

# Setting up a cross-account federation between Amazon Connect and Azure AD using AWS SSO.

26 November 2021 - 6 min. read



In these times, delivering support using multiple communication channels is crucial for a business.

There are already big players in the contact center business, but Amazon Connect is an interesting option to evaluate. It's fully managed, easily scalable, and with an aggressive pricing placement on the market.

Embedded Artificial Intelligence and Machine Learning enable the business to perform sentiment analysis and gain valuable insights.

Every customer has different needs, sometimes leading to unexplored paths in system integration.

In this article, we'll describe a cross-account federation between Amazon Connect, and Azure AD using AWS SSO.

## Scenario Overview

A customer wanted to configure an Amazon Connect instance federating their existing users on Office365 (using the underlying Azure Active Directory service) to keep centralized user management. Another requisite was to have a separate AWS account for the service, to let only some administrators manage services on the account.

Amazon Connect can connect to Active Directory and use it for identity management, but you'll need to use AWS Directory Service. Sometimes it's better to leverage existing Identity providers.

AWS SSO was our service of choice because of the flexibility it offers in configuring SAML applications and account access. As a bonus, we also were able to grant different access levels to multiple AWS accounts using single-sign-on.

As you'll see in this article Amazon Connect doesn't have a native integration with AWS SSO, so we need to configure an application and use it as an identity provider in the destination account.

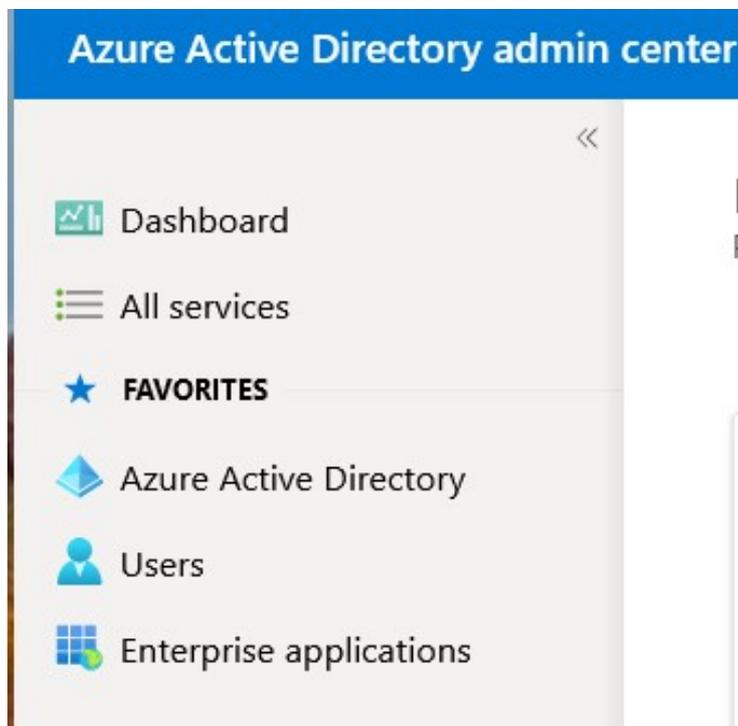
In this article, we will:

- [Configure AWS SSO on the Organization master account to trust Azure Active Directory \(used by Office365\) to authenticate users](#)
- [Configure an Amazon Connect instance with SAML authentication into a different account belonging to the AWS organization \(internal-services\)](#)
- [Create and configure an AWS SSO SAML application for Amazon Connect](#)
- [Configure an Identity Provider on the internal-services account to trust the SAML application for cross-account authentication](#)
- [Add the required roles in the internal-service account to authenticate federated users](#)
- [Test the configuration](#)

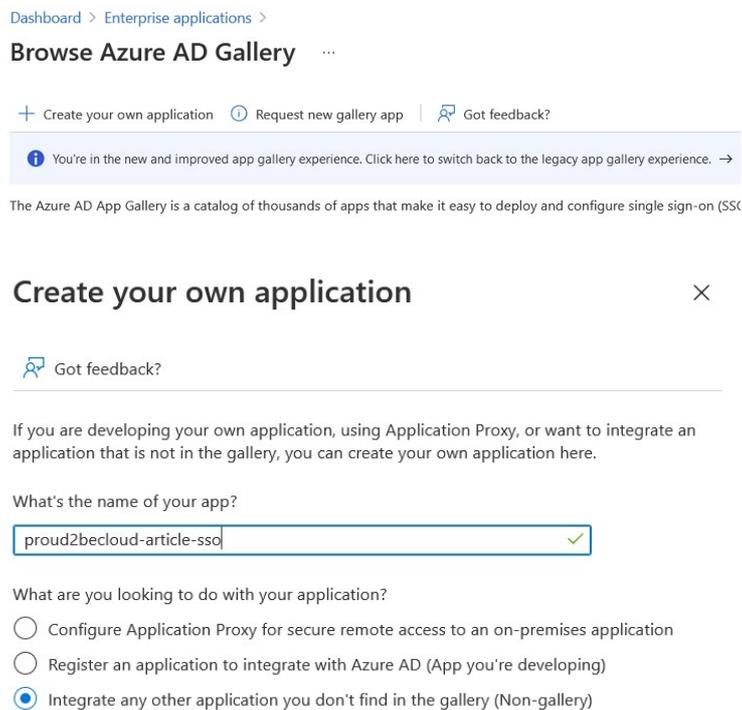
## **AWS SSO Setup**

Log into the [Azure Active Directory admin center](#)

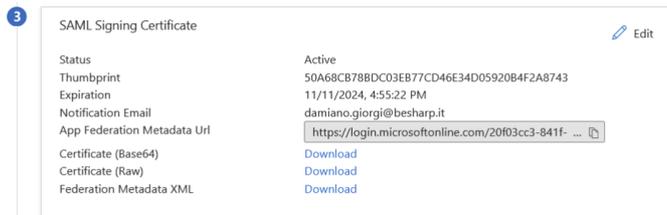
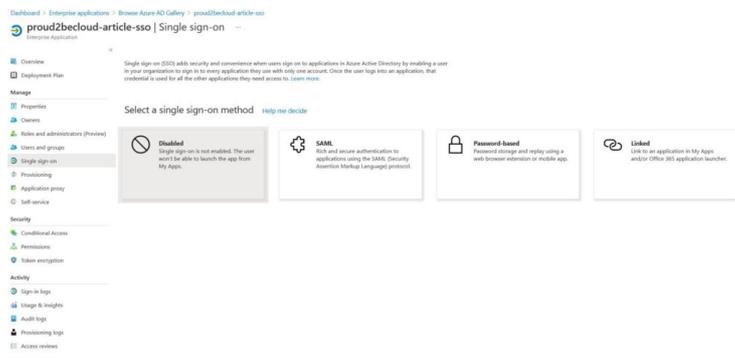
Select "Enterprise Applications"



Select “Create your own application” and give it a unique name

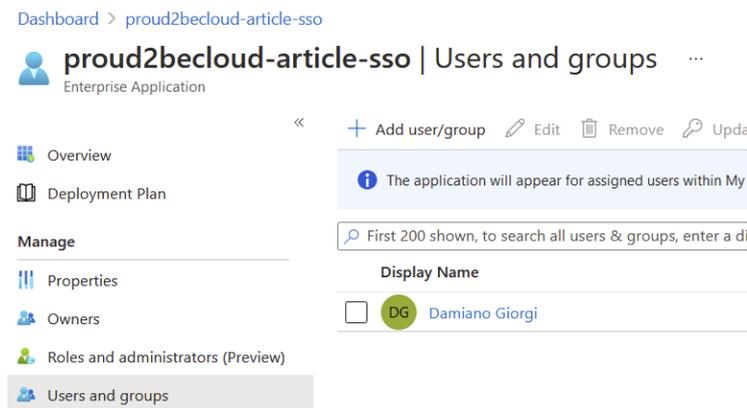


After a little while the application will be ready, we need to set-up **Single sign-on**, click on the menu and then select “**SAML**”

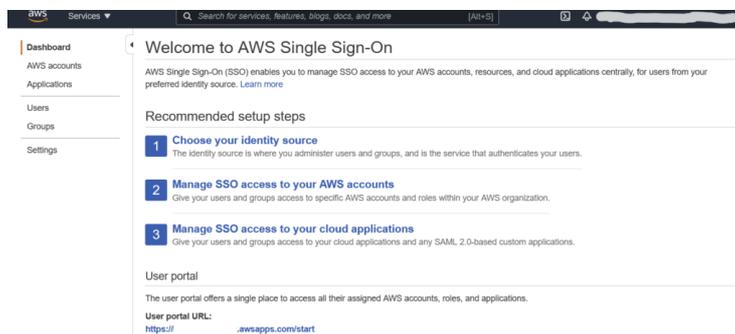


Click on the Download link for “**Federation Metadata XML**” and store the file in a secure place, don’t share this file with anyone !

Assign users to the SSO application to enable them



Log in into the **AWS Organizations management** account for and select “**AWS Single Sign On**”



If you previously configured AWS SSO you can change your identity source or configure a new one on the “Settings” page.

**Identity source**

Your identity source is where you administer your users and groups, and where AWS SSO authenticates your users. You can choose between AWS SSO, SAML 2.0-compatible identity provider (IdP), or Active Directory (AD). [Learn more](#)

<b>Identity source</b>	AWS SSO   <a href="#">Change</a>
<b>Authentication</b>	AWS SSO
<b>Provisioning</b>	AWS SSO
<b>Identity store ID</b>	d-93670984fa <a href="#">🔗</a>
<b>Attributes for access control</b>	Disabled   <a href="#">Enable</a>

## Change identity source



Choose identity source    Review

### Choose where your identities are sourced

Your identity source is the place where you administer and authenticate identities. You use AWS SSO to manage permissions for identities from your identity source to access AWS accounts, roles, and applications. [Learn more](#)

- AWS SSO**  
You will administer all users, groups, credentials, and multi-factor authentication assignments in AWS SSO. Users sign in through the AWS SSO user portal.
- Active Directory**  
You will administer all users, groups, and credentials in AWS Managed Microsoft AD, or you can connect AWS SSO to your existing Active Directory using AWS Managed Microsoft AD or AD Connector. Users sign in through the AWS user portal.
- External identity provider**  
You will administer all users, groups, credentials, and multi-factor authentication in an external identity provider (IdP). Users sign in through your IdP sign-in page to access the AWS SSO user portal, assigned accounts, roles, and applications.

### Configure external identity provider

AWS SSO works as a SAML 2.0 compliant service provider to your external identity provider (IdP). To configure your IdP as your AWS SSO identity source, you must establish a SAML trust relationship by exchanging meta data between your IdP and AWS SSO. While AWS SSO will use your IdP to authenticate users, the users must first be provisioned into AWS SSO before you can assign permissions to AWS accounts and resources. You can either provision users manually from the Users page, or by using the automatic provisioning option in the Settings page after you complete this wizard. [Learn more](#)

#### Service provider metadata

Your identity provider (IdP) requires the following AWS SSO certificate and metadata details to trust AWS SSO as a service provider. You may copy and paste, or type this information into your IdP's service provider configuration interface, or you may download the AWS SSO metadata file and upload it into your IdP.

[AWS SSO SAML metadata](#)    [Download metadata file](#)

[Show individual metadata values](#)

#### Identity provider metadata

AWS requires specific metadata provided by your identity provider (IdP) to establish trust. You may copy and paste from your IdP, type the metadata in manually, or upload a metadata exchange file that you download from your IdP.

**IdP SAML metadata\***    proud2becloud-article-ss0.xml    [Browse...](#)

If you don't have a metadata file, you can manually type your metadata values

Select “**External identity provider**”, download the metadata file and, as previously, store it in a secure place. Upload the metadata file you downloaded from the Azure SAML application page

Go back to the Azure Active Directory Administration Console and click on “**upload metadata file**”

Dashboard > proud2becloud-article-ss0 >

**proud2becloud-article-ss0 | SAML-based Sign-on** ...

Enterprise Application

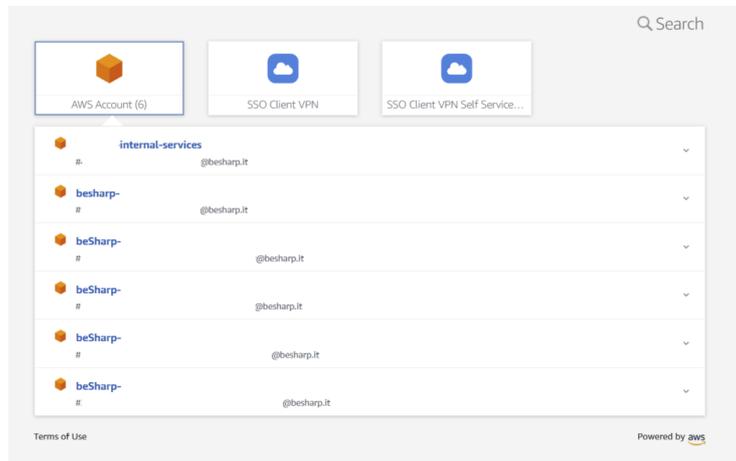
«    [Upload metadata file](#)    [Change single sign-on mode](#) ...

Select the file downloaded from the AWS Console and proceed.

We just exchanged the required configuration information to federate Azure Active Directory users with AWS SSO.

Back on the Azure Console you can try the application, you should now be able to login with your current Azure Active Directory credentials.

If you assign AWS accounts and different applications to your users you should be able to see them



You can also enable **auto-provisioning** for users and enable selected groups to access your AWS accounts.

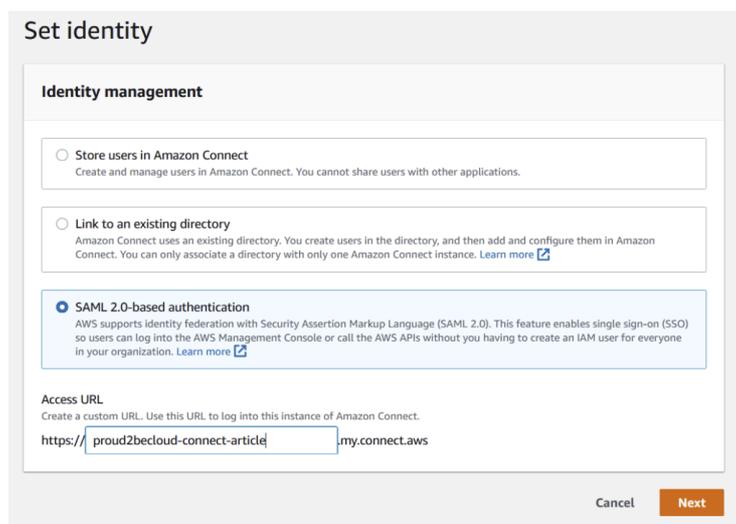
## Amazon Connect Setup

We'll use a different AWS account (**internal-services**) to configure Amazon Connect. With SSO and Organizations we can enable fine-grained access to different teams and isolate duties.

On the internal-service account search for “Amazon Connect” and click on “Create new instance”.

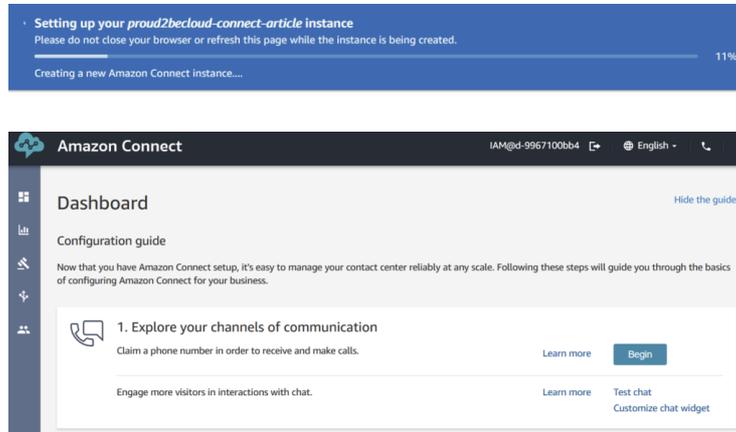
Select “**SAML 2.0-based authentication**” for identity management and assign a name to your instance.

Once you set an authentication mechanism in Amazon Connect you can't change it.



Continue with the configuration wizard steps with your preferences.

In a matter of minutes, your amazon connect instance should be ready.



**Please note:** since Connect doesn't support user auto-provisioning you'll need to create a user with the same username you defined in Azure Active Directory

## SSO Integration Setup

On the management account go back on the AWS SSO Console click on "Applications" and "Add a new Application", search for "**Amazon Connect**"

Configure Proud2beCloud Amazon Connect

AWS SSO works as an identity provider (IdP) for any SAML 2.0-compliant cloud applications. To configure this application for SSO access, you must establish a trust relationship between AWS SSO and your cloud application (service provider) through a SAML metadata exchange. You can view instructions on this page and find metadata details for your provider. [View instructions](#)

Details

Display name\* Proud2beCloud Amazon Connect ⓘ

Description Proud2beCloud Amazon Connect Application ⓘ

The description you type here does not appear in the user portal. However, it will be visible in the AWS SSO console and when using the AWS SSO APIs.

AWS SSO metadata

Your cloud application may require the following certificate and metadata details to recognize AWS SSO as the identity provider.

AWS SSO SAML metadata file	<a href="https://portal.sso.eu-west-1.amazonaws.com/san">https://portal.sso.eu-west-1.amazonaws.com/san</a>	Copy URL	Download
AWS SSO sign-in URL	<a href="https://portal.sso.eu-west-1.amazonaws.com/san">https://portal.sso.eu-west-1.amazonaws.com/san</a>	Copy URL	
AWS SSO sign-out URL	<a href="https://portal.sso.eu-west-1.amazonaws.com/san">https://portal.sso.eu-west-1.amazonaws.com/san</a>	Copy URL	
AWS SSO issuer URL	<a href="https://portal.sso.eu-west-1.amazonaws.com/san">https://portal.sso.eu-west-1.amazonaws.com/san</a>	Copy URL	
AWS SSO certificate	Download certificate		

Application properties

Your cloud application may optionally take additional settings to configure your user experience. [Learn more](#)

Application start URL ⓘ

Relay state ⓘ

Session duration\* 1 hour ▾

Application metadata

AWS SSO requires specific metadata about your cloud application before it can trust this application. You can type this metadata manually or upload a metadata exchange file.

Application ACS URL\* <https://signin.aws.amazon.com/saml> ⓘ

Application SAML audience\* urn:amazon:webservices

If you have a metadata file, you can upload it now instead.

\* Required fields

Cancel Save changes

Give your application a name and, once again, download the metadata file.

On the **internal-services** account, go to IAM and go in the "**Identity Providers**" section, click on "Add provider" and upload the metadata file

## Add an Identity provider

## Configure provider

## Provider type

 SAML

Establish trust between your AWS account and a SAML 2.0 compatible Identity Provider such as Shibboleth or Active Directory Federation Services.

 OpenID Connect

Establish trust between your AWS account and Identity Provider services, such as Google or Salesforce.

## Provider name

Enter a meaningful name to identify this provider

Proud2beCloudAmazonConnect

Maximum 128 characters. Use alphanumeric or '\_' characters.

## Metadata document

This document is issued by your IdP.

Choose file

File needs to be a valid UTF-8 XML document.

Amazon Connect\_ins-2393b1763b136ed4.xml

## Add tags (Optional)

Tags are key-value pairs that you can add to AWS resources to help identify, organize, or search for resources.

No tags associated with the resource.

Add tag

You can add up to 50 more tags

Cancel

Add provider

## Setup roles

Once the identity provider has been created we'll need to setup the necessary roles and policies to let SSO users access the service

On the internal-services account go to IAM -> Roles and create a new Role. Select **"SAML 2.0 federation"** for the type of trusted identity and choose the identity provider you have just created.

## Create role

1 2 3 4

## Select type of trusted entity

 <b>AWS service</b> EC2, Lambda and others	 <b>Another AWS account</b> Belonging to you or 3rd party	 <b>Web identity</b> Cognito or any OpenID provider	 <b>SAML 2.0 federation</b> Your corporate directory
--	---	---	---

Allows users that are federated with SAML 2.0 to assume this role to perform actions in your account. [Learn more](#)

## Choose a SAML 2.0 provider

If you're creating a role for API access, choose an Attribute and then type a Value to include in the role. This restricts access to users with the specified attributes.

SAML provider: Proud2beCloudAmazonConnect

[Create new provider](#) [Refresh](#)

Allow programmatic access only

Allow programmatic and AWS Management Console access

Attribute: SAML:aud

Value\*: https://signin.aws.amazon.com/saml

Condition: [Add condition \(optional\)](#)

Create a new policy to let the role get a "Federation Token" from the Amazon Connect instance, use this json template:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {

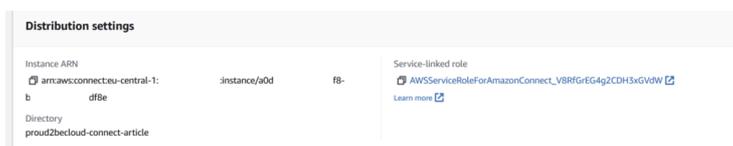
```

```

        "Sid": "Statement1",
        "Effect": "Allow",
        "Action": "connect:GetFederationToken",
        "Resource": [
            "arn:aws:connect:region:Account-id:instance/amazonconnectinstanceid/user/${aws:userid}"
        ]
    }
]
}

```

You can find the instance id by clicking on the connect instance and copying the last part of the **“Instance ARN”** field, use the internal-services accountid and the connect region for the remaining fields:



Once the appropriate role and policy have been created we can go back to the AWS SSO Console on the management account to modify the Connect application to finish the configuration.

Edit the configuration, leave the **“Application start URL”** field blank, for **“Relay state”** use:

<https://region.console.aws.amazon.com/connect/federate/amazonconnectid>

### Configure Proud2beCloud Amazon Connect

Details

**Display name\*** Proud2beCloud Amazon Connect ⓘ

**Description** Proud2beCloud Amazon Connect Application

The description you type here does not appear in the user portal. However, it will be visible in the AWS SSO console and when using the AWS SSO APIs.

**AWS SSO metadata**

Your cloud application may require the following certificate and metadata details to recognize AWS SSO as the identity provider.

**AWS SSO SAML metadata file** <https://portal.sso.eu-west-1.amazonaws.com/saml> [Copy URL](#) [Download](#)

**AWS SSO sign-in URL** <https://portal.sso.eu-west-1.amazonaws.com/saml> [Copy URL](#)

**AWS SSO sign-out URL** <https://portal.sso.eu-west-1.amazonaws.com/saml> [Copy URL](#)

**AWS SSO issuer URL** <https://portal.sso.eu-west-1.amazonaws.com/saml> [Copy URL](#)

**AWS SSO certificate** [Download certificate](#)

**Application properties**

Your cloud application may optionally take additional settings to configure your user experience. [Learn more](#)

**Application start URL** ⓘ

**Relay state** <https://eu-central-1.console.aws.amazon.com/cor>

**Session duration\*** 12 hours

**Application metadata**

AWS SSO requires specific metadata about your cloud application before it can trust this application. You can type this metadata manually or upload a metadata exchange file.

**Application ACS URL\*** <https://signin.aws.amazon.com/saml> ⓘ

**Application SAML audience\*** urn:amazon:webservices

If you have a metadata file, you can upload it now instead.

\* Required fields [Cancel](#) [Save changes](#)

Go to **“Attribute Mappings”** and add a new mapping:

Set <https://aws.amazon.com/SAML/Attributes/Role> as **“User attribute in the application”** field and **arn:aws:iam::**internal-services-account-id**:saml-provider/**saml-provider-name**,arn:aws:iam::**internal-services-account-id**:role/**amazon-connect-federation-role**** for the **“Maps to this string value or user attribute in AWS SSO”** field.

### Proud2beCloud Amazon Connect

Configuration for Proud2beCloud Amazon Connect has been saved. You can now review attribute mappings for this application.

Configuration | **Attribute mappings** | Assigned users

SAML assertions successfully updated.

Attributes you map here become part of the SAML assertion that is sent to the application. You can choose which user attributes in your application map to corresponding user attributes in your connected directory. [Learn more](#)

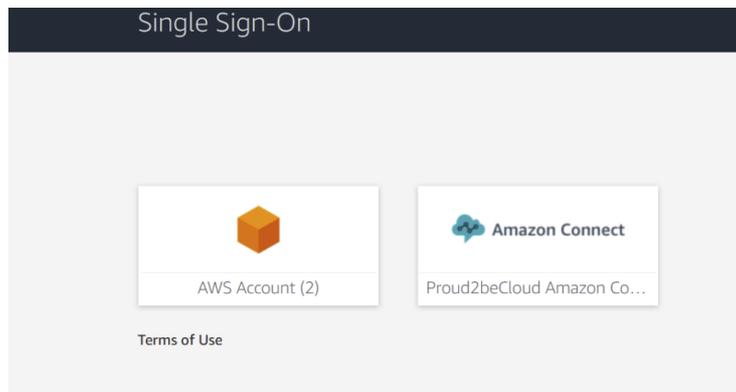
User attribute in the application	Maps to this string value or user attribute in AWS SSO	Format
Subject	\$(user.email)	persistent
<a href="https://aws.amazon.com/SAMLUJA">https://aws.amazon.com/SAMLUJA</a>	\$(user.email)	unspecified
<a href="https://aws.amazon.com/SAMLUJA">https://aws.amazon.com/SAMLUJA</a>	arn:aws:iam:::saml-provider:Proud2beCloudAmazonConn	unspecified

[Add new attribute mapping](#) [Save changes](#)

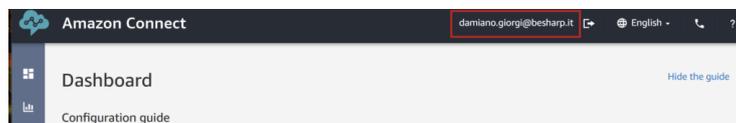
Save changes and on the **“Assigned users”** tab assign users to the Proud2beCloud Amazon Connect application

## Testing

Go to your SSO start url (usually <https://youdefinedname.awsapps.com/start/>) and login with your Azure AD/Office365 credentials, you'll find the Amazon Connect Application.



Clicking on it will take you to the Amazon Connect dashboard with your credentials.



That's it! Your Amazon Connect instance now uses Azure AD to validate user credentials

A cross-account scenario isn't always easy to troubleshoot but with this tutorial, you should be able to get it running in a couple of minutes

Now you can configure Amazon Connect, if you are not able to access your instance using SSO check the IDP provider, configured role, policies, and mappings.

Have you faced any particular use case? Feel free to leave your thoughts in the comments!

And see you again in 14 days on Proud2beCloud!



## Damiano Giorgi

Ex on-prem systems engineer, lazy and prone to automating boring tasks. In constant search of technological innovations and new exciting things to experience. And that's why I love Cloud Computing! At this moment, the only "hardware" I regularly dedicate myself to is

that my bass; if you can't find me in the office or in the band room try at the pub or at some airport, then!

---

Copyright © 2011-2021 by beSharp srl - P.IVA IT02415160189