

Azure Active Directory **Single Sign-On** Deployment Plan

**How to use this guide**

This step-by-step guide walks through the implementation of Single Sign On in a five-step process. The links below take you to each of those steps.



**1**

**[Include](#_Stakeholders_and_Sign-off)**

[Stakeholders](#_Stakeholders_and_Sign-off)

**2**

**[Plan](#_Planning_Your_Implementation)**

[Your project](#_Planning_Your_Implementation)

**3**

**[Design](#_Design)**

[Policies and integration](#_Design)

**5**

**[Manage](#_Operational_Doc_1)**

[Your implementation](#_Operational_Doc_1)



**4**

**[Implement](#_Implementing_Your_Solution)**

[Your design](#_Implementing_Your_Solution)

**Note:**

Throughout this document, you will see items marked as

* **Microsoft Recommends**

These are general recommendations, and you should only implement if they apply to your specific enterprise needs.

Table of Contents

Contents

[Business Value of Single Sign-On 3](#_Toc517766419)

[Planning Your Implementation 4](#_Toc517766420)

[Stakeholders and Sign-off 4](#_Toc517766421)

[General Planning 5](#_Toc517766422)

[Tracking Timelines 5](#_Toc517766423)

[In Scope 5](#_Toc517766424)

[Out of scope 5](#_Toc517766425)

[Licensing 6](#_Toc517766426)

[Planning Single Sign-on 7](#_Toc517766427)

[Considerations for Password-based Single Sign on 7](#_Toc517766428)

[Planning Reporting and Auditing 8](#_Toc517766429)

[Planning Your Security Review 9](#_Toc517766430)

[Designing Your Implementation 9](#_Toc517766431)

[Designing Single Sign on 10](#_Toc517766432)

[Endpoint Information 10](#_Toc517766433)

[Solution Architecture Diagrams and Description 12](#_Toc517766434)

[Azure AD Single Sign-on with Password Hash Sync or cloud-based users Authentication 12](#_Toc517766435)

[Azure AD Single Sign-on with AD FS or other Federation as IdP 13](#_Toc517766436)

[Azure AD Single Sign-on with Pass-through Authentication 13](#_Toc517766437)

[Technical Requirements 15](#_Toc517766438)

[Azure Single Sign-on Requirements 15](#_Toc517766439)

[Attribute Requirements 15](#_Toc517766440)

[Certificate Requirements 16](#_Toc517766441)

[Implementing Your Solution 17](#_Toc517766442)

[Phase 1: Implementation Steps 17](#_Toc517766443)

[Step 1: Identify your Test Users 17](#_Toc517766444)

[Step 2: Configure Azure Single Sign-on 17](#_Toc517766445)

[Phase 2: Change Communications 18](#_Toc517766446)

[Step 1: Provide Internal Change Communication to end users 18](#_Toc517766447)

[Phase 3: Verify End User Scenario for SSO 18](#_Toc517766448)

[Step 1: Create test cases for your application deployment 18](#_Toc517766449)

[Step 2: Document your results 21](#_Toc517766450)

[Step 3: Moving into Production 21](#_Toc517766451)

[Phase 4: Rollback Steps 21](#_Toc517766452)

[Step 1: Identify available options for rolling back during migration or failure 21](#_Toc517766453)

[Operationalize your Implementation 22](#_Toc517766454)

[Purpose of Document 22](#_Toc517766455)

[Required Roles 22](#_Toc517766456)

[Single Sign-on Certificate Lifecycle [Azure Active Directory] 23](#_Toc517766457)

[Access Management 23](#_Toc517766458)

[Troubleshooting Guide & Steps 25](#_Toc517766459)

[Example: Single account not being able to log into the application. 25](#_Toc517766460)

[Example: Complete outage of <<APPLICATION NAME>> - No user can sign in 27](#_Toc517766461)

[Helpful Documentation 28](#_Toc517766462)

# Business Value of Single Sign-On

This document presents an executive summary of the business case for moving forward with enabling Azure Active Directory Single Sign-on (SSO) for application «APPLICATIONNAME» (“The Application”). Many organizations rely upon software as a service (SaaS) applications such as Office 365, Box and Salesforce for end user productivity. Historically, IT staff needs to individually create and update user accounts in each SaaS application, and users must remember a password for each SaaS application. Single sign-on means being able to access all the applications and resources that a user needs to do business, by signing in only once using a single user account. Once signed in, the user can access all needed applications without being required to authenticate (e.g. type a password) a second time.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | \\MAGNUM\Projects\Microsoft\Cloud Power FY12\Design\ICONS_PNG\Increase.png | **INCREASE PRODUCTIVITY**  Enabling **single sign-on** across enterprise applications and Office 365 provides a superior log in experience for existing users, reducing or eliminating log on prompts. The user’s environment feels more cohesive and is less distracting without multiple prompts, or the need to manage multiple passwords. Access control can be managed and approved by the business group, saving IT management costs through self-service and dynamic membership, and improving the overall security of our identity system by ensuring the right people in the business manage access to this application. | | \\MAGNUM\Projects\Microsoft\Cloud Power FY12\Design\ICONS_PNG\Confidentiality.png | **MANAGE RISK**  Coupling Azure AD SSO with **conditional access** policies can offer significantly improved security experiences. These include cloud-scale identity protection, risk-based access control capabilities, native multi-factor authentication support, and conditional access policies which allow for granular control policies based on applications, or on groups that need higher levels of security. | | C:\Users\mitchellg\Desktop\Simple_Licensing.png | **ADDRESS COMPLIANCE AND GOVERNANCE**  Auditing access requests and approvals for the application, as well as understanding overall application usage, becomes easier with Azure Active Directory, which supports native audit logs for every application access request performed. Auditing includes requester identity, requested date, business justification, approval status, and approver identity. This data is also available from an API, which will enables importing this data into a Security Incident and Event Monitoring (SIEM) system of choice. | | \\MAGNUM\Projects\Microsoft\Cloud Power FY12\Design\ICONS_PNG\Within_Your_Reach.png | **MANAGE COST**  Replacing current access management and provisioning process and migration to Azure Active Directory to manage self-service access to the application (as well as other SaaS applications in the future) will allow for significant cost reductions related to running, managing, and maintaining our current infrastructure. Additionally, eliminating application specific passwords eliminates costs related to password reset for that application, and lost productivity while retrieving passwords. | |  |

# Planning Your Implementation

## Stakeholders and Sign-off

The following section serves to identify all the stakeholders that are involved in the project and need to sign off, review, or stay informed. Add stakeholders to the table below as appropriate for your organization.

* + SO = Sign-off on this project
  + R = Review this project and provide input
  + I = Informed of this project

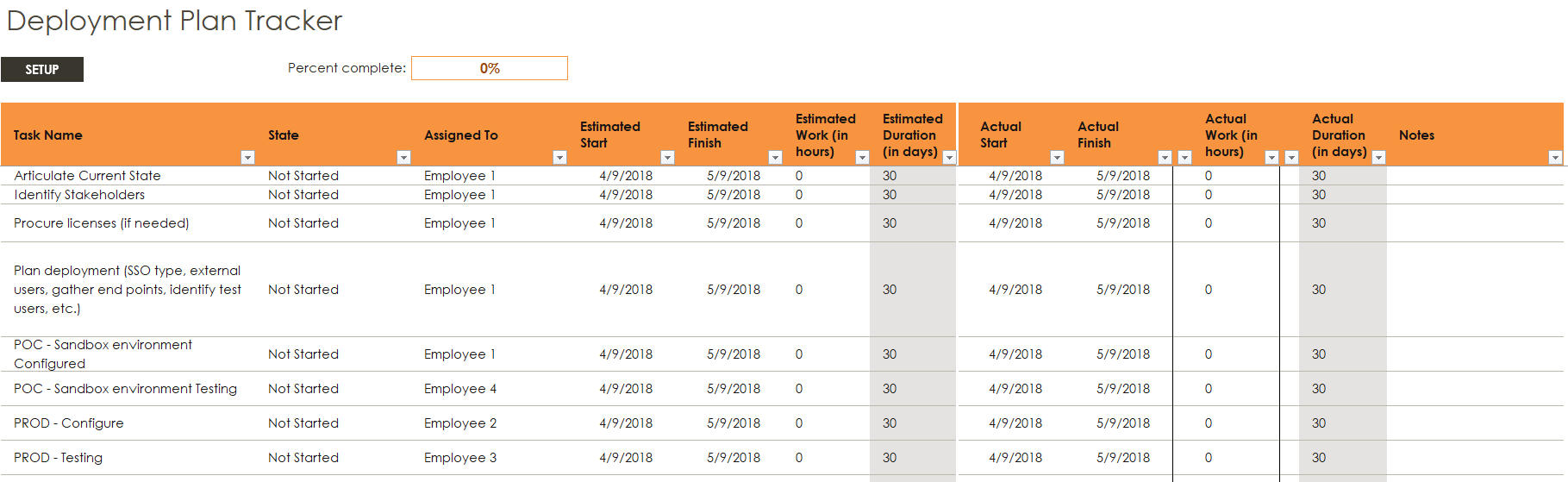
|  |  |  |
| --- | --- | --- |
| Name | Role | Action |
| Enter name and email | **IT Support Manager**  *A representative from the IT support organization who can provide input on the supportability of this change from a helpdesk perspective.* | SO |
| Enter name and email | **Identity Architect or Azure Global Administrator**  *A representative from the identity management team in charge of defining how this change is aligned with the core identity management infrastructure in the customer’s organization.* | SO |
| Enter name and email | **Application Business Owner**  *A representative colleague who can provide input on the user experience and usefulness of this change from an end-user’s perspective and owns the overall business aspect of the application, which may include managing access.* | SO/I |
| Enter name and email | **Security Owner**  *A representative from the security team that can sign off that the plan will meet the security requirements of your organization.* | SO |

## General Planning

### Tracking Timelines

Tracking your plan is an important aspect of project success.

You may use the embedded Deployment Plan Tracker spreadsheet below to monitor and schedule your committed timelines for this project. Begin tracking additional items as you progress through the deployment plan that may require an action or prerequisite:



### In Scope

The following is in scope for this project:

**Single Sign-on**



* Enabling single sign-on to the application using Microsoft Azure Active Directory federation technologies.
* Extension of on-premises AD to include new attributes which will be provisioned to the Azure AD or application environments.
* Enabling the support organization to support and manage this new change, ensuring the right helpdesk processes are in place to ensure on-going end-user success.
* Documenting and testing a recovery plan.
* Approving a business continuity plan.
* Designing operational support for the production service.
* The following environments are in scope for this design:
  + Production
  + Test / QA

### Out of scope

The following are out of scope of this project:

* Enabling any other application for federated single-sign or provisioning.
* Extending the corporate Active Directory system with any additional or new attributes that are require by the application. Any new attributes necessary will be created in Azure Active Directory.
* Disabling of the existing federation relationship between the application and our corporate federation solution.

### Licensing

#### Azure Active Directory Licensing

Single Sign On for pre-integrated SaaS applications is free! However, the number of objects in your directory and the features you wish to deploy may require additional licenses. Common Azure AD scenarios include the following security features:

* [Conditional Access (CA)](https://docs.microsoft.com/azure/active-directory/active-directory-conditional-access-azure-portal) (P1 Required)
* [Azure Multi-Factor Authentication (MFA)](https://docs.microsoft.com/azure/multi-factor-authentication/multi-factor-authentication-how-it-works) (P1 Required)
* [Group based membership](https://docs.microsoft.com/azure/active-directory/active-directory-manage-groups) (P1 required)
* [Identity Protection](https://docs.microsoft.com/azure/active-directory/active-directory-identityprotection) (P2 Required)

The following table describes some of the license requirements that may be relevant. For a full list of license requirements, click [here](https://azure.microsoft.com/pricing/details/active-directory/).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Free | BASIC | PREMIUM P1 | PREMIUM P2 |
| Directory Objects | 500,000 Object Limit | No Object Limit | No Object Limit | |
| Single Sign-On | 10 apps per user (pre-integrated SaaS and developer-integrated apps) | 10 apps per user (free tier + Application proxy apps) | No Limit (free, Basic tiers + Self-Service App Integration templates) | |
| Access based on group membership | Not Available | | Available | |

If you have an existing Enterprise Mobility and Security (EMS) subscription with Microsoft, you may already have Azure AD Premium.

**Enterprise Mobility and Security (EMS) subscriptions:**

* EMS E3 includes P1
* EMS E5 includes P2.

If you have an existing Enterprise Agreement or Server and Cloud Enrollment, you may already have Azure Premium. Check the details of your agreement.

#### Application Licensing

You will also need the appropriate license for your application to meet your business needs.

Discuss with the application owner whether the users assigned to and accessing the application have the appropriate licenses for their roles within the application. If Azure AD manages the automatic provisioning based on roles, the roles that are assigned in Azure AD must align with the correct number of licenses owned within the application; improper number of licenses owned in the application may lead to errors during the provisioning/updating of a user.

## Planning Single Sign-on

An SSO implementation based on federation protocols improves security, reliability, and end user experiences while reducing the amount of work you must do to implement. Many applications are pre-integrated in Azure AD with step-by step guides. See our Azure Marketplace.

* **Federated single sign-on with Azure AD to SaaS applications** enables applications to redirect to Azure AD for user authentication instead of prompting for its own password. This is supported for applications that support protocols such as SAML 2.0, WS-Federation, or OpenID Connect, and is the richest mode of single sign-on.
* **Password-based single sign-on** enables secure application password storage and replay using a web browser extension or mobile app. This leverages the existing sign-in process provided by the application, and enables an administrator to manage the passwords and does not require the user to know the password.
* **Microsoft recommends** using Federated SSO with Azure AD (OpenID Connect / SAML) when an application supports it, instead of password-based SSO and ADFS.

When you enable Federated SSO for your application, Azure AD creates a certificate that is by default valid for three years. You can, however, customize the expiration date for that certificate if desired.

* Learn more: [Azure AD Managing Certificates](https://docs.microsoft.com/azure/active-directory/active-directory-sso-certs)

Whatever your expiration policies, ensure that you have processes in place to renew certificates prior to their expiration.

### Considerations for Password-based Single Sign on

Remove this section if you are doing Federated Single Sign-On

**Preparing user devices for password SSO applications**   
Using Azure AD for password SSO applications requires deploying a browser extension that will securely retrieve the credentials and fill out the login forms. As a result, you should define a mechanism to deploy the extension at scale with [supported browsers](https://docs.microsoft.com/azure/active-directory/active-directory-saas-access-panel-introduction#web-browser-requirements). Options include:

* [Group Policy for Internet Explorer](https://azure.microsoft.com/documentation/articles/active-directory-saas-ie-group-policy/)
* [System Center Configuration Manager (SCCM) for Internet Explorer](https://docs.microsoft.com/sccm/core/clients/deploy/deploy-clients-to-windows-computers)
* [User driven download and configuration for Chrome, Firefox, Edge, or IE](https://docs.microsoft.com/azure/active-directory/active-directory-saas-access-panel-introduction#my-apps-secure-sign-in-extension)

Learn more: [How to configure password single sign on](https://docs.microsoft.com/azure/active-directory/application-config-sso-how-to-configure-password-sso-non-gallery)

**Capturing Login forms metadata for applications that are not in the gallery**   
Microsoft supports capturing metadata on a web application for password vaulting (e.g. capturing the username and password fields). You must navigate to the login URL during the process of configuring the application to capture the forms metadata. Request from the application owner the exact login URL. This information is later used by users during the sign on process, mapping Azure AD credentials to the application during log on.

Learn more: [What is application access and SSO with Azure AD? – Password-based SSO](https://azure.microsoft.com/documentation/articles/active-directory-appssoaccess-whatis/)

**Help desk training**   
When applications change their HTML layout, you might need to re-capture the metadata to adjust for the changes. Common symptoms to watch for:

* Users report that clicking on the application gets “stuck” in the login page
* Users report that username is populated, but not the password or vice versa.

**Shared Accounts**  
From the sign-in perspective, applications with Shared Accounts are not different from a gallery application that uses password SSO for individual users. However, there are some additional steps when planning and configuring an application meant to use shared accounts:

1. Work with application business users to get a specific mapping of:
2. Set of users in the organization who will use the application
3. Existing set of credentials in the application associated with the set of users
4. For each combination determined above, create security groups (in the cloud or on-premises based on your requirements) with target actors.
5. Reset the shared credentials. Once deployed with Azure AD, individuals do not (and should not) need the password of the shared account. Since Azure AD will store the password, consider setting it to be very long and complex.
6. Configure automatic rollover of the password if the application supports it. That way, not even the administrator who did the initial set up will know the password of the shared account.

## Planning Reporting and Auditing

Azure AD provides reports that provide technical and business insights. It is recommended that you work with your business and technical application owners to assume ownership of and consume these reports on a regular basis based on your organization’s requirements. The table below provides some examples of typical reporting scenarios.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | Manage Risk |  | Increase Productivity |  | Governance & Compliance |
|  |  |  |  |  |  |  |
| Report types |  | Application permissions and usage. |  | Account provisioning activity |  | Review who is accessing the applications |
| Potential actions |  | Audit access; revoke permissions |  | Remediate any provisioning errors |  | Revoke access |

Azure AD retains most auditing data for 30 days and makes the data available via Azure Admin Portal or API for you to download into your analysis systems.

Learn more: [View your access and usage reports](https://azure.microsoft.com/documentation/articles/active-directory-view-access-usage-reports/)

### Planning Your Security Review

The following describes security considerations related to this change:

It’s not uncommon for a security review to be required as part of a deployment of a new service. If a security review is required or has not yet been conducted, please review the many Azure AD [whitepapers](https://www.microsoft.com/download/details.aspx?id=36391) that will provides an overview for the identity as a service.

# Designing Your Implementation

This section is used to design the business capabilities that you want to enable. You should add capabilities necessary for your environment. The capabilities necessary can then from the basis for your test planning. When there are choices among options, and Microsoft has a clear recommendation, it is indicated.

Within each table below, indicate in the required column the business capabilities that you want to include in your design. This will form the basis for your implementation

## Designing Single Sign on

Choose which type of Single Sign on you would like to use and remove the ones that do not apply.

|  |  |
| --- | --- |
| SSO type | Description |
| Federated single sign on with Azure AD | * Microsoft Recommended   Enables applications to redirect to Azure AD for user authentication instead of prompting for its own password. This is supported for applications that support protocols such as SAML 2.0, WS-Federation, or OpenID Connect, and is the richest mode of single sign-on. |
| Password based SSO | Enables secure application password storage and replay using a web browser extension or mobile app. This leverages the existing sign-in process provided by the application, but enables an administrator to manage the passwords and does not require the user to know the password. |

### Endpoint Information

Prior to configuring the application, developers will need the following information. Record it here. Note, if you chose password SSO above, you can remove this section.

**Sign-on URL**

* Where the user goes to sign-in to this application. If the application is configured to perform service provider-initiated single sign-on, then when a user navigates to this URL, the service provider will do the necessary redirection to Azure AD to authenticate and log on the user in.

**Identifier**

* The issuer URL should uniquely identify the application for which single sign-on is being configured. This is the value that Azure AD sends back to application as the Audience parameter of the SAML token, and the application is expected to validate it. This value also appears as the Entity ID in any SAML metadata provided by the application.

**Reply URL**

* The issuer URL should uniquely identify the application for which single sign-on is being configured. This is the value that Azure AD sends back to application as the Audience parameter of the SAML token, and the application is expected to validate it. This value also appears as the Entity ID in any SAML metadata provided by the application.

**SAML SSO Sign-Out URL**

* The Sign-On and Sign-Out service URL both resolve to the same endpoint, which is the SAML request-handling endpoint for your instance of Azure AD. The Issuer URL is the value that appears as the "Issuer" inside the SAML token issued to the application.

**SAML Entity ID**

* This value is in any SAML metadata provided by the application. Check the application’s SAML documentation for details on what its Entity ID or Audience value is. This value is mapped to the Reply URL.

Use the following tables to document the endpoints for your deployment:

|  |  |  |
| --- | --- | --- |
| Production Environment: | | Endpoints |
| Sign-on URL (SP-initiated only) |  | |
| Identifier |  | |
| Reply URL |  | |
| SAML SSO Sign-out URL |  | |
| SAML entity ID |  | |

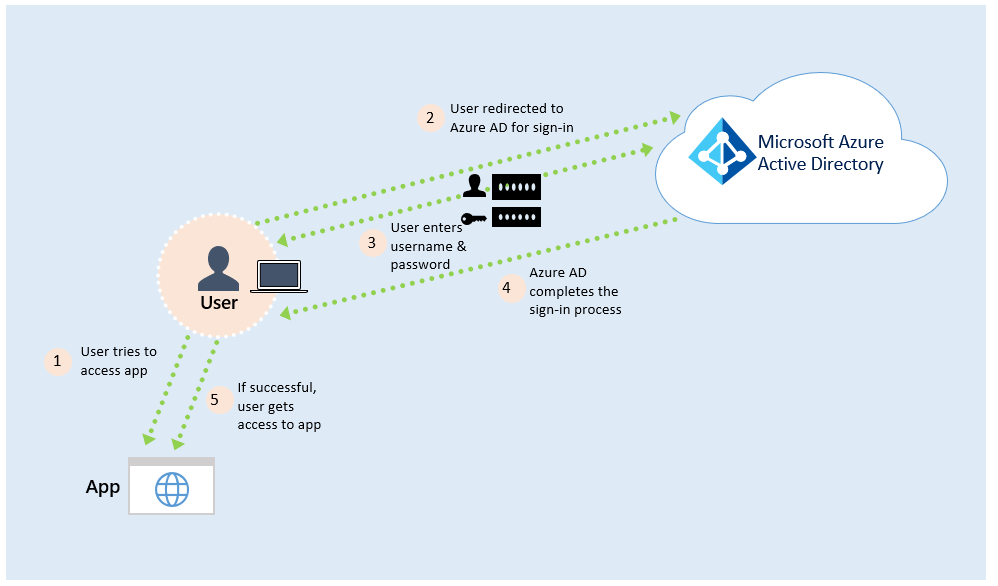
## Solution Architecture Diagrams and Description

Several topologies are represented below. You should choose the one that most closely matches your specific scenario, and delete the rest.

### Azure AD Single Sign-on with Password Hash Sync or cloud-based users Authentication

This is an overview of how the authorization flow for Password Hash Sync or cloud user authentication works.

The following diagram illustrates all the components and the steps involved when a user tries to sign in to an application secured by Azure AD, and if PHS used or cloud-based authentication flow:



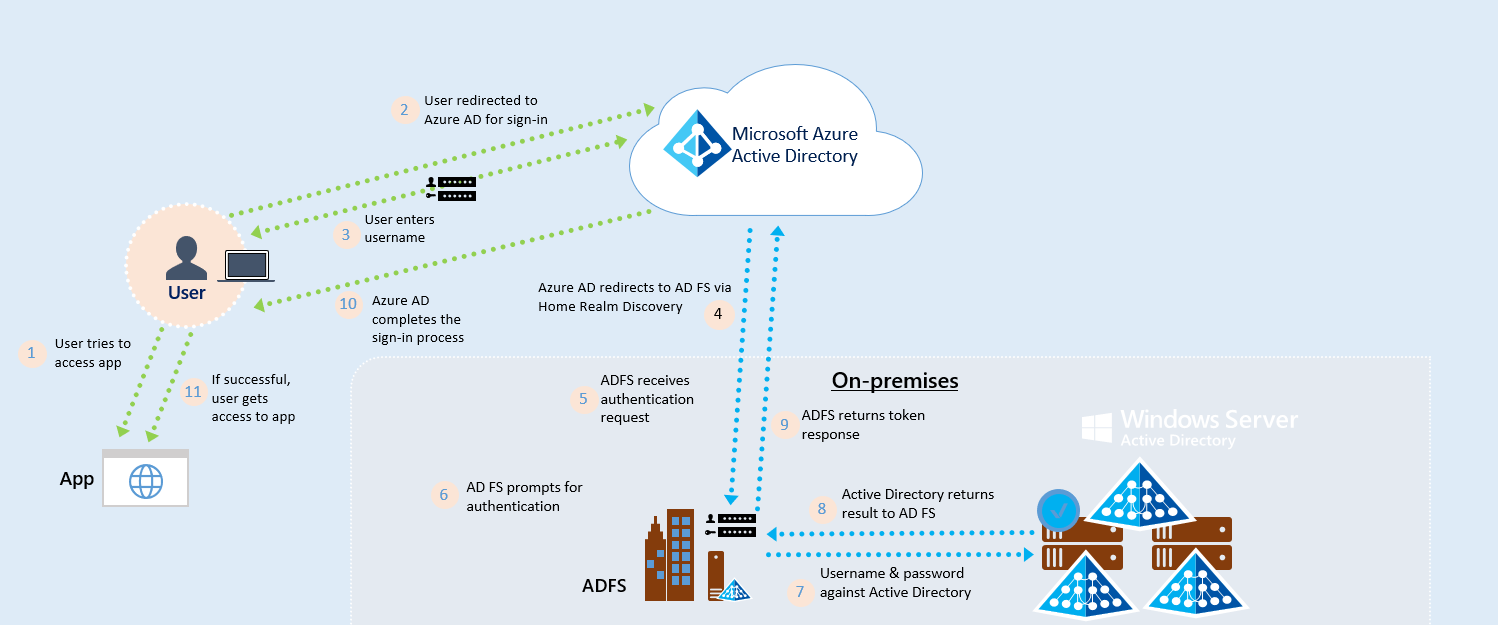
When a user tries to sign in to an application secured by Azure AD, and if the user is authenticating directly with Azure AD, the following steps occur:

1. The user tries to access an application, for example, Outlook Web App or Box.
2. If the user is not already signed in, the user is redirected to the Azure AD User Sign-in page.
3. The user enters their username and password into the Azure AD sign in page, and then selects the Sign in button.
4. Azure AD evaluates the response and responds to the user as appropriate. For example, Azure AD either signs the user in immediately or requests for Azure Multi-Factor Authentication.
5. If the user sign-in is successful, the user can access the application.

### Azure AD Single Sign-on with AD FS or other Federation as IdP

This solution is a combination of hybrid identity sync using Azure AD Connect and maintaining a trust with on-premises federation service as the Identity provider for Azure Active Directory. Azure Active Directory acts as the IdP for the application while acting as the SP for the federation service on-premises.

The following diagram illustrates all the components and the steps involved when a user tries to sign in to an application secured by Azure AD, and if ADFS is enabled on the tenant:

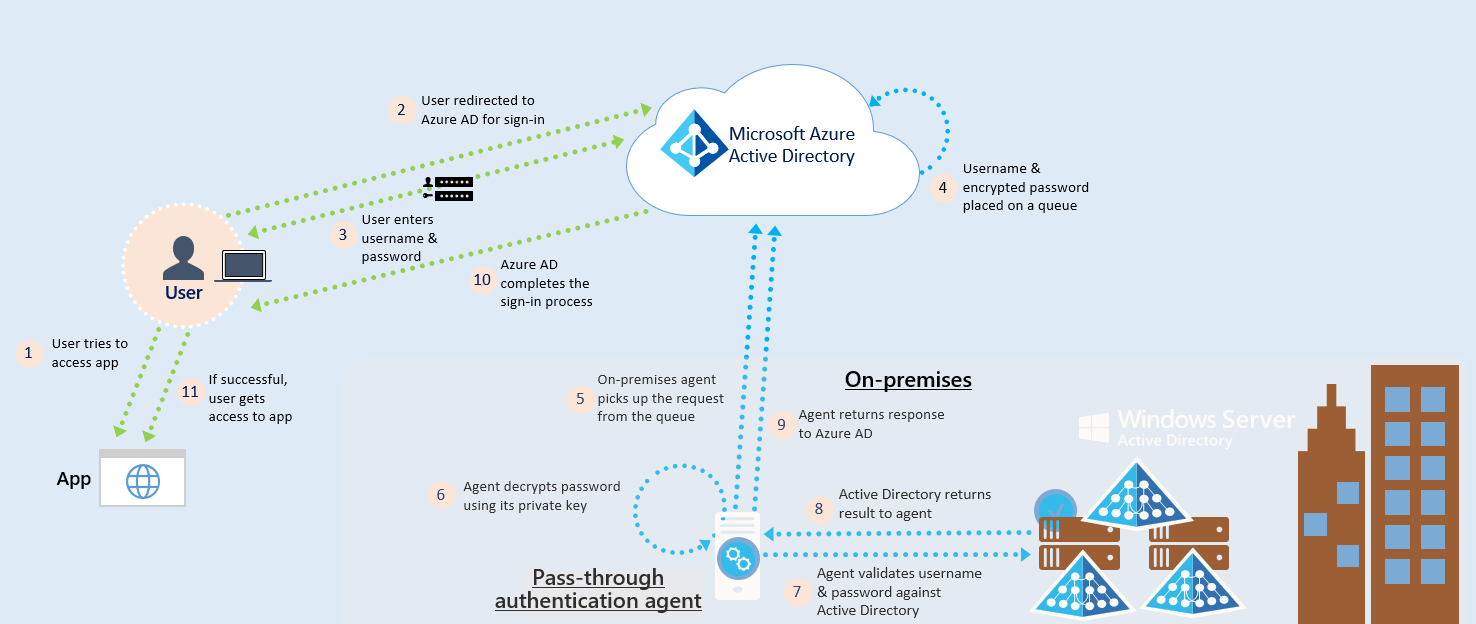


1. The user tries to access an application, for example, Outlook Web App or Box.
2. If the user is not already signed in, the user is redirected to the Azure AD User Sign-in page.
3. The user enters their username into the Azure AD sign in page, and then hits tab.
4. Azure AD detects HRD (Home Realm Discovery) based on the domain and submits a token request to the federated IdP (e.g. AD FS, PingFederate, ect.)
5. AD FS receives the SAML Request
6. AD FS prompts for forms-based authentication with username and password (Windows Integrated Authentication may be performed if applicable)
7. AD FS performs a Username and Password validation to local Active Directory
8. Successful update to AD FS
9. AD FS returns a token Response with claims based on rules for Azure AD’s original request
10. Azure AD evaluates the response and responds to the user as appropriate. For example, Azure AD either signs the user in immediately or requests for Azure Multi-Factor Authentication   
    (*Note: this is where Conditional Access is applied*).
11. If the user sign-in is successful, the user can access the application.

### Azure AD Single Sign-on with Pass-through Authentication

This is an overview of how Azure Active directory (Azure AD) Pass-through Authentication works. For deep technical and security information, see the [Security deep dive](https://docs.microsoft.com/azure/active-directory/connect/active-directory-aadconnect-pass-through-authentication-security-deep-dive) article.

The following diagram illustrates all the components and the steps involved:



When a user tries to sign in to an application secured by Azure AD, and if Pass-through Authentication is enabled on the tenant, the following steps occur:

1. The user tries to access an application, for example, Outlook Web App.
2. If the user is not already signed in, the user is redirected to the Azure AD User Sign-in page.
3. The user enters their username and password into the Azure AD sign in page, and then selects the Sign in button.
4. Azure AD, on receiving the request to sign in, places the username and password (encrypted by using a public key) in a queue.
5. An on-premises Authentication Agent retrieves the username and encrypted password from the queue. Note that the Agent doesn't frequently poll for requests from the queue, but retrieves requests over a pre-established persistent connection.
6. The agent decrypts the password by using its private key.
7. The agent validates the username and password against Active Directory by using standard Windows APIs, which is a similar mechanism to what Active Directory Federation Services (AD FS) uses. The username can be either the on-premises default username, usually userPrincipalName, or another attribute configured in Azure AD Connect (known as Alternate ID).
8. The on-premises Active Directory domain controller (DC) evaluates the request and returns the appropriate response (success, failure, password expired, or user locked out) to the agent.
9. The Authentication Agent, in turn, returns this response back to Azure AD.
10. Azure AD evaluates the response and responds to the user as appropriate. For example, Azure AD either signs the user in immediately or requests for Azure Multi-Factor Authentication.
11. If the user sign-in is successful, the user can access the application.

# Technical Requirements

## Azure Single Sign-on Requirements

The following tables detail the requirements to configure your specific application including the necessary environment(s), endpoints, claim mapping, required attributes, certificates, and protocols used. You will be required to use this information to configure the Single Sign-on portion of your deployment in the [Azure AD admin portal](https://portal.azure.com/#blade/Microsoft_AAD_IAM/StartboardApplicationsMenuBlade/Overview/menuId/).

#### Protocols Supported by Application

For all pre-integrated SaaS apps, Microsoft provides a tutorial and you will not need this information. If the application is not in our application marketplace / gallery, you may need this information.

|  |  |  |
| --- | --- | --- |
| Current State for Authentication | Protocols Supported by Application | Protocol Being Configured with Azure AD |
| Forms-Based  SSO with AD FS  SSO with PingFederate  SSO with Okta | SAML 2.0  Open ID Connect  OAuth  Forms-Based Auth  WS-Fed  WS-Trust | Click to select a protocol. |

### Attribute Requirements

Below, select the attribute matching scheme you will use, and then document the attribute names and mapping.

**Attribute Requirements (if applicable)**

Primary identifier value matches identically to the value in Azure AD

Primary identifier value matches identically to the value in AD

Caps matches between Azure AD and within the application

(**Note:** *Case-sensitivity exists for some applications such as Salesforce with federationID)*

All attributes are available in the application that are required

All attributes are available in Azure AD that are required

**Attribute Mapping**

|  |  |  |
| --- | --- | --- |
| AD Attribute Name | Azure AD Attribute Name  *If Join() or ExtractMailPrefix(), write below values: N/A* | Application Attribute Name |
| <Input attribute name> | **<Input attribute name>**  givenName | **<Input attributes if applicable>** |
|  | Click here to select. |  |
|  | Click here to select. |  |
|  | Click here to select. |  |
|  | Click here to select. |  |

### Certificate Requirements

The certificate for the application must be up to date, or there is a risk of users not being able to access the application. By default, most SaaS application certificates are good for 36 months. However, you may change that length in the application blade. Ensure you document the expiration and know how you will manage your certificate renewal.

|  |  |  |
| --- | --- | --- |
| Length of Certificate | Date of Expiration | Metadata URL for Cert |
| <Input how many months> | **<Input cert expiration date>** | **<Input URL for Cert>** |

There are two ways to manage your certificates.

* **Automatic Cert Rollover:**Microsoft supports [Signing key rollover in Azure AD](https://docs.microsoft.com/azure/active-directory/develop/active-directory-signing-key-rollover). While this is our preferred method for managing certs, not all ISV’s supports this scenario.
* **Manually Updating:** Every application has its own certificate that expires based on how you have defined. Before the application’s cert expires, you must create a new cert and send to the ISV. Alternatively, this can be pulled from the federation metadata. [Read more on federation metadata here.](https://docs.microsoft.com/azure/active-directory/develop/active-directory-federation-metadata)
* **Microsoft Recommends Automatic Cert Rollover**

# Implementing Your Solution

The foundation of proper planning is the basis upon which you can deploy an application successfully with Azure Active Directory. It provides intelligent security and integration that simplifies onboarding while reducing the time for successful deployments. This combination ensure that your application is integrated with ease while mitigating down time for your end users.

Use the following phases to plan for and deploy your solution in your organization:

[Phase 1: Implementation Steps](#_Phase_1:_Implementation)

[Phase 2: Change Communications](#_Phase_2:_Change)

[Phase 3: Verify End User Scenarios for SSO](#_Phase_3:_User)

[Phase 4: Rollback Steps](#_Phase_4:_Rollback)

## Phase 1: Implementation Steps

In this section you will be able to get the instructions to deploy your solution. Use the following steps to implement your project:

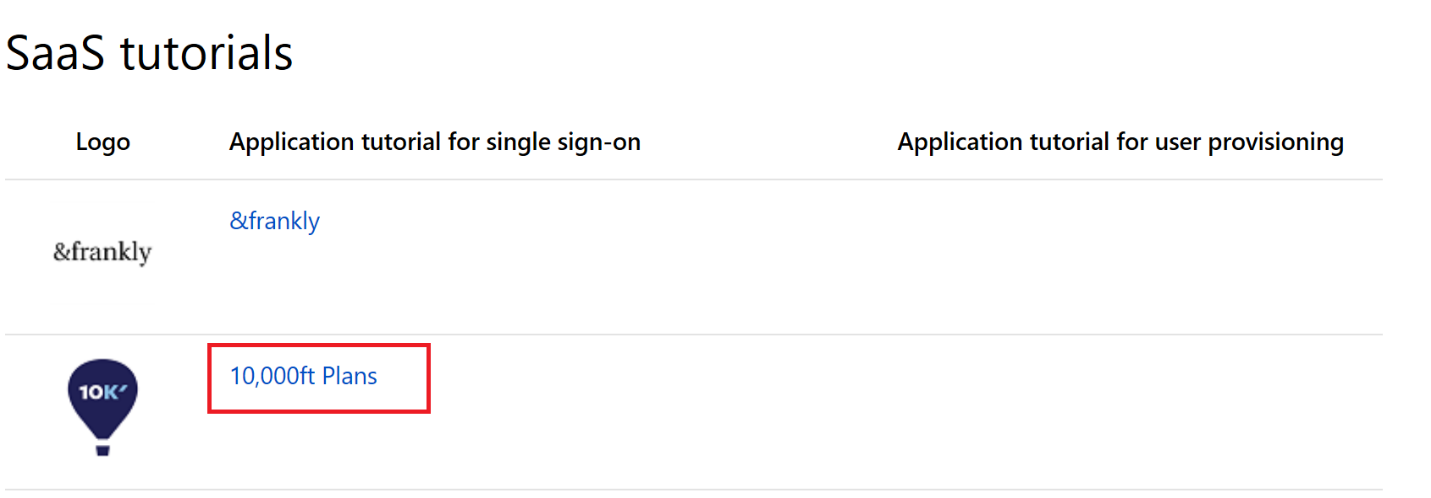
### Step 1: Identify your Test Users

* 1. Reach out to the app owner and request they create a minimum of -3 test users within the application.
  2. Ensure the information that you will be using later as the primary attribute is populated correctly and will match what will be available in Azure AD.
     + **Note** (In most cases this will map to the NameID for SAML-based applications. For JWOT tokens it’s the value name that you provide).
  3. Create the user in Azure AD either manually as a cloud-based user or sync the user from on-premises using Azure AD Connect sync engine. Ensure the information matches that you will be using in the claims being sent to the application.



### Step 2: Configure Azure Single Sign-on

* 1. From the [list of applications](https://docs.microsoft.com/azure/active-directory/active-directory-saas-tutorial-list), locate and open the SSO tutorial for your application, then follow the tutorial’s steps on to successfully configure your SaaS application.

**Example:** 

* 1. If you do not locate your application, navigate to our [Custom Application documentation](https://docs.microsoft.com/azure/active-directory/application-config-sso-how-to-configure-federated-sso-non-gallery). This will walk you through on how to add an application that is not located in the Azure AD gallery.
  2. (Optional) Customize claims issued in the SAML token for enterprise application using [Microsoft’s guidance documentation](https://docs.microsoft.com/azure/active-directory/active-directory-claims-mapping). Ensure these maps to what is expected to be received in the SAML response for your application.
  3. If you encounter issues during configuration, use our guidance on [how to Debug SSO integration](https://docs.microsoft.com/azure/active-directory/develop/active-directory-saml-debugging).
  4. Note: (Custom application is an Azure AD Premium P1 or P2 licenses feature)

## Phase 2: Change Communications

### Step 1: Provide Internal Change Communication to end users

The end user experience will change when accessing your application moving forward. Use the following template example to communicate to end users about these changes to reduce help desk calls and drive positive adoption for your deployment



## Phase 3: Verify End User Scenario for SSO

### Step 1: Create test cases for your application deployment

The following tests will be conducted with both Corporate Own devices and personal devices. These test cases should reflect your Business Use Cases. These will be used to verify whether this solution meets your requirements.

**See Examples below:**

|  |  |  |
| --- | --- | --- |
| Scenarios | Expected Results | Actual Results |
| Authorized User logs into <<APPLICATION NAME>> with IE while on corp (SP-initiated) | User navigates to <<APPLICATION NAME>> URL and initiates SP-initiated flow. IWA occurs with no additional prompts |  |
| Authorized User logs into <<APPLICATION NAME>> with IE while off corp (SP-initiated) with new login attempt | User navigates to <<APPLICATION NAME>> URL and initiates SP-initiated flow. Forms-based prompt at AD FS Sever. User successfully logs in and browser prompts for MFA. |  |
| Authorized User logs into <<APPLICATION NAME>> with IE while off corp (SP-initiated) with a current session and has never performed MFA | User navigates to <<APPLICATION NAME>> URL and initiates SP-initiated flow. User does not receive prompt for first factor. User receives prompt for MFA. |  |
| Authorized User logs into <<APPLICATION NAME>> with IE while off corp (SP-initiated) with a current session and has already performed MFA in this session | User navigates to <<APPLICATION NAME>> URL and initiates SP-initiated flow. User does not receive prompt for first factor. User does not receive MFA. User SSO’s into <<APPLICATION NAME>> |  |
| Authorized User logs into <<APPLICATION NAME>> with Chrome/Firefox/Safari while off corp network (SP-initiated) with a current session and has already performed MFA in this session | User navigates to <<APPLICATION NAME>> URL and initiates SP-initiated flow. User does not receive prompt for first factor. User does not receive MFA. User SSO’s into <<APPLICATION NAME>> |  |
| Authorized User logs into <<APPLICATION NAME>> with Chrome/Firefox/Safari while off corp network (SP-initiated) with new login attempt | User navigates to <<APPLICATION NAME>> URL and initiates SP-initiated flow. Forms-based prompt at AD FS Sever. User successfully logs in and browser prompts for MFA. |  |

|  |  |  |
| --- | --- | --- |
| Scenarios | Expected Results | Actual Results |
| Authorized User logs into <<APPLICATION NAME>> with Chrome/Firefox while on corp network (SP-initiated) with a current session | User navigates to <<APPLICATION NAME>> URL and initiates SP-initiated flow. User does not receive prompt for first factor. User does not receive MFA. User SSO’s into <<APPLICATION NAME>> |  |
| Authorized User logs into <<APPLICATION NAME>> with <<APPLICATION NAME>> mobile app (SP-initiated) with a new login attempt | User navigates to <<APPLICATION NAME>> URL and initiates SP-initiated flow. Forms-based prompt at AD FS Sever. User successfully logs in and ADAL client prompts for MFA. |  |
| Unauthorized User attempts to log into <<APPLICATION NAME>> with login URL (SP-initiated) | User navigates to <<APPLICATION NAME>> URL and initiates SP-initiated flow. Forms-based prompt at AD FS Sever. User fails to login with first factor |  |
| Authorized user attempts to log in but enters an incorrect password | User navigates to <<APPLICATION NAME>> URL and receives bad username/password error. |  |
| Authorized user logs out and then logs in again | If Sign-out URL is configured, user is logged out of all services and prompt to authenticate. If Sign-out URL is **not** configured, user will be automatically logged back in using existing token from the existing Azure AD browser session. |  |
| Authorized user clicks on link in an email and is already authenticated | User clicks on URL and is signed into the application with no additional prompts |  |
| Authorized user clicks on link in an email and is not yet authenticated | User clicks on URL and is prompt to authenticate with first factor. |  |

### Step 2: Document your results

Document the outcomes for both **Expected Results** and **Actual Results** in [Step 1](#_Step_1:_Create). Use this to determine to move forward into production based on your [established timelines](#_Timelines).

### Step 3: Moving into Production

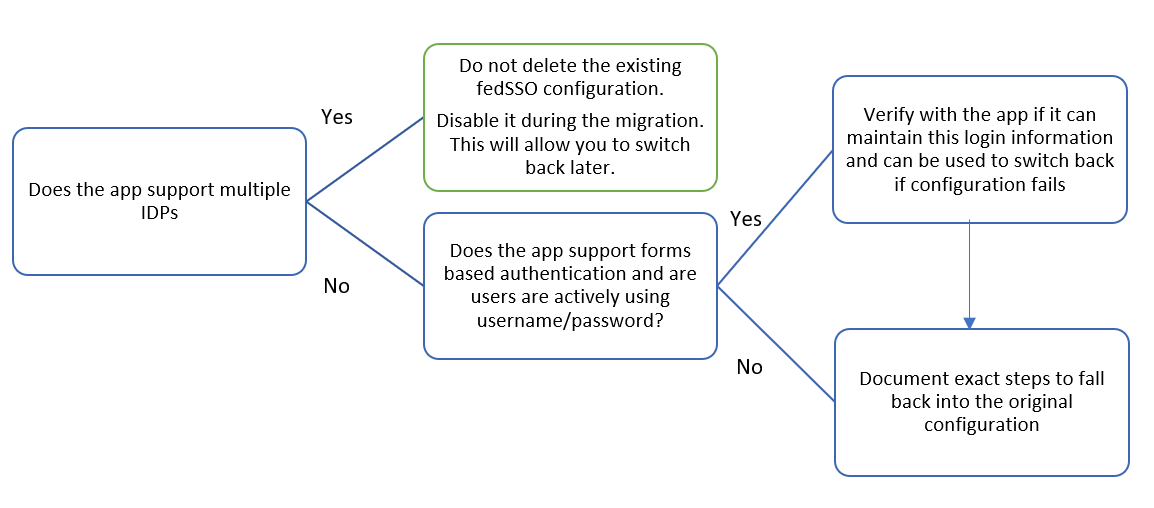
After you complete all of testing based on your test cases, it’s time to move into production with your application.

## Phase 4: Rollback Steps

It’s important to plan what to do in the case during your deployment doesn’t go as planned. If the SSO configuration fails during the deployment, you must understand how to mitigate any outage and reduce impact to your users.

### Step 1: Identify available options for rolling back during migration or failure

The availability of authentication methods within the application will determine your best strategy. Please review the decision tree below to help determine the best way to prepare rollback steps that are available for this application:



# Operationalize your Implementation

## Purpose of Document

The intent for the *Operationalize your Implementation* is to address the day-to-day operations for maintaining the application that has been deployed over the life of the application. This includes the roles, certificate lifecycle, troubleshooting steps, access management and attestation of user access and roles.

## Required Roles

Microsoft recommends using the less role to accomplish the required task within Azure Active Directory. Microsoft recommend [review the different roles that are available](https://docs.microsoft.com/azure/active-directory/active-directory-assign-admin-roles-azure-portal) and choose the right one to solve your needs for each persona for this application. Some roles may need to be applied temporarily and removed after the deployment has been completed.

|  |  |  |  |
| --- | --- | --- | --- |
| Personas | Roles | Azure AD Role (if required) | Assign to |
| Help Desk Admin | Tier 1 Support | None |  |
| Identity Admin | Configure and Debug when issues impact Azure AD | Global Admin |  |
| <<APPLICATION NAME>> Admin | User attestation in <<APPLICATION NAME>>, configuration on users with permissions | None |  |
| Infrastructure Admins | Cert Rollover Owner | Global Admin |  |
| Business Owner/Stakeholder | User attestation in <<APPLICATION NAME>>, configuration on users with permissions | None |  |

* **Microsoft recommends** using [Privileged Identity Management](https://docs.microsoft.com/azure/active-directory/active-directory-privileged-identity-management-configure) to manage your roles to provide additional auditing, control, and access review for users with directory permissions.

## 

## Single Sign-on Certificate Lifecycle [Azure Active Directory]

In this section, it identifies the process to manage the lifecycle of the signing certificate between Azure AD and the application that is being configured with single sign-on. It’s important to identify the right email distribution list to receive alerts for certificate expiration reminders.

|  |  |
| --- | --- |
| Scenarios | Results |
| Owner for updating user properties in <<APPLICATION NAME>> | **Example:**   1. *Request is submitted to Key Users from Support* 2. *Key Users submit a request to ISV Vendor* 3. *ISV Vendor creates a change within <<APPLICATION NAME>>* |
| Owner On call for application break/fix support | **Example:**   1. *Escalate to Key Users* |
| Owner of rolling over certificate | **Example:**  ***Owner: App Owner***  ***Process:***   1. *Notification is received by Support team* 2. *Support Notifies Product Owner/Business Owner* 3. *Business Owner Approves* 4. *Support Creates a ticket to Ticketing System* 5. *Ticket is Assigned to the Infrastructure team* 6. *Infrastructure team will create Cert in Azure AD and deliver the certificate <<APPLICATION NAME>> App Team for update* |
| Notification Email for cert notification: | *Insert a DL that will be monitored 100% of the time* |
| Estimated lifetime of application | *Maximum lifetime a certificate can be 3 years* |

* **Microsoft recommends** establishing a process on how you will handle when there is a certificate change between Azure AD and your application. This can help prevent or minimize an outage due to a certificate expiring or force certificate rollover.

## Access Management

|  |  |
| --- | --- |
| Scenarios | Results |
|  | |
| Name of Group(s) access <<APPLICATION NAME>> |  |
| Group Owner(s) – *if applicable* |  |
| Location of Group | On-Premises and synced  Azure AD – *dynamic or self-service group* |
| Attestation | **Example:**  *Process:*   1. *Annual evaluation from audit team through manual steps*   *Deprovisioning process:*   1. *User is removed from HR System* 2. *Email is sent to Support & Business Owner(s)* 3. *User is manually removed from <<APPLICATION NAME>>*   *Access Review iteration date (e.g. every 120 days): Annually* |

* **Microsoft recommends** choosing a scaled approach when managing access to resources. Common approaches include: utilizing on-prem groups by syncing via Azure AD Connect, [creating Dynamic Groups in Azure AD based on user attributes](https://docs.microsoft.com/azure/active-directory/active-directory-groups-dynamic-membership-azure-portal), or creating [self-service groups](https://docs.microsoft.com/azure/active-directory/active-directory-accessmanagement-self-service-group-management) in Azure AD managed by a resource owner.

## 

## Troubleshooting Guide & Steps

You should create troubleshooting guides for your support organization. In this section you will find examples of Tier 1, 2, and 3 guides for common scenarios. You should modify these to fit your organization.

### Example: Single account not being able to log into the application.

|  |  |  |  |
| --- | --- | --- | --- |
| Tier | Condition: If… | Verification | Details / actions |
| 1 | ***Tier 1*** *– you will need to verify the user’s access status, state of the account (enable/disable) from Azure AD & application attempting to access, and whether there is conflict between multiple accounts.* | | |
| User is unable to locate the application | NA | Provide the Single Sign-on URL for the application.  Additionally, provide instructions to Access Panel |
| User is subject to integrated windows authentication (IWA) | Verify user is using correct credential for the device. |  |
| User owns multiple accounts | Verify that user is using the correct account / URL pair. | Provide [instructions to log](https://support.microsoft.com/help/4026200/windows-browse-inprivate-in-microsoft-edge) into a private session to reduce conflict of existing sessions |
| User can login to network but not application. | Step 1: Verify that user is assigned to the application. | In the [**Azure Portal**](https://aad.portal.azure.com), select **Enterprise applications**, find the application, and see if the user has been directly assigned or has been assigned as a member of a group. |
| Step 2: Verify that user has an account in the application. | Step 1: Log in to the application’s administrative portal (the ISV) and verify the user has an account.  Step 2: Verify that the account is not disabled or otherwise inactive. |
| Step 3: Verify that user can login to MyApps and see the application in the Access Panel | Have user log into [Access Panel](https://myapps.microsoft.com) at myapps.microsoft.com. |
| User can log in to application directly, but not through SSO. | Verify that the user has an account provisioned in AAD, and has the account is at the Premium level. | This is only applicable if you allow the users to perform forms-based authentication directly to the application |
| *2* | ***Tier 2*** *– you will need to collect logs of the traffic between the client and application using your preferred method of decrypting authentication traffic.* [*Fiddler*](https://docs.microsoft.com/azure/active-directory/develop/active-directory-saml-debugging) *or* [*SAML Tracer browser extensions*](https://addons.mozilla.org/en-US/firefox/addon/saml-tracer/) *are generally the most common tools.* | | |
| User can login to network but not application. | Step 1: Review the logs on the application that the user is attempting to access | Logs here will generally reveal bad requests. If there is no logs, then either application does not offer these logs or it has not received a Response from Azure AD. |
| Step 2: Validate the certificate has not expired | Step 1: Verify the certificate thumbprint matches between Azure AD and the application that is attempting to be accessed  Step 2: Verify the certificate is still valid and has not been expired |
| Step 3: Validate endpoints are correct | [Compare the endpoints configured on the application object in Azure AD](#_Endpoint_Information) to what has been configured in the ISV (e.g. the application that the user is attempting to access) |
| Step 4: Validate claims (NameID) mapping between AAD and application or the attribute name you have chosen to map to for the application you user is attempting to access | If the attribute that is being sent does not match to what is being expected, you will either receive an error or mismatching may occur and authenticate the wrong user   *(e.g. Azure AD sends “john@contoso.com” and the application is expecting employeeID “john1445”)* |
| *3* | ***Tier 3*** *– at this point all endpoints has been verified, certs are valid, claims are correct. Contact the application owner and begin steps to escalate to Microsoft support to assist resolving the issue* | | |
| Not resolved by Tier 2 | Notify business owner for <<APPLICATION NAME>> |  |
| Not resolved by Tier 2 | Create a support ticket with Microsoft | Include Repro Steps, UPN, CorrelationID, Timestamp, and Fiddler Trace(s).  **Note:** *Make multiple attempts to authenticate and provide timeframe.* |
|  |  |  |

### 

### Example: Complete outage of <<APPLICATION NAME>> - No user can sign in

When no user can sign in to the application, the following steps can help you to identify the issue.

|  |  |
| --- | --- |
| Task | Details / actions |
| Notify Business Owner | Communicate the current state to the Stakeholder for <<APPLICATION NAME>> |
| Collect Fiddler Trace(s) of Repro | <https://docs.microsoft.com/azure/active-directory/develop/active-directory-saml-debugging> |
| Create a Premier Support Ticket | Include Repro Steps, UPN, CorrelationID, Timestamp, and Fiddler Trace(s).  **Note:** *Make multiple attempts to authenticate and provide timeframe.* |
| Review sign-in logs in both Admin Portals | <<APPLICATION NAME>> and Azure AD both has audit logs. Review these logs to determine whether there is a known issue. |
| Review current documentation and ensure configured correctly | https://docs.microsoft.com/en-us/azure/active-directory/active-directory-saas-<<APPLICATION NAME>>-tutorial |
| Validate endpoints are correct | Compare Reply URL, Identifier, and Login URL between <<APPLICATION NAME>> and Azure AD  View in the SAMLResponse/SAMLRequest that information matches |
| Validate Certificate has not expired | This is located in both <<APPLICATION NAME>>’s IdP setting as well in Azure AD’s Admin Portal. If cert has rolled over, you may have to update cert |
| Validate claims mapping | Confirm both values match in Azure AD and <<APPLICATION NAME>> for all claims, especially the primary claim. This can be viewed in Fiddler under SAMLResponse |
| Work with MS to close support ticket | If this application is considered high impact, open a Sev 1 ticket against Premier |

## 

## Helpful Documentation

**General**

[Debug SAML-based SSO](https://docs.microsoft.com/azure/active-directory/develop/active-directory-saml-debugging)

[Customizing claim issued in SAML token](https://docs.microsoft.com/azure/active-directory/develop/active-directory-saml-claims-customization)

[Single Sign-on SAML protocol](https://docs.microsoft.com/azure/active-directory/develop/active-directory-single-sign-on-protocol-reference)

[Single Sign-Out SAML protocol](https://docs.microsoft.com/azure/active-directory/develop/active-directory-single-sign-out-protocol-reference)

[Azure AD B2B](https://docs.microsoft.com/azure/active-directory/active-directory-b2b-what-is-azure-ad-b2b) (for external users such as partners and vendors)

[Azure AD Conditional Access](https://docs.microsoft.com/azure/active-directory/active-directory-conditional-access-azure-portal)

[Azure Identity Protection](https://docs.microsoft.com/azure/active-directory/active-directory-identityprotection)

[SSO access](https://docs.microsoft.com/azure/active-directory/active-directory-appssoaccess-whatis)

[MFA Conditional Access for SaaS](https://docs.microsoft.com/azure/active-directory/active-directory-playbook-building-blocks#mfa-conditional-access-for-saas-applications)

[Configure Token Lifetimes](https://docs.microsoft.com/azure/active-directory/active-directory-configurable-token-lifetimes)

[Claim mapping for Apps via PowerShell](https://docs.microsoft.com/azure/active-directory/active-directory-claims-mapping)

**Application Specific**

<<APPLICATION NAME>> SSO Tutorial: [https://docs.microsoft.com/azure/active-directory/active-directory-saas-<<APPLICATION NAME>>-tutorial](https://docs.microsoft.com/azure/active-directory/active-directory-saas-salesforce-tutorial)

<<APPLICATION NAME>> Provisioning Tutorial:

[https://docs.microsoft.com/azure/active-directory/active-directory-saas-<<APPLICATION NAME>>-provisioning-tutorial](https://docs.microsoft.com/azure/active-directory/active-directory-saas-%3cInsert%20App%20Name%3e-provisioning-tutorial)

**IMPORTANT NOTICES**

© 2018 Microsoft Corporation.  All rights reserved.  This document is provided "as-is." Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it.   
Some examples are for illustration only and are fictitious. No real association is intended or inferred.   
   
This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes. You may modify this document for your internal, reference purposes

**Customize this document**

To customize this document for your use with a specific application, perform a global replace of <<APPLICATION NAME>> with the name of the SaaS application with which you are working.