Configure CMS Scheduler and Schedule a Meeting on Web App

Contents

Introduction

Prerequisites

- Requirements
- Components Used

Background Information

Configure

- Schedule a Meeting (Optional)

Verify

Troubleshoot

Introduction

This document describes how to configure Cisco Meeting Server (CMS) Scheduler on CMS 3.3 and how to schedule a meeting.

Prerequisites

Requirements

Cisco recommends that you have knowledge of these topics:

- Call Bridge
- Web Bridge

Components Used

The information in this document is based on these software and hardware versions:

- CMS Version 3.3
- Cisco Meeting Management (CMM)

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure

that you understand the potential impact of any command.

Background Information

CMS Version 3.3 introduces the ability to schedule meetings and see upcoming meetings in the web app. Web app users can schedule meetings, modify the scheduled meetings, and notify participants via email.

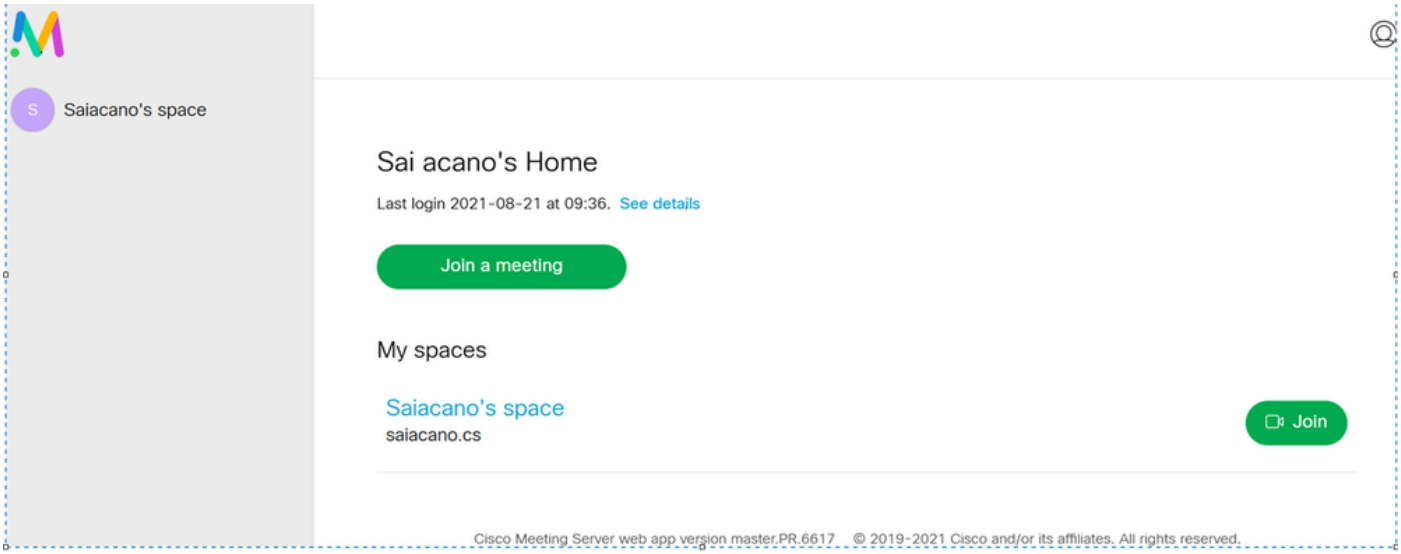
Note: In version 3.4, the Scheduler component was released as a fully supported feature on Meeting Server 1000 and Virtualized deployments. Version 3.5 introduces the support for Scheduler on Meeting Server 2000. It is now supported on Meeting Server 1000, Meeting Server 2000, and Meeting Server on Virtualized deployments.

Note: The scheduler component deletes the temporary spaces that are created when you schedule the meeting through an internal task that runs every 24 hours at 1:15 GMT. If the meeting has ended 24 or more hours before the task is run, the temporary space is removed.

Note: Do not create a localhost A record (or a wildcard * record that matches localhost) in any DNS zone used by the CMS servers. The Scheduler internally resolves its Callbridge API endpoint via localhost, which must resolve to 127.0.0.1; if DNS returns any other address, webbridge discovery fails silently with no webbridge returned and the Scheduler cannot start C2W.

Configure

The web app is configured without a scheduler as shown in the image.



The scheduler is a beta component of CMS 3.3. New Mainboard Management Processor (MMP) command is set to configure the scheduler highlighted as shown in the image.

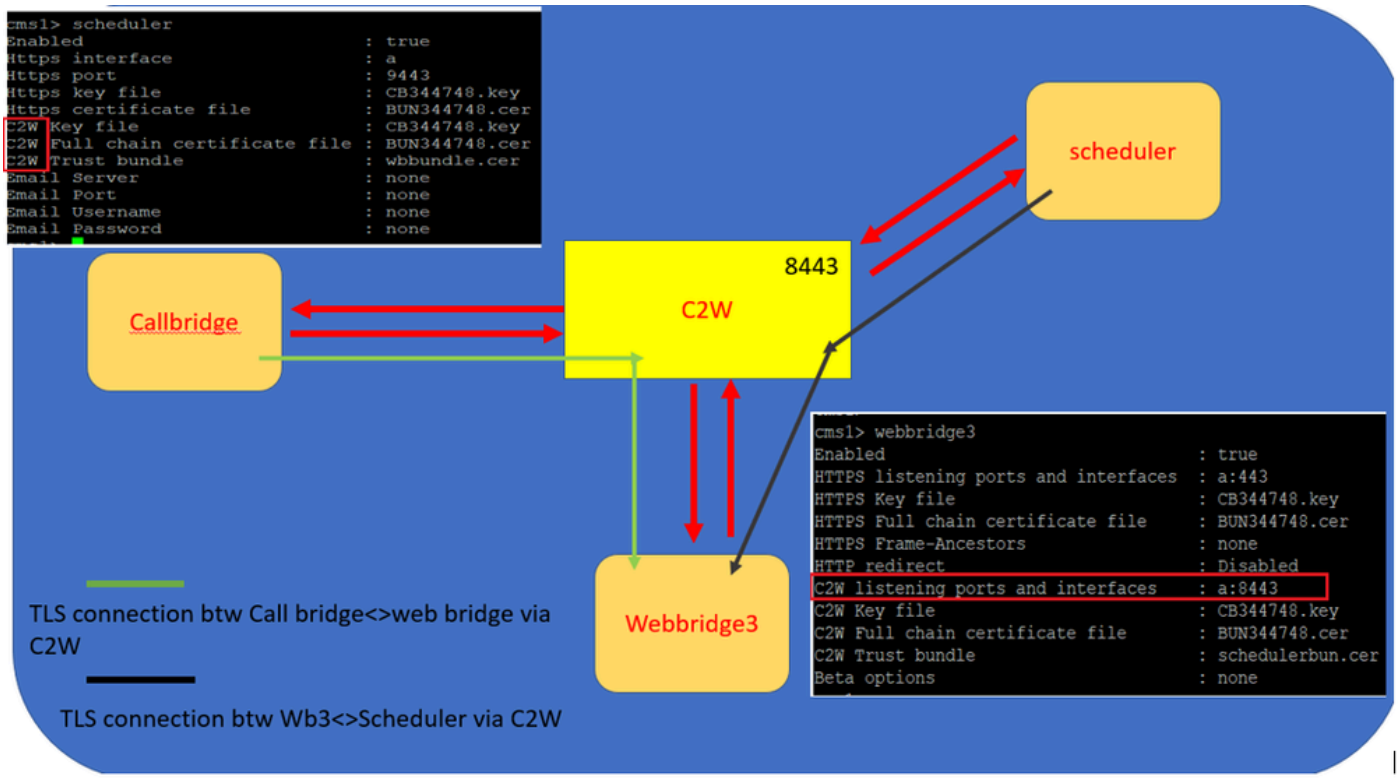
```
cms39> help scheduler
Configure scheduler

Usage:
    scheduler
    scheduler https listen <interface> <port>
    scheduler https listen none
    scheduler https certs <key-file> <crt-fullchain-file>
    scheduler https certs none
    scheduler c2w certs <key-file> <crt-fullchain-file>
    scheduler c2w certs none
    scheduler c2w trust <bundle>
    scheduler c2w trust none
    scheduler email server <hostname|address> <port>
    scheduler email server none
    scheduler email username <smtp username>
    scheduler email remove username
    scheduler email protocol <smtp|smtps>
    scheduler email auth <enable|disable>
    scheduler email starttls <enable|disable>
    scheduler email trust <bundle>
    scheduler email trust none
    scheduler timedLogging
    scheduler timedLogging (webBridge|api|email) <time>
    scheduler enable
    scheduler disable
    scheduler restart
    scheduler status
cms39>
```

Scheduler C2W - Web Bridge Connection Explained

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 also hosts 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 the 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.



Scheduler Connections:

1. Configure C2W Trust:

C2W is a TLS-based WebSocket connection established from the scheduler to each Web Bridge. In this release, each scheduler needs to be able to connect to each Web Bridge in a cluster. The scheduler requires the configuration of a client certificate and key to be used for this connection. Since the Scheduler is required to run on a server which also has a colocated Call Bridge, it is possible to use the Call Bridge certificate and C2W trust cert for the Scheduler service for ease of deployment. This ensures that the certificate used is already included in the Web Bridge C2W trust.

In order to do this, create a certificate and upload it to the Meeting Server via Secure File Transfer Protocol (SFTP) or use the Public Key Infrastructure (PKI) MMP commands to create a certificate:

```
scheduler c2w certs CB344748.key BUN344748.cer
```

Here BUN344748.cer is a full chain certificate. A full chain certificate must be offered by the Scheduler service when you establish a secure connection to Web Bridge servers.

It is important for the scheduler to be able to trust each Web Bridge it connects to. So bundle all Web Bridge certificates and have the scheduler trust Web Bridge Bundle.

Configure the scheduler with the command **scheduler c2w trust webbridge_bundle.cer**.

For example, **scheduler c2w trust wbbundle.cer**, where **wbbundle.cer** is a bundle of trust of all Web Bridge certificates.

It is also necessary for the Web Bridge to be able to trust the scheduler. So, bundle all scheduler certificates and have Web Bridge trust Scheduler Bundle: **webbridge3 c2w trust <crt-bundle>**.
All the necessary certificates for both schedulers and Call Bridges can be included in the **<crt-bundle>**.

For example, **webbridge3 c2w trust schedulerbun.cer** , where **schedulerbun.cer** is a bundle of all scheduler certificates and Call Bridge certificates.

```
cms1> webbridge3
Enabled                                 : true
HTTPS listening ports and interfaces    : a:443
HTTPS Key file                          : CB344748.key
HTTPS Full chain certificate file       : BUN344748.cer
HTTPS Frame-Ancestors                   : none
HTTP redirect                           : Disabled
C2W listening ports and interfaces      : a:8443
C2W Key file                            : CB344748.key
C2W Full chain certificate file         : BUN344748.cer
C2W Trust bundle                        : schedulerbun.cer
Beta options                            : none
cms1>
```

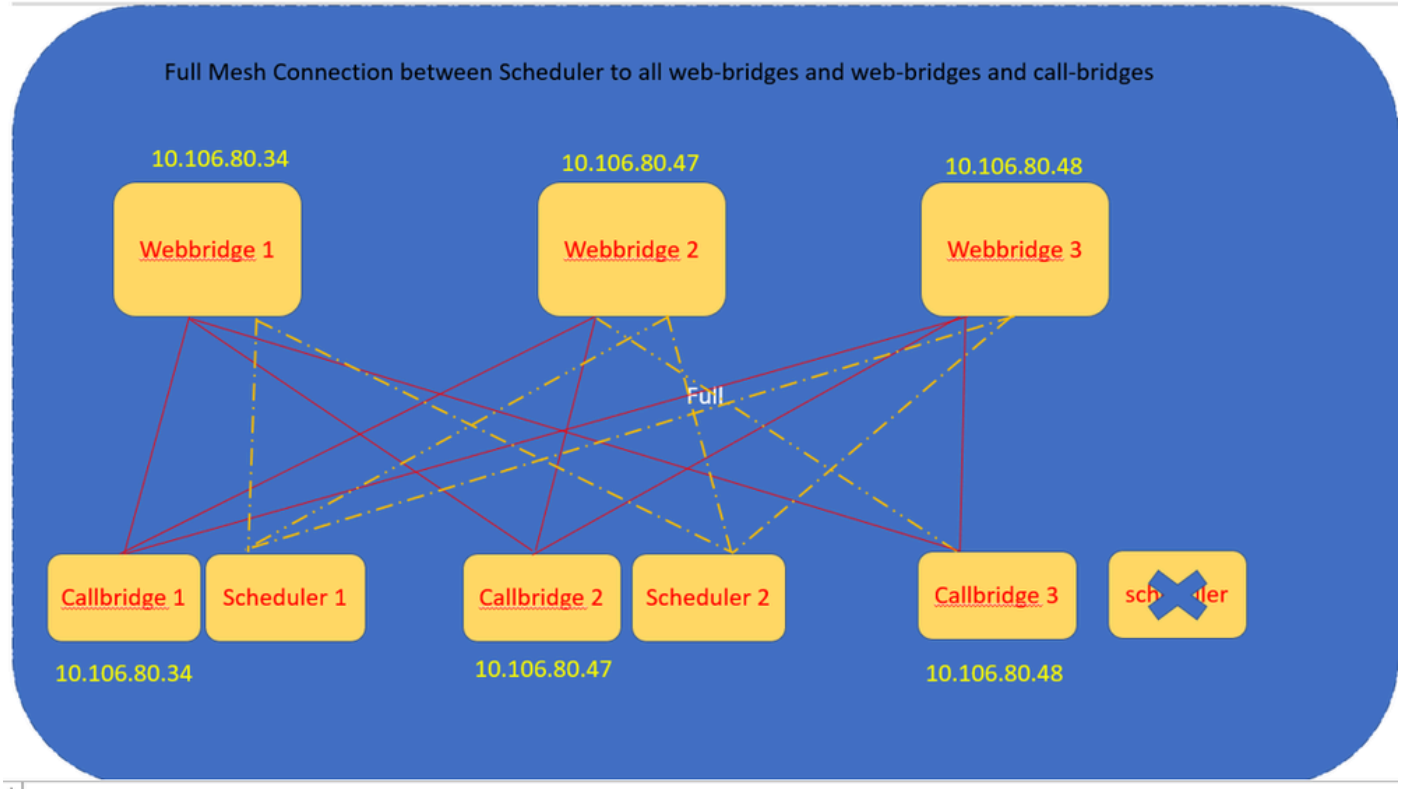
The scheduler maintains full mesh connections with all Web Bridges. In this scenario deployment has:

3 call bridges

3 Web bridges

2 Schedulers

All Call Bridges talk to all Web Bridges. Schedulers 1 and 2 are aware of web-bridge 3 because web-bridge 3 was presented to the scheduler service in the initial API call made to Call Bridge when the scheduler is enabled.



You can also configure the scheduler HTTPS interface. The scheduler has its own HTTPS interface which if enabled, can be used to configure scheduler meetings with the scheduler APIs. Here are the commands to configure:

```
scheduler https listen <interface> <port>

scheduler https certs <key-file> <crt-fullchain-file>
```

```
scheduler https listen a 9443

scheduler https certs CB344748.key  BUN344748.cer
```

Scheduler configured on CMS 1:

```
cms1> scheduler https listen a 9443
cms1> scheduler https certs CB344748.key BUN344748.cer
cms1> scheduler c2w certs CB344748.key BUN344748.cer
cms1> scheduler c2w trust wbbundle.cer
cms1> 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
SUCCESS: scheduler enabled
```

Scheduler enabled on CMS 1:

```
cms1> scheduler
Enabled                          : true
Https interface                  : a
Https port                       : 9443
Https key file                   : CB344748.key
Https certificate file           : BUN344748.cer
C2W Key file                     : CB344748.key
C2W Full chain certificate file : BUN344748.cer
C2W Trust bundle                 : wbbundle.cer
Email Server                     : none
Email Port                       : none
Email Username                   : none
Email Password                   : none
```

Scheduler enabled on CMS 2:

```
cms2> scheduler
Enabled                          : true
Https interface                  : a
Https port                       : 9443
Https key file                   : CB344748.key
Https certificate file           : BUN344748.cer
C2W Key file                     : CB344748.key
C2W Full chain certificate file : BUN344748.cer
C2W Trust bundle                 : wbbundle.cer
Email Server                     : none
Email Port                       : none
Email Username                   : none
Email Password                   : none
```

Logs snippets show:

The list of configured Web Bridges is retrieved by the scheduler with the use of the Call Bridge APIs. Persistent C2W connections are established to each Web Bridge similar to how the Call Bridge also establishes a C2W connection to each Web Bridge.

Scheduler service enabled:

<#root>

Aug 21 11:53:22.408 daemon.info cms1

scheduler_backend[2056]:

INFO CmsWebSchedulerApplication -

Starting CmsWebSchedulerApplication

```
 with PID 1 (/app started by ? in /)
```

The scheduler makes an API query to Call Bridge, a list of Web Bridges configured calls pulled by the scheduler service via API call:

```
Aug 21 11:53:28.999 daemon.info cms1 scheduler_backend[2056]:  INFO  C2WSupervisor - getWebBridges - to

Aug 21 11:53:28.999 daemon.info cms1 scheduler_backend[2056]:  INFO  C2WSupervisor - getWebBridges - ad
```

Connection is attempted by C2W to connect to all Web Bridges:

<#root>

```
Aug 21 11:53:29.011 daemon.info cms1 scheduler_backend[2056]:  INFO  C2WService -

Connecting to webBridge=10.106.80.34:8443



Aug 21 11:53:29.015 daemon.info cms1 scheduler_backend[2056]:  INFO  C2WService -

Connecting to webBridge=10.106.80.47:8443



Aug 21 11:53:29.015 daemon.info cms1 scheduler_backend[2056]:  INFO  C2WService -

 Connecting to webBridge=10.106.80.48:8443
```

```
Aug 21 11:53:29.069 daemon.info cms1 scheduler_backend[2056]:  INFO  C2WService - Received guid b685951

Aug 21 11:53:29.069 daemon.info cms1 scheduler_backend[2056]:  INFO  C2WService - Received guid 09b94d9

Aug 21 11:53:29.070 daemon.info cms1 scheduler_backend[2056]:  INFO  C2WService - Received guid 994190f
```

Scheduler service connects to Web Bridges VIA C2W and provides scheduler TAB:

<#root>

```
Aug 21 11:53:31.016 daemon.info cms1 scheduler_backend[2056]:  INFO  C2WSupervisor - C2W connection for

10.106.80.34:8443 UP
```

```
Aug 21 11:53:31.017 daemon.info cms1 scheduler_backend[2056]:  INFO  C2WSupervisor - C2W connection for
10.106.80.47:8443 UP

Aug 21 11:53:31.017 daemon.info cms1 scheduler_backend[2056]:  INFO  C2WSupervisor - C2W connection for
 10.106.80.48:8443 UP
```

The scheduler maintains FULL MESH connections with all web bridges. This deployment has:

3 Call Bridges

3 Web Bridges

2 Schedulers

All Call Bridges talk to all Web Bridges. Schedulers 1 and 2 are aware of Web Bridge 3 because Web Bridge 3 was presented to the Scheduler service at the time of the initial API call made when the scheduler is enabled.

<#root>

```
Aug 21 11:53:28.999 daemon.info cms1 scheduler_backend[2056]:  INFO  C2WSupervisor -
getWebBridges - totalCount=3

Aug 21 11:53:28.999 daemon.info cms1 scheduler_backend[2056]:  INFO  C2WSupervisor
- getWebBridges - added=3

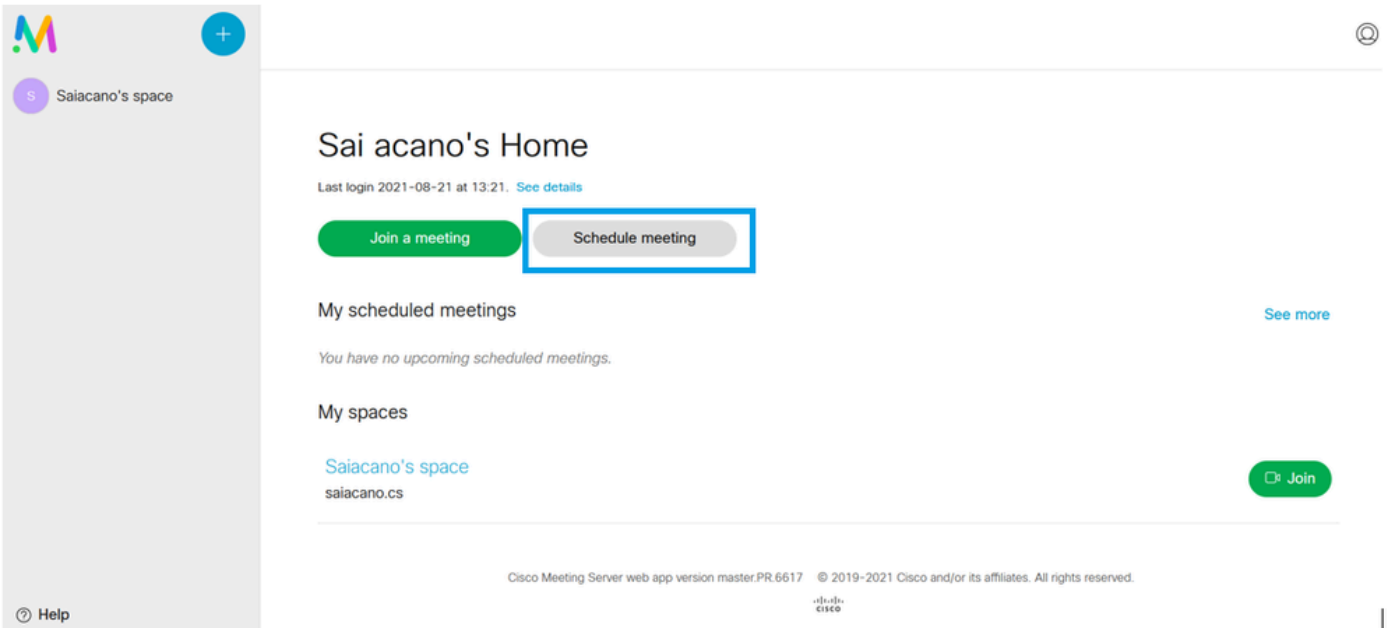
Aug 21 11:53:29.011 daemon.info cms1 scheduler_backend[2056]:  INFO  C2WService - Connecting to webBrid
Aug 21 11:53:29.015 daemon.info cms1 scheduler_backend[2056]:  INFO  C2WService - Connecting to webBrid
Aug 21 11:53:29.015 daemon.info cms1 scheduler_backend[2056]:  INFO  C2WService - Connecting to webBrid
```

Scheduler status:

```
cms1> scheduler status
Status: enabled
Running
Database responsive at start
HTTPS configured
C2W configured
Email server not configured
cms1>
```

Note: You must sign In to be able to access the scheduler functionality and it is not available for the Guest/Join users landing page.

After Scheduler is configured, the client web app schedules a meeting tab.

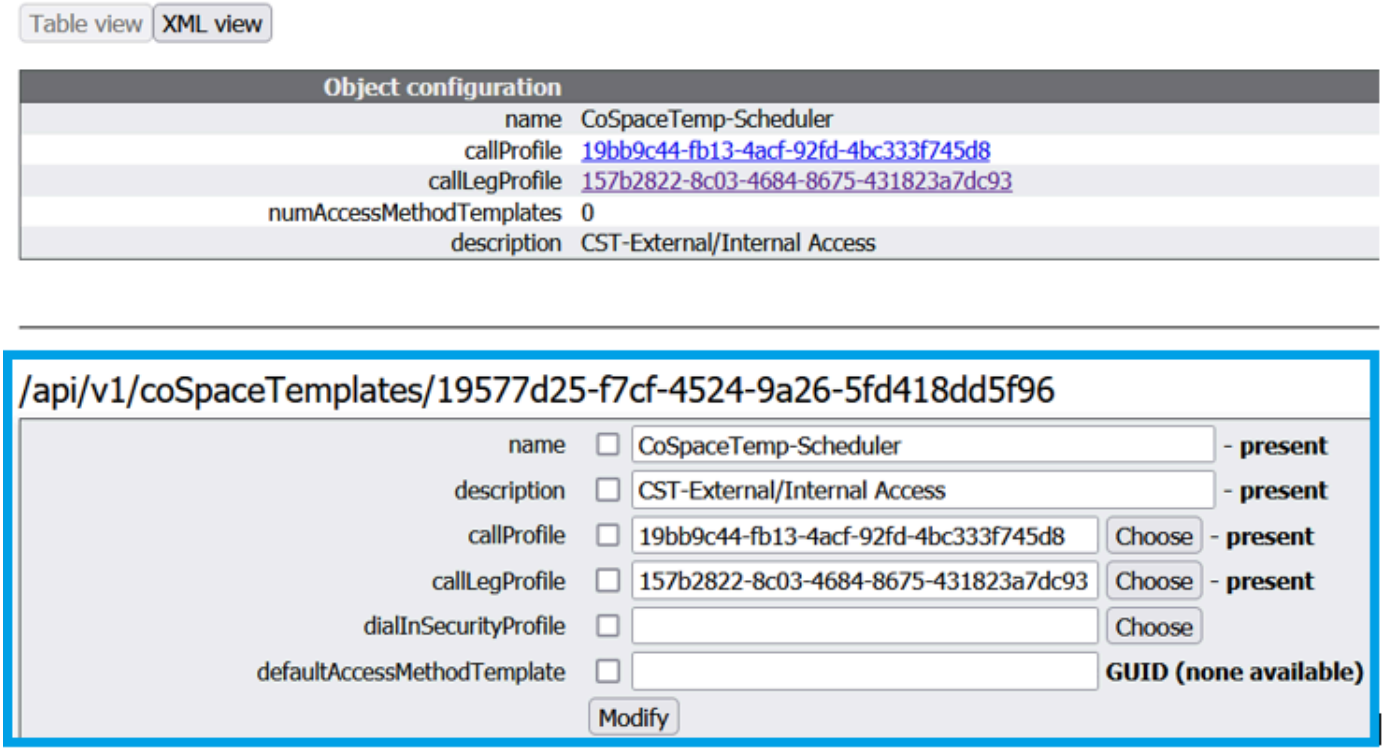


Schedule a Meeting (Optional)

Note: This is your environment-specific configuration.

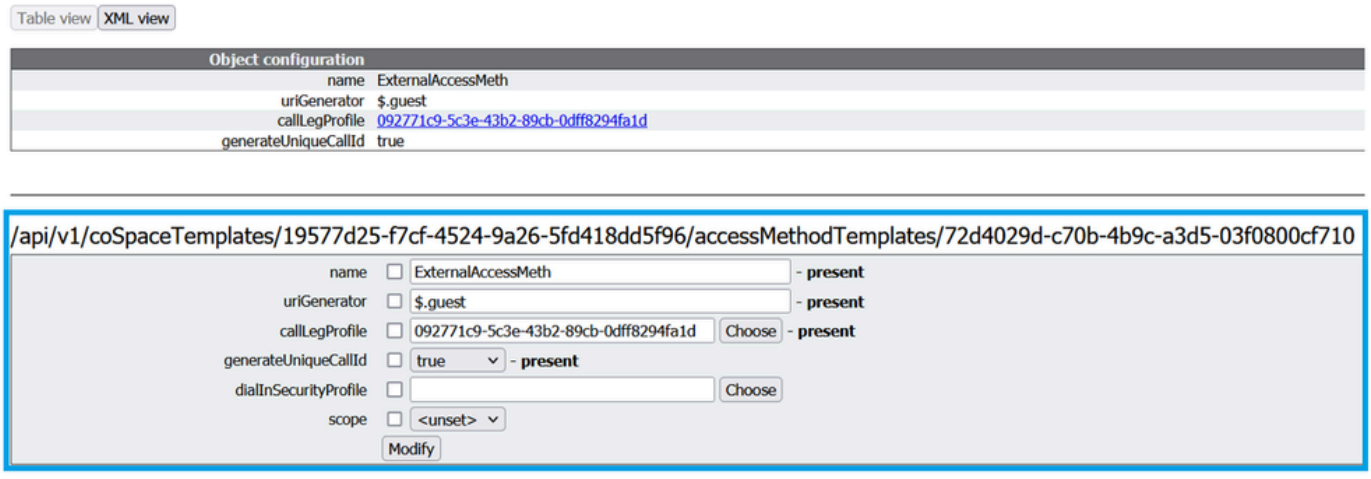
Additionally, you can configure a **CoSpaceTemplates** in order to assign it to the meeting. **CoSpaceTemplates** provides meeting access methods to the organizer and participant.

Create a CoSpace Template:

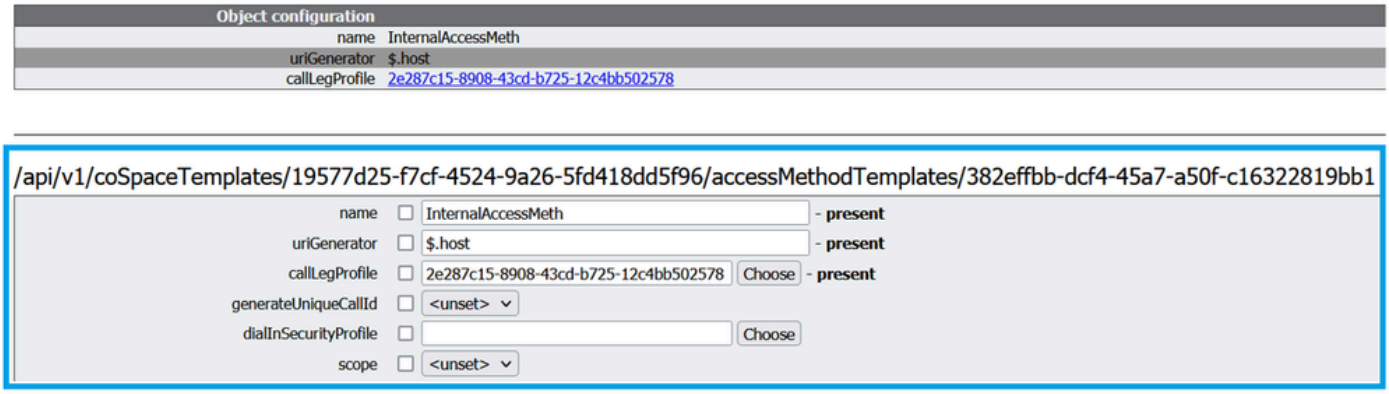


Create an Access method template, and assign it to a **CoSpaceTemplates**:

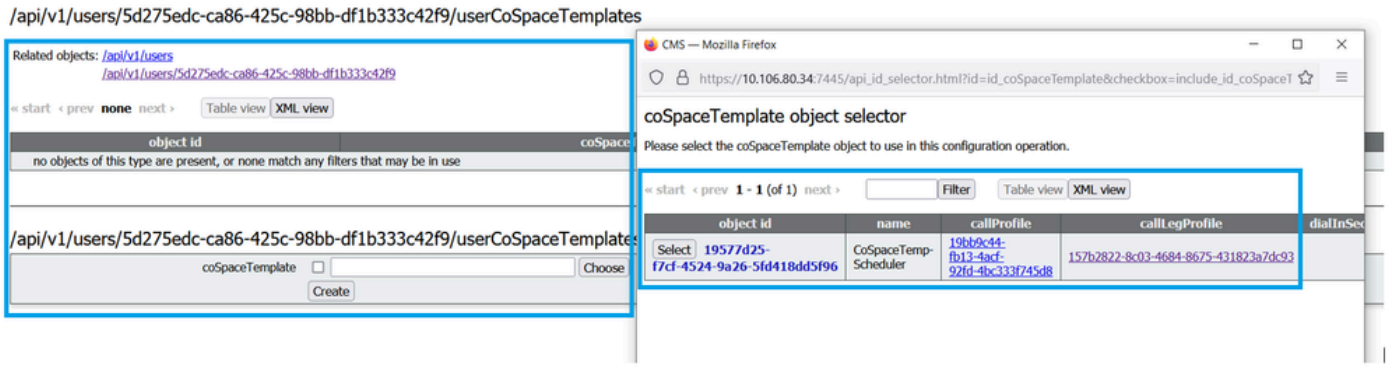
`/api/v1/coSpaceTemplates/19577d25-f7cf-4524-9a26-5fd418dd5f96/accessMethodTemplates`



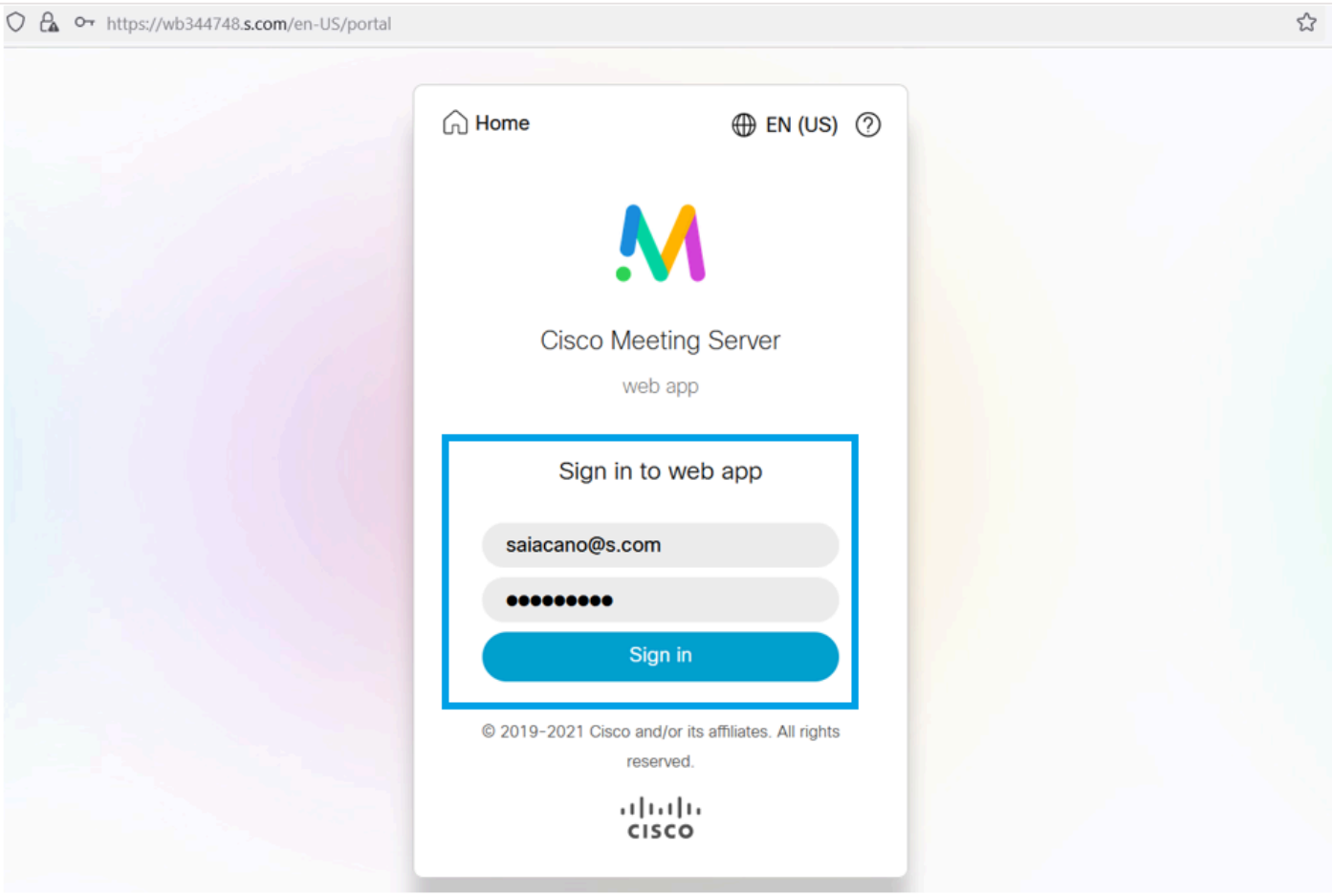
Assign additional access method if you have:



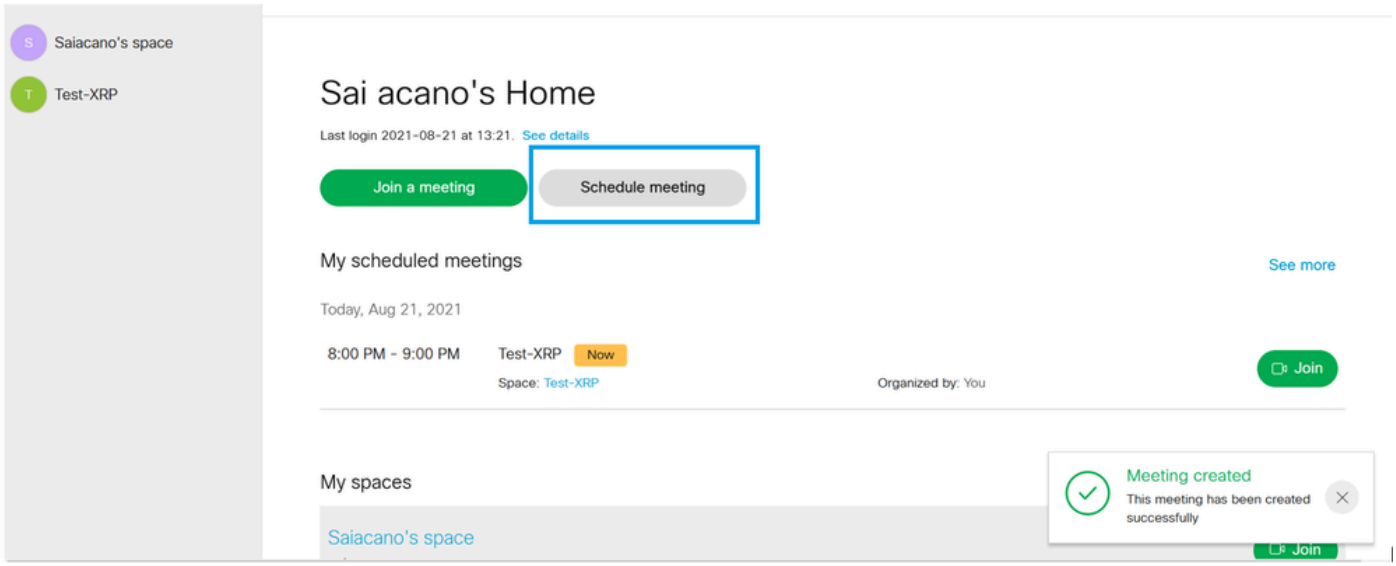
You can now assign this **CoSpaceTemplates** to an LDAP user. For test purposes assign it to 1 user.



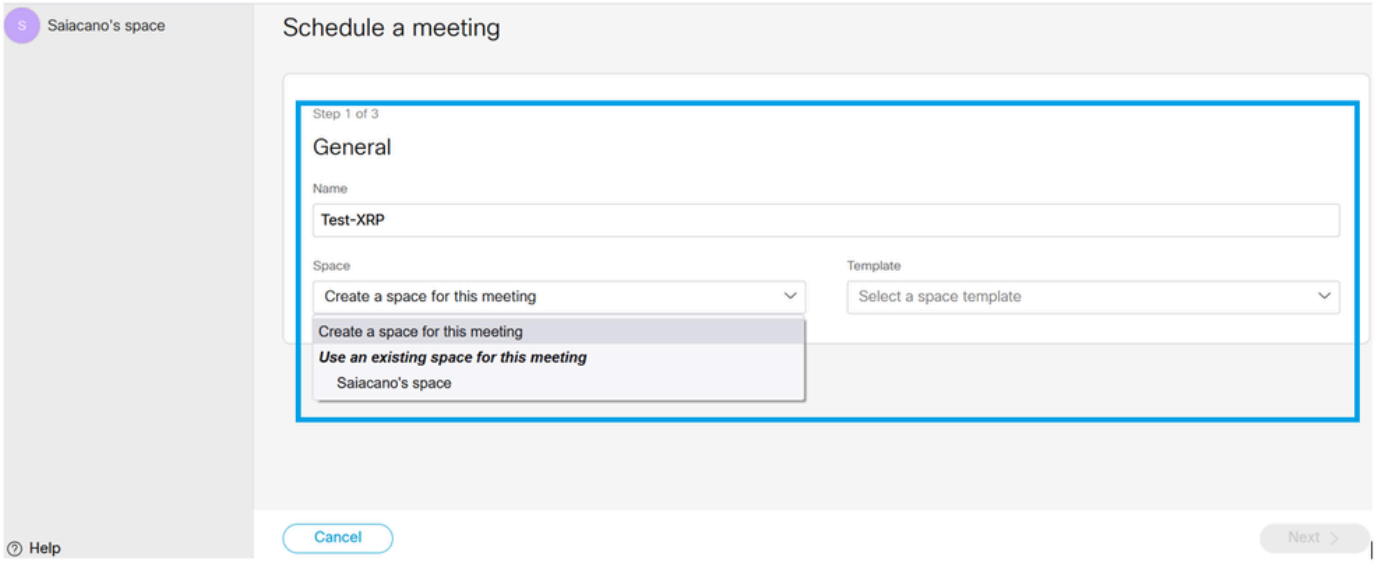
Once the template is assigned to the LDAP user. Sign in on the web app to schedule a meeting.



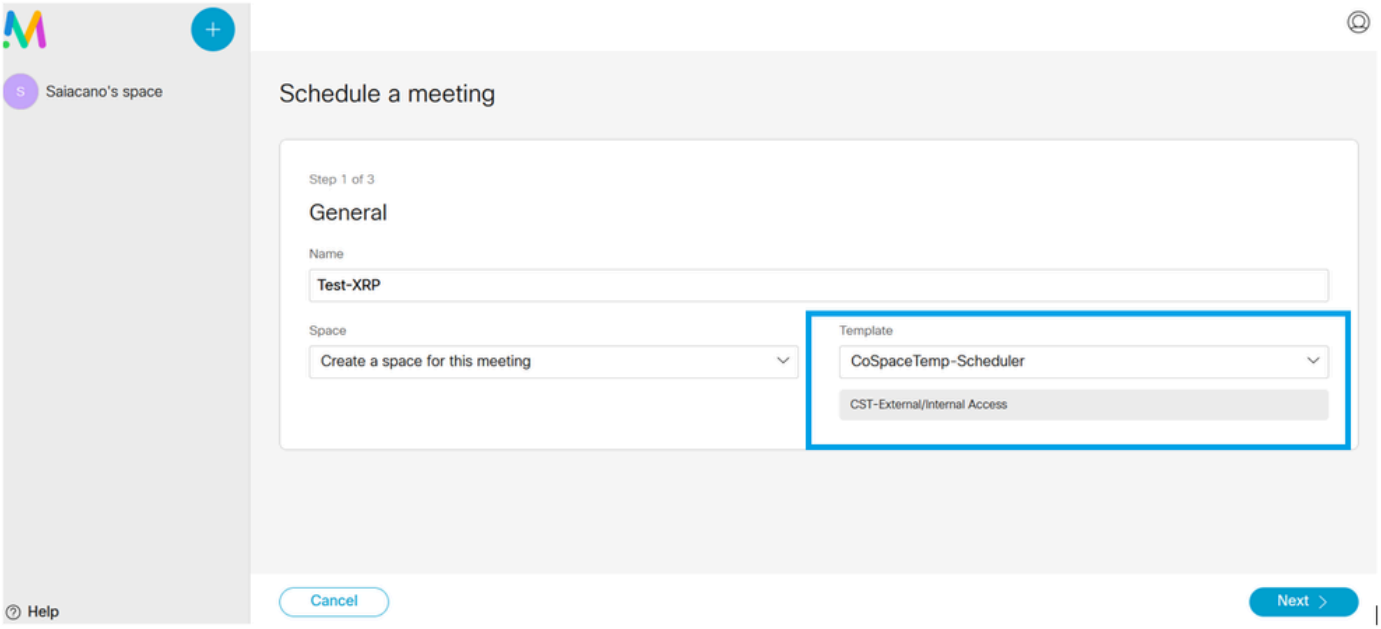
After the user has signed in, click **Schedule meeting** in order to schedule a meeting.



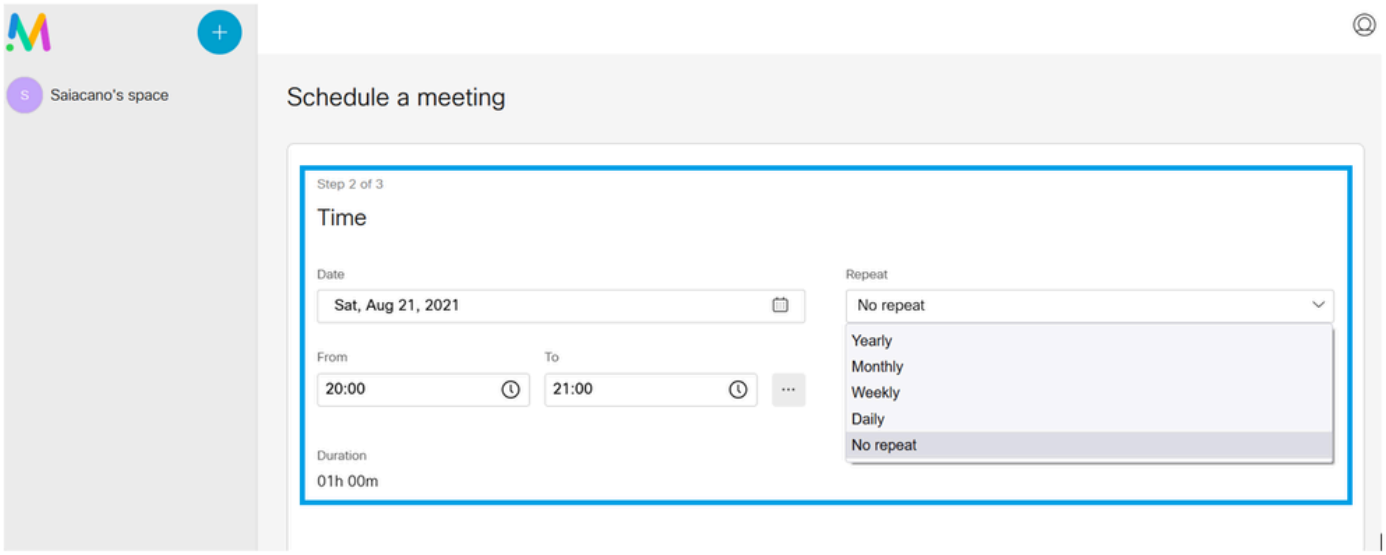
Give a name to the newly scheduled meeting and choose a **CoSpace** that already exists or create a new one.



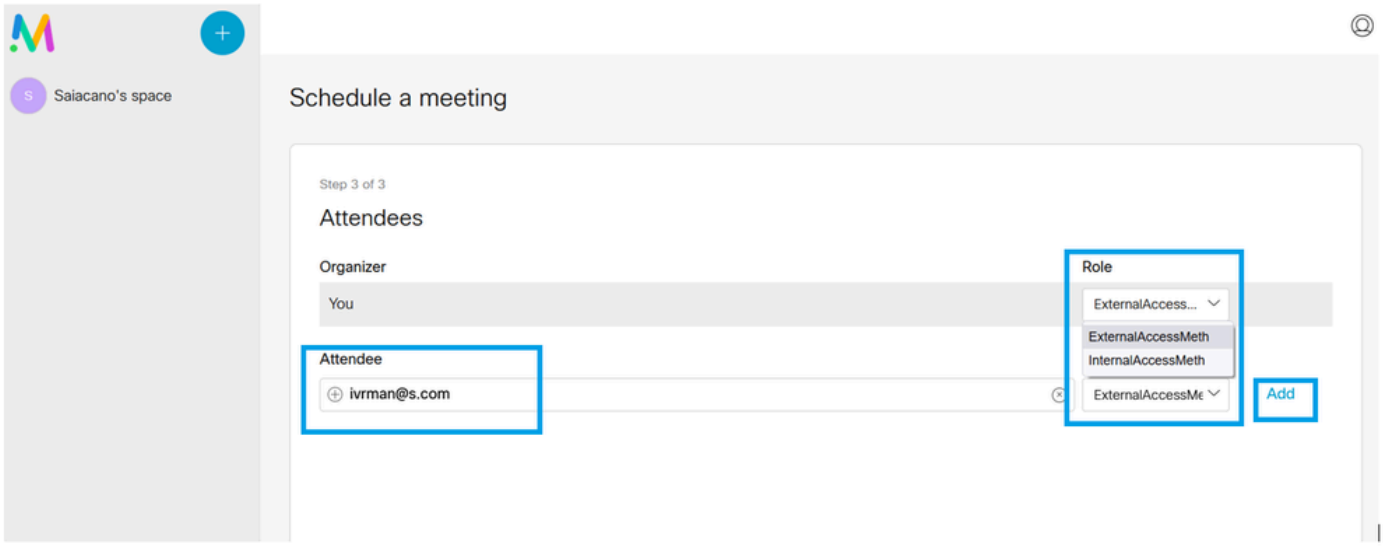
Choose the **CoSpace** template you created earlier:



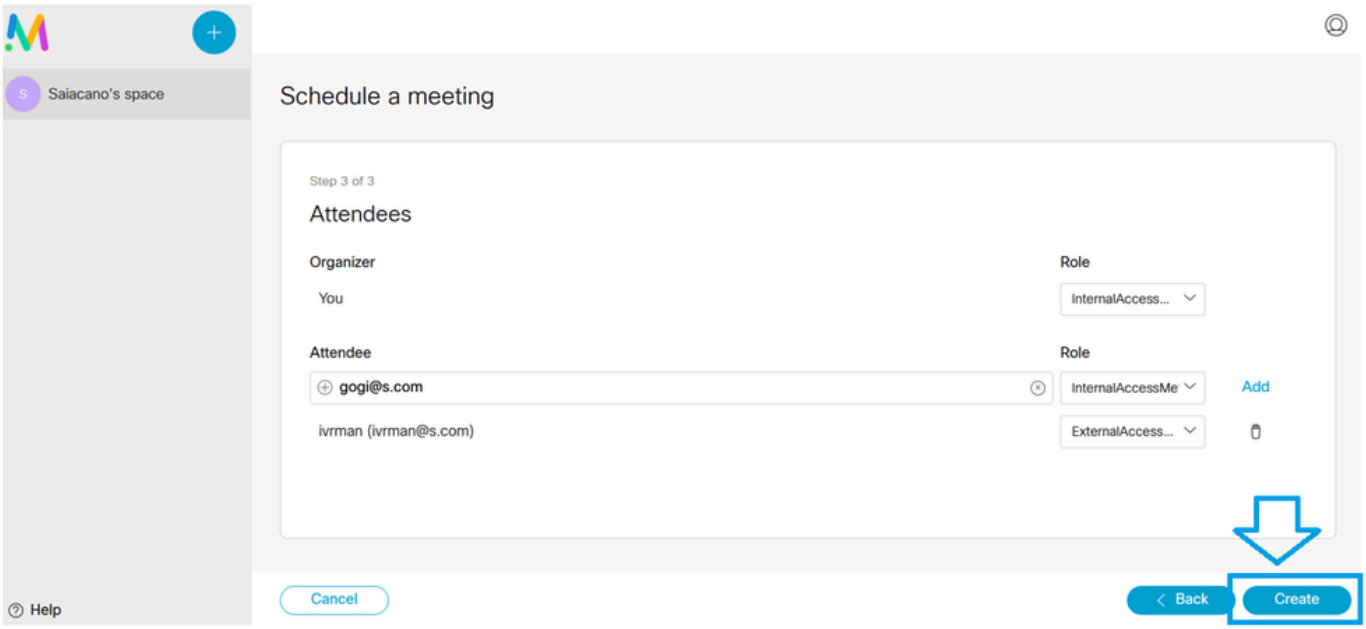
Click **Next** and set a meeting schedule (time/date/repeat or ad-hoc) as shown in the image.



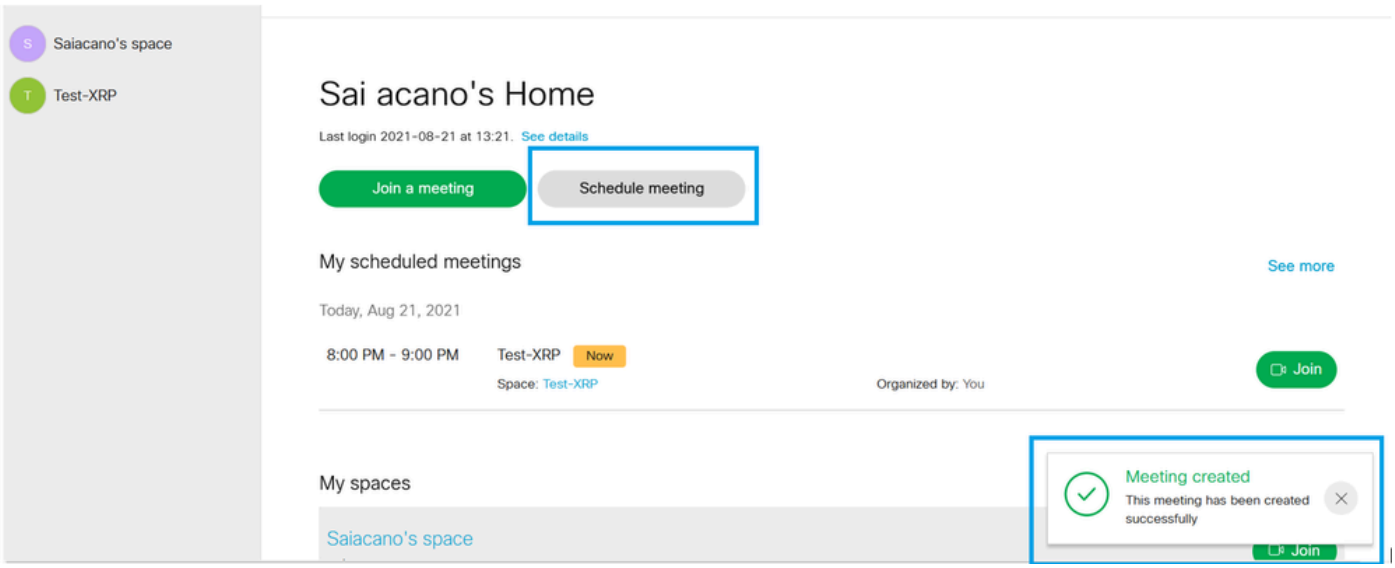
Add participants on the next page. Here you can define which participant has what access method.

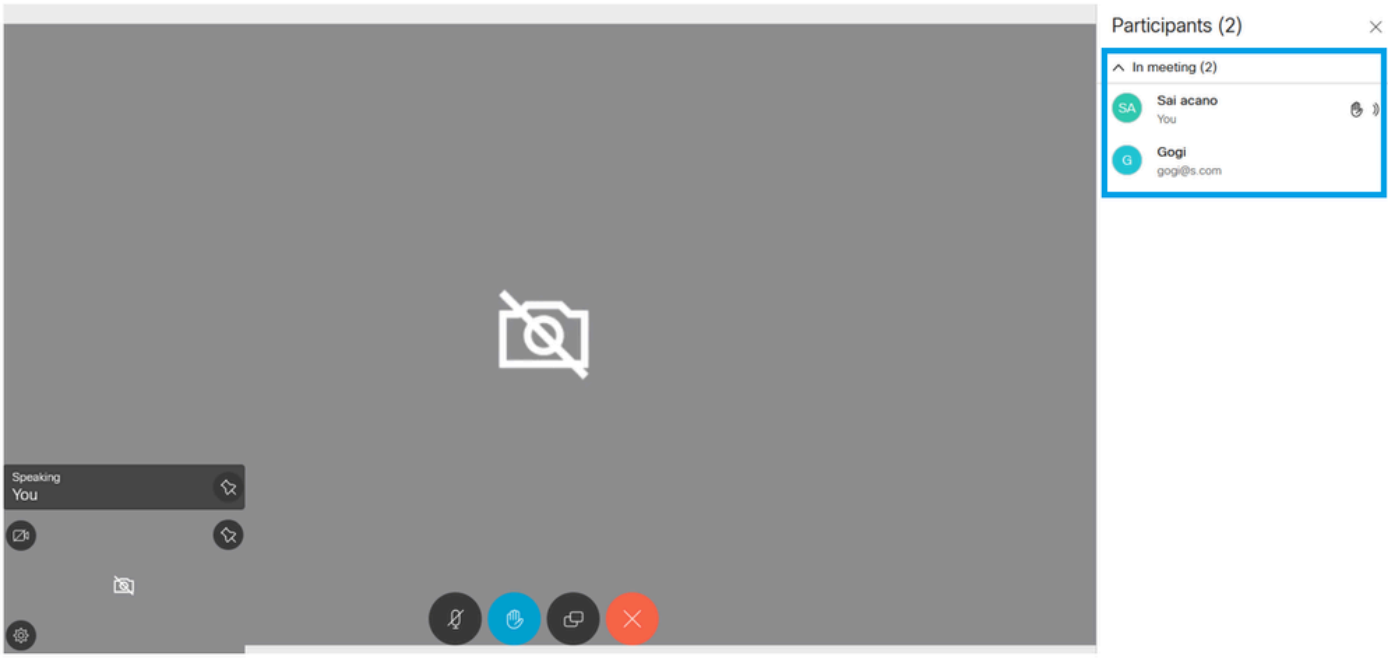


Schedule a meeting and click **Create** in order to populate on the web app.

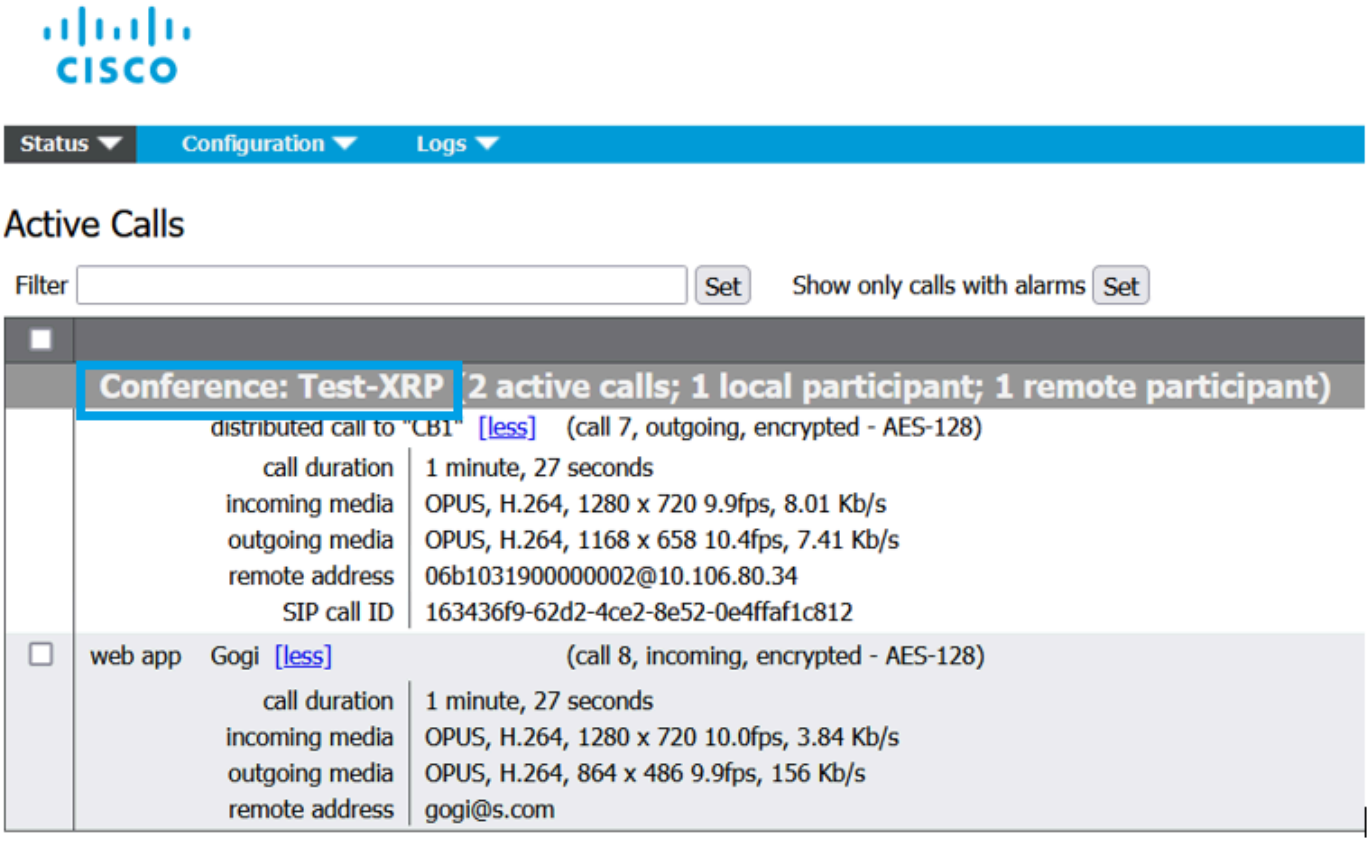


You can then click **Join a meeting** or **Schedule meeting** in order to initiate a meeting as shown in the image.





The scheduled call connects to a cluster of CMS:



Verify

There is currently no verification procedure available for this configuration.

Troubleshoot

There is currently no specific troubleshooting information available for this configuration.