AWS Single Sign-On

User Guide

AWS Single Sign-On: User Guide

Table of Contents

What Is AWS Single Sign-On?

AWS Single Sign-On is a cloud-based single sign-on (SSO) service that makes it easy to centrally manage SSO access to all of your AWS accounts and cloud applications. Specifically, it helps you manage SSO access and user permissions across all your AWS accounts in AWS Organizations. AWS SSO also helps you manage access and permissions to commonly used third-party software as a service (SaaS) applications as well as custom applications that support Security Assertion Markup Language (SAML) 2.0. AWS SSO includes a user portal where your end-users can find and access all their assigned AWS accounts, cloud applications, and custom applications in one place.

AWS SSO Features

AWS SSO provides the following features:

Integration with AWS Organizations

AWS SSO is integrated deeply with AWS Organizations and AWS API operations, unlike other cloud native SSO solutions. AWS SSO natively integrates with AWS Organizations and enumerates all your AWS accounts. If you have organized your accounts under organizational units (OUs) you will see them displayed that way within the AWS SSO console. That way you can quickly discover your AWS accounts, deploy common sets of permissions, and manage access from a central location.

SSO access to your AWS accounts and cloud applications

AWS SSO makes it simple for you to manage SSO across all your AWS accounts, cloud applications, and custom SAML 2.0–based applications. without custom scripts or third-party SSO solutions. Use the AWS SSO console to quickly assign which users should have one-click access to only the applications that you've authorized for their personalized end-user portal.

Create and manage users and groups in AWS SSO

When you enable the service for the first time, we create a default directory for you in AWS SSO. You can use this directory to manage your users and groups directly in the console. Or, if you prefer, you can connect to an existing AWS Managed Microsoft AD directory and manage your users with standard Active Directory management tools provided in Windows Server. If you choose to manage your users in AWS SSO, you can quickly create users and then easily organize them into groups, all within the console.

Leverage your existing corporate identities

AWS SSO is integrated with Microsoft AD through the AWS Directory Service. That means your employees can sign in to your AWS SSO user portal using their corporate Active Directory credentials. To grant Active Directory users access to accounts and applications, you simply add them to the appropriate Active Directory groups. For example, you can grant the DevOps group SSO access to your production AWS accounts. Users added to the DevOps group are then granted SSO access to these AWS accounts automatically. This automation makes it easy to onboard new users and give existing users access to new accounts and applications quickly.

Compatible with commonly used cloud applications

AWS SSO supports commonly used cloud applications such as Salesforce, Box, and Office 365. This cuts the time needed to set up these applications for SSO by providing application integration instructions. These instructions act as guard rails to help administrators set up and troubleshoot these SSO configurations. This eliminates the need for administrators to learn the configuration nuances of each cloud application.

Easy to set up and monitor usage

With AWS SSO, you can enable a highly available SSO service with just a few clicks. There is no additional infrastructure to deploy or AWS account to set up. AWS SSO is a highly available and a completely secure infrastructure that scales to your needs and does not require software or hardware to manage. AWS SSO records all sign-in activity in AWS CloudTrail, giving you the visibility to monitor and audit SSO activity in one place.

Getting Started

In this getting started exercise, you enable AWS Single Sign-On, connect your directory, set up SSO to your AWS accounts, and finally set up SSO to your cloud applications. Although not required, we recommend that you review Understanding Key AWS Single Sign-On Concepts (p. 5) before you begin using the console so that you are familiar with the core features and terminology.

Topics

AWS SSO Prerequisites

Before you can set up AWS SSO, you must:

- Have first set up the AWS Organizations service and have **All features** set to enabled. For more information about this setting, see Enabling All Features in Your Organization in the *AWS Organizations User Guide*.
- Sign in with the AWS Organizations master account credentials before you begin setting up AWS SSO. These credentials are required to enable AWS SSO. For more information, see Creating and Managing an AWS Organization in the *AWS Organizations User Guide*. You cannot set up AWS SSO while signed in with credentials from an Organization’s member account.
- Have chosen a directory store to determine which pool of users has SSO access to the user portal. If you choose to use the default AWS SSO directory for your user store, no prerequisite tasks are required. The AWS SSO directory is created by default once you enable AWS SSO and is immediately ready for use. There is no cost for using this directory type. If you choose to connect to an existing Active Directory for your user store, you must have:
  - An existing AWS Managed Microsoft AD directory set up in AWS Directory Service, and it must reside within your organization's master account. You can connect only one AWS Managed Microsoft AD directory at a time. However, you can change it to a different AWS Managed Microsoft AD directory or change it back to an AWS SSO directory at any time. For more information, see Create a AWS Managed Microsoft AD Directory in the *AWS Directory Service Administration Guide*.
  - You must setup AWS SSO in the region where your AWS Managed Microsoft AD directory is set up. AWS SSO stores the assignment data in the same region as the directory. To administer AWS SSO, you should switch to the region where you have setup AWS SSO. Also, note that AWS SSO’s user portal uses the same access URL as your connected directory.

Enable AWS SSO

When you open the AWS SSO console for the first time, you are prompted to enable AWS SSO before you can start managing it. If you have already chosen this option, you can skip this step. If not, use the procedure below to enable it now. Once enabled, AWS SSO is granted the necessary permissions to create IAM service-linked roles in any of the AWS accounts within your AWS organization. No service-linked roles are created at this time. AWS SSO creates these roles later during the process of setting up SSO access to your AWS accounts (see Set Up SSO to Your AWS Accounts (p. 4)).

To enable AWS SSO

1. Sign in to the AWS Management Console with your AWS Organizations master account credentials.
2. Open the AWS SSO console.
3. Choose **Enable AWS SSO**.
4. If you have not yet set up AWS Organizations, you will be prompted to create an organization. Choose **Create AWS organization** to complete this process.

Choose Your Directory

Choosing a directory determines where AWS SSO looks for users and groups that need SSO access. By default, you get an AWS SSO directory for quick and easy user management. Optionally, you can also connect an AWS Managed Microsoft AD directory with your on-premises Active Directory.

AWS SSO provides users in this directory with a personalized user portal from which they can easily launch multiple AWS accounts or cloud applications. Users sign in to the portal using their corporate credentials or with credentials they set up in AWS SSO. Once they sign in, they have one-click access to all applications and AWS accounts that you have previously authorized.

Depending on which directory type you are trying to set up, review the topics below for guidance:

- Manage Your AWS SSO Directory (p. 6)
- Connect to Your Microsoft AD Directory (p. 9)

For more information about supported directory types, see Manage Your Directory (p. 6).

Set Up SSO to Your AWS Accounts

In this step, you can grant users in your directory with SSO access to one or more AWS consoles for specific AWS accounts in your AWS organization. Afterward, users see only the AWS account icon (for example, Development) that they've been assigned from within their user portal. When they click the icon, they can then choose which IAM role they want to use when signing in to the AWS Management Console for that AWS account.

To get started assigning SSO access to your AWS accounts, see Assign User Access (p. 15).

Set Up SSO to Your Cloud Applications

Depending on which application type you are trying to set up, follow one of the procedures below:

- Add and Configure a Cloud Application (p. 23)
- Add and Configure a Custom SAML 2.0 Application (p. 24)

For more information about supported application types, see Manage SSO to Your Applications (p. 21).

Once you complete the appropriate procedure, you will have successfully configured AWS SSO and set up a trust with your service provider. Your users can now access these applications from within their user portal based on the permissions you assigned.

Understanding Key AWS Single Sign-On Concepts

You'll get more out of AWS Single Sign-On if you become familiar with key concepts relating to SAML federation, user authentication, and IAM permissions.

Topics

SAML Federation

AWS SSO supports identity federation with SAML (Security Assertion Markup Language) 2.0. SAML 2.0 is an industry standard used for securely exchanging SAML assertions that pass information about a user between a SAML authority (called an identity provider or IdP), and a SAML consumer (called a service provider or SP). AWS SSO service uses this information to provide federated single sign-on (SSO) for those users who are authorized to use applications within the AWS SSO user portal.

AWS SSO adds SAML IdP capabilities to either your AWS Managed Microsoft AD or AWS SSO directory. Users can then SSO into services that support SAML, including the AWS Management Console and third-party applications such as Office 365, Concur, and Salesforce. At this time, AWS SSO does not support other directory types or IdPs.

User Authentications

When a user signs in to the user portal using their user name, AWS SSO redirects the request to the AWS SSO authentication service based on the directory associated with the user email address. Once authenticated, users have SSO access to any of the AWS accounts and third-party software-as-a-service (SaaS) applications that show up in the portal without additional sign-in prompts. This means that users no longer need to keep track of multiple account credentials for the various assigned AWS applications that they use on a daily basis.

Permission Sets

A permission set is a collection of administrator-defined policies that AWS SSO uses to determine a user's effective permissions to access a given AWS account. Permission sets can contain either AWS managed policies or custom policies that are stored in AWS SSO. Policies are essentially documents that act as containers for one or more permission statements. These statements represent individual access controls (allow or deny) for various tasks that determine what tasks users can or cannot perform within the AWS account.

Permission sets are stored in AWS SSO and are only used for AWS accounts. They are not used to manage access to cloud applications. Permission sets ultimately get created as IAM roles in a given AWS account, with trust policies that allow users to assume the role through AWS SSO.

Manage Your Directory

You can configure your directory in AWS SSO to determine where your users and groups are stored. Once configured, you can then look up users or groups in your directory to grant them single sign-on access to AWS accounts, cloud applications, or both.

AWS SSO automatically provides you with a directory by default, which you can use to manage your users and groups within AWS SSO. If you choose to store them in AWS SSO, create your users and groups and assign their level of access to your AWS accounts and applications. Alternatively, you can choose to Connect AWS SSO to an On-Premises Active Directory (p. 9) or Connect AWS SSO to an AWS Managed Microsoft AD Directory (p. 9) using AWS Directory Service.

> **Note**
> AWS SSO does not support SAMBA4-based Simple AD as a connected directory.

Topics

Manage Your AWS SSO Directory

AWS Single Sign-On provides you with a default directory where you can store your users and groups. If you choose to store them in AWS SSO, all you need to do is the following:

1. Create your users and groups.
2. Add your users as members to the groups.
3. Assign the groups with the desired level of access to your AWS accounts and applications.

> **Note**
> Users and groups that you create in your AWS SSO directory are available in AWS SSO only.

If you prefer to manage users in AWS Managed Microsoft AD, you can discontinue use of your AWS SSO directory at any time and instead connect AWS SSO to your Microsoft AD using AWS Directory Service. For more information, see Connect to Your Microsoft AD Directory (p. 9).

Topics

Add Users

Use the following procedure to add users to your AWS SSO directory.

To add a user

1. Open the AWS SSO console.

2. From the **Dashboard**, choose **Manage your directory**
3. On the **Directory** page, choose the **Users** tab, and then choose **Add user**.
4. On the **Add user** page, provide the following required information:

   a. **Email address**

   b. **Password** – Choose from one of the following choices to send the user's password.

      i. **Send an email to the user with password setup instructions** – This option automatically sends the user an email addressed from Amazon Web Services and invites the user on behalf of your company to access the AWS SSO user portal.

      ii. **Generate a one-time password that you can share with the user** – This option provides you with the user portal URL and password details that you can manually send to the user from your email address.

   c. **First name**

   d. **Last name**

   e. **Display name**

      Note
      (Optional) You can provide additional attributes such as **Employee ID** and **Office 365 Immutable ID** to help map the user's identity in AWS SSO with certain business applications that the user needs to use.
5. Choose **Next: Groups**.
6. Select one or more groups that you want the user to be a member of, and then choose **Add user**.

Add Groups

Use the following procedure to add groups to your AWS SSO directory.

To add a group

1. Open the AWS SSO console.
2. From the **Dashboard**, choose **Manage your directory**
3. On the **Directory** page, choose the **Groups** tab, and then choose **Create group**.
4. In the **Create group** dialog, enter a **Group name** and **Description**. The description should provide details on what permissions have been (or will be) assigned to the group.
5. Choose **Create**.

Add Users to Groups

Use the following procedure to add users as members of a group that you previously created in your AWS SSO directory.

To add a user as a member of a group

1. Open the AWS SSO console.
2. From the **Dashboard**, choose **Manage your directory**
3. On the **Directory** page, choose the **Groups** tab, and then choose a group from the list.
4. On the group **Details** page, under **Group members**, choose **Add users**.
5. On the **Add users to group** page, locate the users you want to add as members. Then select the check box next to each of them.
6. Choose **Add user**.

Edit User Properties

Use the following procedure to edit the properties of a user in your AWS SSO directory.

To edit user properties

1. Open the AWS SSO console.
2. From the **Dashboard**, choose **Manage your directory**
3. On the **Directory** page, choose the **Users** tab, and then choose the user that you want to edit.
4. On the user **Details** page, choose **Edit user**.
5. On the **Edit user details** page, make the updates to the properties as needed, and then choose **Save changes**.

   Note
   (Optional) You can modify additional attributes such as **Employee ID** and **Office 365 Immutable ID** to help map the user's identity in AWS SSO with certain business applications that users needs to use.

Disable a User

When you disable a user, you can not edit their user details, reset their password, add the user to a group, or view their group membership. Use the following procedure to disable a user in your AWS SSO directory.

To disable a user

1. Open the AWS SSO console.
2. From the **Dashboard**, choose **Manage your directory**
3. On the **Directory** page, choose the **Users** tab, and then choose the user you want to disable.
4. On the **Disable user** dialog, choose **Disable user**.

   Note
   Disabling a user prevents them from being able to sign in to the user portal.

Reset a User Password

Use the following procedure to reset the password for a user in your AWS SSO directory.

To reset a user password

1. Open the AWS SSO console.
2. From the **Dashboard**, choose **Manage your directory**
3. On the **Directory** page, choose the **Users** tab, and then choose the user whose password you want to reset.
4. In the **Reset password** dialog, select one of the following choices, and then choose **Reset password**:

   a. **Send an email to the user with instructions to reset the password** – This option automatically sends the user an email addressed from Amazon Web Services that walks them through how to reset their password.

   b. **Generate a one-time password and share the password with the user** – This option provides you with the password details that you can manually send to the user from your email address.

Connect to Your Microsoft AD Directory

AWS Single Sign-On enables administrators to connect their on-premises Active Directory (AD) or their AWS Managed Microsoft AD directory using AWS Directory Service. This Microsoft AD directory defines the pool of identities that administrators can pull from when using the AWS SSO console to assign single sign-on (SSO) access. After connecting their corporate directory to AWS SSO, administrators can then grant their AD users or groups access to AWS accounts, cloud applications, or both.

AWS Directory Service helps you to set up and run a standalone AWS Managed Microsoft AD directory hosted in the AWS Cloud. You can also use AWS Directory Service to connect your AWS resources with an existing on-premises Microsoft Active Directory. To configure AWS Directory Service to work with your on-premises Active Directory, you must first set up trust relationships to extend authentication from on-premises to the cloud.

> **Note**
> AWS SSO does not support SAMBA4-based Simple AD as a connected directory.

Topics

Connect AWS SSO to an AWS Managed Microsoft AD Directory

Use the following procedure to connect an AWS Managed Microsoft AD directory that is managed by AWS Directory Service to AWS SSO.

To connect AWS SSO to AWS Managed Microsoft AD

1. Open the AWS SSO console.

   > **Note**
   > Make sure that the AWS SSO console is using one of the regions where your AWS Managed Microsoft AD directory is located before you move to the next step.
2. From the **Dashboard**, choose **Manage your directory**
3. On the **Directory** page, do the following:

   a. Under **Available directories**, select the AWS Managed Microsoft AD directory you want AWS SSO to connect to.

   b. Under **User portal URL**, type the prefix to use for the user portal sign-in URL.
4. Choose **Connect directory**.

Connect AWS SSO to an On-Premises Active Directory

Users in your on-premises Active Directory can also have SSO access to AWS accounts and cloud applications in the AWS SSO user portal. To do that, AWS Directory Service has the following two options available:

- **Create a two-way trust relationship** – When two-way trust relationships are created between AWS Managed Microsoft AD and an on-premises Active Directory, on-premises users can sign in with their corporate credentials to various AWS services and business applications. One-way trusts do not work with AWS SSO. For more information about setting up a two-way trust, see When to Create a Trust Relationship in the *AWS Directory Service Administration Guide*.

- **Create an AD Connector** – AD Connector is a directory gateway that can redirect directory requests to your on-premises Active Directory without caching any information in the cloud. For more information, see Connect to a Directory in the *AWS Directory Service Administration Guide*.

  Note
  If you are connecting AWS SSO to an AD Connector directory, any future user password resets must be done from within Active Directory. This means that users will not be able to reset their passwords from the user portal.

  Note
  AWS SSO does not work with SAMBA4-based Simple AD directories.

Attribute Mappings

Attribute mappings are used to map attribute types that exist in AWS SSO with like attributes in an AWS Managed Microsoft AD directory. AWS SSO retrieves user attributes from your Microsoft AD directory and maps them to AWS SSO user attributes. These AWS SSO user attribute mappings are also used for generating SAML assertions for your cloud applications. Each cloud application determines the list of SAML attributes it needs for successful single sign-on.

AWS SSO prefills a set of attributes for you under the **Attribute mappings** tab found on your application's configuration page. AWS SSO uses these user attributes to populate SAML assertions (as SAML attributes) that are sent to the cloud application. These user attributes are in turn retrieved from your Microsoft AD directory. For more information, see Map Attributes in Your Application to AWS SSO Attributes (p. 29).

AWS SSO also manages a set of attributes for you under the **Attribute mappings** section of your directory configuration page. For more information, see Map Attributes in AWS SSO to Attributes in Your AWS Managed Microsoft AD Directory (p. 12).

Supported Directory Attributes

The following table lists all AWS Managed Microsoft AD directory attributes that are supported and that can be mapped to user attributes in AWS SSO.

Supported attributes in your Microsoft AD directory
`${dir:email}`
`${dir:displayname}`
`${dir:distinguishedName}`
`${dir:firstname}`
`${dir:guid}`
`${dir:initials}`
`${dir:lastname}`
`${dir:proxyAddresses}`
`${dir:proxyAddresses:smtp}`

Supported attributes in your Microsoft AD directory
`${dir:proxyAddresses:SMTP}`
`${dir:windowsUpn}`

You can specify any combination of supported Microsoft AD directory attributes to map to a single attribute in AWS SSO. For example, you could choose the `preferredUsername` attribute under the **User attribute in AWS SSO** column. Then map it to either `${dir:displayname}` or `${dir:lastname}${dir:firstname }` or any single supported attribute or any arbitrary combination of supported attributes.

Supported AWS SSO Attributes

The following table lists all AWS SSO attributes that are supported and that can be mapped to user attributes in your AWS Managed Microsoft AD directory. Later, after you set up your application attribute mappings, you can use these same AWS SSO attributes to map to actual attributes used by that application.

Supported attributes in AWS SSO
`${user:AD_GUID}`
`${user:email}`
`${user:familyName}`
`${user:firstName}`
`${user:middleName}`
`${user:name}`
`${user:preferredUsername}`
`${user:subject}`

Default Mappings

The following table shows the default mappings for user attributes in AWS SSO to the user attributes in your AWS Managed Microsoft AD directory. At this time, AWS SSO only supports the list of attributes shown in the **User attribute in AWS SSO** column.

User attribute in AWS SSO	Maps to this attribute in your Microsoft AD directory
`AD_GUID`	`${dir:guid}`
`email`	`${dir:windowsUpn}`
`familyName`	`${dir:lastname}`
`givenName`	`${dir:firstname}`
`middleName`	`${dir:initials}`

User attribute in AWS SSO	Maps to this attribute in your Microsoft AD directory
`name`	`${dir:displayname}`
`preferredUsername`	`${dir:displayname}`
`subject`	`${dir:windowsUpn}`

You can change the default mappings or add more attributes to the SAML assertion based on your requirements. For example, assume that your cloud application requires the users email in the `User.Email` SAML attribute. In addition, assume that email messages are stored in the `windowsUpn` attribute in your Microsoft AD directory. To achieve this mapping, you must make changes in the following two places in the AWS SSO console:

1. On the **Directory** page, under the **Attribute mappings** section, you would need to map the user attribute **`email`** to the **`${dir:windowsUpn}`** attribute (in the **Maps to this attribute in your directory** column)
2. On the **Applications** page, choose the application from the table. Choose the **Attribute mappings** tab. Then map the `User.Email` attribute to the **`${user:email}`** attribute (in the **Maps to this string value or user attribute in AWS SSO** column).

Please note that you must supply each directory attribute in the form ${dir:**`AttributeName`**}. For example, the `firstname` attribute in your Microsoft AD directory becomes `${dir:firstname}`. It is important that every directory attribute have an actual value assigned. Attributes missing a value after `${dir:` will cause user sign-in issues.

Map Attributes in AWS SSO to Attributes in Your AWS Managed Microsoft AD Directory

You can use the following procedure to specify how your user attributes in AWS SSO should map to corresponding attributes in your Microsoft AD directory.

To map attributes in AWS SSO to attributes in your directory

1. Open the AWS SSO console.
2. Choose **Connected directory**.
3. Under **Attribute mappings**, choose **Edit attribute mappings**.
4. On the **Edit attribute mappings** page, find the attribute in AWS SSO that you want to map and then type a value in the text box. For example, you might want to map the AWS SSO user attribute **`email`** to the Microsoft AD directory attribute **`${dir:windowsUpn}`**.
5. Choose **Save changes**.

Change Your Directory Type

You can change where you store users at any time. Use the following procedure to switch from a directory that AWS SSO provides (the default) to an AWS Managed Microsoft AD directory or vice versa.

To change your directory type

1. Open the AWS SSO console.
2. From the **Dashboard**, choose **Manage your directory**

3. On the **Directory** page, select **Change directory**.
4. On the **Change directory** page, select the directory you want to switch to, and then choose **Next**. If you are switching to a Microsoft AD directory, you must choose the available directory from the provided menu.

    Important
    Changing a directory removes all user assignments that were previously assigned. You must manually reapply these once you have successfully changed your directory.
5. Choose **Next: Review**.
6. Once you have read the disclaimer and are ready to proceed, type **CONFIRM**.
7. Choose **Finish**.

Manage SSO to Your AWS Accounts

AWS Single Sign-On is integrated with AWS Organizations so that administrators can pick multiple AWS accounts whose users need single sign-on (SSO) access to the AWS Management Console. These AWS accounts can be either the master account of the AWS Organizations or a member account. A master account is the AWS account that is used to create the organization. The rest of the accounts that belong to an organization are called member accounts. For more information about the different account types, see AWS Organizations Terminology and Concepts in the *AWS Organizations User Guide*.

Once you assign access from the AWS SSO console, you can use permission sets to further refine what users can do in the AWS Management Console. For more information about permission sets, see Permission Sets (p. 16).

Users follow a simple sign-in process:

1. Users use their directory credentials to sign in to the user portal.
2. Users then choose the AWS account name that will give them federated access to the AWS Management Console for that account.
3. Users who are assigned multiple permission sets choose which IAM role to use.

Permission sets are a way to centrally define permissions centrally in AWS SSO so that they can be applied to all of your AWS accounts. These permission sets are provisioned to each AWS account as an IAM role. The user portal gives users the ability to retrieve temporary credentials for the IAM role of a given AWS account so they can use it for short-term access to the AWS CLI. For more information, see How to Get Credentials of an IAM Role for Use with CLI Access to an AWS Account (p. 45).

To use AWS SSO with AWS Organizations, you must first Enable AWS SSO (p. 3), which grants AWS SSO the capability to create Service-Linked Roles (p. 20) in each account in your AWS organization. These roles are not created until after you Assign User Access (p. 15) for a given account.

You can also connect an AWS account that is not part of your organization by setting up the account as a custom SAML application in AWS SSO. In this scenario, you provision and manage the IAM roles and trust relationships that are required to enable SSO access. For more information on how to do this, see Add and Configure a Custom SAML 2.0 Application (p. 24).

Topics

Single Sign-On Access

You can assign users in your connected directory permissions to master or member AWS accounts in your AWS Organizations organization based on common job functions. Or you can use custom permissions to meet your specific security requirements. For example, you can grant database administrators broad permissions to Amazon RDS in development accounts but limit their permissions in production accounts. AWS SSO configures all the necessary user permissions in your AWS accounts automatically.

> **Note**
> Only the IAM account root user or a user who has the **AWSSSOMasterAccountAdministrator** IAM policy attached can grant users in your connected directory permissions to the master AWS

account. For more information on how to delegate these permissions, see Delegate Who Can Assign SSO Access to Users in the Master Account (p. 16).

Assign User Access

Use the following procedure to assign SSO access to users and groups in your connected directory and use permission sets to determine their level of access.

> **Note**
> To simplify administration of access permissions, we recommended that you assign access directly to groups rather than to individual users. With groups you can grant or deny permissions to groups of users rather than having to apply those permissions to each individual. If a user moves to a different organization, you simply move that user to a different group and they automatically receive the permissions that are needed for the new organization.

To assign access to users or groups

1. Open the AWS SSO console.

   > **Note**
   > Make sure that the AWS SSO console is using the Region where your AWS Managed Microsoft AD directory is located before you move to the next step.
2. Choose **AWS accounts**.
3. Under the **AWS organization** tab, in the list of AWS accounts, choose an account to which you want to assign access.
4. On the AWS account details page, choose **Assign users**.
5. On the **Select users or groups** page, type a user or group name and choose **Search connected directory**. Once you have selected all the accounts that you want to assign access to, choose **Next: Permission sets**. You can specify multiple users or groups by selecting the applicable accounts as they appear in search results.
6. On the **Select permission sets** page, select the permission sets that you want to apply to the user or group from the table. Then choose **Finish**. You can optionally choose to **Create a new permission set** if none of the permissions in the table meets your needs. For detailed instructions, see Create Permission Set (p. 16).
7. Choose **Finish** to begin the process of configuring your AWS account.

   > **Note**
   > If this is the first time you have assigned SSO access to this AWS account, this process creates a service-linked role in the account. For more information, see Using Service-Linked Roles for AWS SSO (p. 39).

   > **Important**
   > The user assignment process may take a few minutes to complete. It is important that you leave this page open until the process successfully completes.

Remove User Access

Use this procedure when you need to remove SSO access to an AWS account for a particular user or group in your connected directory.

To remove user access from an AWS account

1. Open the AWS SSO console.
2. Choose **AWS accounts**.
3. In the table, select the AWS account with the user or group whose access you want to remove.

4. On the **Details** page for the AWS account, under **Assigned users and groups**, locate the user or group in the table. Then choose **Remove access**.
5. In the **Remove access** dialog box, confirm the user or group name. Then choose **Remove access**.

Delegate Who Can Assign SSO Access to Users in the Master Account

Assigning single sign-on access to the master account using the AWS SSO console is a privileged action. By default, only an AWS account root user, or a user who has the **AWSSSOMasterAccountAdministrator** AWS managed policy attached, can assign SSO access to the master account. The **AWSSSOMasterAccountAdministrator** provides manage SSO access to the master account within an AWS Organizations organization.

Use the following steps to delegate permissions to manage SSO access to users in your directory.

To grant permissions to manage SSO access to users in your directory

1. Sign in to the AWS SSO console as a root user of the master account or with another IAM user who has IAM administrator permissions to the master account.
2. Use the procedure Create Permission Set (p. 16) to create a permission set. When you get to step 5c, select the option **Attach AWS managed policies**. In the list of IAM policies that appear in the table, choose the **AWSSSOMasterAccountAdministrator** AWS managed policy. This policy grants permissions to any user who will be assigned access to this permission set in the future.
3. Use the procedure Assign User Access (p. 15) to assign the appropriate users to the permission set that you just created.
4. Communicate the following to the assigned users: When they sign in to the user portal and select the **AWS Account** icon, they must choose the appropriate IAM role name to be authenticated with the permissions that you just delegated.

Permission Sets

Permission sets define the level of access that users and groups have to an AWS account. Permission sets are stored in AWS SSO and provisioned to the AWS account as IAM roles. You can assign more than one permission set to a user. Users who have multiple permission sets must choose one when they sign in to the user portal. (Users will see these as IAM roles). For more information, see Permission Sets (p. 5).

Topics

Create Permission Set

Use this procedure to create a permission set based on a custom permissions policy that you create, or on predefined AWS managed policies that exist in IAM, or both.

To create a permission set

1. Open the AWS SSO console.

2. Choose **AWS accounts**.
3. Select the **Permission sets** tab.
4. Choose **Create permission set**.
5. On the **Create new permission set** page, choose from one of the following options, and then follow the instructions provided under that option:

   - **Use an existing job function policy**
     1. Under **Select job function policy**, select one of the default IAM job function policies in the list. For more information, see AWS Managed Policies for Job Functions.
     2. Choose **Create**.
   - **Create a custom permission set**
     1. Under **Create a custom permission set**, type a name that will identify this permission set in AWS SSO. This name will also appear as an IAM role in the user portal for any users who have access to it.
     2. (Optional) You can also type a description. This description will only appear in the AWS SSO console and will not be visible to users in the user portal.
     3. (Optional) Specify the value for **Session duration**. This value is used to determine the length of time a user can be logged on before the console logs them out of their session. For more information, see Set Session Duration (p. 17).
     4. (Optional) Specify the value for **Relay state**. This value is used in the federation process to redirect users within the account. For more information, see Set Relay State (p. 18).
     5. Select either **Attach AWS managed policies** or **Create a custom permissions policy**. Or select both if you need to link more than one policy type to this permission set.
     6. If you chose **Attach AWS managed policies**, under **Attach AWS Managed policies**, select up to 10 job-related or service-specific AWS managed policies from the list.
     7. If you chose **Create a custom permissions policy**, under **Create a custom permissions policy**, paste in a policy document with your preferred permissions. For a list of example policies to use for delegating AWS SSO tasks, see Customer Managed Policy Examples (p. 36).

        For more information about the access policy language, see Overview of Policies in the *IAM User Guide*. To test the effects of this policy before applying your changes, use the IAM policy simulator.
     8. Choose **Create**.

Configure Permission Set Properties

In AWS SSO you can customize the user experience by configuring the following permission set properties.

Topics

Set Session Duration

For each permission set, you can specify a session duration to control the length of time that a user can be signed in to an AWS account. When the specified duration has elapsed, AWS logs the user out of the session. For AWS accounts, AWS SSO uses this setting to set the maximum session duration of the IAM role that you use to generate a user's session. The session duration that you specify for a given permission set applies to both the AWS Management Console and the AWS Command Line Interface (CLI) session.

When you create a new permission set, it comes configured with the default session length of 1 hour (in seconds). The minimum session duration length is 1 hour and can be configured up to 12 hours.

> **Important**
> As a security best practice, we recommend that you do not set the session duration length longer than is needed to perform the role.

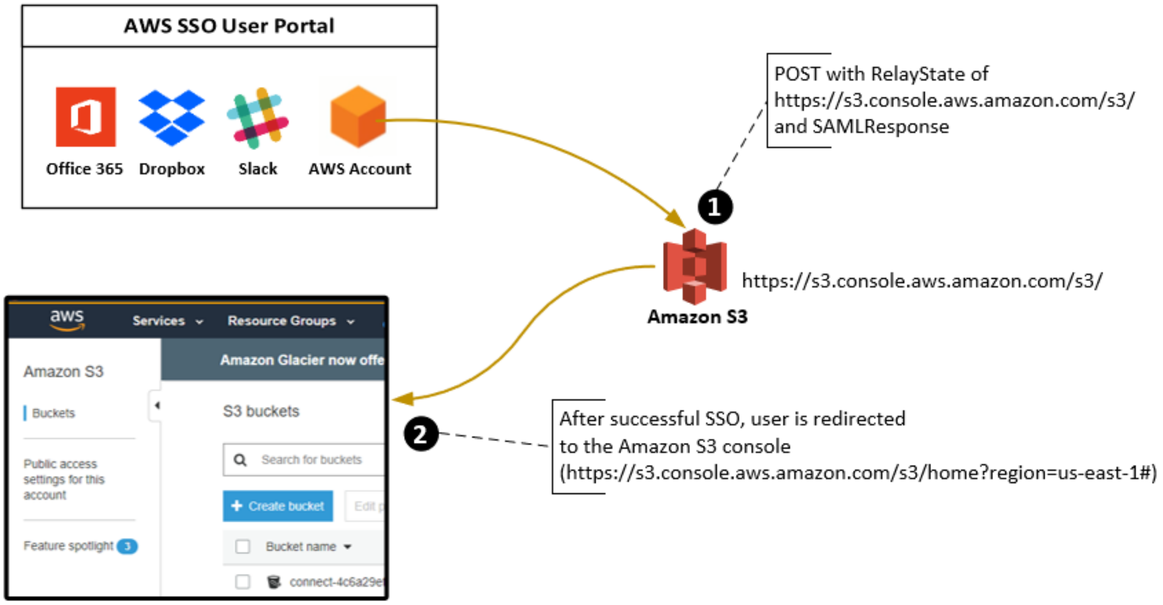
Once a permission set has been created, you can later update it to apply a new session duration. When you reapply the permission set to your AWS accounts, the IAM role's maximum session duration value is updated. Use the following procedure to modify the session duration length for a given permission set.

To set the session duration

1. Open the AWS SSO console.
2. Choose **AWS accounts**.
3. Choose the **Permission sets** tab.
4. Choose the name of the permission set that will have the new session duration.
5. On the **Permissions** tab, next to **Session duration**, choose **Edit**.
6. On the **Edit session duration** page, next to **New session duration**, choose a new session length value, and then choose **Continue**.
7. Select the AWS accounts in the list that you want the new session duration value to apply to, and then choose **Reapply permission set**.

Set Relay State

During the federation authentication process, the relay state redirects users within the AWS Management Console. You can specify a relay state URL to redirect links to any service in the AWS Management Console. For example, the below illustration shows the process for redirecting to the S3 console (https://s3.console.aws.amazon.com/s3/home?region=us-east-1#).


AWS SSO User Portal
Office 365
Dropbox
Slack
AWS Account
POST with RelayState of https://s3.console.aws.amazon.com/s3/ and SAMLResponse
1
Amazon S3
https://s3.console.aws.amazon.com/s3/
aws
Services
Resource Groups
Amazon S3
Amazon Glacier now offe
Buckets
S3 buckets
Public access settings for this account
Search for buckets
Create bucket
Edit
Feature spotlight 3
Bucket name
connect-4c6a29e
2
After successful SSO, user is redirected to the Amazon S3 console (https://s3.console.aws.amazon.com/s3/home?region=us-east-1#)

Use the following procedure to modify the relay state URL for a given permission set.

To set the relay state

1. Open the AWS SSO console.

2. Choose **AWS accounts**.
3. Choose the **Permission sets** tab.
4. Choose the name of the permission set that will have the new relay state URL.
5. On the **Permissions** tab, choose **Edit**.
6. On the **Edit general permission settings** page, next to **Relay state**, type a URL value for any of the AWS services, and then choose **Continue**.
7. Select the AWS accounts in the list that you want the new relay state value to apply to, and then choose **Reprovision**.

Delete Permission Sets

Use this procedure to delete one or more permission sets so that they can no longer be used by any AWS account in the organization.

> **Note**
> All users and groups that have been assigned this permission set, regardless of what AWS account is using it, will no longer be able to sign in.

To delete a permissions set from an AWS account

1. Open the AWS SSO console.
2. Choose **AWS accounts**.
3. Choose the **Permission sets** tab.
4. Select the permission set you want to delete, and then choose **Delete**.
5. In the **Delete permission set** dialog box, choose **Delete**.

IAM Identity Provider

When you add SSO access to an AWS account, AWS SSO creates an IAM identity provider in each AWS account. An IAM identity provider helps keep your AWS account secure because you don't have to distribute or embed long-term security credentials, such as IAM access keys, in your application.

Repair the IAM Identity Provider

Use the following procedure to repair your identity provider in case it was deleted or modified.

To repair an identity provider for an AWS account

1. Open the AWS SSO console.
2. Choose **AWS accounts**.
3. In the table, select the AWS account that is associated with the identity provider that you want to repair.
4. On the AWS account details page, under **IAM identity provider**, choose **Repair identity provider**.

Remove the IAM Identity Provider

Use the following procedure to remove the IAM identity provider from AWS SSO.

To remove the IAM identity provider from AWS SSO

1. Open the AWS SSO management console

2. Choose **AWS accounts**
3. In the table select the AWS account that is associated with the IAM identity provider that you want to remove.
4. On the **Details** page for the AWS account, under **IAM identity provider**, choose **Remove identity provider**.

Service-Linked Roles

Service-linked roles are predefined IAM permissions that allow AWS SSO to delegate and enforce which users have SSO access to specific AWS accounts in your AWS organization. The service enables this functionality by provisioning a service-linked role in every AWS account within its organization. The service then allows other AWS services like AWS SSO to leverage those roles to perform service-related tasks. For more information, see AWS Organizations and Service-Linked Roles.

During the process to Enable AWS SSO (p. 3) for the first time, the AWS Organizations service grants AWS SSO the necessary permissions to create IAM roles in any of its AWS accounts. AWS SSO doesn't create roles in any of the AWS accounts at this point. It only creates a service-linked role in an AWS account after you have used the AWS SSO console to specify which account you want to assign SSO access to. For more information, see Manage SSO to Your AWS Accounts (p. 14).

Service-linked roles that are created in each AWS account are named `AWSServiceRoleForSSO`. For more information, see Using Service-Linked Roles for AWS SSO (p. 39).

Manage SSO to Your Applications

With AWS Single Sign-On, you can easily control who can have single sign-on (SSO) access to your cloud applications. Users get one-click access to these applications after they use their directory credentials to sign in to their user portal.

AWS SSO securely communicates with these applications through a trusted relationship between AWS SSO and the application's service provider. This trust is created when you add the application from the AWS SSO console and configure it with the appropriate metadata for both AWS SSO and the service provider.

After the application has been successfully added to the AWS SSO console, you can manage which users or groups need permissions to the application. By default, when you add an application, no users are assigned to the application. In other words, newly added applications in the AWS SSO console are inaccessible until you assign users to them. AWS SSO supports the following applications types:

- Cloud applications
- Custom Security Assertion Markup Language (SAML 2.0) applications

You can also grant your employees access to the AWS Management Console for a given AWS account in your organization. For more information on how to do this, see Manage SSO to Your AWS Accounts (p. 14).

The following sections explain how to configure user access to your third-party software as a service (SaaS) applications. You can also configure any custom applications that support identity federation with SAML 2.0.

Topics

Cloud Applications

You can use the AWS SSO application configuration wizard to include built-in SAML integrations to many popular cloud applications. Examples include Salesforce, Box, and Office 365. For a complete list of applications that you can add from the wizard, see Supported Applications (p. 21).

Most cloud applications come with detailed instructions on how to set up the trust between AWS SSO and the application's service provider. You can find these instructions on the cloud applications configuration page during the setup process and after the application has been set up. After the application has been configured, you can assign access to the groups or users that require it.

Supported Applications

AWS SSO has built-in support for the following commonly used cloud applications.

Note

AWS Support engineers can assist customers who have Business and Enterprise support plans with some integration tasks that involve third-party software. For a current list of supported platforms and applications, see Third-Party Software Support on the *AWS Support Features* page.

10000ft	Deputy	Heap	Peakon	TalentLMS
4me	Deskpro	Help Scout	Pipedrive	Targetprocess
7Geese	Deskradar	Honey	Pivotal Tracker	TextMagic
Accredible	DigiCert	Hosted Graphite	ProdPad	ThousandEyes
Adobe Creative Cloud	dmarcian	Humanity	Proto.io	Tinfoil Security
Aha	Docebo	IdeaScale	Proxyclick	TitanFile
AlertOps	DocuSign	Igloo	PurelyHR	Trakdesk
AnswerHub	Dome9	Image Relay	Recognize	Trello
AppDynamics	Domo	iSpring	RescueAssist	Uptime
AppFollow	Drift	IT Glue	RingCentral	Uptrends
Asana	Dropbox	Jama Software	Robin	UserEcho
Assembla	Druva inSync	JFrog Artifactory	Rollbar	UserVoice
Atlassian	Duo	Jira	Salesforce	Velpic
BambooHR	EduBrite	Jitbit	Samanage	VictorOps
BenSelect	Egnyte	join.me	ScaleFT	Vtiger
BitaBIZ	eLeaP	Kanbanize	ScreenSteps	Way We Do
Bitglass	Engagedly	Keeper Security	ServiceNow	Weekdone
BlueJeans	Envoy	Kintone	Slack	WhosOnLocation
BMC Remedyforce	Evernote	Klipfolio	Slemma	Workplace by Facebook
Bonusly	Expensify	KnowledgeOwl	Sli.do	Workstars
Box	Expiration Reminder	Kudos	Small Improvements	Wrike
Bugsnag	EZOfficeInventory	LiquidFiles	Smartsheet	xMatters
Buildkite	EZRentOut	LiquidPlanner	SnapEngage	Yodeck
CakeHR	Fastly	Litmos	Split.io	Zendesk
Chartio	Flock	LiveChat	Spotinst	Ziflow
Circonus	Formstack	LogMeInRescue	SproutVideo	Zillable
Cisco Webex	Freshdesk	Lucidchart	Stackify	Zoho

Cisco Meraki	Freshservice	ManageEngine	Status Hero	Zoom
Cisco Umbrella	Front	MangoApps	StatusCast	
Citrix ShareFile	G Suite	Miro	StatusDashboard	
Clarizen	Github	MockFlow	StatusHub	
ClickTime	Gitlab	New Relic	Statuspage	
CloudAMQP	Glasscubes	Nuclino	StoriesOnBoard	
CloudPassage	GorillaStack	Office 365	Stormboard	
CMNTY	GoToMeeting	Opsgenie	SugarCRM	
Confluence	GoToTraining	Pacific Timesheet	Sumo Logic	
Convo	GoToWebinar	PagerDuty	SurveyGizmo	
Coralogix	Grovo	Panopta	SurveyMonkey	
Datadog	HackerOne	Panorama9	Syncplicity	
Declaree	HackerRank	ParkMyCloud	Tableau	

Add and Configure a Cloud Application

Use this procedure when you need to set up a SAML trust relationship between AWS SSO and your cloud application's service provider. Before you begin this procedure, make sure you have the service provider's metadata exchange file so that you can more efficiently set up the trust. If you do not have this file, you can still use this procedure to configure it manually.

To add and configure a cloud application

1. In the AWS SSO console, choose **Applications** in the left navigation pane. Then choose **Add a new application**.
2. In the **Select an application** dialog box, select the application you want to add from the list. Then choose **Add**.
3. On the **Configure <application name>** page, under **Details**, type a **Display name** for the application, such as `Salesforce`.
4. Under **AWS SSO metadata**, do the following:

   a. Next to **AWS SSO SAML metadatafile**, choose **Download** to download the identity provider metadata.

   b. Next to **AWS SSO certificate**, choose **Download certificate** to download the identity provider certificate.

   Note
   You will need these files later when you set up the cloud application from the service provider's website. Follow the instructions from that provider.
5. (Optional) Under **Application properties**, you can specify additional properties for the **Application start URL**, **Relay State**, and **Session Duration**. For more information, see Application Properties (p. 27).
6. Under **Application metadata**, provide the **Application ACS URL** and **Application SAML audience** values.

7. Choose **Save changes** to save the configuration.

Custom SAML 2.0 Applications

You can use the AWS SSO application configuration wizard to add support for applications that allow identity federation using Security Assertion Markup Language (SAML) 2.0. In the console, you set these up by choosing **Custom SAML 2.0 application** from the application selector. Most of the steps for configuring a custom SAML application are the same as configuring a cloud application.

However, you also need to provide additional SAML attribute mappings for a custom SAML application. These mappings tell AWS SSO how to populate the SAML assertion correctly for your application. You can provide this additional SAML attribute mapping when you set up the application for the first time. You can also provide SAML attribute mappings on the application detail page that is accessible from the AWS SSO console.

Add and Configure a Custom SAML 2.0 Application

Use this procedure when you need to set up a SAML trust relationship between AWS SSO and your custom application's service provider. Before you begin this procedure, make sure that you have the service provider's certificate and metadata exchange files so that you can finish setting up the trust.

To add and configure a custom SAML application

1. In the AWS SSO console, choose **Applications** in the left navigation pane. Then choose **Add a new application**.
2. In the **Select an application** dialog box, select **Custom SAML 2.0 application** from the list. Then choose **Configure application**.
3. On the **Configure <Custom app name>** page, under **Details**, type a **Display name** for the application, such as `MyApp`.
4. Under **AWS SSO metadata**, do the following:

   a. Next to **AWS SSO SAML metadatafile**, choose **Download** to download the identity provider metadata.

   b. Next to **AWS SSO certificate**, choose **Download certificate** to download the identity provider certificate.

   Note
   You will need these files later when you set up the custom application from the service provider's website.
5. (Optional) Under **Application properties**, you can specify additional properties for the **Application start URL**, **Relay State**, and **Session Duration**. For more information, see Application Properties (p. 27).
6. Under **Application metadata**, provide the **Application ACS URL** and **Application SAML audience** values.
7. Choose **Save changes** to save the configuration.

Manage AWS SSO Certificates

AWS SSO uses certificates to set up a SAML trust relationship between AWS SSO and your cloud application's service provider. When you add an application in AWS SSO, an AWS SSO certificate is automatically created for use with that application during the setup process. By default, this auto-generated AWS SSO certificate is valid for a period of five years.

As an AWS SSO administrator, you'll occasionally need to replace older certificates with newer ones for a given application. For example, you might need to replace a certificate when the expiration date on the certificate approaches. The process of replacing an older certificate with a newer one is referred to as *certificate rotation*.

Topics

Considerations Before Rotating a Certificate

Before you start the process of rotating a certificate in AWS SSO, consider the following:

- The certification rotation process requires that you reestablish the trust between AWS SSO and the service provider. To reestablish the trust, use the procedures provided in Rotate an AWS SSO Certificate (p. 25).
- Updating the certificate with the service provider may cause a temporary service disruption for your users until the trust has been successfully reestablished. Plan this operation carefully during off peak hours if possible.

Rotate an AWS SSO Certificate

Rotating an AWS SSO certificate is a multistep process that involves the following:

- Generating a new certificate
- Adding the new certificate to the service provider’s website
- Setting the new certificate to active
- Deleting the inactive certificate

Use all of the following procedures in the following order to complete the certificate rotation process for a given application.

Step 1: Generate a new certificate.

New AWS SSO certificates that you generate can be configured to use the following properties:

- **Validity period** – Specifies the time allotted (in months) before a new AWS SSO certificate expires.
- **Key size** – Determines the number of bits that a key must use with its cryptographic algorithm. You can set this value to either 1024-bit RSA or 2048-bit RSA. For general information about how key sizes work in cryptography, see Key size.
- **Algorithm** – Specifies the algorithm that AWS SSO uses when signing the SAML assertion/response. You can set this value to either SHA-1 or SHA-256. AWS recommends using SHA-256 when possible, unless your service provider requires SHA-1. For general information about how cryptography algorithms work, see Public-key cryptography.

1. Open the AWS SSO console.

2. Choose **Applications**.
3. In the list of applications, choose the application that you want to generate a new certificate for.
4. On the application details page, choose the **Configuration** tab. Under **AWS SSO metadata**, choose **Manage certificate**.
5. On the **AWS SSO certificate** page, choose **Generate new certificate**.
6. In the **Generate new AWS SSO certificate** dialog box, specify the appropriate values for **Validity period**, **Algorithm**, and **Key size**. Then choose **Generate**.

Step 2: Update the service provider's website.

Use the following procedure to reestablish the trust with the application's service provider.

> **Important**
> When you upload the new certificate to the service provider, your users might not be able to get authenticated. To correct this situation, set the new certificate as active as described in the next step.

1. In the AWS SSO console, choose the application that you just generated a new certificate for.
2. On the application details page, choose **Edit configuration**.
3. Choose **View instructions**, and then follow the instructions for your specific application service provider’s website to add the newly generated certificate.

Step 3: Set the new certificate to active.

An application can have up to two certificates assigned to it. Whichever certificate is set as active, AWS SSO will use it to sign all SAML assertions.

1. Open the AWS SSO console.
2. Choose **Applications**.
3. In the list of applications, choose your application.
4. On the application details page, choose the **Configuration** tab. Under **AWS SSO metadata**, choose **Manage certificate**.
5. On the **AWS SSO certificate** page, select the certificate you want to set to active, choose **Actions**, and then choose **Set as active**.
6. In the **Set the selected certificate as active** dialog, confirm that you understand that setting a certificate to active may require you to re-establish the trust, and then choose **Make active**.

Step 4: Delete the old certificate.

Use the following procedure to complete the certificate rotation process for your application. You can only delete a certificate that is in an **Inactive** state.

1. Open the AWS SSO console.
2. Choose **Applications**.
3. In the list of applications, choose your application.
4. On the application details page, select the **Configuration** tab. Under **AWS SSO metadata**, choose **Manage certificate**.
5. On the **AWS SSO certificate** page, select the certificate you want to delete. Choose **Actions** and then choose **Delete**.
6. In the **Delete certificate** dialog box, choose **Delete**.

Certificate Expiration Status Indicators

While on the **Applications** page in the properties of an application, you may notice colored status indicator icons. These icons appear in the **Expires on** column next to each certificate in the list. The following describes the criteria that AWS SSO uses to determine which icon is displayed for each certificate.

- **Red** – Indicates that a certification is currently expired.
- **Yellow** – Indicates that a certification will expire in 90 days or less.
- **Green** – Indicates that a certification is currently valid and will remain valid for at least 90 more days.

To check the current status of a certificate

1. Open the AWS SSO console.
2. Choose **Applications**.
3. In the list of applications, review the status of the certificates in the list as indicated in the **Expires on** column.

Application Properties

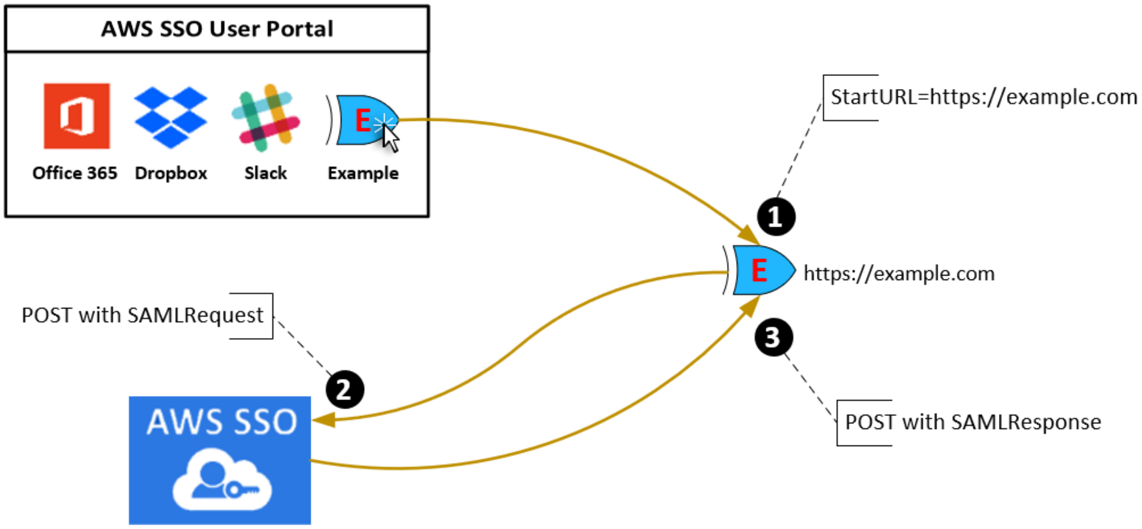
In AWS SSO you can customize the user experience by configuring the following additional application properties.

Application Start URL

You use an application start URL to start the federation process with your application. The typical use is for an application that supports only service provider (SP)-initiated binding.

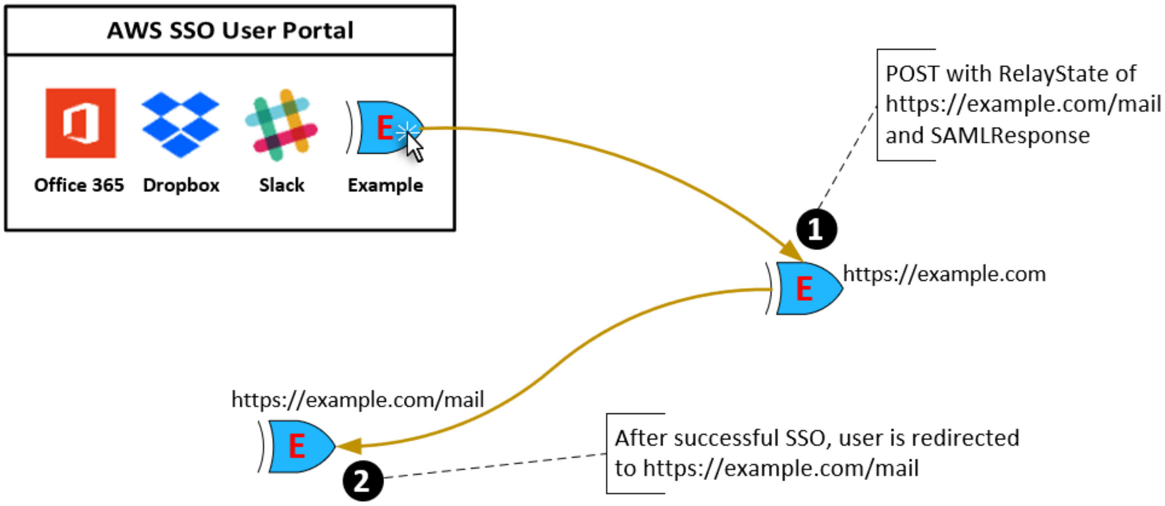
The following steps and diagram illustrates the application start URL authentication workflow when a user chooses an application in the user portal:

1. The user's browser redirects the authentication request using the value for the application start URL (in this case https://example.com).
2. The application sends an `HTML POST` with a `SAMLRequest` to AWS SSO.
3. AWS SSO then sends an `HTML POST` with a `SAMLResponse` back to the application.



Relay State

During the federation authentication process, the relay state redirects users within the application. For SAML 2.0, this value is passed, unmodified, to the application. After the application properties are configured, AWS SSO sends the relay state value along with a SAML response to the application.


AWS SSO User Portal
Office 365
Dropbox
Slack
Example
E
POST with RelayState of https://example.com/mail and SAMLResponse
1
E
https://example.com
https://example.com/mail
E
2
After successful SSO, user is redirected to https://example.com/mail

Session Duration

Session duration is the length of time that the application user sessions are valid for. For SAML 2.0, this is used to set the `NotOnOrAfter` date of the SAML assertion's elements; `saml2:SubjectConfirmationData` and `saml2:Conditions`.

Session duration can be interpreted by applications in any of the following ways:

- Applications can use it to determine how long the SAML assertion is valid. Applications do not consider session duration when deciding the time allowed for the user.
- Applications can use it to determine the maximum time that is allowed for the user's session. Applications might generate a user session with a shorter duration. This can happen when the application only supports user sessions with a duration that is shorter than the configured session length.
- Applications can use it as the exact duration and might not allow administrators to configure the value. This can happen when the application only supports a specific session length.

For more information about how session duration is used, see your specific application’s documentation.

Assign User Access

Use the following procedure to assign users SSO access to cloud applications or custom SAML 2.0 applications.

> **Note**
> To help simplify administration of access permissions, we recommend that you assign access directly to groups rather than to individual users. With groups you can grant or deny permissions to groups of users, rather than having to apply those permissions to each individual. If a user moves to a different organization, you simply move that user to a different group. The user then automatically receives the permissions that are needed for the new organization.

To assign access to users or groups

1. Open the AWS SSO console.

   Note
   Make sure that the AWS SSO console is using the Region where your AWS Managed Microsoft AD directory is located before taking the next step.
2. Choose **Applications**.
3. In the list of applications, choose an application to which you want to assign access.
4. On the application details page, choose the **Assigned users** tab. Then choose **Assign users**.
5. In the **Assign users** dialog box, enter a user or group name. Then choose **Search connected directory**. You can specify multiple users or groups by selecting the applicable accounts as they appear in search results.
6. Choose **Assign users**.

Remove User Access

Use this procedure to remove user access to cloud applications or custom SAML 2.0 applications.

To remove user access from an application

1. Open the AWS SSO console.
2. Choose **Applications**.
3. In the list of applications, choose an application whose access you want to remove.
4. On the application details page, choose the **Assigned users** tab. Select the user or group that you want to remove and then choose **Remove**.
5. In the **Remove access** dialog box, verify the user or group name. Then choose **Remove access**.

Map Attributes in Your Application to AWS SSO Attributes

Some service providers require custom SAML assertions to pass additional data about your user sign-ins. In that case, use the following procedure to specify how your applications user attributes should map to corresponding attributes in AWS SSO.

To map application attributes to attributes in AWS SSO

1. Open the AWS SSO console.
2. Choose **Applications**.
3. In the list of applications, choose the application where you want to map attributes.
4. On the application details page, choose the **Attribute mappings** tab.
5. Choose **Add new attribute mapping**
6. In the first text box, enter the application attribute.
7. In the second text box, enter the attribute in AWS SSO that you want to map to the application attribute. For example, you might want to map the application attribute **`Username`** to the AWS SSO user attribute **`email`**. To see the list of allowed user attributes in AWS SSO, see the table in Attribute Mappings (p. 10).
8. In the third column of the table, choose the appropriate format for the attribute from the menu.
9. Choose **Save changes**.

Authentication and Access Control for AWS SSO

Access to AWS SSO requires credentials that AWS can use to authenticate your requests. Those credentials must have permissions to access AWS resources, such as an AWS SSO application.

Authentication to the AWS SSO user portal is controlled by the directory that you have connected to AWS SSO. However, authorization to the AWS accounts that are available to end users from within the user portal is determined by two factors:

1. Who has been assigned access to those AWS accounts in the AWS SSO console. For more information, see Single Sign-On Access (p. 14).
2. What level of permissions have been granted to the end users in the AWS SSO console to allow them the appropriate access to those AWS accounts. For more information, see Permission Sets (p. 16).

The following sections explain how you as an administrator can control access to the AWS SSO console or can delegate administrative access for day-to-day tasks from the AWS SSO console.

- Authentication (p. 30)
- Access Control (p. 31)

Authentication

You can access AWS as any of the following types of identities:

- **AWS account root user** – When you first create an AWS account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you **do not use the root user for your everyday tasks**, even the administrative ones. Instead, adhere to the best practice of using the root user only to create your first IAM user. Then securely lock away the root user credentials and use them to perform only a few account and service management tasks.
- **IAM user** – An IAM user is an identity within your AWS account that has specific custom permissions (for example, permissions to create a directory in AWS SSO). You can use an IAM user name and password to sign in to secure AWS webpages like the AWS Management Console, AWS Discussion Forums, or the AWS Support Center.

  In addition to a user name and password, you can also generate access keys for each user. You can use these keys when you access AWS services programmatically, either through one of the several SDKs or by using the AWS Command Line Interface (CLI). The SDK and CLI tools use the access keys to cryptographically sign your request. If you don't use AWS tools, you must sign the request yourself. AWS SSO supports *Signature Version 4*, a protocol for authenticating inbound API requests. For more information about authenticating requests, see Signature Version 4 Signing Process in the *AWS General Reference*.

- **IAM role** – An IAM role is an IAM identity that you can create in your account that has specific permissions. It is similar to an *IAM user*, but it is not associated with a specific person. An IAM role enables you to obtain temporary access keys that can be used to access AWS services and resources. IAM roles with temporary credentials are useful in the following situations:

  - **Federated user access** – Instead of creating an IAM user, you can use existing user identities from AWS Directory Service, your enterprise user directory, or a web identity provider. These are known as *federated users*. AWS assigns a role to a federated user when access is requested through an identity provider. For more information about federated users, see Federated Users and Roles in the *IAM User Guide*.

  - **AWS service access** – You can use an IAM role in your account to grant an AWS service permissions to access your account's resources. For example, you can create a role that allows Amazon Redshift to access an Amazon S3 bucket on your behalf and then load data from that bucket into an Amazon Redshift cluster. For more information, see Creating a Role to Delegate Permissions to an AWS Service in the *IAM User Guide*.

  - **Applications running on Amazon EC2** – You can use an IAM role to manage temporary credentials for applications that are running on an EC2 instance and making AWS API requests. This is preferable to storing access keys within the EC2 instance. To assign an AWS role to an EC2 instance and make it available to all of its applications, you create an instance profile that is attached to the instance. An instance profile contains the role and enables programs that are running on the EC2 instance to get temporary credentials. For more information, see Using an IAM Role to Grant Permissions to Applications Running on Amazon EC2 Instances in the *IAM User Guide*.

Access Control

You can have valid credentials to authenticate your requests, but unless you have permissions you cannot create or access AWS SSO resources. For example, you must have permissions to create an AWS SSO connected directory.

The following sections describe how to manage permissions for AWS SSO. We recommend that you read the overview first.

Overview of Managing Access Permissions to Your AWS SSO Resources

Every AWS resource is owned by an AWS account, and permissions to create or access the resources are governed by permissions policies. An account administrator can attach permissions policies to IAM identities (that is, users, groups, and roles). Some services (such as AWS Lambda) also support attaching permissions policies to resources.

> **Note**
> An *account administrator* (or administrator user) is a user with administrator privileges. For more information, see IAM Best Practices in the *IAM User Guide*.

When granting permissions, you decide who is getting the permissions, the resources they get permissions for, and the specific actions that you want to allow on those resources.

Topics

AWS SSO Resources and Operations

In AWS SSO, the primary resources are application instances, profiles, and permission sets.

Understanding Resource Ownership

A *resource owner* is the AWS account that created a resource. That is, the resource owner is the AWS account of the *principal entity* (the account, an IAM user, or an IAM role) that authenticates the request that creates the resource. The following examples illustrate how this works:

- If the AWS account root user creates an AWS SSO resource, such as an application instance or permission set, your AWS account is the owner of that resource.
- If you create an IAM user in your AWS account and grant that user permissions to create AWS SSO resources, the user can then create AWS SSO resources. However, your AWS account, to which the user belongs, owns the resources.
- If you create an IAM role in your AWS account with permissions to create AWS SSO resources, anyone who can assume the role can create AWS SSO resources. Your AWS account, to which the role belongs, owns the AWS SSO resources.

Managing Access to Resources

A *permissions policy* describes who has access to what. The following section explains the available options for creating permissions policies.

> **Note**
> This section discusses using IAM in the context of AWS SSO. It doesn't provide detailed information about the IAM service. For complete IAM documentation, see What Is IAM? in the *IAM User Guide*. For information about IAM policy syntax and descriptions, see AWS IAM Policy Reference in the *IAM User Guide*.

Policies that are attached to an IAM identity are referred to as *identity-based* policies (IAM policies). Policies that are attached to a resource are referred to as *resource-based* policies. AWS SSO supports only identity-based policies (IAM policies).

Topics

Identity-Based Policies (IAM Policies)

You can attach policies to IAM identities. For example, you can do the following:

- **Attach a permissions policy to a user or a group in your account** – An account administrator can use a permissions policy that is associated with a particular user to grant permissions for that user to add an AWS SSO resource, such as a new application.
- **Attach a permissions policy to a role (grant cross-account permissions)** – You can attach an identity-based permissions policy to an IAM role to grant cross-account permissions. For example, the administrator in Account A can create a role to grant cross-account permissions to another AWS account (for example, Account B) or an AWS service as follows:
  1. Account A administrator creates an IAM role and attaches a permissions policy to the role that grants permissions to resources in Account A.
  2. Account A administrator attaches a trust policy to the role identifying Account B as the principal who can assume the role.
  3. Account B administrator can then delegate permissions to assume the role to any users in Account B. Doing this allows users in Account B to create or access resources in Account A. The principal in the trust policy can also be an AWS service principal if you want to grant an AWS service permissions to assume the role.

  For more information about using IAM to delegate permissions, see Access Management in the *IAM User Guide*.

The following permissions policy grants permissions to a user to run all of the actions that begin with `List`. These actions show information about an AWS SSO resource, such as an application instance or permissions set. Note that the wildcard character (*) in the `Resource` element indicates that the actions are allowed for all AWS SSO resources that are owned by the account.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":"sso:List*",
         "Resource":"*"
      }
   ]
}
```

For more information about using identity-based policies with AWS SSO, see Using Identity-Based Policies (IAM Policies) for AWS SSO (p. 34). For more information about users, groups, roles, and permissions, see Identities (Users, Groups, and Roles) in the *IAM User Guide*.

Resource-Based Policies

Other services, such as Amazon S3, also support resource-based permissions policies. For example, you can attach a policy to an S3 bucket to manage access permissions to that bucket. AWS SSO doesn't support resource-based policies.

Specifying Policy Elements: Actions, Effects, Resources, and Principals

For each AWS SSO resource (see AWS SSO Resources and Operations (p. 32)), the service defines a set of API operations. To grant permissions for these API operations, AWS SSO defines a set of actions that you can specify in a policy. Note that performing an API operation can require permissions for more than one action.

The following are the basic policy elements:

- **Resource** – In a policy, you use an Amazon Resource Name (ARN) to identify the resource to which the policy applies. For AWS SSO resources, you always use the wildcard character (*) in IAM policies. For more information, see AWS SSO Resources and Operations (p. 32).
- **Action** – You use action keywords to identify resource operations that you want to allow or deny. For example, the `sso:DescribePermissionsPolicies` permission allows the user permissions to perform the AWS SSO `DescribePermissionsPolicies` operation.
- **Effect** – You specify the effect when the user requests the specific action—this can be either allow or deny. If you don't explicitly grant access to (allow) a resource, access is implicitly denied. You can also explicitly deny access to a resource, which you might do to make sure that a user cannot access it, even if a different policy grants access.
- **Principal** – In identity-based policies (IAM policies), the user that the policy is attached to is the implicit principal. For resource-based policies, you specify the user, account, service, or other entity that you want to receive permissions (applies to resource-based policies only). AWS SSO doesn't support resource-based policies.

To learn more about IAM policy syntax and descriptions, see AWS IAM Policy Reference in the *IAM User Guide*.

Specifying Conditions in a Policy

When you grant permissions, you can use the access policy language to specify the conditions that are required for a policy to take effect. For example, you might want a policy to be applied only after a specific date. For more information about specifying conditions in a policy language, see Condition in the *IAM User Guide*.

To express conditions, you use predefined condition keys. There are no condition keys specific to AWS SSO. However, there are AWS condition keys that you can use as appropriate. For a complete list of AWS keys, see Available Global Condition Keys in the *IAM User Guide*.

Using Identity-Based Policies (IAM Policies) for AWS SSO

This topic provides examples of permissions policies that an account administrator can attach to IAM identities (that is, users, groups, and roles).

> **Important**
> We recommend that you first review the introductory topics that explain the basic concepts and options available for you to manage access to your AWS SSO resources. For more information, see Overview of Managing Access Permissions to Your AWS SSO Resources (p. 31).

The sections in this topic cover the following:

- Permissions Required to Use the AWS SSO Console (p. 35)
- AWS Managed (Predefined) Policies for AWS SSO (p. 35)
- Customer Managed Policy Examples (p. 36)

The following shows an example of a permissions policy.

```
{
  "Version" : "2012-10-17",
  "Statement" : [
    {
      "Action" : [
        "sso:CreateApplicationInstance",
        "sso:UpdateResponseConfig",
        "sso:UpdateResponseSchemaConfig",
        "sso:UpdateSecurityConfig",
        "sso:UpdateServiceProviderConfig",
        "sso:UpdateApplicationInstanceStatus",
        "sso:UpdateApplicationInstanceDisplay",
        "sso:CreateProfile",
        "sso:SetupTrust"
      ],
      "Effect" : "Allow",
      "Resource" : "*"
    },

    {
      "Action" : [
        "organizations:xxx",
        "organizations:yyy"
       ],
      "Effect" : "Allow",
      "Resource" : "*"
    },

    {
      "Action" : [
        "ds:AuthorizeApplication"
      ],
      "Effect" : "Allow",
      "Resource" : "*"
    }
  ]
}
```

The policy includes the following:

- The first statement grants permission to manage profile associations to users and groups within your directory. It also grants permission to read all of the AWS SSO resources.
- The second statement grants permissions to search the directory for users and groups. This is required before you can create profile associations.

The policy doesn't specify the `Principal` element because in an identity-based policy you don't specify the principal who gets the permission. When you attach a policy to a user, the user is the implicit principal. When you attach a permission policy to an IAM role, the principal identified in the role's trust policy gets the permissions.

Permissions Required to Use the AWS SSO Console

For a user to work with the AWS SSO console, that user must have permissions listed in the preceding policy.

If you create an IAM policy that is more restrictive than the minimum required permissions, the console won't function as intended for users with that IAM policy.

AWS Managed (Predefined) Policies for AWS SSO

AWS addresses many common use cases by providing standalone IAM policies that are created and administered by AWS. Managed policies grant necessary permissions for common use cases so you can avoid having to investigate what permissions are needed. For more information, see AWS Managed Policies in the *IAM User Guide*.

Customer Managed Policy Examples

In this section, you can find example user policies that grant permissions for various AWS SSO actions.

Examples

Example 1: Allow a User to Set Up and Enable AWS SSO

The following permissions policy grants permissions to allow a user to open the AWS SSO console and enable the service. In order to do so, permissions such as those granted to the AWS Organizations master account are also required.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action": [
                sso:StartSSO,
                sso:GetSSOStatus
                  ],
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action": [
                organizations:DescribeAccount,
                organizations:EnableAWSServiceAccess
                  ],
         "Resource":"*"

      }
   ]
}
```

Example 2: Allow a User to Manage Your AWS SSO Connected Directory

The following permissions policy grants permissions to a user to manage your connected directory.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action": [
                sso:AssociateDirectory,
                sso:DisassociateDirectory,
                sso:ListDirectoryAssociations,
                sso:UpdateDirectoryAssociation
                  ],
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action": [
                ds:DescribeDirectories
                  ],
         "Resource":"*"
      }
   ]
}
```

Example 3: Allow a User to Manage Applications in AWS SSO

The following permissions policy grants permissions to allow a user to create and manage application instances, profiles, and certificates in the AWS SSO console.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action": [
                sso:ListApplicationTemplates,
                sso:GetApplicationTemplate
                sso:ListApplicationInstances,
                sso:GetApplicationInstance,
                sso:CreateApplicationInstance,
                sso:UpdateApplicationInstanceStatus,
                sso:UpdateApplicationInstanceDisplayData,
                sso:UpdateApplicationInstanceServiceProviderConfiguration,
                sso:UpdateApplicationInstanceResponseConfiguration,
                sso:UpdateApplicationInstanceResponseSchemaConfiguration,
                sso:UpdateApplicationInstanceSecurityConfiguration,
                sso:DeleteApplicationInstance,
                sso:ImportApplicationInstanceServiceProviderMetadata,
                sso:CreateProfile,
                sso:UpdateProfile,
                sso:DeleteProfile,
                sso:GetProfile,
                sso:ListProfiles,
                sso:ListApplicationInstanceCertificates,
                sso:CreateApplicationInstanceCertificate,
                sso:UpdateApplicationInstanceActiveCertificate,
                sso:DeleteApplicationInstanceCertificate
                  ],
         "Resource":"*"
      }
   ]
}
```

Example 4: Allow a User to Manage Permissions for Your AWS Accounts in AWS SSO

The following permissions policy grants permissions to allow a user to create and manage permission sets for your AWS accounts in the AWS SSO console.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action": [
                sso:ListApplicationInstances,
                sso:GetApplicationInstance,
                sso:CreateApplicationInstance,
                sso:UpdateApplicationInstanceStatus,
                sso:UpdateApplicationInstanceDisplayData,
                sso:UpdateApplicationInstanceServiceProviderConfiguration,
                sso:UpdateApplicationInstanceResponseConfiguration,
                sso:UpdateApplicationInstanceResponseSchemaConfiguration,
                sso:UpdateApplicationInstanceSecurityConfiguration,
                sso:DeleteApplicationInstance,
                sso:ImportApplicationInstanceServiceProviderMetadata,
                sso:CreateProfile,
                sso:UpdateProfile,
                sso:DeleteProfile,
                sso:GetProfile,
                sso:ListProfiles,
                sso:ListApplicationInstanceCertificates,
                sso:CreateApplicationInstanceCertificate,
                sso:UpdateApplicationInstanceActiveCertificate,
                sso:DeleteApplicationInstanceCertificate,
                sso:CreatePermissionSet,
                sso:GetPermissionSet,
                sso:ListPermissionSets,
                sso:DeletePermissionSet,
                sso:PutPermissionsPolicy,
                sso:DeletePermissionsPolicy,
                sso:DescribePermissionsPolicies,
                sso:GetTrust,
                sso:CreateTrust,
                sso:UpdateTrust,
                sso:DeleteTrust
                  ],
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action": [
                organizations:DescribeOrganization
                  ],
         "Resource":"*"
      }
   ]
}
```

Example 5: Allow a User to Manage Access for Your Applications in AWS SSO

The following permissions policy grants permissions to allow a user to manage who can access your applications in the AWS SSO console.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action": [
                sso:ListApplicationInstances,
                sso:ListProfileAssociations,
                sso:AssociateProfile,
                sso:DisassociateProfile
                  ],
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action": [
                ds:DescribeDirectories
                  ],
         "Resource":"*"
      }
   ]
}
```

Example 6: Allow a User to Find Which Cloud Applications Are Preintegrated with AWS SSO

The following permissions policy grants permissions to allow a user to locate what cloud applications are preintegrated with AWS SSO using the Add Application wizard.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action": [
                sso:ListApplicationTemplates,
                sso:GetApplicationTemplate
                  ],
         "Resource":"*"
      }
   ]
}
```

Example 7: Allow a User to Add Users and Groups in AWS SSO

The following permissions policy grants permissions to allow a user to open the AWS SSO console and add users and groups in the directory that AWS SSO provides by default.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action": [
                "sso-directory:*"
                  ],
         "Resource":"*"
      }
   ]
}
```

Using Service-Linked Roles for AWS SSO

AWS Single Sign-On uses AWS Identity and Access Management (IAM) service-linked roles. A service-linked role is a unique type of IAM role that is linked directly to AWS SSO. It is predefined by AWS SSO and includes all the permissions that the service requires to call other AWS services on your behalf. For more information, see Service-Linked Roles (p. 20).

A service-linked role makes setting up AWS SSO easier because you don't have to manually add the necessary permissions. AWS SSO defines the permissions of its service-linked role, and unless defined otherwise, only AWS SSO can assume its role. The defined permissions include the trust policy and the permissions policy, and that permissions policy cannot be attached to any other IAM entity.

For information about other services that support service-linked roles, see AWS Services That Work with IAM and look for the services that have **Yes** in the **Service-Linked Role** column. Choose a **Yes** with a link to view the service-linked role documentation for that service.

Service-Linked Role Permissions for AWS SSO

AWS SSO uses the service-linked role named **AWSServiceRoleForSSO** to grant AWS SSO permissions to manage AWS resources, including IAM roles, policies, and SAML IdP on your behalf.

The AWSServiceRoleForSSO service-linked role trusts the following services to assume the role:

- `AWS SSO`

The AWSServiceRoleForSSO service-linked role permissions policy allows AWS SSO to complete the following on roles on the path "/aws-reserved/sso.amazonaws.com/" and with the name prefix "AWSReservedSSO_":

- `iam:AttachRolePolicy`
- `iam:CreateRole`
- `iam:DeleteRole`
- `iam:DeleteRolePolicy`
- `iam:DetachRolePolicy`
- `iam:GetRole`
- `iam:ListRolePolicies`
- `iam:PutRolePolicy`
- `iam:ListAttachedRolePolicies`

The AWSServiceRoleForSSO service-linked role permissions policy allows AWS SSO to complete the following on SAML providers with name prefix as "AWSSSO_":

- `iam:CreateSAMLProvider`
- `iam:GetSAMLProvider`
- `iam:UpdateSAMLProvider`
- `iam:DeleteSAMLProvider`

The AWSServiceRoleForSSO service-linked role permissions policy allows AWS SSO to complete the following on all organizations:

- `organizations:DescribeAccount`
- `organizations:DescribeOrganization`
- `organizations:ListAccounts`

The AWSServiceRoleForSSO service-linked role permissions policy allows AWS SSO to complete the following on all IAM roles (*):

- `iam:listRoles`

The AWSServiceRoleForSSO service-linked role permissions policy allows AWS SSO to complete the following on “arn:aws:iam::*:role/aws-service-role/sso.amazonaws.com/AWSServiceRoleForSSO”:

- `iam:GetServiceLinkedRoleDeletionStatus`
- `iam:DeleteServiceLinkedRole`

You must configure permissions to allow an IAM entity (such as a user, group, or role) to create, edit, or delete a service-linked role. For more information, see Service-Linked Role Permissions in the *IAM User Guide*.

Creating a Service-Linked Role for AWS SSO

You don't need to manually create a service-linked role. When a user who is signed in with the AWS organization’s master account assigns access to an AWS account for the first time, AWS SSO creates the service-linked role automatically in that AWS account.

> **Important**
> If you were using the AWS SSO service before December 7, 2017, when it began supporting service-linked roles, then AWS SSO created the AWSServiceRoleForSSO role in your account. To learn more, see A New Role Appeared in My IAM Account.

If you delete this service-link role and then need to create it again, you can use the same process to recreate the role in your account.

Editing a Service-Linked Role for AWS SSO

AWS SSO does not allow you to edit the AWSServiceRoleForSSO service-linked role. After you create a service-linked role, you cannot change the name of the role because various entities might reference the role. However, you can edit the description of the role using IAM. For more information, see Editing a Service-Linked Role in the *IAM User Guide*.

Deleting a Service-Linked Role for AWS SSO

You don't need to manually delete the AWSServiceRoleForSSO role. When an AWS account is removed from an AWS organization, AWS SSO automatically cleans up the resources and deletes the service-linked role from that AWS account.

You can also use the IAM console, the IAM CLI, or the IAM API to manually delete the service-linked role. To do this, you must first manually clean up the resources for your service-linked role and then you can manually delete it.

> **Note**
> If the AWS SSO service is using the role when you try to delete the resources, then the deletion might fail. If that happens, wait for a few minutes and try the operation again.

To delete AWS SSO resources used by the AWSServiceRoleForSSO

1. Remove User Access (p. 15) for all users and groups that have access to the AWS account.
2. Delete Permission Sets (p. 19) that you have associated with the AWS account.
3. Remove the IAM Identity Provider (p. 19) to delete the trust between AWS SSO and the AWS account.

To manually delete the service-linked role using IAM

Use the IAM console, the IAM CLI, or the IAM API to delete the AWSServiceRoleForSSO service-linked role. For more information, see Deleting a Service-Linked Role in the *IAM User Guide*.

Using the User Portal

Your user portal provides you with single sign-on access to all your AWS accounts and most commonly used cloud applications such as Office 365, Concur, Salesforce, and many more. From here you can quickly launch multiple applications simply by choosing the AWS account or application icon in the portal. The presence of icons in your portal means that an administrator or designated help desk employee from your company has granted you access to those AWS accounts or applications. It also means that you can access all these accounts or applications from the portal without additional sign-in prompts.

Contact your administrator or help desk to request additional access in the following situations:

- You don't see an AWS account or application that you need access to.
- The access that you have to a given account or application is not what you expected.

Topics

Tips for Using the Portal

Like any business tool or application that you use on a daily basis, the user portal might not work as you expected. If that happens, try these tips:

- Occasionally, you may need to sign out and sign back in to the user portal. This might be necessary to access new applications that your administrator recently assigned to you. This is not required, however, because all new applications are refreshed every hour.
- When you sign in to the user portal, you can open any of the applications listed in the portal by choosing the application’s icon. After you are done using the application, you can either close the application or sign out of the user portal. Closing the application signs you out of that application only. Any other applications that you have opened from the user portal remain open and running.
- Before you can sign in as a different user, you must first sign out of the user portal. Signing out from the portal completely removes your credentials from the browser session.

How to Accept the Invitation to Join AWS SSO

If this is your first time signing into the user portal, check your email for instructions on how to activate your account.

To activate your account

1. Depending on the email you received from your company, choose one of the following methods to activate your account so that you can start using the user portal.

   a. If you received an email with the subject **Invitation to join AWS Single Sign-On**, open it and choose **Accept invitation**, which takes you to the **Single Sign-On** page. Here you specify a password, which you use each time you sign in to the portal. Once you have provided a password and have confirmed it, choose **Update User**.
   b. If you were sent an email from your company's IT support or IT administrator, follow the instructions they provided to activate your account.
2. Once you activate your account by providing a new password, the user portal signs you in automatically. If this does not occur, you can manually sign in to the user portal using the instructions provided in the next step.

How to Sign In to the User Portal

By this time, you should have been provided a specific sign-in URL to the user portal by an administrator or help desk employee. Once you have this, you can proceed with the following steps to sign in to the portal.

To sign in to the user portal

1. In your browser window, paste in the sign-in URL that you were provided. Then press **Enter**. We recommend that you bookmark this link to the portal now so that you can quickly access it later.
2. Sign in using your standard company user name and password. If you are prompted for a **Verification code**, check your email and then copy and paste the code into the sign-in page.

   Note
   Verification codes are typically sent through email, but the delivery method can vary. Check with your administrator for details.
3. Once signed in, you can access any AWS account and application that appears in the portal. Simply choose an icon.

How to Sign Out of the User Portal

When you sign out from the portal, your credentials are completely removed from the browser session.

> **Note**
> If you want to sign in as a different user, you must first sign out of the user portal.

To sign out of the user portal

- In the user portal, choose **Sign out** from the upper right corner of the portal.

How to Search for an AWS Account or Application

If your list of applications or AWS accounts is too large to find what you need, you can use the **Search** box.

To search for an AWS account or application in the user portal

1. While signed into the portal, choose the **Search** box.
2. Type the name of the application. Then press **Enter**.

How to Reset Your Password

From time to time you may need to reset your password, depending on your company policies.

To reset your password

1. Open a browser and go to the sign-in page for your user portal.
2. Under the **Sign In** button, choose **Forgot Password?**.
3. Provide your **Username** and type the characters for the provided image to confirm that you are not a robot. Then choose **Recover Password**. This sends an email to you with the subject **AWS Directory Service Reset Password Request**.
4. Once you receive the email, choose **Reset Password**.
5. On the **Single Sign-On** page, need to specify a new password for the portal. Once you have provided a password and have confirmed it, choose **Reset Password**.

How to Get Credentials of an IAM Role for Use with CLI Access to an AWS Account

Use this procedure in the user portal when you need temporary security credentials for short-term access to resources in an AWS account using the AWS CLI. The user portal makes it easy for you to quickly select an AWS account and get the temporary credentials for a given IAM role. You can then copy the necessary CLI syntax (including all necessary credentials) and paste them into your AWS CLI command prompt.

By default, credentials retrieved through the user portal are valid for 1 hour. You can change this value up to 12 hours. Once you have completed this procedure, you can run any AWS CLI command (that your administrator has given you access to) until those temporary credentials have expired.

> **Note**
> Before you get started with the steps in this procedure, you must first install the AWS CLI. For instructions, see Installing the AWS Command Line Interface.

To get temporary credentials of an IAM role for CLI access to an AWS account

1. While signed into the portal, choose the **AWS Accounts** icon to expand the list of accounts.
2. Choose the AWS account from which you want to retrieve access credentials. Then, next to the IAM role name (for example **Administrator**), choose **Command line or programmatic access**.
3. In the **Get credentials** dialog box, choose either **MacOS and Linux** or **Windows**, depending on the operating system where you plan to use the CLI command prompt.
4. Depending on how you want to use the temporary credentials, choose one or more of the following options:

   - If you need to run commands from the AWS CLI in the selected AWS account, under **Option 1: Set AWS environment variables**, pause on the commands. Then choose **Copy**. Paste the commands into the CLI terminal window and press **Enter** to set the necessary environment variables.
   - If you need to run commands from multiple command prompts in the same AWS account, under **Option 2: Add a profile to your AWS credentials file**, pause on the commands. Then choose **Copy**. Paste the commands into your AWS credentials file to set up a newly named profile. For more information, see Configuration and Credential Files in the *AWS CLI User Guide*. Modifying the credential files in this way enables the `--profile` option in your AWS CLI command so that you can use this credential. This affects all command prompts that use the same credential file.
   - If you need to access AWS resources from an AWS service client, under **Option 3: Use individual values in your AWS service client**, choose **Copy** next to the commands you need to use. For more

information, see Getting Temporary Credentials with AWS STS in the *AWS CLI User Guide* or see Tools for Amazon Web Services.

5. Continue using the AWS CLI as necessary for your AWS account until the credentials have expired.

Enable Two-Step Verification

By default, when a user signs in to the user portal, they sign in with their email address and password (the first step). This is the standard authentication mechanism used in AWS SSO. But if the two-step verification option is enabled, users receive a verification code (the second step) sent to their assigned email address. Users must use this verification code to be authenticated to the user portal. These factors together provide additional security by preventing access to your user portal unless users supply valid user credentials and a valid verification code.

Note
When a verification code is sent to a user, it is only valid for 10 minutes.

Considerations Before Using Two-Step Verification in AWS SSO

Before you enable email-based verification as your two-step verification method, consider the following information:

- All users must have first verified their email address before they can begin using email-based verification during sign-in.
- Do not enable two-step verification if your users require signing in to the user portal to access to their email. For example, your users might use Office 365 on the user portal to read their email. In this case, users would not be able to retrieve the verification code and would be unable to sign in to the user portal.
- If you are already using RADIUS MFA that you configured with AWS Directory Service, then you do not need to enable two-step verification within AWS SSO. Two-step verification is an alternative to RADIUS MFA for Microsoft Active Directory users of AWS SSO. For more information, see About RADIUS MFA (p. 48).

Verification Modes

Verification modes help you determine the level of security you want to enforce across all your users during sign-in. The default mode when you first configure AWS SSO is **Disabled**. While in this mode, no two-step verification method is enabled so users continue to sign in using their user name, password and/or RADIUS MFA as normal. You can use either of the following modes to enable two-step verification:

- **Context-aware**
- **Always-on**

Note
If you have configured AWS SSO to use a connected directory and decide to enable either **Context-aware** or **Always-on** mode, your users must sign in to the user portal using the down-level logon name format (DOMAIN\UserName). This restriction does not apply when you are using an AWS SSO directory. With an AWS SSO directory, users can sign in using either their down-level logon name format or their UPN logon name format (UserName@Corp.Example.com). For general information about sign in formats, see User Name Formats on the Microsoft documentation website.

Context-aware

In this mode, AWS SSO analyzes the sign-in context (browser, location, and devices) for each user to determine whether the user is signing in with a previously trusted context. If a user is signing in from an unknown location or is using an unknown device, SSO prompts the user to verify their second-factor of authentication. The user is prompted for a verification code in addition to their email address and password credentials.

This mode provides additional protection while making it easier for users who frequently sign in from their offices because they do not need to complete two-step verification on every sign-in. SSO prompts users with two-step verification during initial sign-ins to create a baseline for successful sign-ins. Once a stable baseline is established, AWS SSO uses the baseline to determine a “trusted” sign-in and does not challenge users for a verification code. Users are only required to provide additional verification when their sign-in context changes. Such changes include signing in from a new device, a new browser, or an unknown location.

> **Note**
> Changing from **Disabled** mode to **Context-aware** mode overrides existing RADIUS MFA settings that are configured in AWS Directory Service while signing in to AWS SSO for this directory. For more information, see About RADIUS MFA (p. 48).

Always-on

In this mode, AWS SSO requires that all users provide a verification code on every sign-in. You should use this mode if you have organizational or compliance policies that require your users to complete two-step verification every time they sign in to the user portal. For example, PCI DSS strongly recommends two-step verification during every sign-in to access applications that support high-risk payment transactions.

About RADIUS MFA

Remote Authentication Dial-In User Service (RADIUS) is an industry-standard client–server protocol that provides authentication, authorization, and accounting management to enable users to connect to network services. AWS Directory Service includes a RADIUS client that connects to the RADIUS server upon which you have implemented your MFA solution. For more information, see Enable Multi-Factor Authentication for AWS Managed Microsoft AD .

RADIUS MFA and AWS SSO

You can use either RADIUS MFA or two-step verification in AWS SSO for user sign-ins to the user portal, but not both. Two-step verification in AWS SSO is an alternative to RADIUS MFA in cases where you are looking for AWS native two-factor authentication for access to the portal.

When you enable two-step verification in AWS SSO, your users require a verification code when they sign in to the SSO user portal. If you had previously used RADIUS MFA, enabling two-step verification in AWS SSO effectively overrides RADIUS MFA for users who sign into the AWS SSO user portal. However, RADIUS MFA continues to challenge users when they sign in to all other apps that work with AWS Directory Service, such as Amazon WorkDocs.

If your two-step verification is **Disabled** on the AWS SSO console and you have configured RADIUS MFA with AWS Directory Service, RADIUS MFA governs user portal sign-in. This means that AWS SSO falls back to RADIUS MFA configuration if two-step verification is disabled.

Topics

- How to Disable Two-Step Verification (p. 49)

How to Enable Two-Step Verification

Use the following procedure to enable two-step verification in the AWS SSO console. Before you enable two-step verification, we recommend that you first review details about Verification Modes (p. 47).

> **Note**
> If your users can only access their email through AWS SSO, then you should not enable two-step verification. For more information, see Considerations Before Using Two-Step Verification in AWS SSO (p. 47).

To enable two-step verification

1. Open the AWS SSO console.
2. In the left navigation pane, choose **Settings**.
3. On the **Settings** page, choose **Configure**.
4. On the **Configure two-step verification** page, choose one of the following Verification Modes (p. 47) based on your business needs:
   - **Context-aware**
   - **Always-on**
5. Choose **Save Changes**.

How to Disable Two-Step Verification

Use the following procedure to disable two-step verification in the AWS SSO console.

To disable two-step verification

1. Open the AWS SSO console.
2. In the left navigation pane, choose **Settings**.
3. On the **Settings** page, choose **Configure**.
4. On the **Configure two-step verification** page, choose **Disabled**.
5. Choose **Save Changes**.

Logging AWS SSO API Calls with AWS CloudTrail

AWS SSO is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service in AWS SSO. If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, Amazon CloudWatch Logs, and Amazon CloudWatch Events. Using the information collected by CloudTrail, you can determine the request that was made to AWS SSO, the IP address from which the request was made, who made the request, when it was made, and additional details.

To learn more about CloudTrail, see the AWS CloudTrail User Guide.

AWS SSO Information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in AWS SSO, that activity is recorded in a CloudTrail event along with other AWS service events in **Event history**. You can view, search, and download recent events in your AWS account. For more information, see Viewing Events with CloudTrail Event History.

For an ongoing record of events in your AWS account, including events for AWS SSO, create a trail. A *trail* enables CloudTrail to deliver log files to an Amazon S3 bucket. By default, when you create a trail in the console, the trail applies to all AWS Regions. The trail logs events from all Regions in the AWS partition and delivers the log files to the Amazon S3 bucket that you specify. Additionally, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see the following:

- Overview for Creating a Trail
- CloudTrail Supported Services and Integrations
- Configuring Amazon SNS Notifications for CloudTrail
- Receiving CloudTrail Log Files from Multiple Regions and Receiving CloudTrail Log Files from Multiple Accounts

When CloudTrail logging is enabled in your AWS account, API calls made to AWS SSO actions are tracked in log files. AWS SSO records are written together with other AWS service records in a log file. CloudTrail determines when to create and write to a new file based on a time period and file size.

The following actions are supported:

- `AssociateDirectory`
- `AssociateProfile`
- `CreateApplicationInstance`
- `CreateApplicationInstanceCertificate`
- `CreatePermissionSet`
- `CreateProfile`
- `DeleteApplicationInstance`
- `DeleteApplicationInstanceCertificate`
- `DeletePermissionsPolicy`
- `DeletePermissionSet`

- `DeleteProfile`
- `DescribePermissionsPolicies`
- `DisassociateDirectory`
- `DisassociateProfile`
- `GetApplicationInstance`
- `GetApplicationTemplate`
- `GetPermissionSet`
- `GetSSOStatus`
- `ImportApplicationInstanceServiceProviderMetadata`
- `ListApplicationInstances`
- `ListApplicationInstanceCertificates`
- `ListApplicationTemplates`
- `ListDirectoryAssociations`
- `ListPermissionSets`
- `ListProfileAssociations`
- `ListProfiles`
- `PutPermissionsPolicy`
- `StartSSO`
- `UpdateApplicationInstanceActiveCertificate`
- `UpdateApplicationInstanceDisplayData`
- `UpdateApplicationInstanceServiceProviderConfiguration`
- `UpdateApplicationInstanceStatus`
- `UpdateApplicationInstanceResponseConfiguration`
- `UpdateApplicationInstanceResponseSchemaConfiguration`
- `UpdateApplicationInstanceSecurityConfiguration`
- `UpdateDirectoryAssociation`
- `UpdateProfile`

Every log entry contains information about who generated the request. The identity information in the log helps you determine whether the request was made by an AWS account root user or with IAM user credentials. You can also learn whether the request was made with temporary security credentials for a role or federated user or by another AWS service. For more information, see the CloudTrail userIdentity Element.

You can create a trail and store your log files in your Amazon S3 bucket for as long as you want. You can also define Amazon S3 lifecycle rules to archive or delete log files automatically. By default, your log files are encrypted with Amazon S3 server-side encryption (SSE).

To be notified of log file delivery, configure CloudTrail to publish Amazon SNS notifications when new log files are delivered. For more information, see Configuring Amazon SNS Notifications for CloudTrail.

You can also aggregate AWS SSO log files from multiple AWS regions and multiple AWS accounts into a single Amazon S3 bucket. For more information, see Receiving CloudTrail Log Files from Multiple Regions and Receiving CloudTrail Log Files from Multiple Accounts.

Understanding AWS SSO Log File Entries

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. An event represents a single request from any source and includes information about the requested action, the date and time of the action, request parameters, and so on. CloudTrail log files are not an ordered stack trace of the public API calls, so they do not appear in any specific order.

The following example shows a CloudTrail log entry for an administrator (samadams@example.com) that took place in the AWS SSO console:

```
{
   "Records":[
      {
         "eventVersion":"1.05",
         "userIdentity":{
            "type":"IAMUser",
            "principalId":"AIDAJAIENLMexample",
            "arn":"arn:aws:iam::08966example:user/samadams",
            "accountId":"08966example",
            "accessKeyId":"AKIAIIJM2K4example",
            "userName":"samadams"
         },
         "eventTime":"2017-11-29T22:39:43Z",
         "eventSource":"sso.amazonaws.com",
         "eventName":"DescribePermissionsPolicies",
         "awsRegion":"us-east-1",
         "sourceIPAddress":"203.0.113.0",
         "userAgent":"Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36
 (KHTML, like Gecko) Chrome/62.0.3202.94 Safari/537.36",
         "requestParameters":{
            "permissionSetId":"ps-79a0dde74b95ed05"
         },
         "responseElements":null,
         "requestID":"319ac6a1-d556-11e7-a34f-69a333106015",
         "eventID":"a93a952b-13dd-4ae5-a156-d3ad6220b071",
         "readOnly":true,
         "resources":[

         ],
         "eventType":"AwsApiCall",
         "recipientAccountId":"08966example"
      }
   ]
}
```

The following example shows a CloudTrail log entry for an end-user (bobsmith@example.com) action that took place in the AWS SSO user portal:

```
{
   "Records":[
      {
         "eventVersion":"1.05",
         "userIdentity":{
            "type":"Unknown",
            "principalId":"example.com//S-1-5-21-1122334455-3652759393-4233131409-1126",
            "accountId":"08966example",
            "userName":"bobsmith@example.com"
         },
         "eventTime":"2017-11-29T18:48:28Z",
         "eventSource":"sso.amazonaws.com",
         "eventName":"https://portal.sso.us-east-1.amazonaws.com/instance/appinstances",
         "awsRegion":"us-east-1",
         "sourceIPAddress":"203.0.113.0",
         "userAgent":"Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36
 (KHTML, like Gecko) Chrome/62.0.3202.94 Safari/537.36",
         "requestParameters":null,
         "responseElements":null,
         "requestID":"de6c0435-ce4b-49c7-9bcc-bc5ed631ce04",
         "eventID":"e6e1f3df-9528-4c6d-a877-6b2b895d1f91",
         "eventType":"AwsApiCall",
         "recipientAccountId":"08966example"
      }
   ]
}
```

AWS SSO Region Availability

AWS SSO is available in several commonly used AWS Regions. This availability makes it easier for you to configure user access to multiple AWS accounts and business applications. When your users sign in to the user portal, they can select the AWS account that they have permission to. Then they can access the AWS Management Console. For a full list of the Regions that AWS SSO supports, see AWS Regions and Endpoints.

AWS SSO Region Data

When you first enable AWS SSO, all the data that you configure in AWS SSO is stored in the Region where you configured it. This data includes directory configurations, permission sets, application instances, and user assignments to AWS account applications. If you are using the AWS SSO directory, all users and groups that you create in AWS SSO are also stored in the same Region.

AWS Organizations only supports one AWS SSO Region at a time. If you want to make AWS SSO available in a different Region, you must first delete your current AWS SSO configuration. Switching to a different Region also changes the URL for the user portal.

Delete Your AWS SSO Configuration

When an AWS SSO configuration is deleted, all the data in that configuration is deleted and cannot be recovered. The following table describes what data is deleted based on the directory type that you have currently configured in AWS SSO.

What data gets deleted	Connected directory (AWS Managed Microsoft AD or AD Connector)	AWS SSO directory
All permission sets you have configured for AWS accounts	**X**	**X**
All applications you have configured in AWS SSO	**X**	**X**
All user assignments you have configured for AWS accounts and applications	**X**	**X**
All users and groups in the directory	**N/A**	**X**

Use the following procedure when you need to delete your current AWS SSO configuration.

To delete your AWS SSO configuration

1. Open the AWS SSO console.
2. In the left navigation pane, choose **Settings**.

3. On the **Settings** page, under **Delete AWS SSO configuration**, choose **Delete AWS SSO**.
4. On the **Delete AWS SSO configuration** page, select each of the check boxes to acknowledge you understand the data that will be deleted. Type `DELETE` in the text box, and then choose **Delete AWS SSO**.

Limits in AWS SSO

The following tables describe limits within AWS SSO.

Application Limits

Resource	Default Limit
File size of service provider SAML certificates (in PEM format)	2 kb

AWS Account Limits

Resource	Default Limit
Maximum number of permission sets in AWS SSO	500
Number of permission sets allowed per AWS account	20
Number of references to AWS managed policies per permission set	10
Number of inline policies per permission set	1
Maximum size of inline policy per permission set	10,000 bytes
Number of IAM roles in the AWS account that can be repaired at a time *	1
Number of directories that you can have at a time	1

* Permission sets are provisioned in an AWS account as IAM roles. For more information, see Permission Sets (p. 5).

Connected Directory Limits

Resource	Default Limit
Number of unique Active Directory groups that can be assigned *	1500
Number of connected directories that you can have at a time	1

* Users within their Active Directory can belong to many directory groups. However within AWS SSO, they can have up to 1500 of their Active Directory groups assigned for using applications.

AWS SSO Directory Limits

Resource	Default Limit
Number of unique groups that can be assigned *	100
Number of AWS SSO directories that you can have at a time	1
Maximum number of users supported in AWS SSO	500
Maximum number of groups supported in AWS SSO	100

* Users within an AWS SSO directory can have up to 100 of their groups assigned for using applications.

Additional Limits

Resource	Default Limit
Total number of AWS accounts or applications that can be configured *	500
Maximum number of unique groups that can be used to evaluate the permissions for a user **	500

* Only 500 AWS accounts or applications (total combined) are supported. For example, you might configure 275 accounts and 225 applications, resulting in a total of 500 accounts and applications.

** Before displaying the user's available AWS accounts and application icons in the user portal, AWS SSO evaluates the user's effective permissions by evaluating their group memberships. Only 500 unique groups can be used to determine a user's effective permissions.

Troubleshooting AWS SSO Issues

The following can help you troubleshoot some common issues you might encounter while setting up or using the AWS SSO console.

Users can’t sign in when their user name is in UPN format

Users might not be able to sign in to the user portal based on the format they use to enter in their user name on the sign in page. For the most part, users can sign in to the user portal using either their plain user name, their down-level logon name (DOMAIN\UserName) or their UPN logon name (UserName@Corp.Example.com). The exception to this is when AWS SSO is using a connected directory that has been enabled with two-step verification and the verification mode has been set to either **Context-aware** or **Always-on** . In this scenario, users must sign in with their down-level logon name (DOMAIN\UserName). For more information, see Verification Modes. For general information about user name formats used to sign in to Active Directory, see User Name Formats on the Microsoft documentation website.

I cannot get my cloud application configured correctly

Each service provider of a preintegrated cloud application in AWS SSO has its own detailed instruction manual. You can access the manual from the **Configuration** tab for that application in the AWS SSO console.

If the problem is related to setting up the trust between the service provider's application and AWS SSO, make sure to check the instruction manual for troubleshooting steps.

I don't know what data is in my SAML assertion that would be passed to the service provider

Use the following steps in the user portal to view what data in the SAML assertion will be sent to the application's service provider for the currently signed-in user. This procedure displays the contents in the browser window before sending it to the provider.

1. While you are signed into the portal, hold the **Shift** key and then choose the application.
2. Examine the information on the page titled **You are now in administrator mode**.
3. If the information looks good, you can choose **Send to <application>** to send the assertion to the service provider and review the outcome of the response.

Document History

The following table describes the documentation for this release of AWS Single Sign-On.

- **Latest documentation update:** January 16, 2019

update-history-change	update-history-description	update-history-date
New setting to add two-step verification	Added content on how to enable two-step verification for users.	January 16, 2019
Support for session duration on AWS accounts	Added content on how to set the session duration for an AWS account.	October 30, 2018
New option to use AWS SSO directory	Added content for choosing either AWS SSO directory or connecting to an existing AD directory.	October 17, 2018
Support for relay state and session duration on applications	Added content about relay state and session duration for cloud applications.	October 10, 2018
Additional support for new cloud applications	Added 4me, BambooHR, Bonusly, Citrix ShareFile, ClickTime, Convo, Deputy, Deskpro, Dome9, DruvaInSync, Egnyte, Engagedly, Expensify, Freshdesk, IdeaScale, Igloo, Jitbit, Kudos, LiquidFiles, Lucidchart, PurelyHR, Samanage, ScreenSteps, Sli.do, SmartSheet, Syncplicity, TalentLMS, Trello, UserVoice, Zoho, OpsGenie, DigiCert, WeekDone, ProdPad, and UserEcho to the application catalog.	August 3, 2018
Support for SSO access to master accounts	Added content about how to delegate SSO access to users in a master account.	July 9, 2018
Support for new cloud applications	Added DocuSign, Keeper Security, and SugarCRM to the application catalog.	March 16, 2018
Get temporary credentials for CLI access	Added information about how to get temporary credentials to run AWS CLI commands.	February 22, 2018
New guide	This is the first release of the AWS SSO User Guide.	December 7, 2017

AWS Glossary

For the latest AWS terminology, see the AWS Glossary in the *AWS General Reference*.