1/coSpaces/0ded3322-6df5-4cb9-aa98-14db67f8b3ca

Related objects: /api/v1/coSpaces
/api/v1/coSpaces/0ded3322-6df5-4cb9-aa98-14db67f8b3ca/accessMethods
/api/v1/coSpaces/0ded3322-6df5-4cb9-aa98-14db67f8b3ca/coSpaceUsers
/api/v1/coSpaces/0ded3322-6df5-4cb9-aa98-14db67f8b3ca/diagnostics
/api/v1/coSpaces/0ded3322-6df5-4cb9-aa98-14db67f8b3ca/emailInvitation
/api/v1/coSpaces/0ded3322-6df5-4cb9-aa98-14db67f8b3ca/meetingEntryDetail
/api/v1/coSpaces/0ded3322-6df5-4cb9-aa98-14db67f8b3ca/metadata

Table view XML view

Object configuration	
name	jdoe Meeting Space
autoGenerated	true
uri	jdoe.space
callId	51001
callLegProfile	7129d09a-4caf-42f0-a12a-e96f4b09ce3f
callProfile	1204f56e-3028-4e6d-bbcd-062781193ad3
nonMemberAccess	true
ownerId	d53948af-0931-4499-9680-91752be4d153
ownerJid	jdoe@lab.local
streamUrl	rtmp://52.223.195.116/app/live_756394105_duRbtEdvtMoUtzragMznyFwrvaGFaH
secret	K6sBNdYW6sNJRfgaeh9RCw
numAccessMethods	1

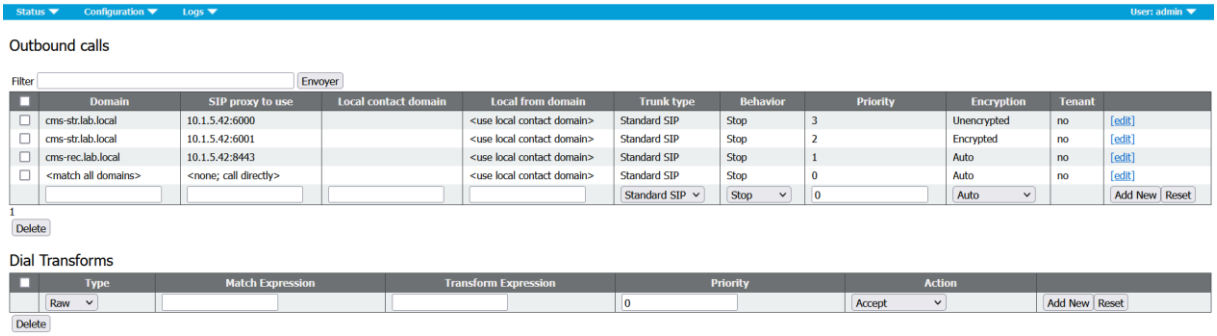
Configure an outboundDialPlan rule to match the domain used in streamerUri to route.

On CMS GUI navigate to Configuration > Outbound calls.
Create a new Outbound rules with the following informations:

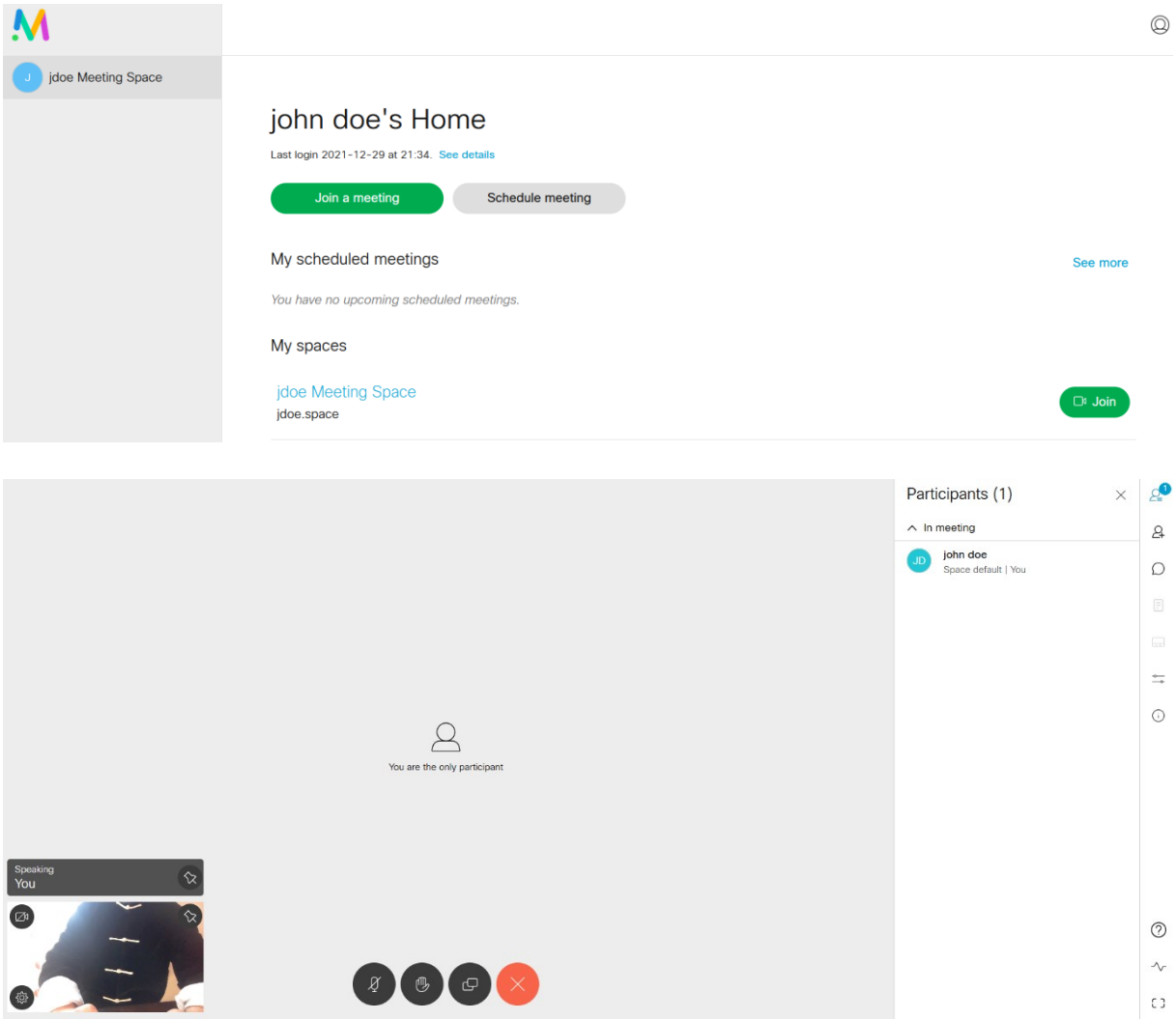
- **Domain:** cms-str.lab.local
- **SIP proxy to use:** 10.1.5.42:6000
- Encryption: Unencrypted
- click **And New**

- **Domain:** cms-str.lab.local
- **SIP proxy to use:** 10.1.5.42:6001
- Encryption: Encrypted
- click **And New**

For SIP streamer, if default ports for SIP (5060,5061) are not used, then It is mandatory to specify ports in the configuration of the streamer and include the following port number to connect to the "sip proxy to use" field when outboundDialPlanRule is configured for the service.



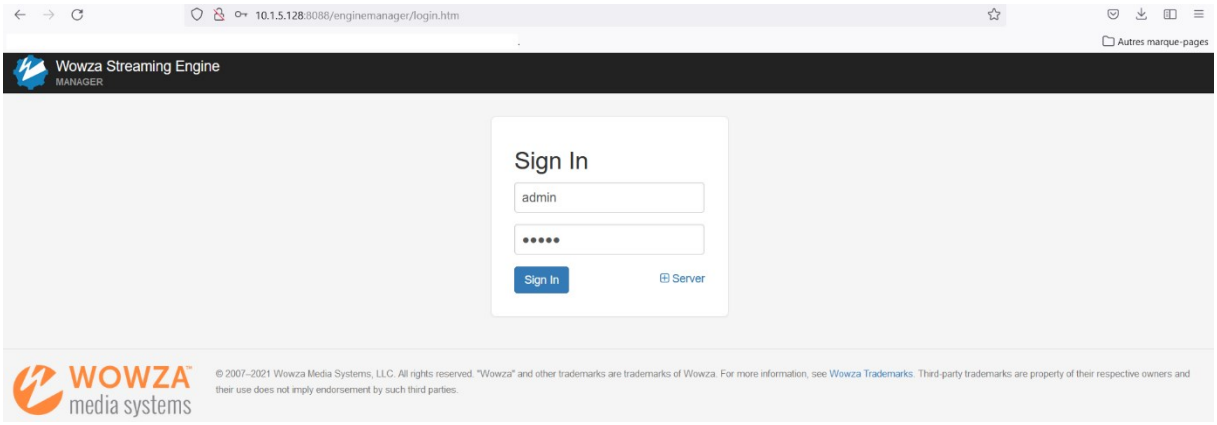
From the **jdoe-pc**, access the jdoe space and click the **Join** button.

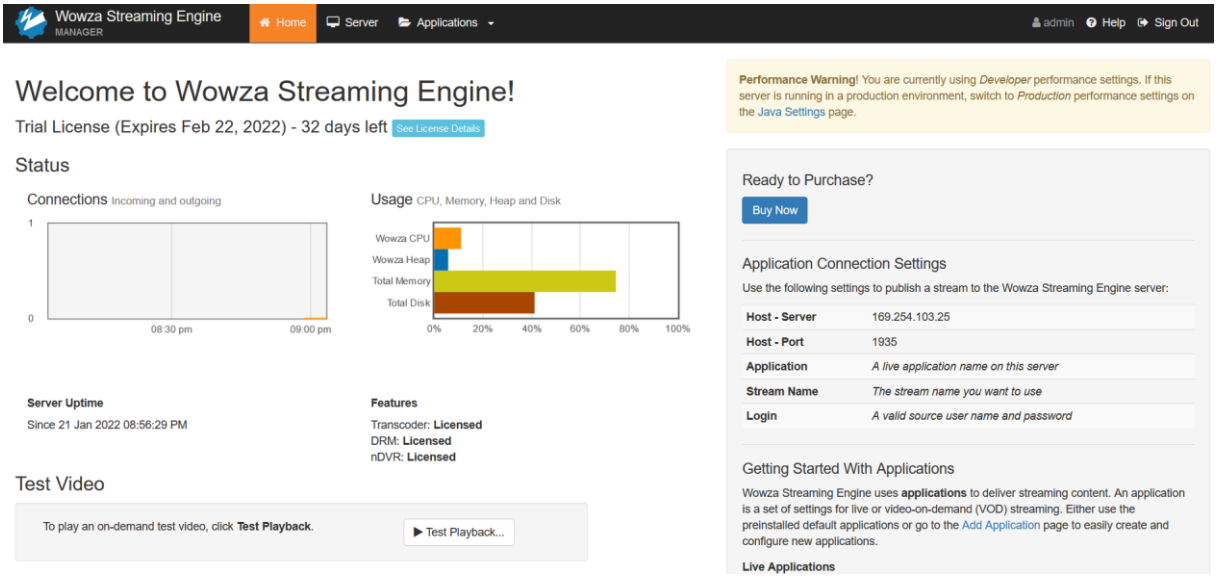


Live Streaming with Wowza Media Server

By default, Wowza Streaming Engine requires that RTMP-based encoders such Cisco Meeting Server provide a source username and password before they can connect to a live application and publish a live stream. Complete the following steps to create a source account and manage source authentication.

First to connect to Wowza Streaming Engine Manager, from the Admin-PC, open a web browser and type the url **http://10.1.5.128:8088**.



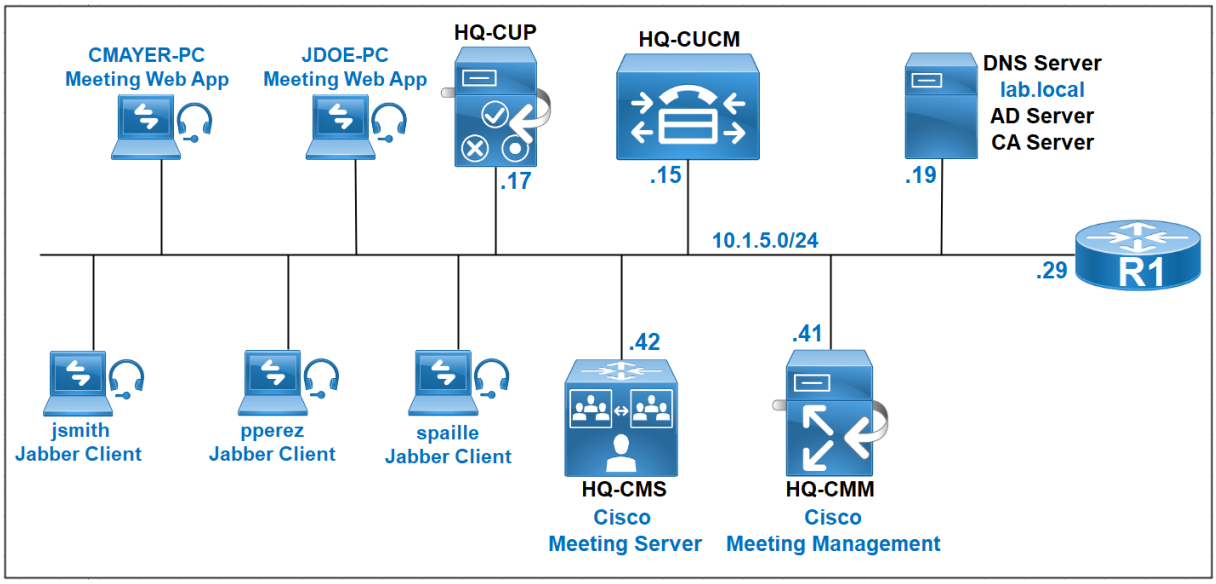


In Wowza Streaming Engine Manager, click **Server** and then click **Source Authentication**.

Click **Add Source**.

Add Source **UserName : cmsuser** and **Password : cisco**. Click **Add**. This account will be used later on the cospaces of Cisco Meeting Server. This source account is used to authenticate connections from Cisco Meeting Server to live applications in Wowza Streaming Engine.

Project 6: Scheduler Service Certificates



Version 3.3 introduces the ability to schedule meetings and see upcoming meetings in web app. Web app users can schedule meetings, modify the scheduled meetings, and notify participants via email. Scheduler is a new component that enables scheduling meetings, and is enabled by the new scheduler MMP commands.

When the scheduler is enabled, it makes API requests to the Call Bridge over the loopback interface. It is therefore a requirement that the scheduler is deployed on a Meeting Server which is also hosting a Call Bridge. It is not possible to configure the scheduler to use a remote Call Bridge.
C2W connections are established to each Web Bridge similar to how the Call Bridge also establishes a C2W connection to each Web Bridge. No explicit configuration is required to enable connection between the scheduler and Call Bridge, because this happens automatically over the loopback interface. Similarly, the C2W connections are all automatic but it is necessary to configure a trust bundle between the scheduler and Web Bridges.

Verify the appropriate certificates are uploaded to the Cisco CMS.

```
hq-cms>
hq-cms> pki list
User supplied certificates and keys:
Chain-CA.cer
cmscert.cer
cmscert.key
Root-CA.cer
CMS-Chain.cer
hq-cms>
```

Verify the CallBridge service is enabled.

```
hq-cms>
hq-cms> callbridge
Listening interfaces  : a
Preferred interface   : none
Key file              : cmscert.key
Certificate file      : cmscert.cer
Address               : none
CA Bundle file        : Chain-CA.cer
C2W trusted certs                    : Chain-CA.cer
Callbridge cluster trusted certs     : none
hq-cms>
```

Verify the WebBridge3 service is enabled.

```
hq-cms>
hq-cms> webbridge3
Enabled                                 : true
HTTPS listening ports and interfaces    : a:443
HTTPS Key file                          : cmscert.key
HTTPS Full chain certificate file       : CMS-Chain.cer
HTTPS Frame-Ancestors                   : none
HTTP redirect                           : Disabled
C2W listening ports and interfaces      : a:9999
C2W Key file                            : cmscert.key
C2W Full chain certificate file         : CMS-Chain.cer
C2W Trust bundle                        : Chain-CA.cer
Beta options                            : none
hq-cms>
```

Configure the scheduler to use the certificate.

```
hq-cms> scheduler c2w certs cmscert.key CMS-Chain.cer
```

It is necessary for the scheduler to be able to trust each Web Bridge it connects to. Upload a trust bundle which contains each Web Bridge certificate, via SFTP.

Configure the scheduler using the command.

```
hq-cms> scheduler c2w trust CMS-Chain.cer
```

The scheduler has its own HTTPS interface which if enabled, can be used to configure scheduler meetings using the scheduler APIs.

Configure the HTTPS server listen interface using the command.

```
hq-cms> scheduler https listen a 9445
```

Configure a certificate key pair for the server using the command:

```
hq-cms> scheduler https certs cmscert.key CMS-Chain.cer
```

```
hq-cms>
hq-cms> scheduler https listen a 9445
hq-cms> scheduler c2w trust CMS-Chain.cer
hq-cms> scheduler https certs cmscert.key CMS-Chain.cer
hq-cms> scheduler c2w certs cmscert.key CMS-Chain.cer
hq-cms>
```

To enable the Scheduler to send email notifications via the SMTP with Auth Login, configure the email server to listen on a specified port for the SMTP protocol, enable the SMTP server to support Auth Login, and configure a user account for authentication. This account configured on the MMP must have appropriate permissions to be able to send emails on behalf of web app users.

Configure the email server and port, and et the email protocol to SMTP.

```
hq-cms> scheduler email server smtp.gmail.com 25
hq-cms> scheduler email protocol smtp
```

From version 3.4 meeting invites can be sent to all the participants from a common email address. The MMP command scheduler email common-address <address@mail.domain> "<Display name>" configures the common email address and a display name on the Meeting Server. The Scheduler sends the meeting invites from the common email address to the participants.

If the common email address is left blank, the Scheduler sends the email invites from the organizer's email address.

To enable the Scheduler to send email notifications via the SMTP, configure the email server to listen on a specified port for the SMTP protocol.

```
hq-cms> scheduler email common-address rmeddane@gmail.com ipdemystify
```

```
hq-cms>
hq-cms> scheduler email server smtp.gmail.com 25
hq-cms> scheduler email protocol smtp
hq-cms> scheduler email common-address rmeddane@gmail.com ipdemystify
hq-cms>
```

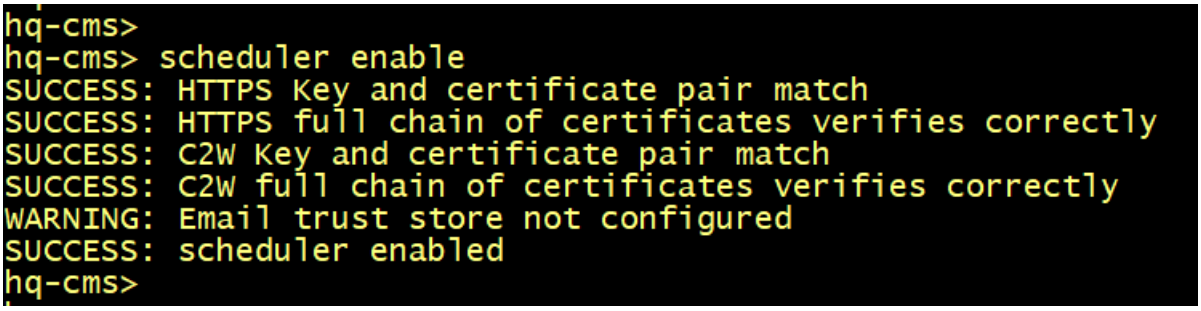
Set the username to be used for authentication.

```
hq-cms> scheduler email username rmeddane
Please enter password:
Please enter password again:
hq-cms>
```

```
hq-cms>
hq-cms> scheduler email username rmeddane
Please enter password:
Please enter password again:
hq-cms>
```

Enable the Scheduler.

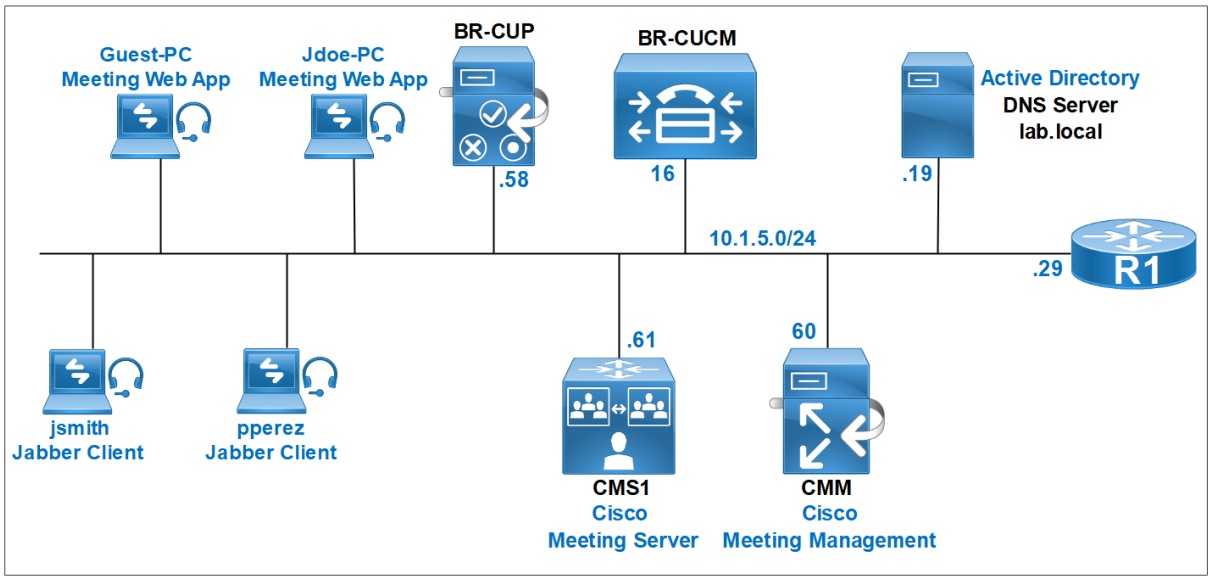
```
hq-cms> scheduler enable
SUCCESS: HTTPS Key and certificate pair match
SUCCESS: HTTPS full chain of certificates verifies correctly
SUCCESS: C2W Key and certificate pair match
SUCCESS: C2W full chain of certificates verifies correctly
WARNING: Email trust store not configured
SUCCESS: scheduler enabled
hq-cms>
```



Check the configuration and status of the scheduler service using the following command.

Project 7: Access Methods CoSpace Users, Blast Dial and AdHoc

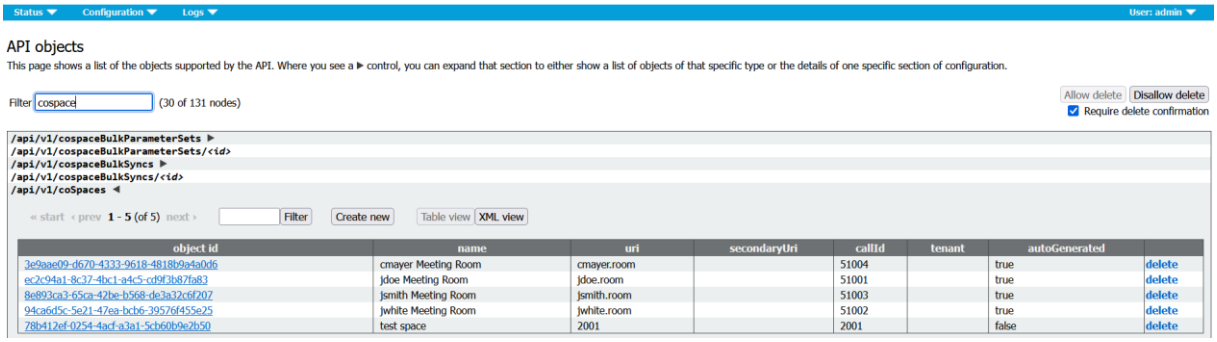
Access Methods and CoSpace Users



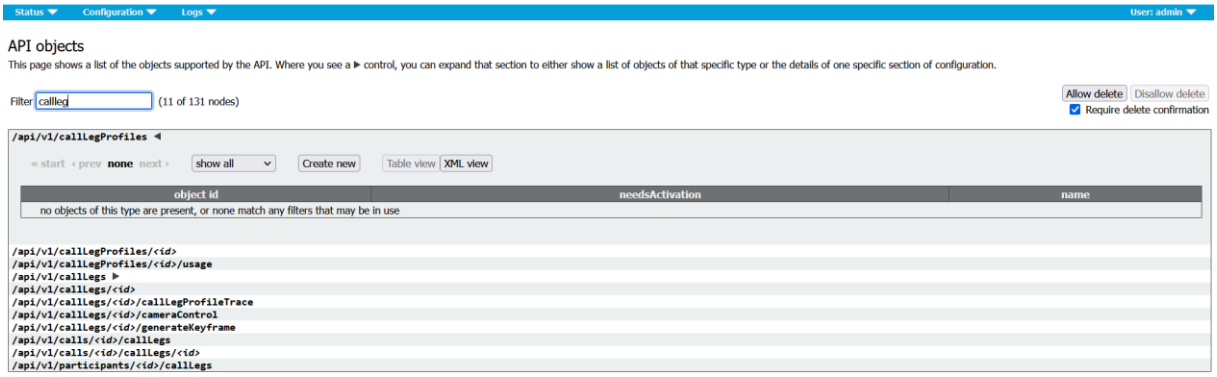
You want in some deployment different URIs or call-IDs are used by guest and host participants to join a meeting, and different rights or privileges, for example add or remove participants, mute audio or video, or when there are only guests joined to the space, they are all put in a lobby room waiting for the host or the owner to join in.

To do this on Cisco Meeting Server, we need a combination of **CallLegProfile** and **AccessMethods**.

Navigate to **Configuration > API**. Ensure that the spaces were created from the imported Active Directory users. Ensure that the **Call ID** fields are populated, each **Call ID** represents the Meeting ID of the corresponding space.



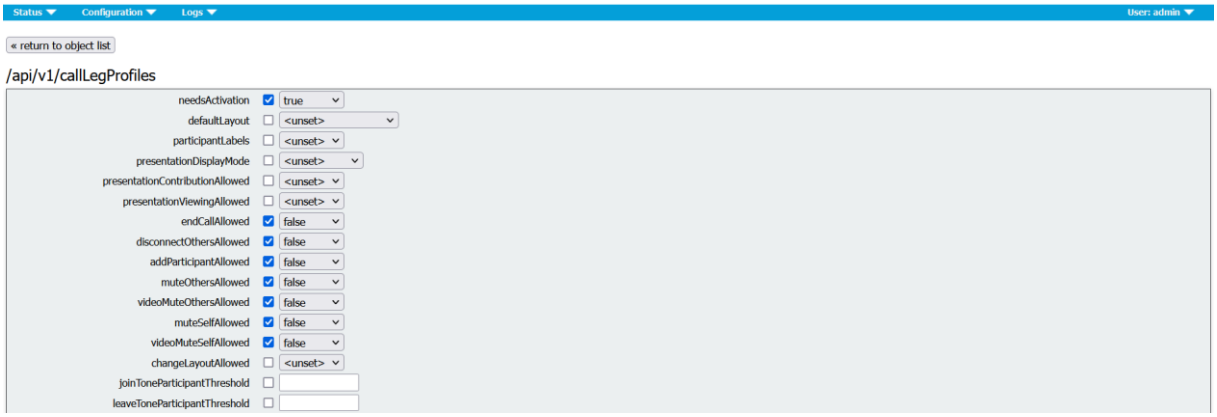
On **CMS1** GUI navigate to **Configuration** > **API**.
In the **Filter** section, type "**CallLegProfiles**". Click **Create New**.



Set the following settings :

- needActivation = **true**
- endCallAllowed = **false**
- disconnectOthersAllowed = **false**
- addParticipantAllowed = **false**
- muteOthersAllowed = **false**
- videoMuteOthersAllowed = **false**
- muteSelfAllowed = **false**
- videoMuteSelfAllowed = **false**

Give a name of **Guest** for this Call Leg Profile.



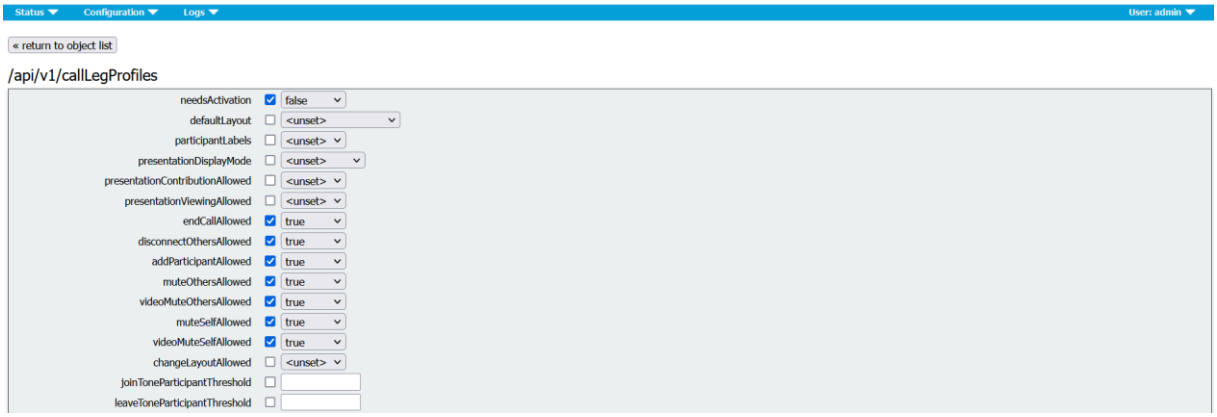
Click **Create New** to create a second **CallLegProfile**.

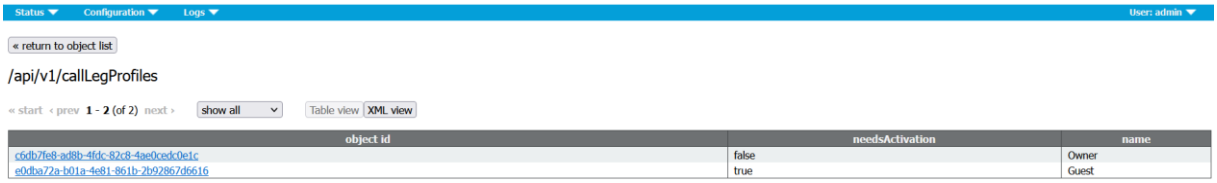
Set the following settings:

- needActivation = **false**
- endCallAllowed = **true**
- disconnectOthersAllowed = **true**
- addParticipantAllowed = **true**
- muteOthersAllowed = **true**
- videoMuteOthersAllowed = **true**
- muteSelfAllowed = **true**
- videoMuteSelfAllowed = **true**

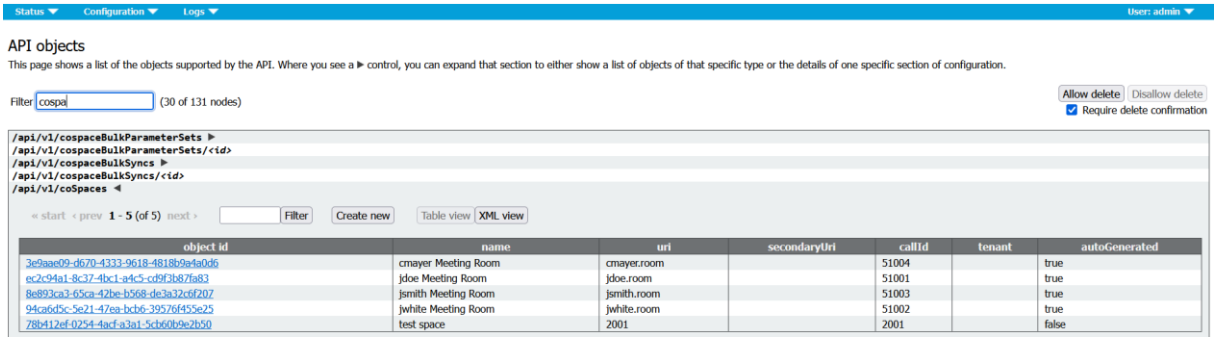
Give a name of **Owner** for this Call Leg Profile.

The **needsActivation** parameter is very important here for the guest/host behavior since if set to true, the participant is unable to receive audio and video or join to join a meeting until one or more activator (host) participants join.

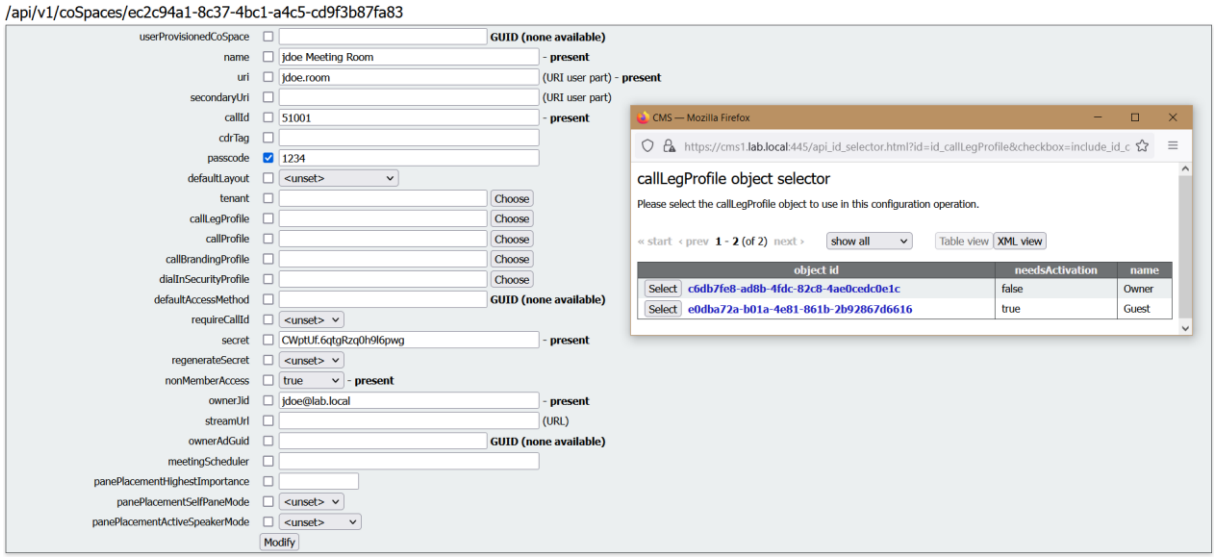


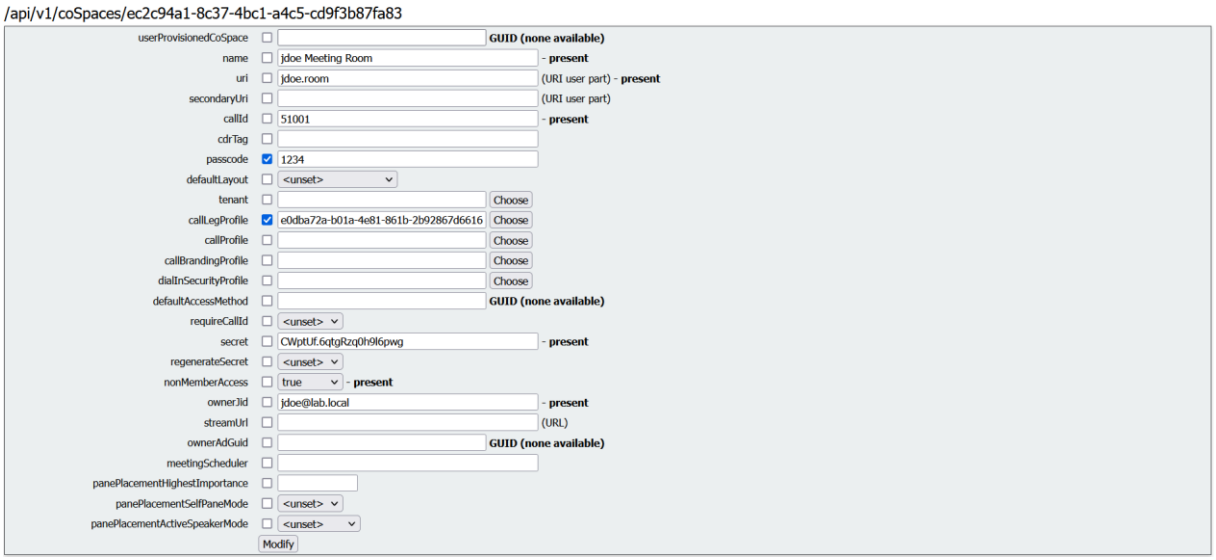


Navigate to **Configuration** > **API**.
In the **Filter** section, type "**CoSpaces**". Locate the **John Doe Space** for example. Click on the **Object ID**.

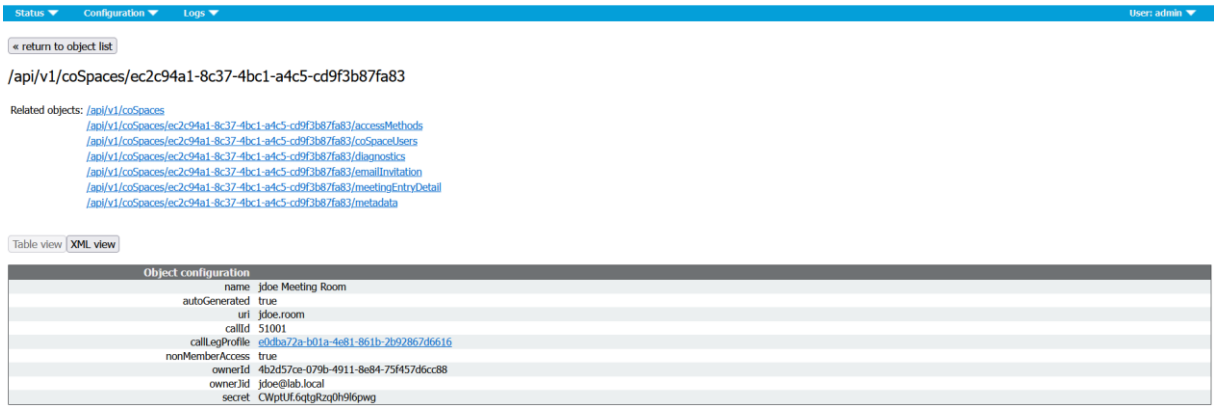


In the **CallLegProfile** field, click the **Choose** button.
A new window appears, click the **Select** button and select the CallLegProfile named **Guest**.
Optionally, set the **passcode 1234** to authenticate the guests who trying to access John Doe's space.



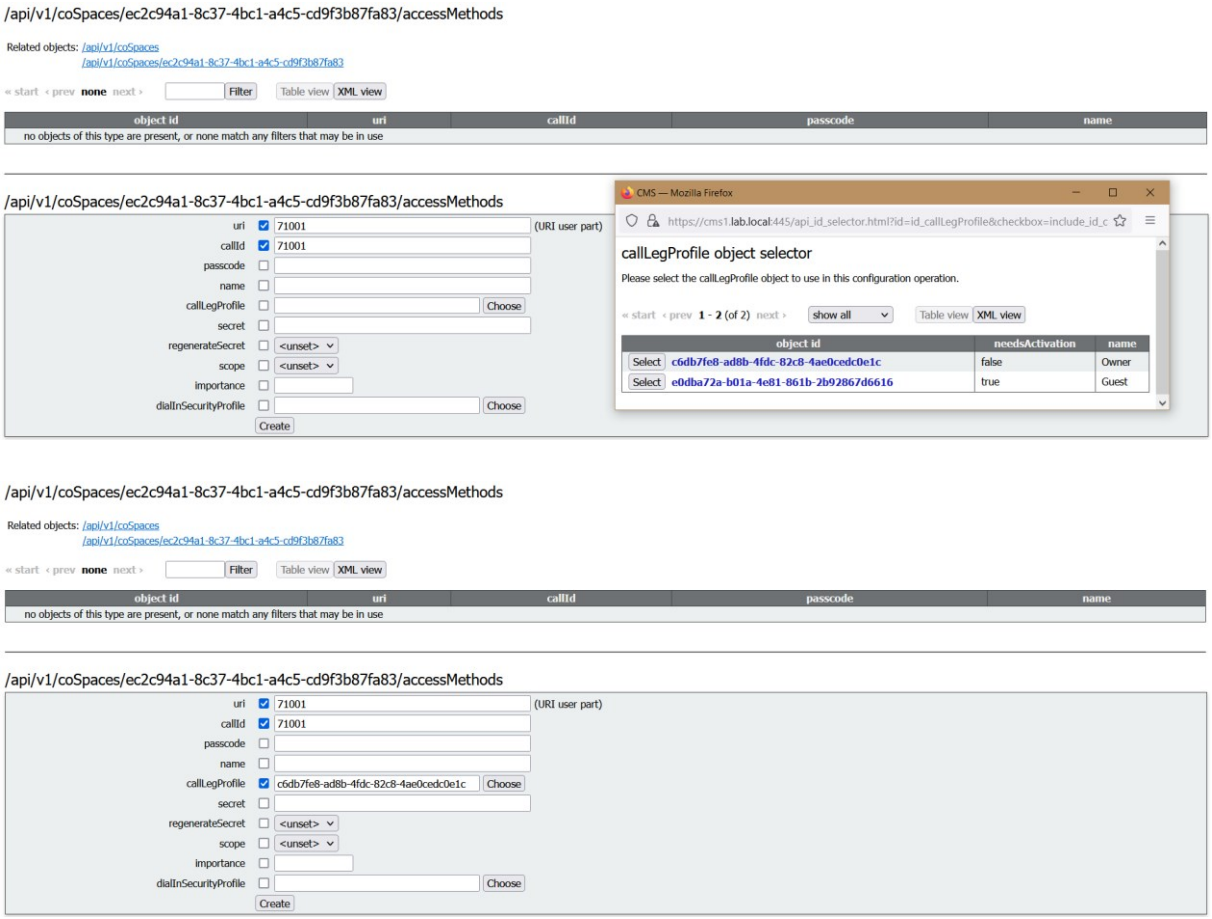


In the **Related Objects**, select the **accessMethods**.

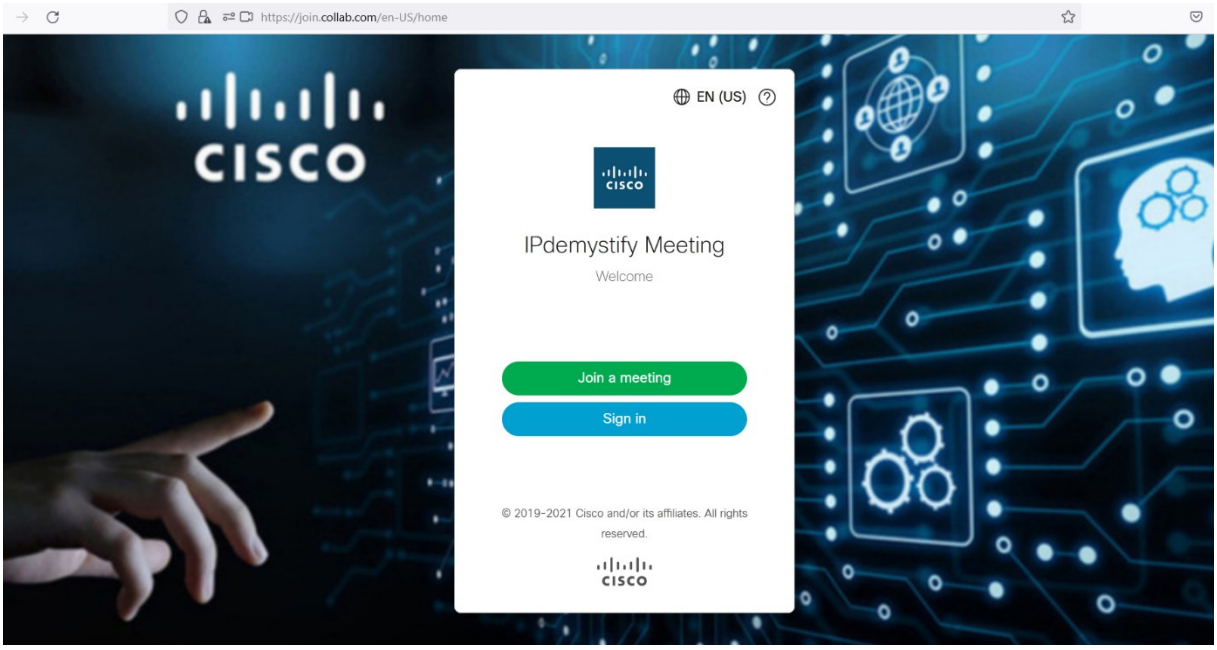


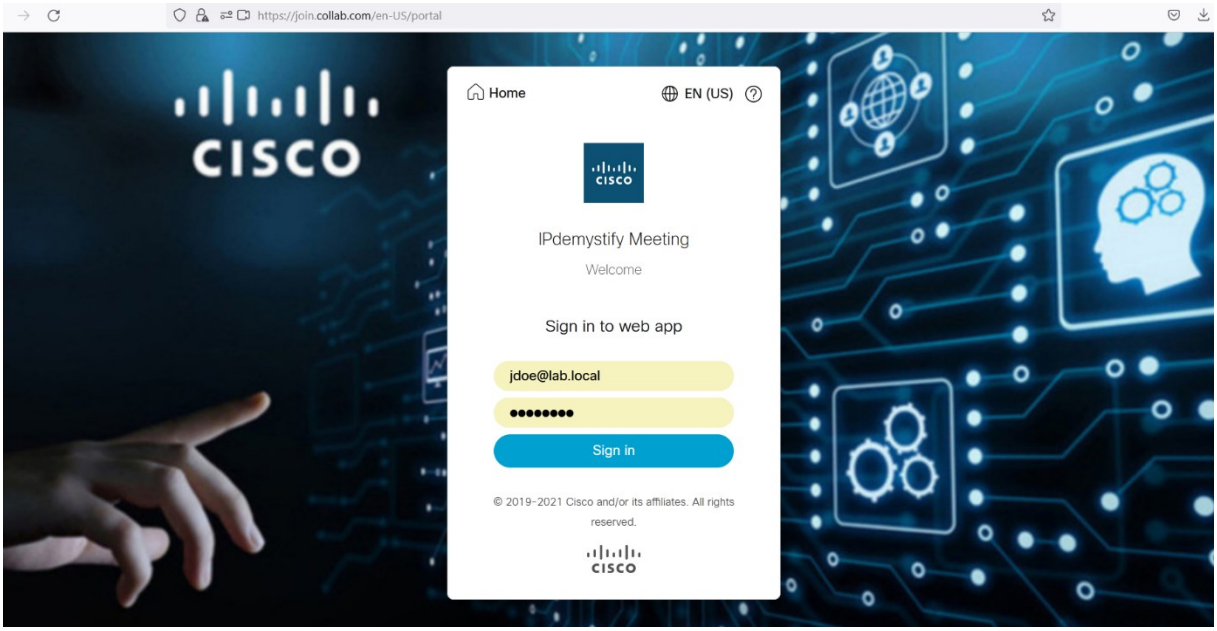
In the **CallLegProfile** field, click the **Choose** button.
A new window appears, click the **Select** button and select the CallLegProfile named **Owner**.

In the **uri** field, enter **71001**, this is the URI user part as shown below.
In the **callId**, enter **71001**, this is the Meeting ID the owner of the space will use to join **John Doe's space** instead of **51001**.

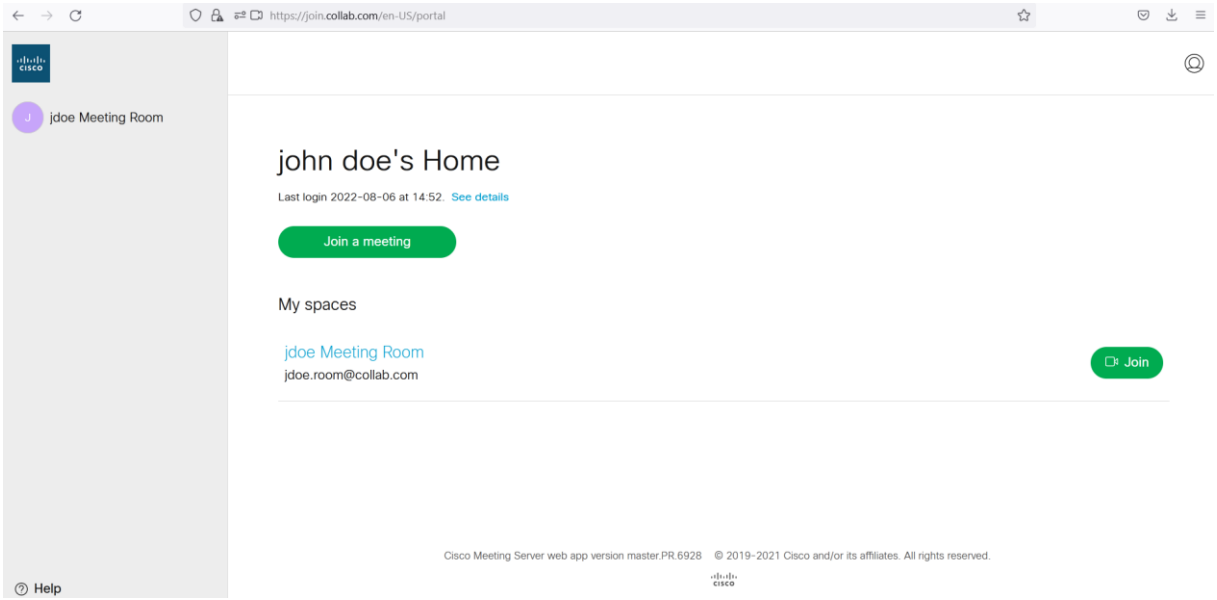


From the **Jdoe-PC**, open a web browser and type the URL **https://join.collab.com.** Click **Sign in** and connect using the username **jdoe@lab.local.**





At the left, click **jdoe Meeting Room**.



You should see two access methods to join the jdoe's space, Role 1 is the access method with a CallLegProfile Guest.

Or you can provide the video address with the URI format so that the guest user can join the meeting using Cisco Jabber or video endpoint such as Cisco Telepresence endpoint.

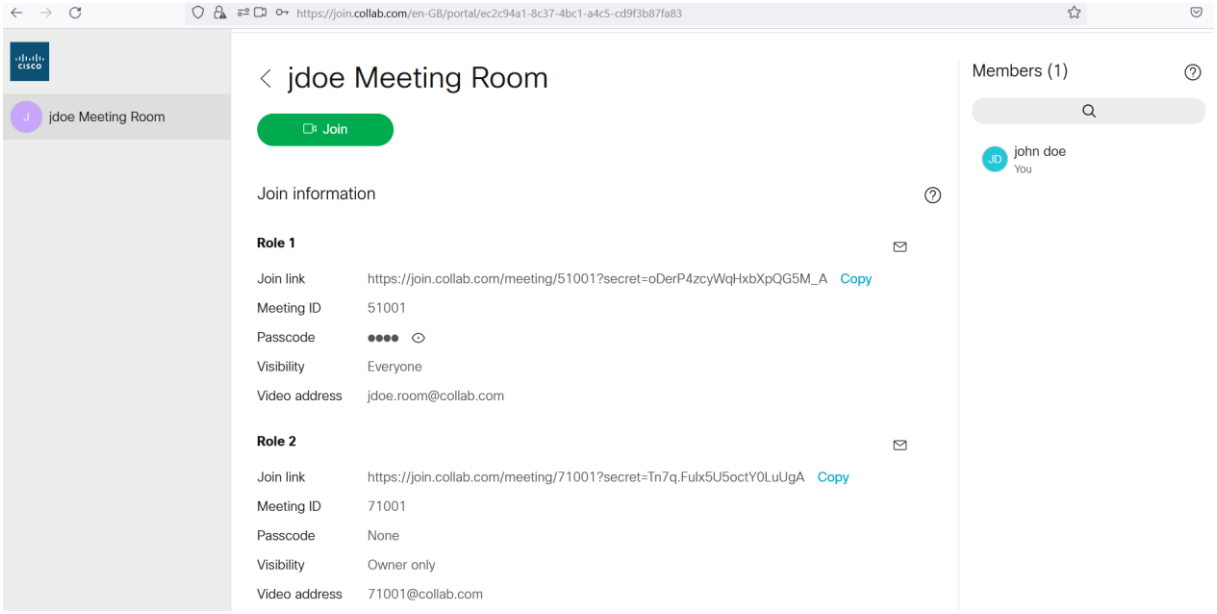
To dial into the jdoe meeting:

With Role 1 as Guest:

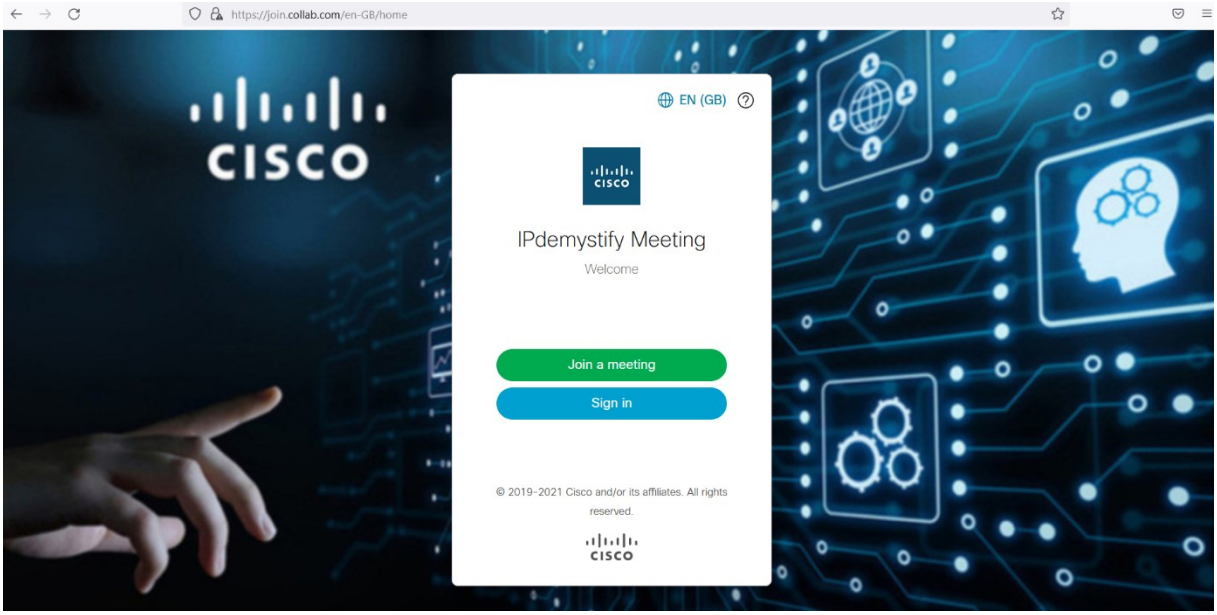
- by dialing to **jdoe.room@collab.com** URI using Video Endpoints
- by entering the **call-ID** value **51001** via web browser (Cisco Meeting web app) with passcode 1234

With Role 2 as Owner:

- by dialing to **71001@collab.com** URI using Video Endpoints
- by entering the **call-ID** value **71001** via web browser (Cisco Meeting web app)

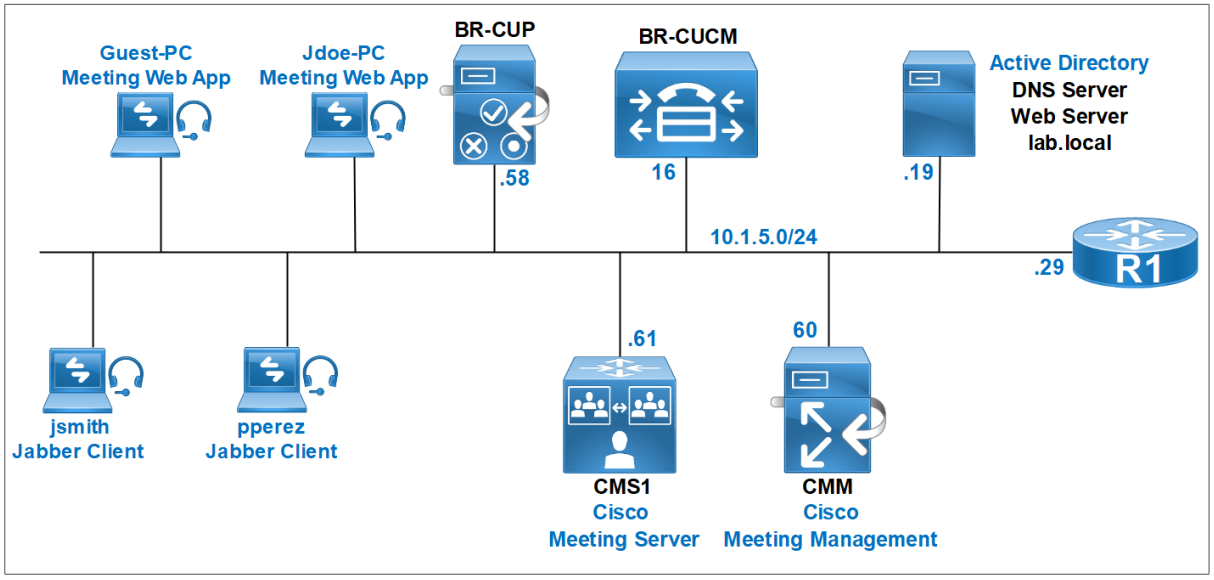


From the Guest-PC, open a web browser and type the URL **https://join.collab.com**. Click the **Join a meeting button**.



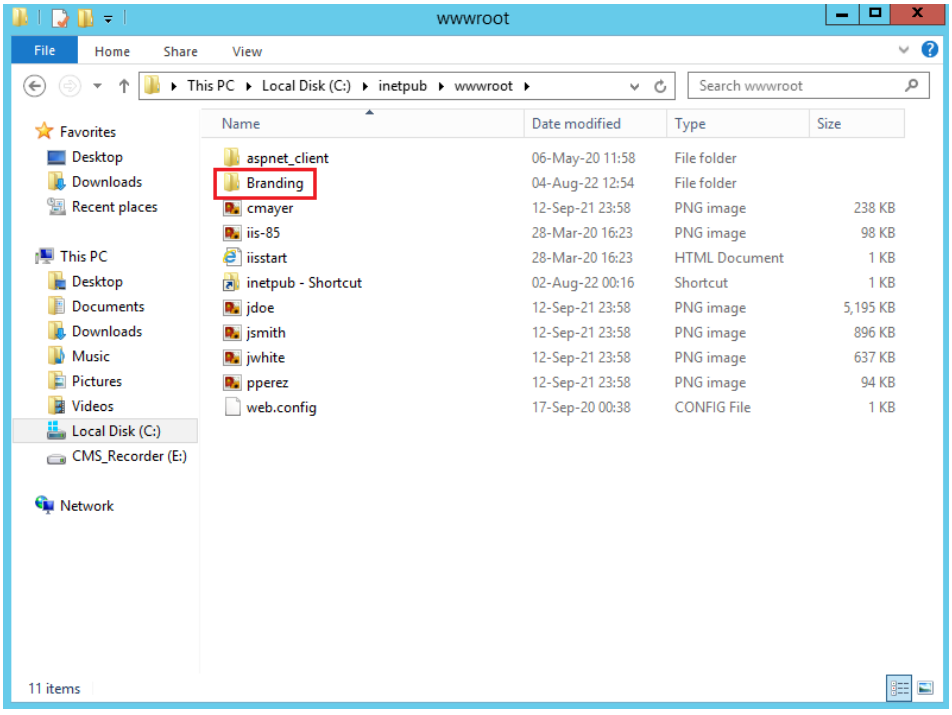
Enter the Meeting ID **51001** with passcode **1234**.

Project 8: Cisco CMS Web Server Hosted Branding



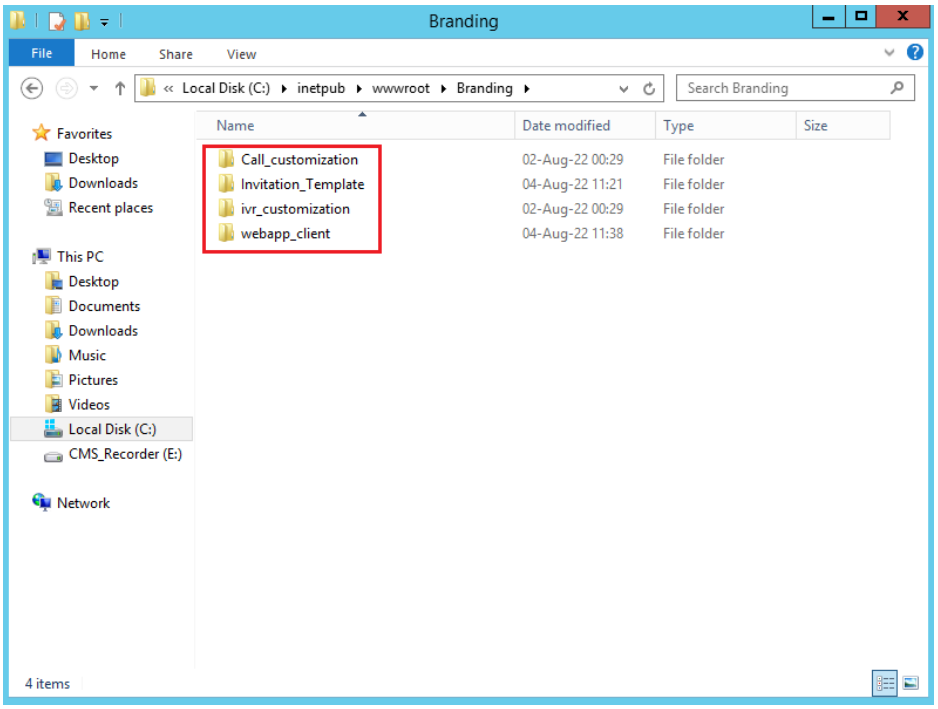
Call Branding feature on Cisco Meeting Server provides the ability to customize the Cisco Web App and SIP calls such as customized background picture and sign in logo or wav file for prompts according to the enterprise activity.

On the web server, create a folder called **Branding**.

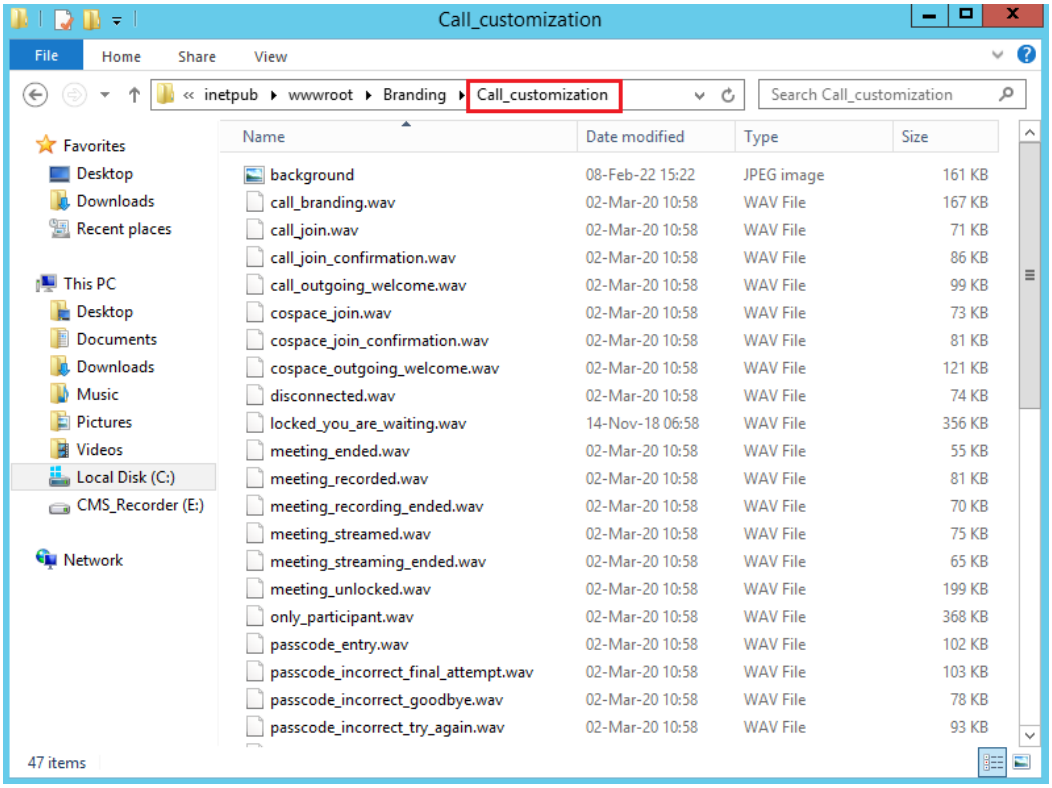


On the previously created folder, create the following subfolders:

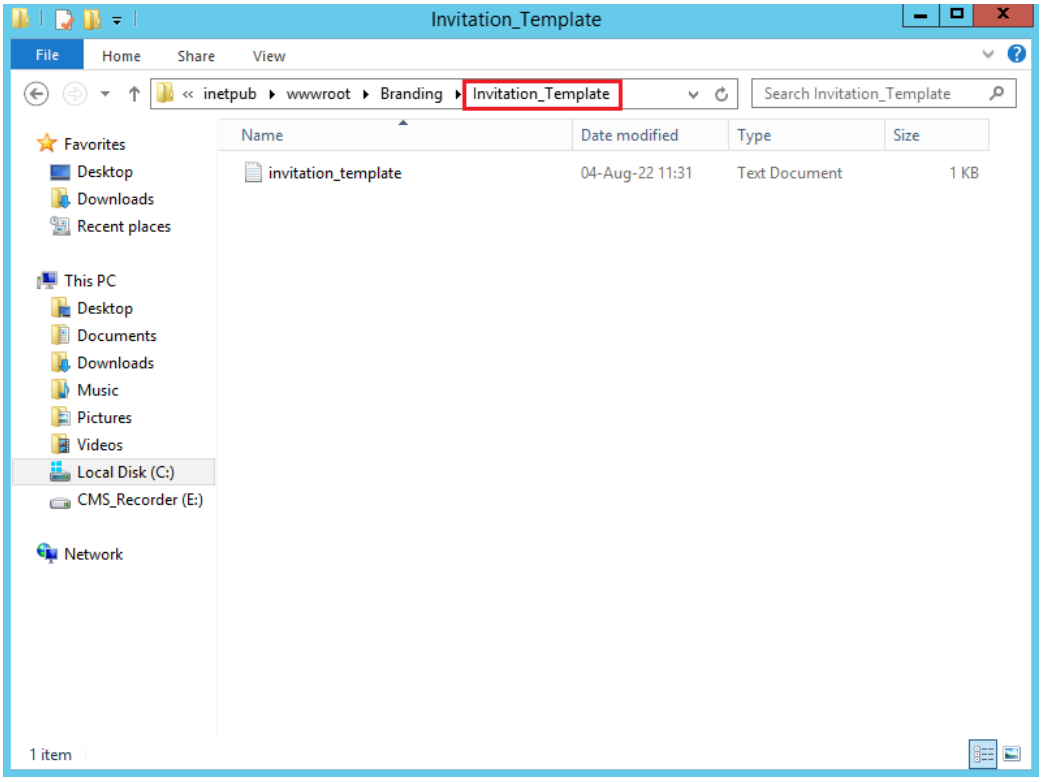
- Call_customization
- Invitation_Template
- Ivr_customization
- Webapp_client



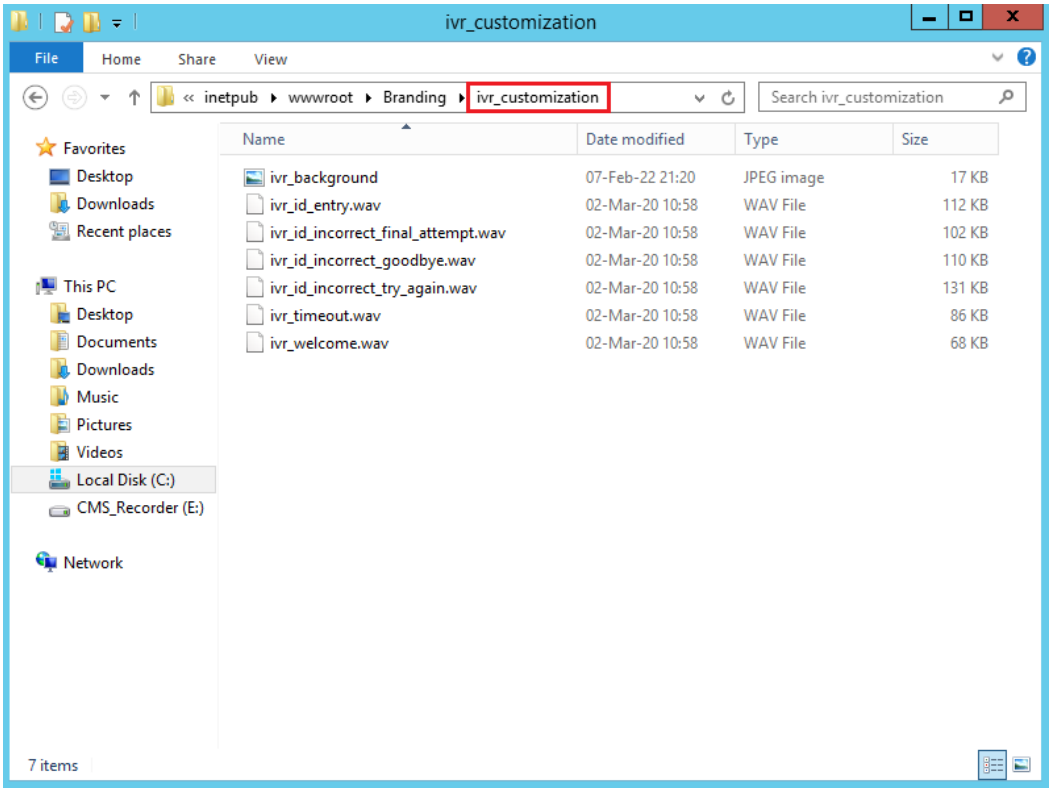
Copy the appropriate wav files and the background image into the subfolder **call_customization**.



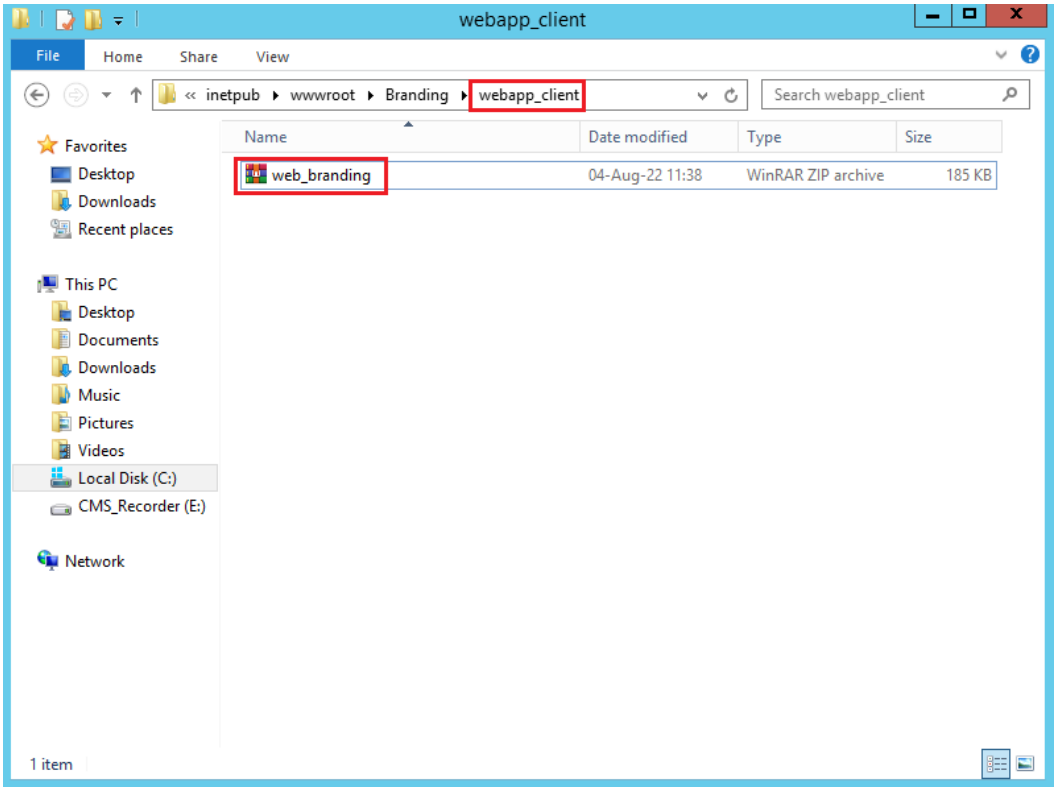
Copy the invitation template text file into the subfolder **Invitation_Template**.

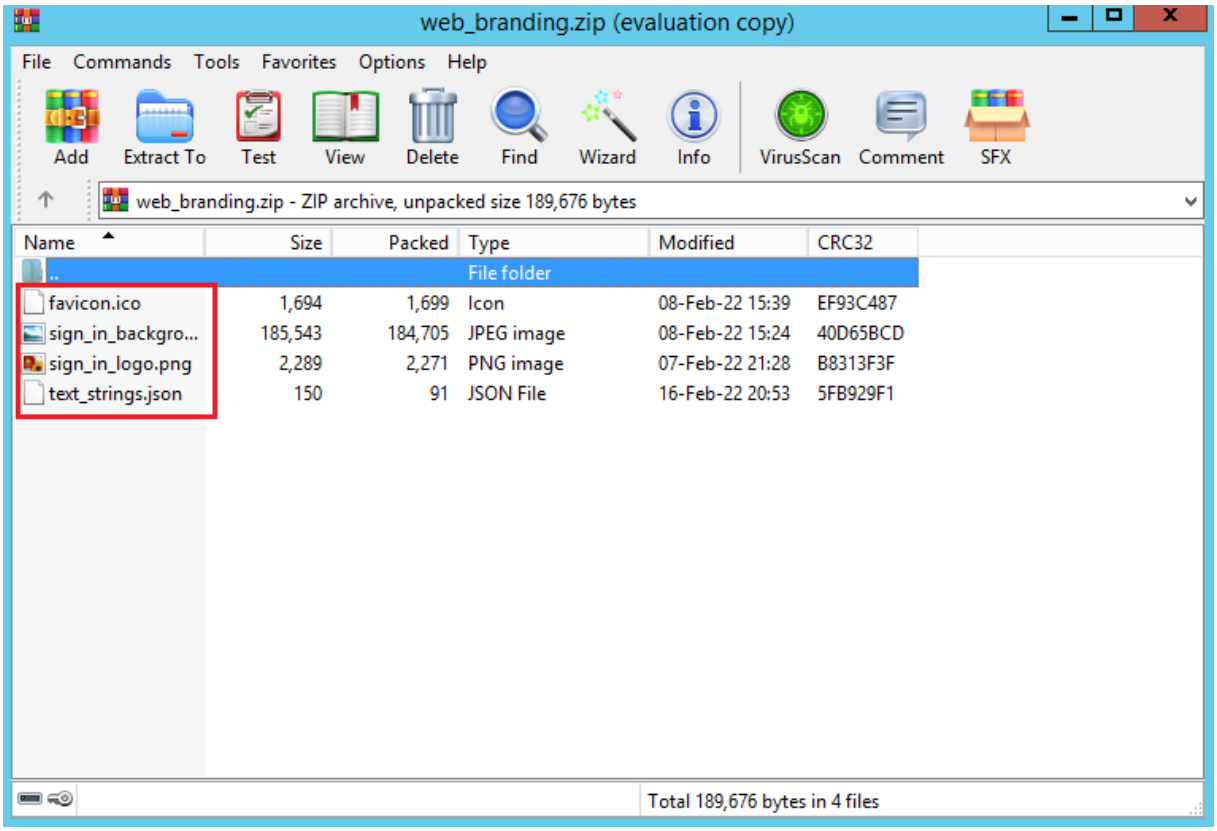


Copy the appropriate wav files and the background image into the subfolder **ivr_customization**.



Files of the subfolder **webapp_client** must be compressed with zip extension.





Access the CMS GUIS, navigate to API configuration, edit the **webbridge Profiles**.

In the **resourceArchive** field, past the URL
http://10.1.5.19/Branding/webapp_client/web_branding.zip.