xtendx®
corporate media solutions

simplex Meeting

External Meeting Authorization Service

xtendx AG | Adlikerstrasse 246, CH-8105 Regensdorf
t +41 43 299 92 92 **e** contact@xtendx.com **e** support@xtendx.com **w** www.xtendx.com

Index

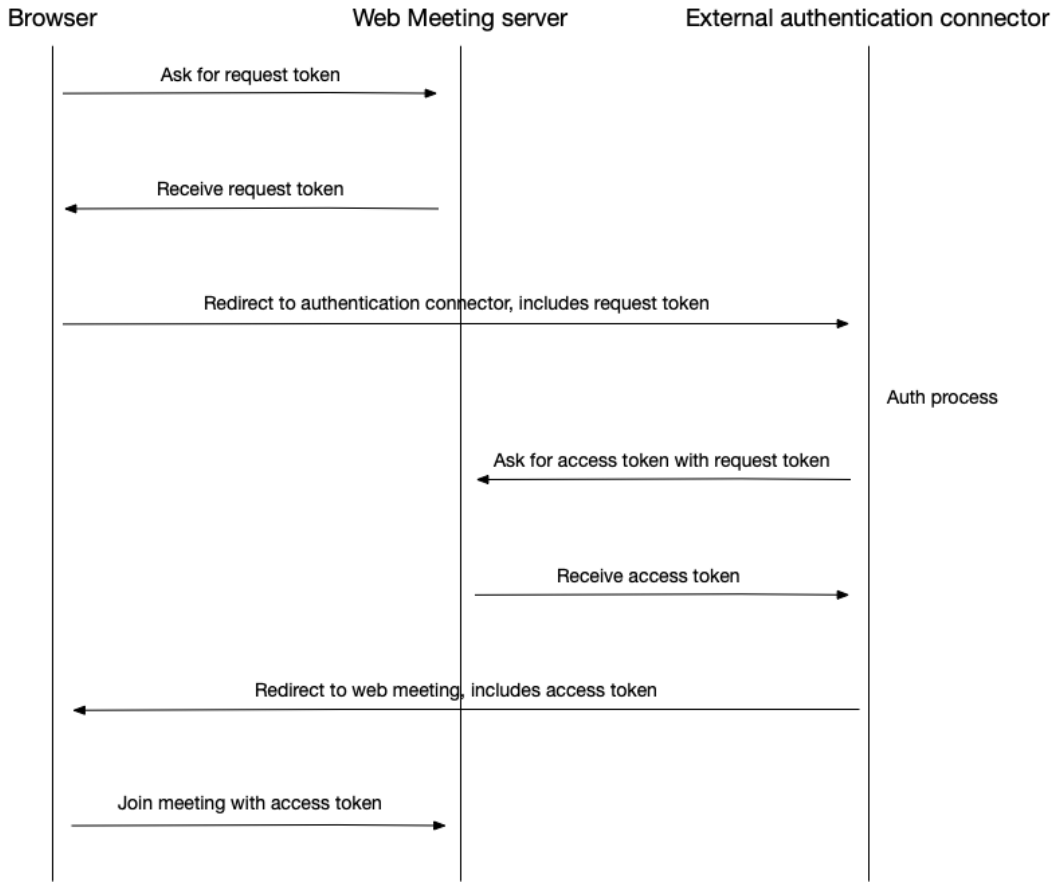
1 Overview

The web meeting platform allows users to restrict access to their meetings in various ways. One way is to delegate the authorization process to an external web service. This external web service will be responsible for the authentication and authorization of meeting participants.

This document describes how such a meeting authorization service needs to be implemented.

2 High level architecture

To join a restricted meeting the web browser requests an “Request Token” from the web meeting platform. The system the automatically redirects the Browser to the “External authentication connector”. This redirect contains the “Request Token”, among other information. The “External authentication connector” will perform the authentication and authorization process. The connector will contact the web meeting server to exchange the “Request Token” for an “Access Token” once it has determined that the user is allowed to join the meeting. To complete the process the browser will be redirected to the Web Meeting platform. This redirect contains the “Access Token” which grants the browser session access to that meeting.



3 Implementation Details

3.1 HTTP GET to the authorization service

The external meeting authorization service needs to implement an “HTTP GET” API and configure this URL in the web meeting platform (see below)

The web browser will be redirect to that API URL. The request will contain at least the following query parameters:

hostname	The host name of the web meeting platform, for example “meeting.example.org”
meetingId	The internal meeting ID of the meeting, for example “5f521a93c20ff6721fbb6a6c”
meetingToken	The human readable meeting ID, for example “0000-0000-0000-0000”
requestToken	The unique request token as a long, random string

Additional query parameters may be passed alongside the mandatory parameters defined above.

Example:

If the new API listens to “https://external.example.org/auth”, then the Web Meeting platform will redirect the user to:

https://external.example.org/auth?hostname=webmeeting.example.com&meetingId=5f521a93c20ff6721fbb6a6c&meetingToken=8320-2640-2482-3499&requestToken=dedf1722-661f-4004-9aaf-d3e56c498859-a27fd10f-b697-4c83-bca0-cb764cfd6c43&optionalParameter1=optionalValue1&optionalParameter2=optionalValue2

3.2 Exchange Request Token for Access Token

Once the service has authorized the user it needs to exchange the Request Token for an Access Token. The Web Meeting platform provides an API for this:

https://<HOSTNAME>/api/v6/meeting-room/auth/<SECRET>/access-token/<MEETING-ID>/<REQUEST-TOKEN>

In the above example call, replace the values between brackets (e.g. <HOSTNAME>) with the following parameters:

HOSTNAME	Hostname from the redirect request above, e.g. meeting.example.org
MEETING_ID	The internal meeting ID of the meeting, for example “5f521a93c20ff6721fbb6a6c”
REQUEST-TOKEN	The unique request token as a long, random string
SECRET	A random string defined by the Service itself, see below for details

The Web Meeting platform will validate the request token to make sure it was issued for the meeting ID. Upon success the API will respond with a JSON document in the following form:


```
{
 "responseCode": 0,
 "data": {
  "meetingId": "5f521a93c20ff6721fbb6a6c",
  "accessToken": "81430667-540e-4755-b32a-b5c51f704c7b-03526573-1494-48fb-a648-e80073275976"
 }
}
```


The responseCode will be 0 (zero), the “data” object will contain an “accessToken” string. This access token will be used in the final step.

3.3 Redirect the user back to the Web Meeting platform

https://<HOSTNAME>/join/<MEETING-TOKEN>?meetingAccessToken=<ACCESS-TOKEN>

Once an Access Token has been received, the user needs to be redirected to the web meeting platform. The URL must be the following:

In the example above, replace the values between brackets (e.g. <HOSTNAME>) with the following parameters:

HOSTNAME	Hostname from the redirect request above, e.g. meeting.example.org
MEETING-TOKEN	The human readable meeting ID, for example “0000-0000-0000-0000”
ACCESS-TOKEN	The unique access token received from the exchange of the Request Token

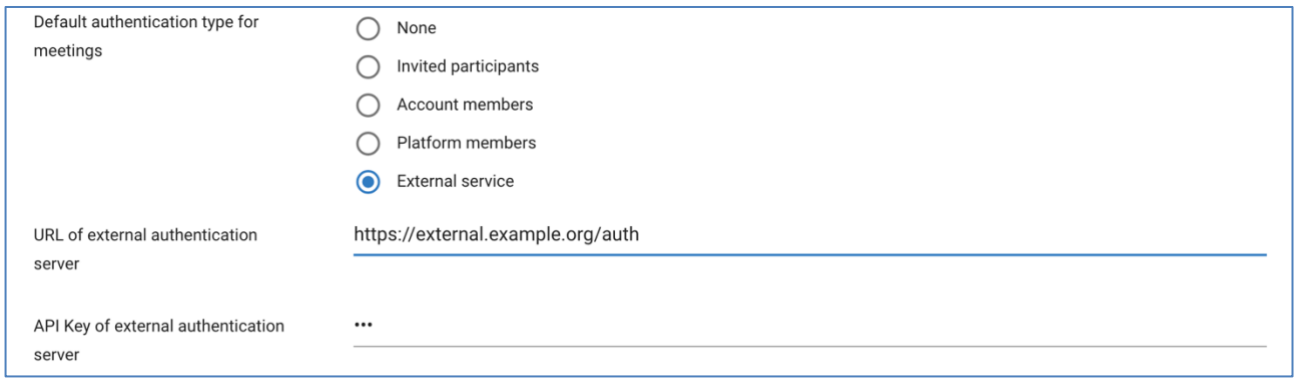
For the convenience of the user, additional, optional query parameters may be passed along:

PARTICIPANT-NAME	The name of the user, will be prefilled on the “Join Meeting” screen
PARTICIPANT-EMAI	The email address of the user, will be prefilled on the “Join Meeting” screen

https://<HOSTNAME>/join/<MEETING-TOKEN>?meetingAccessToken=<ACCESS-TOKEN>&
participantName=<PARTICIPANT-NAME>&participantEmail=<PARTICIPANT-EMAIL>

4 Configure the Web Meeting platform

Once the authorization service was implemented you need to configure the Web Meeting platform accordingly. In “Platform Settings” -> “System Configuration” -> “Meeting Room”, select “External Service” as the “Default authentication type for meetings”. This will open two additional configuration fields:



Type in the full API URL into the field “URL of external authentication server”. You will also need to fill in the SECRET defined above into the “API Key of external authentication server”. The Web Meeting platform will store the SECRET in an encrypted way. All API calls to exchange a “Request Token” for an “Access Token” will be verified against this secret.

5 Example implementation in NodeJS

```
const express = require('express')
const app = express()
const port = 3000

const SECRET = "MySecret"

app.get('auth', (req, res) => {
 // Read mandatory parameters
 const hostname = req.query.hostname;
 const meetingId = req.query.meetingId;
 const meetingToken = req.query.meetingToken;
 const requestToken = req.query.requestToken;

 // TODO: Implement authentication and authorization process here

 // User is authorized to join the meeting, exchange request token for access token
 let url = `https://${hostname}/api/v6/meeting-room/auth/${SECRET}/access-token/${meetingId}/${requestToken}`;
```

```
 const options = {
  url: url,
  method: "GET",
  headers: {
   "Content-Type": "application/json; charset=utf-8"
  },
 };

 request(options, (err, res, body) => {
  const response = JSON.parse(body);

  // Access token received
  const accessToken = response.data.accessToken;

  const redirectUrl = `https://${hostname}/${meetingToken}/join?meetingAccessToken=${accessToken}`;

  res.redirect(redirectUrl)
 });
})

app.listen(port, () => {
 console.log(`Example app listening on https://external.example.org`)
})
```