Configure and Troubleshoot SSO on Cisco Unified Communications Manager (CUCM)

Contents

Introduction

This document describes the SSO feature in CUCM, configuration, tips to troubleshoot, example log analysis, and resources for additional information.

Prerequisites

Requirements

Cisco recommends knowledge of a few Single Sign-On (SSO) terms:

- Security Assertion Markup Language (SAML) - an open standard to exchange authentication and authorization data between parties
- Service Provider (SP) - The SP is the entity that is hosts the service. In this document, Cisco Unified Communications Manager (CUCM) is the service provider
- Identity Provider (IdP) - The IdP is the entity that authenticates client credentials. Authentication is completely transparent to the SP so the credentials can be a smart card, username/password, and so on. Once the IdP authenticates client credentials, it generates an assertion, sends it to the client, and redirects the client back to the SP
- Assertions – A time-sensitive piece of information that the IdP generates after successful authentication of a user. The purpose of the assertion is to provide information about the authenticated user to the SP
- Bindings – defines the transport method used to deliver the SAML protocol messages between entities. Cisco Unified Communications products use HTTP
- Profiles – predefined constraints and combinations of SAML message content (assertions, protocol, bindings) that work to achieve a specific business use-case. This training focuses on the Web browser Single Sign-On profile as that is the method used by CUCM
- Metadata – set of configuration information that is exchanged between parties. Contains information such as supported SAML bindings, operational roles such as IdP or SP, supported identifer attributes, identifier information, and certificate information used to sign and encrypt the request or response.

Components Used

- Cisco Unified Communications Manager (CUCM) 12.5.1.14900-63
- Microsoft Windows Server 2016
- Active Directory Federation Services (AD FS) 4.0

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

The purpose of SSO is to allow users and administrators to access multiple Cisco collaboration applications withou require separate authentications to each one. Enablement of SSO results in several advantages:

- It improves productivity because users do not need to re-enter credentials for the same identity on different products.
- It transfers the authentication from your system that hosts the applications to a third party system. You create a circle of trust between an IdP and a service provider which allows the IdP to authenticate users on behalf of the SP.
- It provides encryption to protect authentication information passed between the IdP, service provider, and user.SSOalso hides authentication messages passed between the IdP and the service provider from any external party.
- It can reduce costs as fewer help desk calls are made for password resets.

Circle of Trust

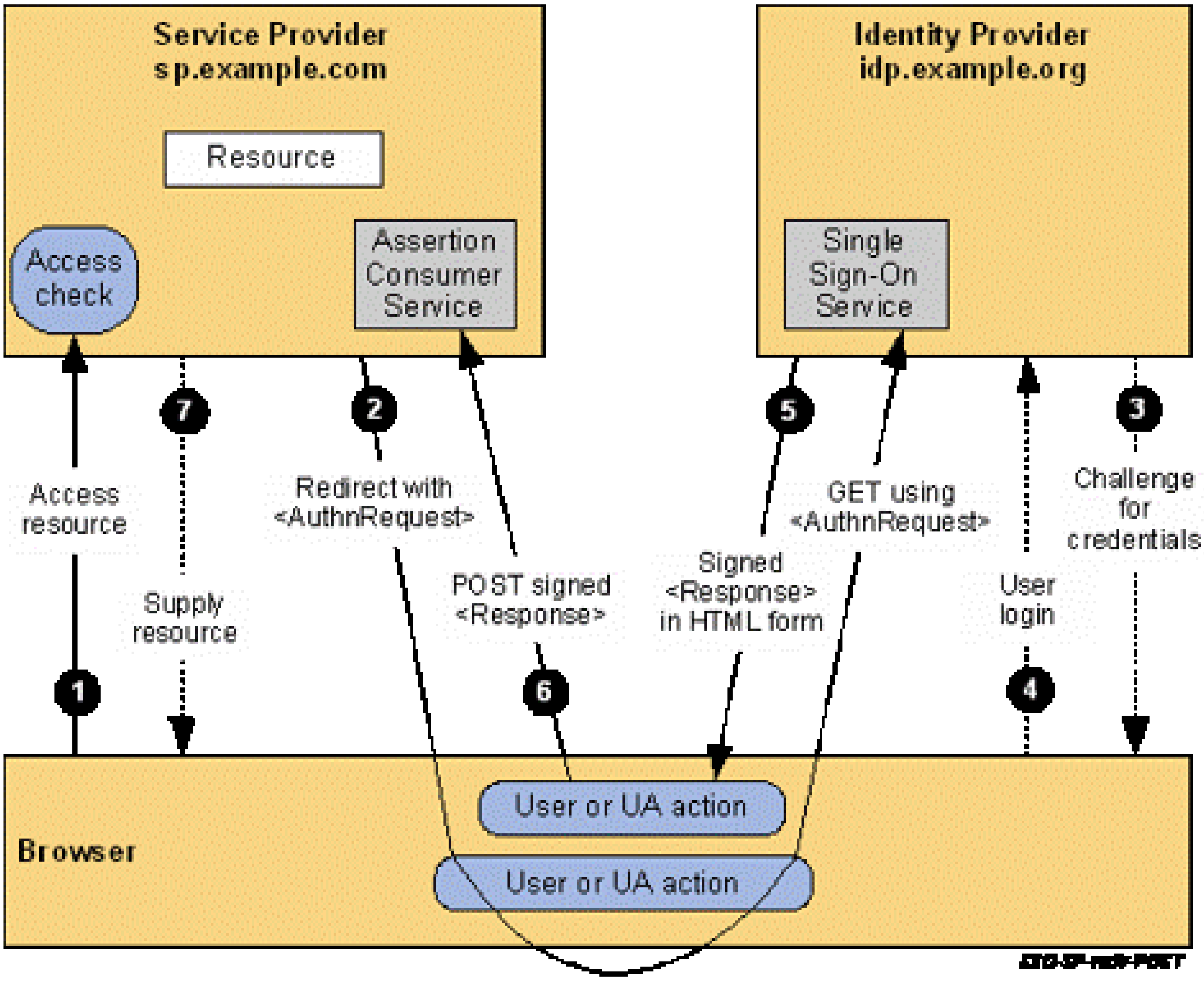
Certificates play a very important role in SSO and they are exchanged between the SP and IdP via metadata files. The SP metadata file contains the service provider signing- and encryption certificate along with some other important information such as the Assertion Consume Service Index values and HTTP POST/REDIRECT information. The IdP metadata file contains its certificate(s) along with some other informataion about the IdP capabilities. You must import the SP metadata into the IdP and import the IdP metadata into the SP to create a circle of trust. Essentially, the SP signs and encrypts any request that it generates with the certificate that the IdP trusts, and the IdP signs and encrypts any assertion (response) it generates with certificate(s) that the SP trusts.

Note: If certain information on the SP changes, such as the hostname/Full Qualified Domain Name (FQDN) or signing/encryption certificate (Tomcat or ITLRecovery), then the circle of trust can be broken. Download a new metadata file from the SP and import it into the IdP. If certain information on the IdP changes, download a new metadata file from the IdP and re-run the SSO test so that you can update the information on the SP. If you are not sure whether your change necessitates a metadata update on the opposite device, it is best to update the file. There is no downside to a metadata update on either side and this is a valid step to troubleshoot SSO issues, especially if there has been a configuration change.

Configure

Network Diagram

The flow for a standard SSO log in is shown in the image:

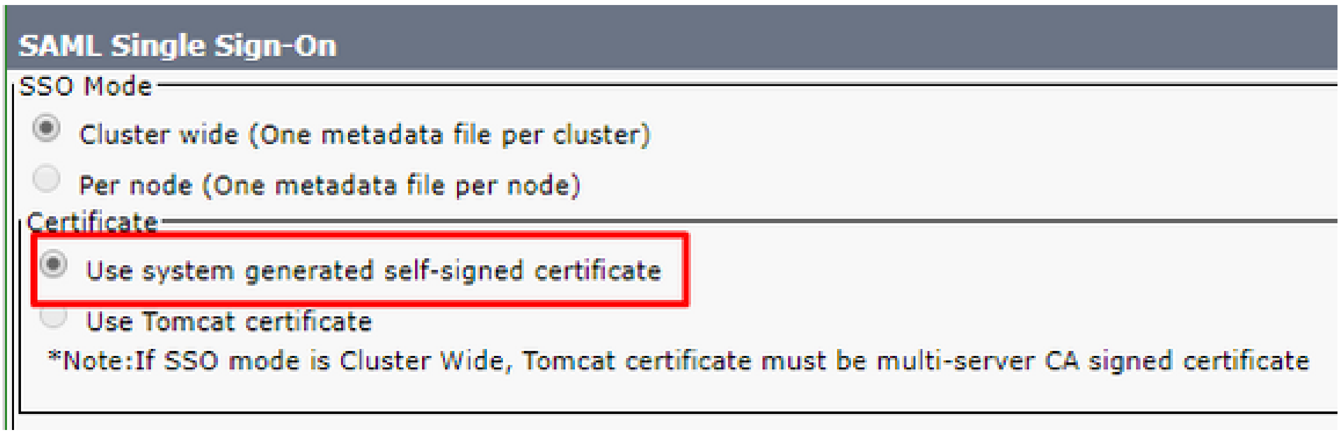


Note: The process in the image is not in order from left to right. Remember that the SP is CUCM and the IdP is the third-party application.

Configuration

From the CUCM perspective, there is very little to configure with regard to SSO. In CUCM 11.5 and higher, you can select Cluster wide or per-node SSO.

- In CUCM 11.5, Cluster wide SSO requires a multi-server tomcat certificate to be installed on all nodes since there is only one metadata file for the whole cluster (and the certificate is stored in that file, so each node must have the same tomcat certificate).
- In CUCM 12.0 and up, you have the option to **Use system generated self-signed certificate** for Cluster wide SSO. This option uses the ITLRecovery certificate rather than tomcat:


SAML Single Sign-On

SSO Mode
[✓ Cluster wide (One metadata file per cluster)]
Per node (One metadata file per node)

Certificate
[✓ Use system generated self-signed certificate]
Use Tomcat certificate
*Note:If SSO mode is Cluster Wide, Tomcat certificate must be multi-server CA signed certificate

- Per-node SSO is the default prior to CUCM 11.5. In a per-node configuration, each node has its own metadata file that needs to be imported to the IdP since any of those nodes can potentially redirect a user for authentication.
- You can also enable SSO for RTMT in CUCM 11.5. This is enabled by default and it is located at **Cisco Unified CM Administration** > **Enterprise Parameters** > **Use SSO for RTMT**.

Note: The note that states **If SSO mode is Cluster Wide, Tomcat certificate must be multi-server CA Signed certificate** is erroneous on 12.0 and 12.5 and a defect has been opened to correct it (Cisco bug ID CSCvr49382).

Aside from these options, the rest of the configuration for SSO is on the IdP. The configuration steps can differ drastically based on which IdP you choose. These documents contain steps to configure some of the more common IdPs:

- Microsoft AD FS Configuration Guide
- Okta Configuration Guide
- PingFederate Configuration Guide
- Microsoft Azure Configuration Guide

Troubleshoot

Data to Collect

In order to troubleshoot an SSO issue, set the SSO traces to debug. The SSO log level cannot be set to debug via GUI. To set the SSO log level to debug, run this command in the CLI: **set samltrace level debug**

Note: This command is not Cluster wide, so it needs to be run on each node that could be involved with an SSO log in attempt.

Once the log level has been set to debug, reproduce the issue and collect this data from CUCM:

- **Cisco SSO logs**
- **Cisco Tomcat logs**

Most SSO issues generate exceptions or errors in the SSO logs but in some circumstances, the Tomcat logs can be helpful as well.

Example Analysis

Device information from TAC lab

CUCM (Service Provider):

- Version: 12.5.1.14900-11
- FQDN: 1cucm1251.sckiewer.lab

Windows Server 2016 (Identity Provider):

- Active Directory Federation Services 3.0
- FQDN: WinServer2016.sckiewer.lab

Log Review for CUCM

```
tomcat/logs/ssosp/log4j/

%%%%% A user has attempted to access Cisco Unified CM Administration
2021-04-30 09:00:53,156 DEBUG [http-bio-443-exec-83] filter.SSOAuthAgentFilter - servlet path :/showHom
2021-04-30 09:00:53,157 DEBUG [http-bio-443-exec-83] filter.SSOAuthAgentFilter - recovery URL :/showRec

%%%%% You can see the SP and IdP EntityIDs here
2021-04-30 09:00:53,194 DEBUG [http-bio-443-exec-83] fappend.SamlLogger - SPSSOFederate: spEntityID is
2021-04-30 09:00:53,194 DEBUG [http-bio-443-exec-83] fappend.SamlLogger - SPSSOFederate: idpEntityID :

%%%%% The client is redirected to the SSO URL listed here
2021-04-30 09:00:53,196 DEBUG [http-bio-443-exec-83] fappend.SamlLogger - SPSSOFederate: SingleSignOnSe

%%%%% CUCM prints the AssertionConsumerService URL and you can see that CUCM uses an HTTP-POST
2021-04-30 09:00:53,196 DEBUG [http-bio-443-exec-83] fappend.SamlLogger - SPSSOFederate: AssertionConsu
2021-04-30 09:00:53,196 DEBUG [http-bio-443-exec-83] fappend.SamlLogger - SPSSOFederate: AssertionConsu
2021-04-30 09:00:53,196 DEBUG [http-bio-443-exec-83] fappend.SamlLogger - SPSSOFederate: AssertionConsu

%%%%% Here CUCM prints the AuthnRequest to the client. The client is redirected to the IdP with a 302 a
2021-04-30 09:00:53,199 DEBUG [http-bio-443-exec-83] fappend.SamlLogger - SPSSOFederate: AuthnRequest:<
ID="s29fd87c888ef6a4bc8c48d7e7087a8aeb997dd76f" Version="2.0" IssueInstant="2021-04-30T13:00:53Z" Desti
<saml:Issuer xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">1cucm1251.sckiewer.lab</saml:Issuer>
<samlp:NameIDPolicy xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol" Format="urn:oasis:names:tc:SAML:
</samlp:AuthnRequest>

%%%%% You can see that CUCM has received an encoded SAML response that is base64 encoded
2021-04-30 09:01:03,986 DEBUG [http-bio-8443-exec-85] authentication.SAMLAuthenticator - SAML Response
```

```
%%%%% Here is the encrypted SAML response from the client. You can see that the InResponseTo value matc
2021-04-30 09:01:04,005 DEBUG [http-bio-8443-exec-85] fappend.SamlLogger - SPACSUtils.getResponse: got
<samlp:StatusCode xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
Value="urn:oasis:names:tc:SAML:2.0:status:Success">
</samlp:StatusCode>
</samlp:Status><EncryptedAssertion xmlns="urn:oasis:names:tc:SAML:2.0:assertion"><xenc:EncryptedData xm

%%%%% Here you can see that the IdP uses a supported binding type
2021-04-30 09:01:04,010 DEBUG [http-bio-8443-exec-85] fappend.SamlLogger - SAML2Utils.verifyResponse:bi

%%%%% The decrypted assertion is printed here. You see that a lot of important information covered late
2021-04-30 09:01:04,090 DEBUG [http-bio-8443-exec-85] authentication.SAMLAuthenticator - <Assertion xml

%%%%% CUCM looks at its current time and makes sure that it is within the validity timeframe of the ass
2021-04-30 09:01:04,090 DEBUG [http-bio-8443-exec-85] authentication.SAMLAuthenticator - Time Valid?:tr
2021-04-30 09:01:04,090 DEBUG [http-bio-8443-exec-85] authentication.SAMLAuthenticator - SAML Authentic
2021-04-30 09:01:04,090 DEBUG [http-bio-8443-exec-85] authentication.SAMLAuthenticator - Attributes: {u

%%%%% CUCM prints the username here
2021-04-30 09:01:04,090 DEBUG [http-bio-8443-exec-85] authentication.SAMLAuthenticator - userid is ::ad
2021-04-30 09:01:04,090 DEBUG [http-bio-8443-exec-85] authentication.SAMLAuthenticator - Realy state is
2021-04-30 09:01:04,091 DEBUG [http-bio-8443-exec-85] authentication.SAMLAuthenticator - http request c

%%%%% The client is redirected to the resource it initially tried to access
2021-04-30 09:01:04,283 INFO [http-bio-8443-exec-85] servlet.RelayToOriginalAppServlet - relayUrl ::/cc
2021-04-30 09:01:04,284 INFO [http-bio-8443-exec-85] servlet.RelayToOriginalAppServlet - redirecting to
```

Closer Look at the SAML Request and Assertion

SAML Request

Analysis and information about the SAML request:

```
AuthnRequest:<samlp:AuthnRequest xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"

%%%%% The ID from the request is returned in the assertion generated by the IdP.  This allows CUCM to c
%%%%% This log snippet was taken from CUCM 12.5, so you use the AssertionConsumerServiceIndex rather th
ID="s29fd87c888ef6a4bc8c48d7e7087a8aeb997dd76f" Version="2.0" IssueInstant="2021-04-30T13:00:53Z" Desti
<saml:Issuer xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">1cucm1251.sckiewer.lab</saml:Issuer>

%%%%% The NameID Format must be transient.
%%%%% The SP Name Qualifier allows us to see which node generated the request.
<samlp:NameIDPolicy xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol" Format="urn:oasis:names:tc:SAML:
</samlp:AuthnRequest>
```

Assertion

Analysis and information about the SAML response:

<#root>

```
<Assertion xmlns="urn:oasis:names:tc:SAML:2.0:assertion" ID="_23d2b89f-7e75-4dc8-b154-def8767a391c" Iss
```

```
%%%%% You can see that the issuer of the assertion was my Windows server
<Issuer>http://WinServer2016.sckiewer.lab/adfs/services/trust</Issuer>
<ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
<ds:SignedInfo>
<ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
<ds:SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"/>
<ds:Reference URI="#_23d2b89f-7e75-4dc8-b154-def8767a391c">
<ds:Transforms>
<ds:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
<ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
</ds:Transforms>
<ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
<ds:DigestValue>aYnlNK8NiHWHshYMggpeDsta2GyUKQI5MmRmx+gI374=</ds:DigestValue>
</ds:Reference>
</ds:SignedInfo>
<ds:SignatureValue>rvkc6QWoTCLDly8/MoRCzGcu0FJr6PSu5BTQt3qp5ua7J/AQbbzWn7gWK6TzI+xcH2478M2Smm5mIVVINXnG
<KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
<ds:X509Data>
<ds:X509Certificate>MIIC8DCCAdigAwIBAgIQQ2RhydxzTY1GQQ88eF3LWjANBgkqhkiG9w0BAQsFADA0MTIwMAYDVQQDEylBREZ
</ds:X509Data>
</KeyInfo>
</ds:Signature>
<Subject>

%%%%% The NameID Format is transient which is what CUCM expects
<NameID Format="urn:oasis:names:tc:SAML:2.0:nameid-format:transient" NameQualifier="http://WinServer201
<SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">

%%%%% You have an InResponseTo value that matches our SAML request, so you can correlate a given assert
<SubjectConfirmationData InResponseTo="s29fd87c888ef6a4bc8c48d7e7087a8aeb997dd76f" NotOnOrAfter="2021-0
</SubjectConfirmation>
</Subject>



%%%%% You can see here that this assertion is only to be considered valid from 13:01:03:891-14:01:03:89



<Conditions NotBefore="2021-04-30T13:01:03.891Z" NotOnOrAfter="2021-04-30T14:01:03.891Z">
<AudienceRestriction>
<Audience>1cucm1251.sckiewer.lab</Audience>
</AudienceRestriction>
</Conditions>



%%%%% AttributeStatement is a required section that provides the ID of the user (admin in this case) an
<AttributeStatement>
<Attribute Name="uid">
<AttributeValue>admin</AttributeValue>
</Attribute>
</AttributeStatement>
<AuthnStatement AuthnInstant="2021-04-30T13:01:03.844Z" SessionIndex="_23d2b89f-7e75-4dc8-b154-def8767a
<AuthnContext>
<AuthnContextClassRef>urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport</AuthnContextCl
</AuthnContext>
</AuthnStatement>
</Assertion> XML Representation
```

Helpful CLI Commands

- utils sso disable - This allows you to disable SSO if it is not functional
- utils sso status - This shows the current status of SSO on the node
- utils sso recovery-url enable - This allows you to disable the recovery URL
- utils sso recovery-url disable - This allows you to enable the recovery URL
- show samltrace level - This shows the current log level for SSO logs
- set samltrace level - This allows you to set the log level for SSO logs. This needs to be set to DEBUG in order for us to effectively troubleshoot issues.

Change from AssertionConsumerServiceURL to AssertionConsumerServiceIndex

When Cluster wide SSO was added in CUCM 11.5, CUCM no longer writes the AssertionConsumerService (ACS) URL in the SAML request. Instead, CUCM writes the AssertionConsumerServiceIndex. See these snippets from a SAML request:

CUCM pre 11.5.1:

```
AssertionConsumerServiceURL="https://1cucm1101.sckiewer.lab:443/ssosp/saml/SSO/alias/1cucm1101.sckiewer
```

CUCM 11.5.1 and up:

```
AssertionConsumerServiceIndex="0"
```

In 11.5 and higher, CUCM expects the IdP to use the ACS Index # from the request in order to look up the ACS URL from the metadata file that was uploaded during the configuration process. This CUCM metadata snippet shows the POST URL (of the Publisher) associated with index 0:

```
<md:AssertionConsumerService index="0" Location="https://cucm14.sckiewer.lab:8443/ssosp/saml/SSO/alias/
```

There is no workaround to change this behavior and the IdP must use the ACS Index values rather than the ACS URL. More information can be found here, Cisco bug ID CSCvc56596.

Common Issues

Unable to Access OS Administration or Disaster Recovery

In CUCM 12.x, the Cisco Unified OS Administration and Disaster Recovery System web applications utilize SSO. If log in attempts to these applications fail with a 403 error after you enable SSO, it is likely due to the CUCM platform being unable to find the user ID. This occurs because these applications do not reference the enduser table used by CM Administration, Serviceability, and Reporting. Because of this, the user ID that the IdP has authenticated does not exist on the CUCM platform side so CUCM returns a 403 Forbidden. This document details how to add the appropriate users into the system so that platform applications use SSO successfuly.

NTP Failure

SSO is time-sensitive because the IdP attaches a 'validity timeframe' to assertions. In order to verify whether the time is the issue in your case, you can look for this section in the SSO logs:

```
2021-04-30 09:01:04,090 DEBUG [http-bio-8443-exec-85] authentication.SAMLAuthenticator - Time Valid?:tr
2021-04-30 09:01:04,090 DEBUG [http-bio-8443-exec-85] authentication.SAMLAuthenticator - SAML Authentic
```

If you find **Time Valid?:false** in your SSO logs, investigate the Conditions section of the assertion to identify the timeframe that the assertion must be considered valid:

```
<Conditions NotBefore="2021-04-30T13:01:03.891Z" NotOnOrAfter="2021-04-30T14:01:03.891Z">
<AudienceRestriction>
<Audience>1cucm1251.sckiewer.lab</Audience>
</AudienceRestriction>
</Conditions>
```

You can see in the example snippet that this assertion is only valid from 13:01:03:8917 to 14:01:03:8917 on 4/30/2021. In a failure scenario, refer to the time that CUCM received this assertion and verify that it is within the validity period from the assertion. If the time that CUCM processed the assertion is outside the validity period, this is the cause of your issue. Ensure that CUCM and the IdP both sync to the same NTP server since SSO is very time-sensitive.

Invalid Attribute Statement

Refer to the analysis of the assertion here and see the note about the attribute statement. Cisco Unified Communications products require an attribute statement to be provided by the IdP, but sometimes the IdP does not send one. For reference, this is a valid AttributeStatement:

```
<AttributeStatement>
<Attribute Name="uid">
<AttributeValue>admin</AttributeValue>
</Attribute>
</AttributeStatement>
```

If you see an assertion from the IdP, but the attribute statement is omitted, work with the vendor of your IdP software to make the necessary changes so that it provides this statement. The fix differs based on the IdP and in some scenarios, more information can be sent in this statement than you see in the snippet. As long as there is an Attribute Name set to uid and an AttributeValue that matches a user with the correct privileges in the CUCM database, the log in is successful.

Two Signing Certificates - AD FS

This issue is specific to Microsoft AD FS. When the signing certificate on AD FS is close to expiration, the Windows Server automatically generates a new certificate, but leave the old certificate in place until it expires. When this occurs, the AD FS metadata contains two signing certificates. The error message you can see when you attempt to run the SSO test during this timeframe is, **Error while processing SAML response**.

Note:Error while processing SAML response can also be presented for other issues so do not assume that this is your issue if you see this error. Be sure to check the SSO logs to verify.

If you see this error, review the SSO logs and look for this:

```
2018-12-26 13:49:59,581 ERROR [http-bio-443-exec-45] authentication.SAMLAuthenticator - Error while pro
com.sun.identity.saml2.common.SAML2Exception: The signing certificate does not match what's defined in t
```

This error indicates that the IdP metadata imported into CUCM contains a signing certificate that does not match what the IdP used in this SAML exchange. This error usually occurs because AD FS has two signing certificates. When the original cert is close to expiration, AD FS automatically generates a new certificate. You must download a new metadata file from AD FS, verify that it only has one signing and encryption certificate and import it into CUCM. Other IdPs also have signing certificates that need to be updated so it is possible that someone updated it manually but simply has not imported the new metadata file that contains the new certificate to CUCM.

If you encounter the errors mentioned:

- If you use AD FS, refer to Cisco bug ID CSCuj66703
- If you do NOT use AD FS, collect a new metadata file from the IdP and import it into CUCM

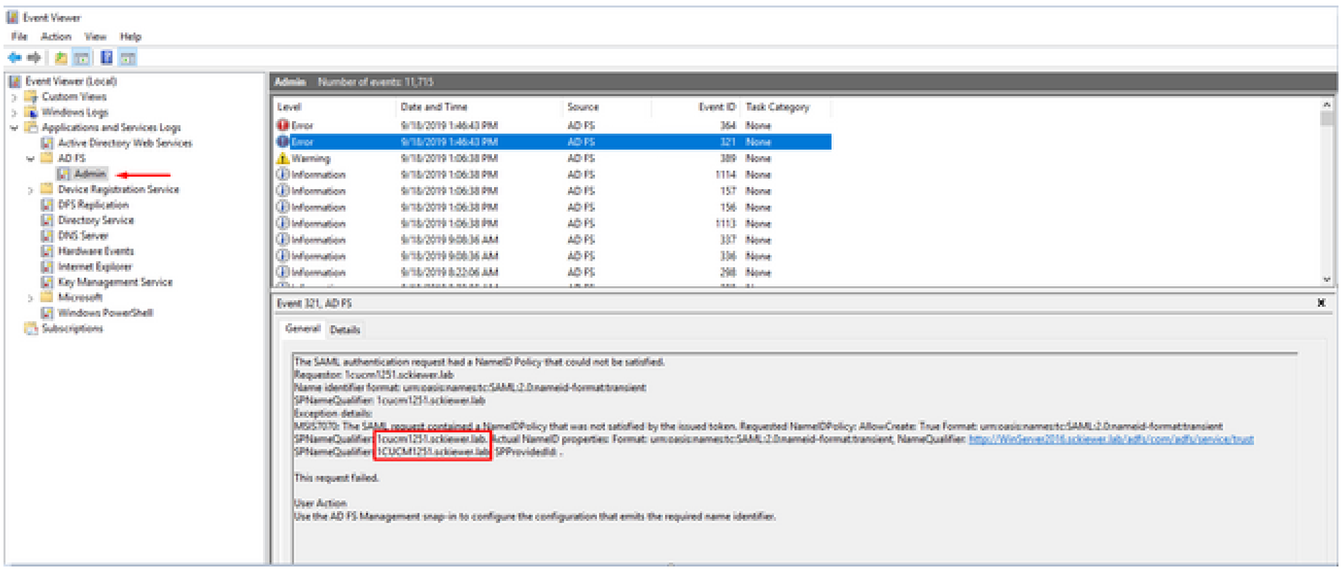
Invalid Status code in Response

This is a common error in deployments with AD FS:

```
Invalid Status code in Response. This may be caused by a configuration error in the IDP. Please check th
```

In almost all cases, this is an issue with the claim rule on the AD FS side. Paste the rule into notepad first, add your entityIDs, and then paste the rule from notepad into AD FS. In some scenarios, a copy/paste directly from your email or browser can omit some of the punctuation and cause a syntax error.

Another common issue is with the claim rule is that the capitalization of either the IdP or SP FQDNs does not match the entityID in the metadata files. Check the Event Viewer logs on the Windows Server to determine if this is your issue.



Admin Number of events: 11,715

Level	Date and Time	Source	Event ID	Task Category
Error	9/18/2019 1:46:43 PM	AD FS	364	None
Error	9/18/2019 1:46:43 PM	AD FS	321	None
Warning	9/18/2019 1:06:38 PM	AD FS	389	None
Information	9/18/2019 1:06:38 PM	AD FS	1114	None
Information	9/18/2019 1:06:38 PM	AD FS	157	None
Information	9/18/2019 1:06:38 PM	AD FS	156	None
Information	9/18/2019 1:06:38 PM	AD FS	1113	None
Information	9/18/2019 9:08:36 AM	AD FS	337	None
Information	9/18/2019 9:08:36 AM	AD FS	336	None
Information	9/18/2019 8:22:06 AM	AD FS	298	None

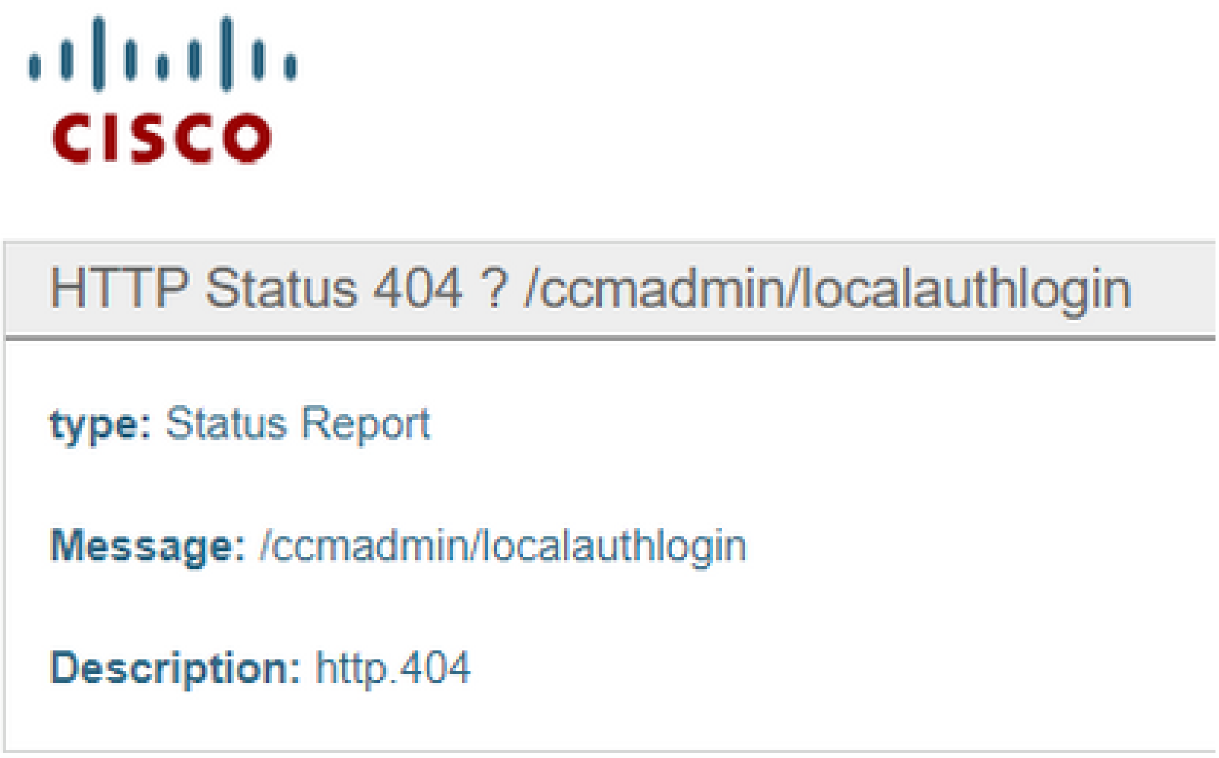
Event 321, AD FS

```
The SAML authentication request had a NameID Policy that could not be satisfied.
Requestor: 1cucm1251.sckiewer.lab
Name identifier format: urn:oasis:names:tc:SAML:2.0:nameid-format:transient
SPNameQualifier: 1cucm1251.sckiewer.lab
Exception details:
MSIS7070: The SAML request contained a NameIDPolicy that was not satisfied by the issued token. Requested NameIDPolicy: AllowCreate: True Format: urn:oasis:names:tc:SAML:2.0:nameid-format:transient
SPNameQualifier: 1cucm1251.sckiewer.lab. Actual NameID properties: Format: urn:oasis:names:tc:SAML:2.0:nameid-format:transient, NameQualifier: http://WinServer2016.sckiewer.lab/adfs/com/adfs/service/trust
SPNameQualifier: 1CUCM1251.sckiewer.lab SPProvidedId: .

This request failed.

User Action
Use the AD FS Management snap-in to configure the configuration that emits the required name identifier.
```

You can see in the image that the Requested NameID is 1cucm1251.sckiewer.lab while the Actual NameID is 1CUCM1251.sckiewer.lab. The Requested NameID must match the entityID in the SP metadata file while the Actual NameID is set in the claim rule. To fix this issue, I need to update the claim rule with a lower case FQDN for the SP.

SSO Status Mismatch Between CLI & GUI

In some cases, **utils sso status** and the GUI can show different information with regard to whether SSO is enabled or disabled. The easiest way to fix this is to disable and re-enable SSO. There are quite a few files and references that update through the enablement process, so it is not feasible to try to manually update all of those files. In most cases, you can log into the GUI and disable and re-enable with no issues, however, it is possible to see this error when you try to access the publisher via Recovery URL or the main link:



HTTP Status 404 ? /ccmadmin/localauthlogin

type: Status Report

Message: /ccmadmin/localauthlogin

Description: http.404

You can check the GUI to see if the recovery URL is an option and you can also check the **utils sso status** output from the CLI:

```
admin:utils sso status
SSO Status: SAML SSO Enabled
IdP Metadata Imported Date = Fri Apr 09 09:09:00 EDT 2021
SP Metadata Exported Date = Fri Apr 02 15:00:42 EDT 2021
SSO Test Result Date = Fri Apr 09 09:10:39 EDT 2021
SAML SSO Test Status = passed
Recovery URL Status = enabled
Entity ID = http://WinServer2016.sckiewer.lab/adfs/services/trust
```

Next, Check the process node table. In this example, you can see that SSO is disabled in the database (see the tkssomode value for 1cucm1251.sckiewer.lab on the far right):

```
admin:run sql select pkid,name,tkssomode from processnode
pkid                                    name                    tkssomode
====================================    ==================      =========
00000000-1111-0000-0000-000000000000    EnterpriseWideData         0
04bff76f-ba8c-456e-8e8f-5708ce321c20    1cucm1251.sckiewer.lab     0


admin:run sql select * from typessomode
enum name       moniker
==== ========== ===================
0    Disable    SSO_MODE_DISABLE
1    Agent Flow SSO_MODE_AGENT_FLOW
2    SAML       SSO_MODE_SAML
```

In order to fix this, set the tkssomode field on the process node table back to 2 so that you can log in via the recovery URL:

```
admin:run sql update processnode set tkssomode='2' where name ='1cucm1251.sckiewer.lab'
Rows: 1

admin:run sql select pkid,name,tkssomode from processnode
pkid                                    name                    tkssomode
====================================    ==================      =========
00000000-1111-0000-0000-000000000000    EnterpriseWideData         0
04bff76f-ba8c-456e-8e8f-5708ce321c20    1cucm1251.sckiewer.lab     2
```

At this point, test the Recovery URL and proceed with a **Disable** > **Re-enable of SSO** which triggers CUCM to update all of the references in the system.

Related Information

- **SAML SSO Deployment Guide for Cisco Unified Communications Applications, Release 12.5(1)**
- **Security Assertion Markup Language (SAML) V2.0 Technical Overview**
- **Technical Support & Documentation - Cisco Systems**