SailPoint IdentityIQ Azure Active Directory Connector

The following topics are discussed in this document:

Supported Features

The Azure Active Directory connector supports the following features:

- Account Management for User, User in Federated Domain, and Guest User (B2B)
- Account Management for Local User (B2C)
- Account - Group Management
  - Group Management
  - Group Management for Microsoft Teams
- Service Principal Management
- OAuth2.0 Authentication
- Azure Endpoint Configuration
- Other

Account Management for User, User in Federated Domain, and Guest User (B2B)

This section describes the supported features for User, User in federated domain, and Guest User (B2B).

Operations	User	User in federated domain	Guest User (B2B)
Aggregation	Yes	Yes	Yes
Delta Aggregation	Yes	Yes	Yes
Partitioning Aggregation	Yes	Yes	Yes
Get Account	Yes	Yes	Yes
Pass Through Authentication	Yes	No	Yes
Create	Yes	Yes	Yes
Update	Yes	No	Yes
Delete	Yes	No	Yes
Enable/Disable Users	Yes	No	Yes
Set Password	Yes	No	Yes
Add/Remove Entitlements • Add\Remove individual license plans • Add\Remove license packs • Add\Remove Roles • Add\Remove user's group membership • Add/Remove ServicePrincipal Names	Yes	Yes	Yes

Account Management for Local User (B2C)

Azure Active Directory B2C is primarily for businesses and developers that create customer-facing applications. With Azure Active Directory B2C, users can use Azure Active Directory as the full-featured identity system for their application, while letting customers sign in with an identity they already have established (like Facebook or Gmail).

B2C is intended for inviting customers of your mobile and web applications, whether individuals, institutional or organizational customers into your Azure Active Directory.

Supported identities are Consumer users with local application accounts (any email address or user name) or any supported social identity with direct federation. A new/separate Azure Active Directory application must be created to manage B2C tenant.

If **Manage B2C Tenant** is selected, entitlements are not fetched by default. To fetch entitlements set **fetchB2CMembership** attribute to true in application debug page as mentioned in "Additional Configuration Parameters" section.

Operations	**Manage B2C Tenant**	
	Local User	**Social User**
Aggregation	Yes	Yes
Partitioning Aggregation	Yes	Yes
Create	Yes	No
Update	Yes	Yes
Delete	Yes	Yes
Enable/Disable Users	Yes	Yes
Add/Remove Entitlements • Add\Remove Roles • Add\Remove user's group membership • Add/Remove ServicePrincipal Names	Yes	Yes

Note: **Delta Aggregation is not applicable if 'Manage B2C Tenant' is selected in "Configuration Parameters".**

Account - Group Management

This section describes the features related to the account-group management.

Group Management

Azure Active Directory Connector supports managing following group types as an entitlement and as separate group objects. These different type of groups are created for different purpose which are described as follows:

- Microsoft 365 groups (formerly Office 365 groups) are used for collaboration between internal and external users of the company.
- Distribution List (Distribution groups) are used for sending notifications to a group of people.
- Security groups are used for granting access to resources such as SharePoint sites.
- Mail-enabled security groups are used for granting access to resources such as SharePoint, and emailing notifications to those users.

Following table describes different conditions/configurations depending on which connector will aggregate / able to provision different types of groups:

Group Types	Filter condition to support Group Aggregation/Provisioning		
	Default Aggregation	**skipMailEnabledGroup=true in application debug page**	**'Manage Office 365 Groups' selected in Configuration Parameters**
Security	Yes	Yes	Yes
Mail Enabled Security	Yes	No	Yes
Office 365	No	No	Yes
Distribution List	No	No	No

Group Management for Microsoft Teams

Microsoft Teams, also referred to as simply Teams, is a unified communication and collaboration platform that combines persistent workplace chat, video meetings, file storage (including collaboration on files), and application integration.

Teams are built on the top of Microsoft 365 groups. Microsoft 365 Group forms the directory object representing Teams. All identity and access controls for Teams are performed on Microsoft 365 Group.

As Azure Active Directory Connector deals with directory objects, Microsoft Teams are represented as Microsoft 365 Groups.

The **Manage Office 365 Groups** checkbox must be selected as it is aggregated as Microsoft 365 Group. Following operations are supported:

- Creation of Teams as Microsoft 365 Group
- Add\Remove Owners and Members

Note: To delete Teams, respective Microsoft Office 365 Group must be deleted.

Service Principal Management

Azure Service Principal is a security identity used by user-created applications, services, and automation tools to access specific Azure resources. Service Principal is a **user identity** (login and password or certificate) with a specific role, and controlled permissions to access your resources. Security is improved if minimum permissions level are granted to perform management tasks.

Following operations are supported for ServicePrincipal object type:

- Aggregation
- View details of **ServicePrincipal** (like object properties, members and so on)
- Provision and Revoke access request for **ServicePrincipal**

OAuth2.0 Authentication

Azure Active Directory Connector supports OAuth2.0 authentication with the following grant types:

- Client Credentials

- SAML Bearer Assertion

Client Credentials

For the default grant type client credentials based authentication, following are the required configurations:

- Obtain **Client ID**, **Client Secret** from Azure Active Directory by registering the application.
- Assign the required permission to application. For more information, see "Assigning Application Permission from Portal" .
- The following permissions do not allow the connector to manage users with administrative roles. To manage users with administrative roles, the application created on Azure must have **User Account Administrator** or **Company Administrator** role assigned using Windows Azure Active Directory Module for Windows PowerShell. For more information, see "Assigning Application Role using PowerShell" .

Permission	Type	Purpose
Azure Active Directory Graph API		
Directory.ReadWriteAll	Delegated	Read, Update, Change Password, Delete User
Microsoft Graph API		
Directory.ReadWriteAll	Delegated	Read, Update, Delete Group Add Membership
Groups.ReadWriteAll	Application	Teams Creation

SAML Bearer Assertion

The SAML Bearer Assertion grant type authentication involves password-based user authentication with ADFS environment. SAML assertion issued by ADFS after authentication is used to obtain access token from Azure Active Directory.

The SAML Bearer Assertion authentication requires the following additional configurations to be performed:

- AAD Connect configured with Azure Active Directory along with ADFS.
- Obtain Client ID, Client Secret from Azure Active Directory by registering the application.
- Assign required permissions to application:

Permission	Type	Purpose
Azure Active Directory Graph API		
Directory.AccessAsUser	Delegated	Change Password, Delete User
Directory.ReadWriteAll	Delegated	Read User, Update User
Microsoft Graph API		
Directory.ReadWriteAll	Delegated	Read, Update, Delete Group Add Membership
Groups.ReadWriteAll	Application	Teams Creation

- The authentication user must be synchronized in Azure Active Directory.
- Assign the user with **User Administrator** permission Role in Azure Active Directory.

Note: To manage users with administrative roles assign user with ‘Global Administrator’ role.

- The ADFS endpoint required to authenticate user must be enabled.
- The ADFS service communication certificate must be installed on IdentityIQ machine.
- The ADFS machine time zone must be in sync with Azure time zone, that is, UTC

Azure Endpoint Configuration

To meet security and compliance needs of federal agencies, state and local governments, Microsoft provides separate instance of the Microsoft Azure service. Few example of such instances are Azure Government, Azure China. As these instances are separated from general Azure services, endpoint (host address) for such instances might also be different.

With endpoint configuration functionality, Azure Active Directory Connector can be configured to communicate with these instances.

Following attributes must be configured in application debug page:

Attributes	Description
azureADGraphResourceBase	Base resource URL to be used for Azure AD Graph API rest calls. Following example points base to default Azure AD Graph resource base: `<entry key="azureADGraphResourceBase" value="https://graph.windows.net"/>`
azureADGraphTokenBase	Base token URL to be used to get access token for Azure AD Graph API rest calls Following example points base to default Azure AD Graph token base: `<entry key="azureADGraphTokenBase" value="https://login.windows.net"/>`
msGraphResourceBase	Base resource URL to be used for Microsoft Graph API rest calls. Following example points base to default Microsoft Graph resource base: `<entry key="msGraphResourceBase" value="https://graph.microsoft.com"/>`
msGraphTokenBase	Base token URL to be used to get access token for Microsoft Graph API rest calls Following example points base to default Microsoft Graph token base: `<entry key="msGraphTokenBase" value="https://login.microsoftonline.com"/>`

Note: Actual values of endpoints can be found in Microsoft documentation for specific instance. For connector operation, values for the above attributes must be provided.

Other

The other feature supported by Azure Active Directory is executing native before/after scripts for provisioning requests.

This feature requires installation and registration of IQService. For more information, see “IQService”.

References

- "Delta Aggregation"
- "Partitioning Aggregation"

Prerequisites

- Ensure a client application has been registered on your Azure Management portal as a web application or web API, and you have access to the Client ID and Client Secret for this application.

  To use Graph API, a client application must be registered on the Azure management portal. This application is responsible for calling Web APIs on behalf of the connector. The application's client ID and client secret key are required while configuring the application.

  To register an application on Azure, perform the following:

  a. User can use any of the following Azure management portal to do the configuration:

     https://portal.azure.com

     OR

     https://aad.portal.azure.com
  b. Select **Azure Active Directory** in the left pane.
  c. Click on **App registrations**.
  d. Click **New registration**.
  e. On the **Register an application** page, in the Name field, enter the name of the application that you want to set up. For example, SailPointAzureADManagement.
  f. In the Supported account types, set up accounts on the basis of users, eligible to avail that application or the API.
  g. (*Optional*) Set up the URL in **Redirect URL**, to have the successful response after authentication. You can use the following format: **http://*domainName*/GraphWebapp**

     Note: Azure Active Directory connector does not use the URL mentioned above, the above example is just a place-holder and does not impact functionality.
  h. Click **Register**. An Application is created. On the Application page the **Application (client) ID**, and other details are displayed. Note down this ID.
  i. On the left-hand panel, select **Certificates & secrets**. On the Certificates & secrets page, in the **Client secrets** section, click **New client secret**.
  j. On the **Add a client secret** page, enter the **Description** to generate a secret, choose the validity duration in the **Expires** list. Click **Add**. Note down the value of the secret that you have just created.

  Note: If Azure Active Directory Connector is behind proxy server, see the "Special Java Considerations" section of the *SailPoint IdentityIQ Installation Guide*.
- The required host values (**https://&lt;host&gt;/**) by the Azure Active Directory Connector to interact with the managed system are as follows:

  - https://login.windows.net
  - https://graph.windows.net
  - https://graph.microsoft.com
  - https://login.microsoftonline.com

Administrator Permissions

Following permissions must be granted to the client application created in Azure:

- Read Directory Data
- Read and Write Directory Data

To grant permissions to the client application:

- Click **API permissions** in Azure Active Directory console. Click **Add a permission**.
- On the **Request API permissions** page, you will see a list of supported APIs. Under **Supported legacy APIs**, click **Azure Active Directory Graph**.
- Choose **Application permissions** under **What type of permissions does your application require?**
- Under **Select permissions**, choose Directory. Select **Read Directory Data** and **Read and Write Directory Data** permissions. Click **Add permissions**.

  In Grant consent, click **Grant admin consent** for your configuration and directory. On the pop-up dialog box, click **Yes**.

To manage Guest User (B2B), perform the steps to provide permissions for Microsoft Graph API:

- Click **API permissions** in Azure Active Directory console. Click **Add a permission**.
- On the **Request API permissions** page, you will see a list of supported APIs, click **Microsoft Graph API**.
- Choose **Application permissions** under **What type of permissions does your application require?**
- Under **Select permissions**, choose **User.** Select **User.Invite.All** permission.
- Choose **Delegated permissions** under **What type of permissions does your application require**?
- Under **Select permissions**, choose **User.** Select **User.Read** permission. Click **Add permissions**.

  In Grant consent, click **Grant admin consent** for your configuration and directory. On the pop-up dialog box, click **Yes**.

To perform Set Password and Delete user operations, application created on Azure must have **User Account Administrator** role assigned using **Windows Azure Active Directory Module for Windows PowerShell**.

To manage users with administrative roles, application created on Azure must have **User Account Administrator** or **Company Administrator** role assigned using **Windows Azure Active Directory Module for Windows PowerShell**. Following are the prerequisite for executing the PowerShell commands.

Note: These prerequisites are not required for the connector to function. These can be installed on any system for temporary use to provide required role to the application on Azure.

- **Microsoft Online Services Sign-In Assistant for IT Professionals RTW**
- **Azure Active Directory Module for Windows PowerShell**

After installing the prerequisites, open **Azure Active Directory Module for Windows PowerShell** console and execute the following commands:

- `Connect-msolservice`, press enter, provide Azure administrator credentials.
- Execute `Get-MsolServicePrincipal | ft DisplayName, AppPrincipalId -Autosize`
- Locate your application name and copy the **ObjectId** value.
- Execute `$ClientObjID = '<copied objectId of the application in the previous step>'`
- Execute `$webApp = Get-MsolServicePrincipal -AppPrincipalId $ClientObjID`
- Execute `Add-MsolRoleMember -RoleName "Company Administrator" -RoleMemberType ServicePrincipal -RoleMemberObjectId $webapp.ObjectId`

Configuration Parameters

This section contains the information that this connector uses to connect and interact with the application. Each application type requires different information to create and maintain a connection.

The Azure Active Directory connector uses the following connection parameters:

Attributes	Description
Connector Credentials	
Authentication Method	Authentication method supported by the managed system. Default is **OAuth 2.0**
Grant Type	Grant type to be used for the authentication: • Client Credentials • SAML Bearer Assertion
Azure AD application client ID*	Client ID of the application created on the Azure Active Directory for using Graph REST API.
Azure AD application client secret key*	Client secret of the Azure Active Directory application.
Azure AD domain name*	Name of the Azure Active Directory domain to be managed. For example, `contoso.onmicrosoft.com`
*Applicable only if **Grant Type** is selected as **SAML Bearer Assertion***	
Authorization Endpoint*	Endpoint URL of authorization server.
Username*	Username for authorization.
Password*	Password of the user for authorization.
Request Body*	Request body for SAML assertion. For more information on SAML Request Body, see "SAML Request Body" .
Additional Configuration	
Page Size	Number of records per page. Default: 500
IQService Host	FQDN/IP of the system where IQService is installed.
IQService Port	The TCP/IP port on which IQService is listening for requests. **Note: If 'Use TLS' is enabled, then ensure to configure corresponding IQService TLS port.**
IQService User	User registered with IQService for Client Authentication.
IQService Password	Password of registered user for Client Authentication.
Use TLS for IQService	Indicates whether this is a TLS communication between IdentityIQ and IQService. **Note: If 'Use TLS for IQService' is enabled, 'IQService User' and 'IQService Password' attributes are mandatory.**
Manage Office 365 Groups	Enables aggregation and provisioning of Office 365 groups.

Attributes	Description
Manage B2C Tenant	Indicates whether application is used to manage B2C tenant.

Note: Attributes marked with * sign are the mandatory attributes.

Note: To enable native before/after script execution for provisioning requests, IQService Host and IQService Port parameters must be configured.

Note: For more information on enabling the Client Authentication and TLS communication, see "IQService".

Additional Configuration Parameters

Attributes	Description
createAccountTimelag	Time in seconds to wait after create account and before calling get account. Default: 20 seconds For example, `<entry key="createAccountTimelag" value="20"/>`
maxReadTimeout	Time in seconds to wait for getting response from the REST call before the read operation times out. Default: 180 seconds. For example, `<entry key="maxReadTimeout" value="200"/>`
maxRetryCount	Indicates the number of time read operation must be performed on the errors that appear. Default: 5 For example, `<entry key="maxRetryCount" value="6"/>`
retryableErrorsOnAgg	List of errors which must be retried if occurred during aggregation or get operation. Type: List of strings
userPartitions	List of filters to be used for creating partitions during partitioned aggregation. For more information, see "Partitioning Aggregation" .
userFilters	Filter that defines the scoping condition for Accounts to be applied during account aggregation to limit set of data. Following is an example for **userFilters** configuration attribute: `<entry key="userFilters" value="(startswith(displayName,'A')and accountEnabled eq true)"/>`
groupFilters	Filter that defines the scoping condition for Groups to be applied during account-group aggregation to limit set of data. Following is an example for **groupFilters** configuration attribute: `<entry key="groupFilters" value="(startswith(displayName,'A') or startswith(displayName,'B'))"/>`
partitionHost	(*Applicable only for partitioned account delta aggregation*) The **partitionHost** is the host on which all delta partitions must be executed. Following is an example for **partitionHost** configuration attribute: `<entry key="partitionHost" value= "myhost"/>`

Attributes	Description
skipMailEnabledGroup	If set to true, Mail Enabled Groups would be skipped during aggregation. Default value is false. For example, `<entry key="skipMailEnabledGroup" value="true"/>`
aggregateAllGroups	If it is set to true, all types of groups (that are, Security, Office 365 (Unified), Distribution List, Mail Enabled Security) would be aggregated. Default value is false. For example, `<entry key="aggregateAllGroups" value="true"/>`
fetchB2CMembership	(*Applicable for B2C tenant only*) If set to true B2C user memberships would be fetched for all users. Default value is false. For example, `<entry key="fetchB2CMemberships" value="true"/>`
teamsConfiguration	(*Applicable only when **teamsEnabled** is selected for Group creation*) Teams configuration with which all teams are created. Default settings are provided in **"Default Settings for Teams Configuration"** below.

Note: To aggregate EmployeeID attribute from Azure Active Directory, add the following entry key in the application debug page and add the employeeID attribute in account schema (for example, Name: employeeID, Type: string):

```
<entry key="api-version" value="1.6"/>
```

The employeeID attribute is fetched only by Azure API version 1.6.

Default Settings for Teams Configuration

Teams can be created by selecting **Group Type** as **Office 365** and selecting **teamsEnabled** in Group Creation. For Teams creation owner having **Teams** license must be provided. By default, Teams are created with the following settings:

```
"memberSettings": {
        "allowCreateUpdateChannels": true,
        "allowDeleteChannels": true,
        "allowAddRemoveApps": true,
        "allowCreateUpdateRemoveTabs": true,
        "allowCreateUpdateRemoveConnectors": true
    },
    "guestSettings": {
        "allowCreateUpdateChannels": false,
        "allowDeleteChannels": false
    },
    "messagingSettings": {
        "allowUserEditMessages": true,
        "allowUserDeleteMessages": true,
        "allowOwnerDeleteMessages": true,
        "allowTeamMentions": true,
            "allowChannelMentions": true
        },
        "funSettings": {
            "allowGiphy": true,
            "giphyContentRating": "strict",
            "allowStickersAndMemes": true,
            "allowCustomMemes": true
        }
}
```

To create Teams with custom settings, add the following entry key in the application debug page:

```
<key="teamsConfiguration" value="{"memberSettings": {"allowCreateUpdateChannels":
false,  "allowDeleteChannels": true, "allowAddRemoveApps": false,
"allowCreateUpdateRemoveTabs": false, "allowCreateUpdateRemoveConnectors": true}}"
/>
```

Note:

- The custom settings must be in valid JSON format and HTML escaping must be performed before adding to XML, that is, double quotes must be replaced with **&quot;**
- As per Microsoft documentation, Team creation within 15 minutes of Microsoft 365 group creation can fail with 404 error code due to replication delays. In such scenario Microsoft recommends to retry with 10 seconds delay between calls. SailPoint Azure Active Directory Connector follows this recommended pattern.

Schema Attributes

This section describes the different schema attributes.

Custom Attributes

In addition to the schema attributes listed in the following tables, the connector supports managing the extended attributes that are registered on the client application on Azure. Creation of Local User (B2C) also supports custom attributes.

Account Attributes

The following table lists the account attributes:

Name	Description
accountEnabled	True if the user is enabled; otherwise, false.
assignedLicenses	List of the licenses that are assigned to the user.
assignedPlans	Plans that are assigned to the user (Entitlement).

Name	Description
city	City in which the user is located.
country	Country/region in which the user is located.
department	Name for the department the user belongs to.
dirSyncEnabled	Indicates whether this user was synced from the on-premises directory.
disabledPlans	Plans that are not assigned to the user.
displayName	Name displayed in the address book for the user.
facsimileTelephoneNumber	Telephone number of the user's business fax machine.
givenName	First name of an user.
groups	Groups assigned to an user (Entitlement).
servicePrincipals	Enterprise Applications assigned to a user.
immutableId	Property used to associate an on-premises Active Directory user account to their Azure AD user account.
jobTitle	User’s job title.
lastDirSyncTime	Indicates the last time at which the user was synchronized with the on-premises directory.
mail	The SMTP address for the user. For example, `john@contoso.onmicrosoft.com`
mailNickname	Mail alias for the user.
signInNames	Specifies the collection of sign-in names for a local account in an Azure Active Directory B2C tenant.
userIdentities	Specifies the collection of userIdentities for a social user account in an Azure Active Directory B2C tenant.
creationType	Indicates whether the user account is a local account for an Azure Active Directory B2C tenant.
manager	Manager of the user. (Type: String). **Note: Azure Active Directory Connector now provides support for provisioning of the ‘manager’ attribute. For more information, see "Support for Provisioning Operations of ‘manager’ Attribute" .**
mobile	Primary cellular telephone number for the user.
objectId	Unique identifier for the user.
onPremisesSecurityIdentifie r	Contains the on-premises security identifier (SID) for the user that was synchronized from on-premises to the cloud.
otherMails	A list of additional email addresses for the user.
passwordPolicies	Specifies password policies for the user.
physicalDeliveryOfficeName	Office location in the user's place of business.
postalCode	ZIP OR postal code for the user's postal address.
preferredLanguage	Preferred written or spoken language for a person.

Name	Description
proxyAddresses	Proxy addresses. For example, `["SMTP: bob@contoso.com", "smtp: bob@sales.contoso.com"]`
roles	Administrator Role assigned to user (Entitlement).
sipProxyAddress	Specifies the voice over IP (VOIP) session initiation protocol (SIP) address for the user.
state	State or province in the user's address.
streetAddress	Street address of the user's place of business.
surname	Last name of the user.
telephoneNumber	Primary telephone number of the user's place of business.
usageLocation	A two letter country code indicating usage location.
userPrincipalName	User principal name (UPN) of the user.
userType	Type of the user.

Group Attributes

Name	Description
Native Object Type = group	
description	Description for the group.
dirSyncEnabled	Indicates whether this group was synced from the on-premises directory.
displayName	Display name for the group.
lastDirSyncTime	Indicates the last time at which the group was synced with the on-premises directory.
mail	SMTP address for the group.
mailEnabled	Specifies whether the group is mail-enabled
mailNickname	Mail alias for the group.
objectId	Group ID.
onPremisesSecurityIdentifier	Contains the on-premises security identifier (SID) for the group that was synchronized from on-premises to the cloud.
owners	Owner of the group.
proxyAddresses	Proxy addresses of the group.
securityEnabled	Specifies whether the group is a security group.
teamsEnabled	Specifies whether teams has been enabled for the group.
groupTypes	Type of the group. Blank for Security and /or **Unified** for Office 365 type of groups. For example, Unified for Office 365 group

Name	Description
Native Object Type = servicePrincipal	
displayName	ServicePrincipal name.
servicePrincipalType	ServicePrincipal type.
objectId	ServicePrincipal ID.
owners	ServicePrincipal Owners. Type: String, Multi-Valued
appRoles	ServicePrincipal Roles. Type: String, Multi-Valued

Provisioning Policy Attributes

This section lists different policy attributes for Azure Active Directory Connector.

Note: The attributes marked with * sign are required attributes.

Create Account Policy

Following table describes various attributes in the create account policy.

Attribute	Description
accountType*	To create user, set this value to **User**. Default is User
userPrincipalName*	User principal name (UPN) of the user. For example, `jeff@contoso.onmicrosoft.com`
password*	Password for the new account.
displayName*	Display name of the account.
mailNickname*	Mail alias for the account.
accountEnabled	Set it to false to create disabled account. Default: True
forceChangePasswordNextLogin	If true, asks user to change password on next login. Default: True
department	Department in which the user works.
jobTitle	User's job title.
isFederatedDomain	Set it true to create federated domain user. If this is checked and **immutableId** is not set then random **immutableId** value will be used.

Attribute	Description
immutableId	This property is used to associate an on-premises Active Directory user account to their Azure AD user account; Populate this attribute with objectGUID of account from on-premises Active Directory to create federated user synchronized with on –premises Active Directory user.
passwordPolicies	Specifies password policies for the user For example: DisablePasswordExpiration, DisableStrongPassword
otherMails	Additional email addresses for the user.
givenName	First name of the user.
surname	Surname of the user.
usageLocation	A two letter country code (ISO standard 3166). Required for users that will be assigned licenses.
country	Country/region in which the user is located. For example, **US** or **UK**
state	State or province in the user's address.
city	City in which the user is located.
streetAddress	Street address of the user's place of business.
postalCode	Postal code for the user's postal address.
physicalDeliveryOfficeName	Office location in the user's place of business.
preferredLanguage	Preferred language for the user. Should follow ISO 639-1 Code. For example, `en-US`
telephoneNumber	Primary telephone number of the user's place of business.
mobile	Primary cellular telephone number for the user.
facsimileTelephoneNumber	Telephone number of the user's business fax machine.
servicePrincipalRoleId	ServicePrincipal Role Id. By default not present in the schema. It is required if you want to assign ServicePrincipal during account creation. For more information, see "Assigning ServicePrincipal" .

Guest User (B2B) Support

Azure Active Directory Connector supports creation of Guest User (B2B) by sending invitations. Creation of Guest User (B2B) varies from normal user creation in terms of attributes provided during creation.

Create Guest User (B2B) Account Policy

Following table describes various attributes in the create Guest User (B2B) account policy.

Attribute	Description
accountType*	To create Guest User (B2B), set this value to **Guest User B2B**. Default is User
invitedUserEmailAddress*	Email address of the user.
inviteRedirectUrl*	The URL that the user will be redirected to after redemption.
sendInvitationMessage*	Set it to **False** if invitation email need not to be sent to the user. Default is True
invitedUserDisplayName	The display name of the user being invited.
invitedUserUsageLocation	A two letter country code indicating usage location (ISO standard 3166).
servicePrincipalRoleId	ServicePrincipal Role Id. By default not present in the schema. It is required if you want to assign ServicePrincipal during account creation. For more information, see "Assigning ServicePrincipal" .

Local User (B2C) Support

Azure Active Directory Connector supports creation of Local Users. Creation of Local User account varies from normal user in terms of attributes provided during creation. Account creation also supports custom attribute.

Create Local User (B2C) Account Policy

Following table describes various attributes in the create Local User (B2C) account policy.

Attribute	Description
accountType*	To create Local User (B2C), set this value to **Local User B2C**. Default is User
signInNameType*	Sign-in type for user in your Azure directory
signInNameValue*	Sign-in name for user.
localAccountDisplayName*	Display name of account.
b2cPassword*	Password for the new account.
enableLocalAccount	Set it to false to create disabled account. Default: True
b2cForceChangePasswordNextLogin	If true, asks user to change password on next login. Default: True

Attribute	Description
servicePrincipalRoleId	ServicePrincipal Role Id. By default not present in the schema. It is required if you want to assign ServicePrincipal during account creation. For more information, see "Assigning ServicePrincipal" .

Note: Custom user attribute can be added in B2C create account policy by appending suffix "_C" to the attribute.

Create Group Policy

Following table describes various attributes in the create group policy.

Attribute	Description
displayName*	Display name of the group.
owners	Owner of the group.
mailNickname*	The mail alias for the group.
groupTypes	Type of the group to be created, that is, Security or Office 365
teamsEnabled	Specifies whether teams has been enabled for the group.
addOwnerAsMember	Specifies if Owners must be added as Members in group.

Update Group Policy

Following table describes various attributes in the update group policy.

Attribute	Description
description	Description of the group.
owners	Owner of the group.
mailEnabled	True if it is Office 365 or mail enabled security group. Read only.
securityEnabled	True if it is security group or Office 365 type of group. Read only.
groupTypes	Type of the group, that is, Blank for Security and /or Unified for Office 365 type of groups. Read only.

Attribute	Description
teamsEnabled	Specifies whether teams has been enabled for the group. For Teams creation owner having **Teams** license must be provided. Teams can be enabled for existing Microsoft 365 group by selecting **Teams Enabled**. **Note:** For **SAML Bearer Authentication** method, Authorization User is the default owner of a Team but must be added as owner to enable Teams for existing Microsoft 365 Group.

Additional Information

This section describes the additional information related to the Azure Active Directory Connector.

SAML Request Body

The SAML Request Body must consists of the following:

- SAML Request must have MicrosoftOnline as the intended audience
- Username in request must be replaced by the **$username$** place holder
- Password in request must be replaced by **$password$** place holder

Following is an example of a sample SAML Request Body:

Authorization Endpoint: https://<YOUR_DOMAIN>/adfs/services/trust/2005/usernamemixed

SAML Request:
```
<s:Envelope xmlns:s='http://www.w3.org/2003/05/soap-envelope'
xmlns:a='http://www.w3.org/2005/08/addressing'
xmlns:u='http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility
-1.0.xsd'>
  <s:Header><a:Action
s:mustUnderstand='1'>http://schemas.xmlsoap.org/ws/2005/02/trust/RST/Issue</a:Actio
n>
<a:ReplyTo><a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address></a
:ReplyTo>
        <a:To
s:mustUnderstand='1'>https://<YOUR_DOMAIN>/adfs/services/trust/2005/usernamemixed</
a:To>
          <o:Security s:mustUnderstand='1'
xmlns:o='http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-
1.0.xsd'>
            <o:UsernameToken u:Id='uuid-<user’s object id>'>
        <o:Username>$username$</o:Username>
        <o:Password>$password$</o:Password></o:UsernameToken>
    </o:Security>
    </s:Header>
    <s:Body>
            <trust:RequestSecurityToken
xmlns:trust='http://schemas.xmlsoap.org/ws/2005/02/trust'>
```

```
            <wsp:AppliesTo xmlns:wsp='http://schemas.xmlsoap.org/ws/2004/09/policy'>
            <a:EndpointReference>
            <a:Address>urn:federation:MicrosoftOnline</a:Address>
            </a:EndpointReference>
            </wsp:AppliesTo>

<trust:KeyType>http://schemas.xmlsoap.org/ws/2005/05/identity/NoProofKey</trust:Key
Type>

<trust:RequestType>http://schemas.xmlsoap.org/ws/2005/02/trust/Issue</trust:Request
Type>
            </trust:RequestSecurityToken>
  </s:Body>
</s:Envelope>
```

Assigning ServicePrincipal

While providing role id from policy, if user has configured policy for specifying role id, then this role id would be used for provisioning of all **ServicePrincipals** in a request. Hence, user must provision one **ServicePrincipal** in one request.

While providing an access to a **ServicePrincipal**, it is mandatory to provide an access role for the User. These access roles may differ depending upon the **ServicePrincipal** type. IdentityIQ would use one of the access role it receives for each **ServicePrincipal** during the aggregation of **ServicePrincipals**.

Note: The 'servicePrincipalRoleId' attribute must be added in 'Update Account Policy' while assigning ServicePrincipals from Request Access.

Assigning Application Permissions

This section provides details about assigning application permissions from Portal and using PowerShell.

Assigning Application Permission from Portal

- Click **API permissions** section registered App in Azure Active Directory console. Click **Add a permission**.
- On the **Request API permissions** page a list of supported APIs is listed.
    - **Microsoft Graph** can be found under Commonly used Microsoft API
    - **Azure Active Directory Graph** can be found under Supported legacy API.
- Select the type of permission as **Delegated** or **Application** under **What type of permissions does your application require?**
- Under **Select permissions**, select the required permissions.
- Click **Add permissions**.

In Grant consent, click **Grant** Admin Consent for your configuration and directory. On the pop-up dialog box, click **Yes**.

Assigning Application Role using PowerShell

Following are the prerequisite for the PowerShell commands.

- Microsoft Online Services Sign-In Assistant for IT Professionals RTW
- Azure Active Directory Module for Windows PowerShell

Note: These prerequisites are not required for the Azure Active Directory Connector to function. These can be installed on any system for temporary use to provide required role to the application on Azure.

After installing the prerequisites, open **Azure Active Directory Module for Windows PowerShell** console and execute the following commands:

- `Connect-msolservice`, press enter, provide Azure administrator credentials.
- Execute `Get-MsolServicePrincipal | ft DisplayName, AppPrincipalId -Autosize`
- Locate your application name and copy the **ObjectId** value.
- Execute `$ClientObjID = '<copied objectId of the application in the previous step>'`
- Execute `$webApp = Get-MsolServicePrincipal -AppPrincipalId $ClientObjID`
- Execute `Add-MsolRoleMember -RoleName "Company Administrator" -RoleMemberType ServicePrincipal -RoleMemberObjectId $webapp.ObjectId`

Managing Licenses

Azure Active Directory Connector supports assigning different Azure services licenses to the users. Connector provides options to assign license either by individual plan or as a whole license pack.

- **Assigning license plan**: Office 365 license pack consist of licenses for individual services. For example, Exchange Online, SharePoint Online and so on.

  The connector models **assignedPlans** attribute from account schema as an entitlement. It can be requested as an entitlement during **Create** or **Update** operations for Identities.
- **Assigning license pack**: To assign license pack, set **assignedLicenses** attribute from account schema as **Managed**, **Entitlement**, **Multi-Valued**, So that it request-able as an entitlement.

Note: It is recommended that either 'assignedPlans' or 'assignedLicenses' must be promoted as an entitlement to avoid conflicts.

Note: To provision the licenses or plans to user, set the user's 'usageLocation' property correctly.

IQService Before/After Script Attributes

Attributes to be used only for customization (for example, IQService Before/After Script), must be passed as Attributes map of the **AccountRequest** instead of AttributeRequest. For more information, see the example of **AccountRequest** xml provided in the IQService Before/After Scripts.

Upgrade Considerations

- To manage Office 365 type of groups after upgrading IdentityIQ to version 8.1 Patch 1, add the following:

Attribute Name	Group Schema	Create Group Policy	Update Group Policy
groupTypes	• Property: Multi-Valued • Data Type: string • Description: Type of the group	• Review Required: true • Required: true • Type: String • Allowed Values: Security, Office 365	• Multi-Valued: true • Type: String • Read Only: True
securityEnabled	NA	NA	• Type: Boolean • Read Only: True

- For creating and enabling Teams after upgrading IdentityIQ to version 8.1 Patch 1, add the following:

Attribute Name	Action	Group Schema	Create Group Policy
teamsEnabled	Distinguish Teams group	Type: Boolean	NA
	Enabling Teams creation	NA	• Type: Boolean • Review Required: True
	Enabling Teams creation for existing Microsoft Office 365 Groups	NA	• Type: Boolean • Review Required: True
Owners	NA	• Property: Multi-Valued • Data Type: string • Description: Owners of **the group**	Read-Only: False

Support for Provisioning Operations of 'manager' Attribute

Azure Active Directory Connector now provides support for create, update and delete operations for the **manager** attribute when the value of the attribute can be passed in **userPrincipalName** or **objectId** format. This would require addition of **manager** attribute in the provisioning policy attributes.

Delta Aggregation

Delta Aggregation is supported for SailPoint Azure Active Directory. On Full Aggregation, the respective delta link of account and group aggregation are stored in the Application object which are used by Delta Aggregation to retrieve the changed data into IdentityIQ. Same values are updated after each respective delta aggregation.

Note: **- Account Delta Aggregation does not capture role assignment changes.**
- Delta Aggregation triggered without first full aggregation would trigger respective full aggregation by default.

Partitioning Aggregation

Azure Active Directory Connector supports partitioning aggregation based on search filters. To use partitioning feature perform the following:

1. Enable Partitioning on the aggregation task definition page by selecting the **Enable Partitioning** check box.
2. Add the following application configuration attribute:

   ```
   <entry key="userPartitions">
   ```

   The **userPartitions** configuration attribute is a multi-valued attribute. It's value consists of different search filters for the attributes which are filterable like accountEnabled, city, displayName, mail, usageLocation and so on.

   For example,

   ```
   <entry key="userPartitions">
       <value>
           <List>
                <String>startswith(displayName,'J')</String>
                <String>startswith(givenName,'Smith')</String>
                <String>accountEnabled eq true</String>
               <String>userPrincipalName eq 'Paul@contoso.onmicrosoft.com'</String>
           </List>
       </value>
   </entry>
   ```

For large environments, for faster delta aggregation of the accounts, the connector supports partition delta aggregation.

Supported operators are

- Logical operators: **and**, **or**
- Comparison operators: **'eq'(equal to)**, **ge' (greater than or equal to)** and **'le'(Less than or equal to)**
- **startswith**
- **any** is supported while querying multi valued properties

For example,

- `proxyAddresses/any(c:c eq 'smtp:Mary@contoso.com')`
- `proxyAddresses/any(c:startswith(c,'smtp:Mary@contoso.com'))`

Troubleshooting

1 - While adding a Service Principal Name to a user, the add entitlement request fails with an error message

While adding a Service Principal Name to a user, the add entitlement request fails with the following error message:

```
Response Code - 400 Error - Permission being assigned was not found on application
```

Resolution: Verify if the user already has the Service Principal Name added in entitlements.

2 - Test Connection fails with an error message when Azure Active Directory application is checked for ‘SAML Bearer Assertion’ Grant Type

Test Connection fails with the following error message when Azure Active Directory application is checked for ‘SAML Bearer Assertion’ Grant Type:

```
OAuth2Exception [toString()=connector.common.oauth2.OAuth2Exception: Unable to
generate access token. Response returned:

{"error":"invalid_grant","error_description":"AADSTS50008: The SAML token is
invalid.\r\nTrace ID: a74df376-3ede-4c17-ba34-b352079e3300\r\nCorrelation ID:
f8c370ec-4ef6-48a4-a393-0297a5ce3b20\r\nTimestamp: 2020-05-04
05:58:16Z","error_codes":[50008],"timestamp":"2020-05-04
05:58:16Z","trace_id":"a74df376-3ede-4c17-ba34-b352079e3300","correlation_id":"f8c3
70ec-4ef6-48a4-a393-0297a5ce3b20","error_uri":"https://login.microsoftonline.com/er
ror?code=50008"}

]
```

Resolution: Verify if the time zone of ADFS machine is in synchronize with Azure time zone, that is, UTC. If not, change the ADFS machine time and re-start the ADFS services.

3 - Test Connection/Account Aggregation fails with an error message when Azure Active Directory application is checked for ‘SAML Bearer Assertion’ Grant Type

Test Connection/Account Aggregation fails with the following error message when Azure Active Directory application is checked for **SAML Bearer Assertion** Grant Type:

```
Error - invalid_grant: AADSTS5000811: Unable to verify token signature. The signing
key identifier does not match any valid registered keys.
```

Resolution: Microsoft recommends to execute the following command from PowerShell running on ADFS server to manually renew token signing certificates:

```
Update-MSOLFederatedDomain –DomainName <domain>
```