

# Using SAML SSO with OPS-COM

## What is Single Sign-On (SSO)

Single Sign-On, or SSO, is a way to make it easier for your users to access OperationsCommander using your existing managed accounts. Your users will not have to remember a separate username and password and will instead login directly to your corporate service provider.

## Configuring SAML Setup

Important - You must first follow the instructions to setup login sources found [here](#).

## Service Provider Fields

The **Unique Identifier** is part of the XML communication between OPSCOM and your SAML system. It is supplied by your SAML system and it is what OPS-COM uses to match against our UniqueID field.

The **Entity ID** for **Service Provider** defines the SAML integration path of the URL in the metadata. If there is more than one SAML integration in the system, each ID needs to be unique. The value supplied ends up in the path like this: "<https://client.ops-com.com/auth/saml2/> ENTITY\_ID\_FIELD /acs"

The **x509 certificate** can be generated and added to the service provider. You, the Identity Provider provide this.

## Identity Provider Fields

These fields come from the system you are working with, such as SAML, when communicating with OPS-COM. For example, SAML should display its metadata under **Federation → Show Metadata**

on the SAML installation page.

Once the settings have been completed and saved, you will have access to the **MetaData**, **Synchronization**, and **Translations** tabs.

Login Source: Saml 2.0

Back

Settings

Metadata

Synchronization

Translations

Name

Saml2 - Login

Login Source

saml2

Login Source coincides with the login\_source value used in the [User Push API](#). The value can be anything other than the reserved word OPSCOM.

This is the login source for 2 users. Changing it can prevent those users from logging in.

Unique ID Field

uid

SAML identity providers can differ so please enter the field you wish OPSCOM to refer to for the unique identifier.

Service Provider

Entity ID

TomahawkUTesting

# Metadata

The Metadata tab provides the XML that would be provided to the Service Provider.

## Login Source: Saml 2.0

[Back](#)[Settings](#)[Metadata](#)[Translations](#)

### Metadata URL

<https://tomahawku-test.preview.parkadmin.com/auth/saml2/TomahawkUTesting/metadata>

```
1 <?xml version="1.0"?>
2 <md:EntityDescriptor xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
3   validUntil="2020-05-22T13:14:11Z"
4   cacheDuration="PT604800S"
5   entityID="TomahawkUTesting">
6   <md:SPSSODescriptor AuthnRequestsSigned="false" WantAssertionsSigned="false" protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol">
7     <md:SingleLogoutService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect"
8       Location="https://tomahawku-test.preview.parkadmin.com/auth/saml2/TomahawkUTesting/sls" />
9     <md:NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-format:persistent</md:NameIDFormat>
10    <md:AssertionConsumerService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
11      Location="https://tomahawku-test.preview.parkadmin.com/auth/saml2/TomahawkUTesting/acs"
12      index="1" />
13  </md:SPSSODescriptor>
14  <md:ContactPerson contactType="support">
15    <md:GivenName>OPS-COM Support</md:GivenName>
16    <md:EmailAddress>support@ops-com.com</md:EmailAddress>
17  </md:ContactPerson>
18 </md:EntityDescriptor>
```

## Sample XML File

The following is an example of a response from an external system to OPS-COM. In this case, it is a SimpleSAMLPhp service set up as the identity provider. At the bottom, are several attributes within an `saml:AttributeStatement` tag. These are required for our system to match to a user within our system. The one field that matters in this attribute section is the value being used as the permanently-unique identifier for a user. In this case it is "uid".

Since "uid" is being sent back, then the setup for Identity Provider Fields should have "uid" as the Unique ID Field. If the unique ID is something else, such as SAMaccountName, then that should be used for the UniqueID.

```
... DEV-2K8 - DEBUG: Saml2 Incoming User Array ( [uid] => Array ( [0] => 6ddf4027-3397-4e45-8628-0189f60fe91e ) [full name] => Array ( [0] => Sarah Knowles ) [email] => Array ( [0] => sknowles@tomahawk.ca ) ) []
```

```

<? xml version = "1.0" ?>
< samlp:Response xmlns:samlp = "urn:oasis:names:tc:SAML:2.0:protocol" xmlns:saml =
"urn:oasis:names:tc:SAML:2.0:assertion" ID = "_aa1963115aa6490e728c7376f4c8849813bbb..." >
...
< saml:Assertion xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xs = "
http://www.w3.org/2001/XMLSchema" ID = "_9efd79bf6425983ee9176f3d33a99d1a9176180..." >
...
< saml:Subject >
< saml:NameID SPNameQualifier = "MinionOpsComStaff" Format = "urn:oasis:names:tc:SAML:2.0:nameid-
format:transient" >_7a426e0be71f14c1f349db00d7d543b6f7dcb52baa</ saml:NameID >
< saml:SubjectConfirmation Method = "urn:oasis:names:tc:SAML:2.0:cm:bearer" >
< saml:SubjectConfirmationData NotOnOrAfter = "2021-08-24T16:00:41Z" Recipient = "https://minion-
3.dev.parkadmin.com/auth/saml2/MinionOpsComStaff/acs" InResponseTo = "ONELOGIN_bb8a09203c888cf59af4c621a71cfa8f7559c016"
/>
</ saml:SubjectConfirmation >
</ saml:Subject >
< saml:Conditions NotBefore = "2021-08-24T15:55:11Z" NotOnOrAfter = "2021-08-24T16:00:41Z" >
< saml:AudienceRestriction >
< saml:Audience >MinionOpsComStaff</ saml:Audience >
</ saml:AudienceRestriction >
</ saml:Conditions >
< saml:AuthnStatement AuthnInstant = "2021-08-24T15:34:46Z" SessionNotOnOrAfter = "2021-08-24T23:34:46Z"
SessionIndex = "_a7a68666092117d24aab8adecf1b0830622855b85..." >
< saml:AuthnContext >
< saml:AuthnContextClassRef >urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport</
saml:AuthnContextClassRef >
</ saml:AuthnContext >
</ saml:AuthnStatement >

< saml:AttributeStatement >
< saml:Attribute Name = "uid" NameFormat = "urn:oasis:names:tc:SAML:2.0:attrname-format:basic" >
< saml:AttributeValue xsi:type = "xs:string" >6ddf4027-3397-4e45-8628-0189f60fe91e</ saml:AttributeValue >
</ saml:Attribute >
< saml:Attribute Name = "full name" NameFormat = "urn:oasis:names:tc:SAML:2.0:attrname-format:basic" >
< saml:AttributeValue xsi:type = "xs:string" >Sarah Knowles</ saml:AttributeValue >
</ saml:Attribute >
< saml:Attribute Name = "email" NameFormat = "urn:oasis:names:tc:SAML:2.0:attrname-format:basic" >
< saml:AttributeValue xsi:type = "xs:string" >sknowles@tomahawk.ca</ saml:AttributeValue >
</ saml:Attribute >
</ saml:AttributeStatement >

</ saml:Assertion >
</ samlp:Response >

```

# Translations

Translations can be used to change the text displayed on your login button from the user side. We can create as many different translations as we have available on our system.

For this example, we have English and French.

Login Source: Saml 2.0

Back

Settings

Metadata

Translations

Token	Text	Language	Translation
login_with_button	Login With Button	English (en)	<input type="text" value="Login With SAML"/>
		Français (fr_ca)	<input type="text" value="Connectez-vous avec SAML"/>

Save Changes

Reset

# Synchronization

The synchronization tab allows you to create users in OPS-COM when they login from SAML if they do not already exist by mapping your user attributes to our system. This also lets you update existing users information in the system.

In this example, any field that is mapped and has a value from your SSO side should get updated to the value from SAML.

To begin, ensure that you enable **Auto Create/Update User**. Keep in mind that these are sample values from our test system, and that your SAML system may differ.

✔ Auto Create/Update User

Field Mapping

Map the attributes from the Identity Provider to this service provider. In the example below, address\_city and first\_name are attributes supplied by the Identity Provider. OPS-COM will know to map that to the internal field name.

```
<saml:AttributeStatement>
  <saml:Attribute Name="first_name" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic">
    <saml:AttributeValue xsi:type="xs:string">John</saml:AttributeValue>
  </saml:Attribute>
  <saml:Attribute Name="last_name" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic">
    <saml:AttributeValue xsi:type="xs:string">Smith</saml:AttributeValue>
  </saml:Attribute>
  <saml:Attribute Name="address_city" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic">
    <saml:AttributeValue xsi:type="xs:string">Borden</saml:AttributeValue>
  </saml:Attribute>
  ...
</saml:AttributeStatement>
```

Enabled enabled

User Type user\_type

First Name full name

Middle Name

Last Name

Username username

Email Address email

Address address\_street

City address\_city

Province address\_province

Postal Code address\_postal

Phone Number phone

profile.student\_number student\_number

Employee Number employee\_number

Employer employer

Building building

Supervisor Name supervisor\_name

Supervisor Title supervisor\_title

Save Changes

Reset

After you have supplied the information in each field, you can click **Save Changes** and your users will begin to be created/updated.

If any of the supplied fields are incorrect, then the information will be blank when the user logs in, or it will be unchanged if the user already existed.

---

Revision #4

Created 21 May 2024 11:20:41

Updated 5 May 2025 09:49:14 by Shannon Jones