Integrating Amazon Web Services

Version: 8.4

Copyright and Trademark Notices

Contents

Integrating SailPoint and Amazon Web Services **1**

Supported Features 1

Prerequisites 4

Required Permissions 5

Connecting SailPoint and Amazon Web Services **17**

Configuring the Connector in SailPoint 17

Configuration Parameters 19

Review and Test 23

Schema Objects and Attributes **24**

Identity Attributes 25

Account Attributes 25

Group Attributes 26

Provisioning Policy Attributes **30**

Account 30

Account-Group 31

Multiple Group Objects Support **34**

Group Entitlements 34

Using Multiple Group Entitlements with a Pre-existing Source 35

Operation Specific Service IAM User permissions **50**

Organization APIs 55

(Optional) Upgrade Consideration **57**

Troubleshooting **58**

Integrating SailPoint and Amazon Web Services

Revised Date: 14 September 2023

> **Note**
>
> IdentityIQ Connector information is now available as online help and PDF. The online help describes the latest updates for the connector.
>
> To find documents related to a specific version of IdentityIQ, refer to the Supported Connectors for IdentityIQ page on Compass.
>
> Configuration details for connectors may vary not only by release version but also by patch version. Be sure to refer to the correct documentation for your specific release and patch level.

The SailPoint Amazon Web Services (AWS) Connector enables organizations to extend existing identity lifecycle and compliance management capabilities within SailPoint to mission-critical AWS IaaS environments to provide a central point of visibility, administration, and governance across the entire enterprise. This includes policy discovery and access history across all organization accounts, provisioning AWS entities and objects, access review and certification, and federated access support.

IdentityIQ for Amazon Web Services manages the AWS Organizations entities such as Service Control Policies, Organization Units and AWS Accounts. It also manages the IAM (Identity Access Management) entities such as Users, Groups, Roles, Inline policies, Managed policies (AWS and Customer managed) under each AWS Account.

This document is designed to give specific information about the requirements and field definitions needed to get a working instance of an Amazon Web Services (AWS) source.

> **Important**
>
> You must have an IdentityIQ Cloud Governance license to enable cloud governance features. If you already have a CAM license, no additional license purchase required. Contact your SailPoint Customer Success Manager to request access and for more information.

What's New in 8.4

- Supports AWS GovCloud (US) Regions.

Supported Features

The AWS source supports the following features:

- Load AWS Accounts
- Provision AWS Accounts
- Access Certifications (certification of entitlements connected to AWS accounts)
- Password management

Account Management

- Manage IAM Users under the AWS Account as Accounts
- Aggregate, Refresh Accounts
- Create, Update, Delete

> **Note**
> For more information on enabling and disabling, see IAM User Status.

- Change Password
- Add/Remove Entitlements (Groups, AWS Managed Policies, Customer Managed Policies, Inline Policies)
- Create, Update, and Delete Inline Policies for IAM Users.

IAM Entities

- **IAM Groups**: Aggregate, Refresh Group, Create, Update, Delete, Create-Update-Delete Inline Policy
- **AWS Managed Policy Management**: Aggregate, Refresh
- **Customer Managed Policies**: Aggregate, Refresh, Create, Update

> **Note**
> Updating the **Customer Managed Policy** creates a new policy version.

- **Inline Policies**: Aggregate, Refresh, Update for User and Group
- **Role Management**: Aggregate, Refresh, Update (Add/ Remove AWS Managed Policy or Customer Managed Policy from Role)

Tags Management

The AWS connector supports the aggregation and refresh of tags attribute for the following entities:

- IAM User
- IAM Role
- Customer Managed Policy
- Service Control Policy
- Organization Unit
- AWS Account

Organization Entities

The AWS Connector supports the following on Organization Entities (managed as group object only):

- AWS Accounts Management: Aggregate, Refresh
- Organization Unit Management: Aggregate, Refresh
- Service Control Policy Management: Aggregate, Refresh

Permissions Management

AWS Connector supports JSON Policy for Permission Policy and Trust Policy as direct permission.

The Permission Policy for the following AWS entities are represented as Permissions:

- AWS Managed Policies
- Customer Managed Policies
- Inline Policies
- Service Control Policies

The Trust Policy for the following AWS entity is represented as direct permission:

- Roles

> **Note**
> *Role aggregation takes care of aggregating the trust polices (entities that can assume a role) as direct permission.

IAM User Status

The following are the SailPoint operations with the corresponding IAM User Status:

Enable

- Set Console Password (This would also activate the Signing Certificate if it is associated with an IAM User.)
- Activates Last Created Access Keys
- Activates Last CreatedAWS CodeCommit HTTPS Credentials
- Activates Last CreatedAWS CodeCommit SSH Keys
- Activates Signing Certificates

Disable

- Deletes Console Password
- Inactivates Both Access Keys
- Inactivates Both AWS CodeCommit HTTPS Credentials
- Inactivates All AWS CodeCommit SSH Keys
- Inactivates Signing Certificates

> **Note**
> When you enable an AWS IAM (Identity Access Management) user for the AWS source it activates the Signing Certificate (if the certificate is associated with that user) along with setting the console password. When you disable the user, the source will deactivate the Signing Certificate.

Prerequisites

- IAM role authentication requires an AWS EC2 instance to perform aggregation and provisioning operations.
- Based on authentication method, create an IAM user or IAM role and assign required permission to it so that it uses all the cross-account roles.
- Create a customer/Inline managed policy in each AWS account that you want to manage with policy document specified in Multiple Group Object Source Policies and Non Multiple-group Object Source Policies.

- Create Cross Account Roles in each AWS account that you want to manage and attach the appropriate policies to the role.
- Based on the authentication method, create the IAM user or IAM role and assign required permission to it so that it can assume all cross-account roles.
- For each AWS cross-account role, establish the trust relationship with the IAM user or IAM role.

> **Note**
> Ensure you create the cross-account role across the AWS Accounts with the same name and assign the permissions as mentioned.

- The AWS System Administrator can refine the Permission Policies as needed.

Required Permissions

This section provides the administrator permissions for the following users authentication method:

- IAM Role Authentication Method
- IAM User Authentication Method

See Non Multiple-group Object Source Policies or Multiple Group Object Source Policies for examples of required policies.

Set Up Service User or Service Role

You must attach this policy document using the Inline policy to the service user or service role.

```
 {

"Version": "2012-10-17",

"Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": [
          "sts:AssumeRole"
        ],
        "Resource": "arn:aws:iam::<<AWS account ID>>:role/<<cross account role
name>>"
    }
  ]
```

```
}
<<cross account role name>> is a cross account role that has the customer
managed/inline policies mentioned above and it enables the service
account/role to perform all the necessary tasks needed for the source.
```

> **Note**
> This policy must be created under each AWS account that you want to manage. The <<AWS account ID>> above is the ID of the Master Account ID.

GetUserInlinePolicy Document

`iam:GetUser` API permission is required when the authentication method is IAM User and you want to manage organization group objects.

```
{

"Version": "2012-10-17",

"Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": [
          "iam:GetUser"
        ],
        "Resource": "arn:aws:iam::<Service User's AWS Account Id>:user/<<Service
User name>>"
    }
  ]
}
```

IAM Role Authentication Method

- IAM Role based Authentication can be used when SailPoint is hosted on the AWS EC2 instance.
- The EC2 instance must not have IAM User AWS credentials stored as credential chain.
- The EC2 instance can be present in any of the AWS Accounts (that is, either the Management AWS Account or in Member AWS Account).

See Non Multiple-group Object Source Policies or Multiple Group Object Source Policies for examples of these policies.

Trust relationship

The role must be added in the Account from where the data would be aggregated

```
arn:aws:iam::AccountId1:role/<Cross Role created in AWS accounts>
```

> **Note**
> The External ID can also be provided while creating the Role.

Assume Role Permissions for EC2 Instances

For Role Authentication, the role associated with the EC2 instance must have the assume role permissions with the common role across the AWS accounts from where the data must be aggregated.

For example, the following is the JSON format for the policy permissions:

```
{
  "Version": "2012-10-17",
  "Statement": [
     {
       "Effect": "Allow",
       "Principal": {
       "AWS": [
              "arn:aws:iam::AccountId1:role/<Cross Role created in AWS accounts>"
             ]
               },
                "Action": "sts:AssumeRole",
                "Condition": {
                "StringEquals": {
                                "sts:ExternalId": "External ID associated with
role"
           }
         }
     }
   ]
}
```

IAM User Authentication Method

Customer Managed Policies must be created and attached to the AWS Service IAM User and Role respectively as mentioned in the table below.

> **Note**
> The AWS System Administrator can refine the Permission Policies as needed.

> **Note**
> If 'Include AWS Account IDs' list is specified and organization schema is not present in the application, then 'iam:GetUser' API permission is not required for AWS Service IAM User.

The description for the policy name and role that are used is as follows:

- **SPServiceIAMUser**: an IAM account in the management (or designated Service IAM User) account that is used as the connector's service account to your AWS environment.
- **SPOrganizationPolicy**: allows management of Organization entities. This will only be created if the ServiceIAMUser is created in your organization's management AWS account.
- **SPAggregationPolicy**: allows mostly read access in order to aggregate IAM entities from your AWS environment.
- **SPProvisioningPolicy**: allows write access for provisioning IAM entities back to your AWS environment.
- **SPServiceIAMUserAccess**: a role that will have the above mentioned policies and will allow the ServiceIAMUser to perform all the necessary tasks needed for the connector to work.

See Non Multiple-group Object Source Policies or Multiple Group Object Source Policies for examples of these policies.

Create Cross Account Roles

To aggregate the data present in AWS accounts in an organization, the AWS Connector uses the assume role functionality of the AWS System. This functionality helps return data from different AWS accounts.

Create the cross-account role to allow the IAM user or IAM role from one AWS Account to access the resources in another AWS Account.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          arn:aws:iam::<IAM role AWS Account ID>:role/<IAM role assciated with EC2
instance>,
          OR
          arn:aws:iam::<IAM user AWS Account ID>:user/<IAM user name>
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Trusted entities in the above case can be the IAM user or IAM Role associated with the EC2 instance based on the authentication method selected.

Non Multiple-group Object Source Policies

SPAggregationPolicy

This aggregation policy must be assigned to the role of AWS accounts you want to manage.

```
"Version": "2012-10-17",
"Statement": [
{
"Sid": "VisualEditor0", "Effect": "Allow", "Action": [
            "iam:GetPolicyVersion",
            "iam:ListServiceSpecificCredentials",
            "iam:ListMFADevices",
            "iam:ListSigningCertificates",
            "iam:GetGroup",
            "iam:ListSSHPublicKeys",
            "iam:ListAttachedRolePolicies",
            "iam:ListAttachedUserPolicies",
            "iam:ListAttachedGroupPolicies",
            "iam:ListRolePolicies",
            "iam:ListAccessKeys",
            "iam:ListPolicies",
            "iam:GetRole",
            "iam:GetPolicy",
            "iam:ListGroupPolicies",
            "iam:ListRoles",
            "iam:ListUserPolicies",
            "iam:GetUserPolicy",
            "iam:ListGroupsForUser",
            "iam:ListAccountAliases",
            "iam:ListUsers",
            "iam:ListGroups",
            "iam:GetGroupPolicy",
            "iam:GetUser",
            "iam:GetRolePolicy",
            "iam:GetLoginProfile",
            "iam:ListEntitiesForPolicy",
            "iam:GetAccessKeyLastUsed",
            "iam:ListUserTags",
           "iam:ListRoleTags",
           "iam:ListPolicyTags"
],
        "Resource": "*"
    }
] }
```

SPProvisioningPolicy

Must be assigned to the Role of AWS Account which needs to be managed.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
              "iam:UpdateLoginProfile",
              "iam:UpdateAccessKey",
              "iam:CreateUser",
              "iam:CreateAccessKey",
              "iam:CreateLoginProfile",
              "iam:RemoveUserFromGroup",
              "iam:AddUserToGroup",
              "iam:DeleteLoginProfile",
              "iam:CreatePolicyVersion",
              "iam:PutUserPolicy",
              "iam:AttachGroupPolicy",
              "iam:AttachUserPolicy",
              "iam:DetachGroupPolicy",
              "iam:DetachUserPolicy",
              "iam:DeleteGroupPolicy",
              "iam:DeleteUserPolicy"

          ],
          "Resource": "*"
      }
  ]
}
```

Multiple Group Object Source Policies

Examples of policies for the respective policy names:

For AWS Service IAM User:

SPServiceIAMUSer

```
{   "Version": "2012-10-17",
  "Statement": [
   {
    "Sid": "VisualEditor0",
    "Effect": "Allow",
    "Action": [
      "sts:AssumeRole"
        ],
    "Resource":         "arn:aws:iam::*:role/SPServiceUserAccountAccess"
   }
  ]
}
```

> **Note**
> The above role name is an example. Replace **SPServiceUserAccountAccess** with the specific role name that was created on your AWS system.

For Role:

SPOrganizationPolicy

Required for Multiple Group Object Source and must be assigned to the Role of the AWS Account which needs to be managed.

```
{"Version": "2012-10-17",
 "Statement": [
   {
    "Sid": "VisualEditor0",
    "Effect": "Allow",
    "Action": [
      "organizations:ListPoliciesForTarget",
      "organizations:ListAccountsForParent",
      "organizations:ListRoots",
      "organizations:ListAccounts",
      "organizations:ListTargetsForPolicy",
      "organizations:DescribeOrganization",
      "organizations:DescribeOrganizationalUnit",
      "organizations:DescribeAccount",
      "organizations:ListParents",
      "organizations:ListOrganizationalUnitsForParent",
      "organizations:DescribePolicy",
    "organizations:ListPolicies",
     "organizations:ListTagsForResource"
],
  "Resource": "*"
    }
] }
```

SPAggregationPolicy

Required for Multiple Group Object Source and must be assigned to the Role of the AWS Account which needs to be managed.

```
"Version": "2012-10-17",
"Statement": [
{
"Sid": "VisualEditor0", "Effect": "Allow", "Action": [
            "iam:GetPolicyVersion",
            "iam:ListServiceSpecificCredentials",
```

```
            "iam:ListMFADevices",
            "iam:ListSigningCertificates",
            "iam:GetGroup",
            "iam:ListSSHPublicKeys",
            "iam:ListAttachedRolePolicies",
            "iam:ListAttachedUserPolicies",
            "iam:ListAttachedGroupPolicies",
            "iam:ListRolePolicies",
            "iam:ListAccessKeys",
            "iam:ListPolicies",
            "iam:GetRole",
            "iam:GetPolicy",
            "iam:ListGroupPolicies",
            "iam:ListRoles",
            "iam:ListUserPolicies",
            "iam:GetUserPolicy",
            "iam:ListGroupsForUser",
            "iam:ListAccountAliases",
            "iam:ListUsers",
            "iam:ListGroups",
            "iam:GetGroupPolicy",
            "iam:GetUser",
            "iam:GetRolePolicy",
            "iam:GetLoginProfile",
            "iam:ListEntitiesForPolicy",
            "iam:GetAccessKeyLastUsed",
            "iam:ListUserTags",
           "iam:ListRoleTags",
           "iam:ListPolicyTags"
],
        "Resource": "*"
    }
] }
```

SPProvisioningPolicy

Required for Multiple-group Object Source and must be assigned to the Role of the AWS Account which needs to be managed.

```
{    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "iam:UpdateLoginProfile",
                "iam:CreateGroup",
                "iam:DeleteAccessKey",
                "iam:DeleteGroup",
                "iam:AttachUserPolicy",
```

```
                "iam:DeleteUserPolicy",
                "iam:UpdateAccessKey",
                "iam:AttachRolePolicy",
                "iam:DeleteUser",
                "iam:CreateUser",
                "iam:CreateAccessKey",
                "iam:CreatePolicy",
                "iam:CreateLoginProfile",
                "iam:RemoveUserFromGroup",
                "iam:AddUserToGroup",
                "iam:DetachRolePolicy",
                "iam:DeleteSigningCertificate",
                "iam:AttachGroupPolicy",
                "iam:DeleteRolePolicy",
                "iam:DetachGroupPolicy",
                "iam:DetachUserPolicy",
                "iam:DeleteGroupPolicy",
                "iam:DeleteLoginProfile"
            ],
            "Resource": "*"
        }
    ]
}
```

Note

- For all provisioning operations, in addition to the provisioning policy permissions listed for SPProvisioningPolicy the permissions for Refresh Operations are also required.
- For more information on operation specific administrator permissions required for IAM and Organization APIs, see Operation Specific Service IAM User permissions.

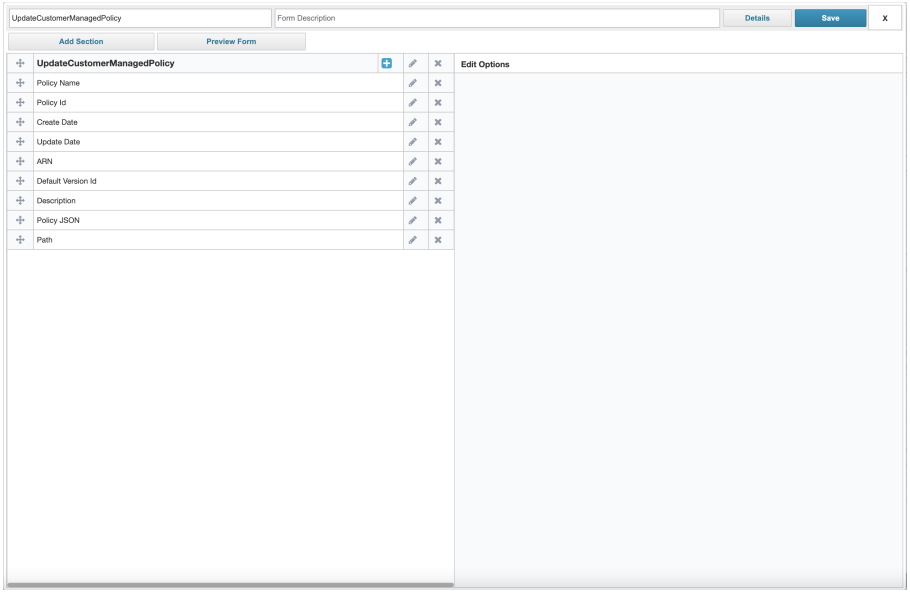
Update of Customer Managed Policy

To add a customer managed policy, complete the following:

1. Add an update policy form for the Customer Managed Policy group type.
2. Add the following attribute in the Policy Form:
    - `PolicyName`
    - `PolicyId`
    - `CreateDate`

- `UpdateDate`
- `ARN`
- `DefaultVersionId`
- `Description`
- `PolicyJSON`
- `Path`

3. Make all fields read-only except `PolicyJSON` as the connector only supports Policy JSON update.



> **Note**
> AWS supports up to 5 policy versions for the customer managed policy. If you want to add more versions, you must delete an older version associated with the policy.

Creating and Updating Inline Policy for Group and User

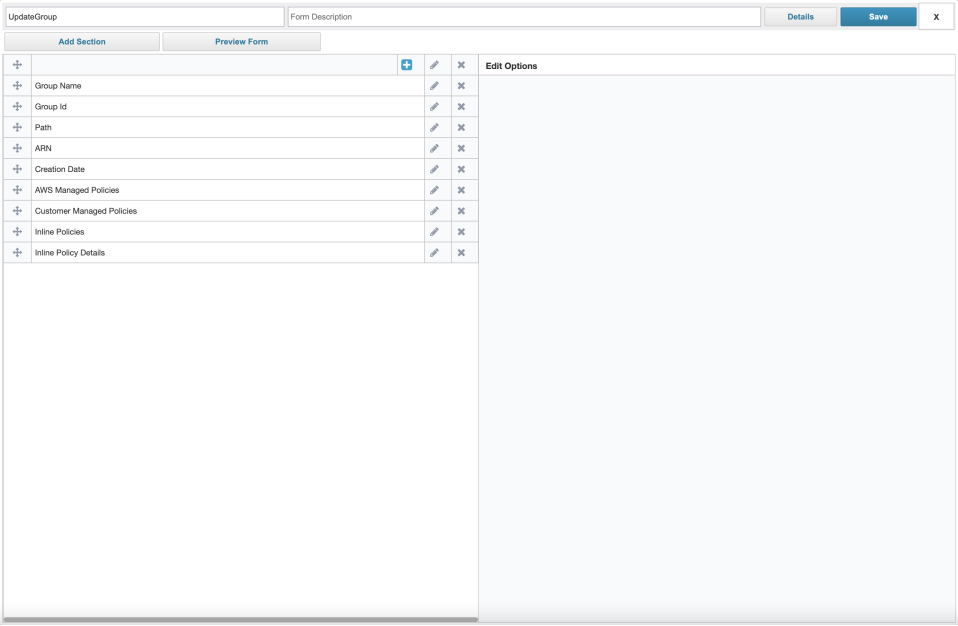
Use the following for creating inline policies for Group and User.

Inline Policy for Creating the IAM Group

In the existing Policy Form for UpdateGroup, add the following attribute:

```
inlinePolicyDetail(type=string and multi-valued, Editable=true)
```

Screenshot



- Inline Policy for Creating the IAM User

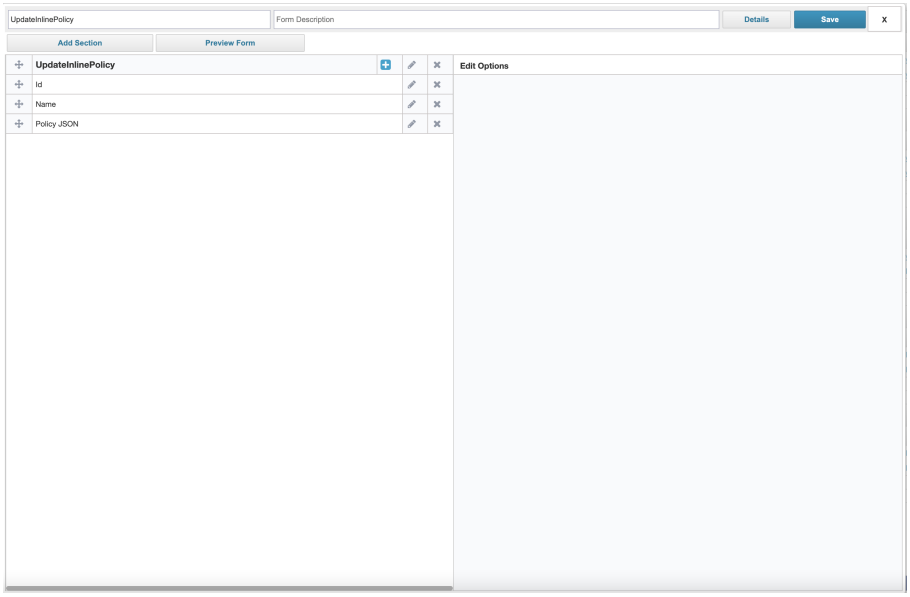
  To create an Inline Policy for a user, add the `inlinePolicyDetails` attribute in the account schema and make it multi-valued.

  Inline Policy for Updating the Group and User

  Add the following attribute in the inlinePolicy update form:

  - `id`
  - `Name`
  - `Policy JSON`

Screenshot



Updating of the inline policy attached to a Group and User only supports the `PolicyJSON` attribute. To perform an update, add the `featureString` for `InlinePolicy` schema and make the `PolicyJSON` attribute editable in the Inline Policy Update Form.

Use the following to add the `featureString`:

```
featureString="PROVISIONING, NO_GROUP_PERMISSIONS_PROVISIONING"
```

To create the inline policy for the Group and User, provide the JSON in the following format:

```
{
   "policyName": "InlinePolicyName",
    "policyDocument":
    {
       "Version": "2012-10-17",
       "Statement": [
       {
          "Sid": "VisualEditor0",
          "Effect": "Allow",
          "Action": "iam:ListRoles",
          "Resource": "*"
        }
      ]
    }
}
```

Connecting SailPoint and Amazon Web Services

To connect SailPoint and Amazon Web Services, perform the following tasks:

Configuring the Connector in SailPoint 17

Configuration Parameters 19

Review and Test 23

Configuring the Connector in SailPoint

An application is an instance of third-party software connected to IdentityIQ. The connector is configured to seamlessly provide governing and provisioning access to the application. The connector configuration includes all of the configuration and connection details required to connect IdentityIQ to the application.

> **Note**
>
> This procedure provides the basic information necessary to connect your connector. For additional information, refer to the Application Configuration Guide PDF for your deployed version of IdentityIQ found in the IdentityIQ Product Guides page on Compass.
>
> To view the latest online IdentityIQ guides, refer to the Documentation Portal.

> **Caution**
> Do not open the application configuration in multiple tabs or browsers. Doing so may overwrite changes made in the other.

1. Go to **Applications > Application Definition**.
2. Select **Add New Application**.
3. The **Edit Application** page opens to the **Details** page. Enter the following information:
    - **Name** – The name of the application. This is the named used to identify the application throughout IdentityIQ.

> **Note**
> IdentityIQ does not support application names that start with a numeric value or that are longer than 31 characters.

- **Owner** – The owner of the application. The owner specified here is responsible for certifications and account group certifications requested on this application if no revoker is specified.

  Application ownership can be assigned to an individual identity or to a workgroup. If the application ownership is assigned to a workgroup, all members share certification responsibilities, are assigned certification requests associated with the application, and all can take action on those requests.

- **Application Type** – The dropdown list contains the applications to which IdentityIQ can connect. This list will grow and change to meet the needs of IdentityIQ users.

- **Description** – The brief description of the application. Use the language selector to enter the description in multiple languages. The dropdown list displays languages supported by your instance of IdentityIQ.

- **Revoker** – The default IdentityIQ user or workgroup to be assigned revocation requests associated with entitlements on this application. If no user is specified in this field, all revocation requests are assigned to the to application owner by default.

- **Proxy Application** – Specify an application to manage accounts and provide connector and schema settings for this application. The proxy application is an application that handles the processing (aggregation and provisioning) on behalf of your application.

- **Profile Class** – A class used to associate this application with a larger set of applications for role modeling purposes.

- **Authoritative Application** – Select this option if this application in an authoritative application. An authoritative source is a repository for employee information for your enterprise that represents the primary and most trusted information about identities, such as a human resources application.

- **Case Insensitive** – Select this option to remove case sensitivity and ignore capitalization differences within values.

- **Native Change Detection** – Select this option if this application should be included when IdentityIQ performs native change detection during aggregation.

  - **Native Change Operations** – Select which operations are included when detecting native change. If no operations are selected, native change detection is disabled.

  - **Attributes to Detect** – Indicates which attributes are compared when accounts are modified. If the Entitlement option is selected, all entitlement attributes are included. If you select User Defined, enter the name of the attributes to compare in the Attribute Names box.

- **Maintenance Enabled** – Select this option to exclude this application from provisioning and aggregation during the defined maintenance period.

  For more information, refer to Application Maintenance Windows.

  - **Maintenance Expiration** – The date at which the maintenance will end. If no date is defined, this application will be in maintenance indefinitely.

- **Extended Attributes** – This section displays any extended attributes that were configured for your deployment of IdentityIQ.

- For more information on the fields displayed on the **Details** page, refer to the IdentityIQ Application Configuration Guide for your release.

4. Select **Configuration** and enter the information required for IdentityIQ to connect and interact with the application. The information required varies by application.
5. Select **Save**.

Configuration Parameters

> **Note**
> Parameters with * are mandatory parameters.

The configuration parameters of AWS are as follows:

Authentication Method

Select the method that would be used to securely connect to AWS:

- IAM User
- IAM Role

Applicable if Authentication Method is selected as IAM User

Access Key ID*

Enter the Access Key ID of the Service IAM User.

Secret Access Key*

Enter the Secret Access Key of the Service IAM User.

Role Name

Enter the role name that is created in all the AWS Accounts that are to be aggregated.

If the Amazon Resource Name (ARN) of the role contains a path, then it should be created with same path and name in all the AWS accounts. The input value must be provided as follows:

```
<entire Role Path>/<Role Name>.
```

Manage All Accounts

When checked, will manage IAM entities from all the AWS accounts.

Exclude AWS Account IDs

Lists all the AWS Account IDs, separated by a comma, that are to be excluded.

Include AWS Account IDs

Lists all the AWS Account IDs, separated by a comma, that are to be included.

Region

Enter the Region as per your AWS instance. For example, "us-east-1" for AWS commercial cloud and "us-gov-west-1" for AWS GovCloud (US).

Page Size

The maximum size of each data set when querying over large number of objects for IAM entities. Default: 100

Applicable if Authentication Method is selected as IAM Role

Role Name

Enter the role name that is created in all the AWS Accounts that are to be aggregated.

If the Amazon Resource Name (ARN) of the role contains a path, then it should be created with same path and name in all the AWS accounts. The input value must be provided as follows:

```
<entire Role Path>/<Role Name>.
```

External ID

Enter the External ID that is used in an IAM role trust policy to designate who can assume the role.

> **Note**
> This is mandatory if the external ID condition is provided to the IAM Role trust policy. This condition defines how and when trusted entities can assume the role.

Management Account ID

Enter the Management Account ID of the AWS organization.

> **Note**
> Applicable if the **Manage All Accounts** checkbox is selected or Organization entities are present in the application schema.

Manage All Accounts

When checked, will manage IAM entities from all the AWS accounts.

Exclude AWS Account IDs

Lists all the AWS Account IDs, separated by a comma, that are to be excluded.

Include AWS Account IDs

Lists all the AWS Account IDs, separated by a comma, that are to be included.

Region

Enter the Region as per your AWS instance. For example, "us-east-1" for AWS commercial cloud and "us-gov-west-1" for AWS GovCloud (US).

Page Size

The maximum size of each data set when querying over large number of objects for IAM entities. Default: 100

Additional Configuration Parameters

Following are the additional configuration parameters that can be set in the application debug page:

assumeRoleDurationInSeconds

Default value: 3600

The duration, in seconds, of the role session. The value can range from 900 seconds (15 minutes) up to the maximum session duration setting for the role.

Set the value of the `assumeRoleDurationInSeconds` parameter as follows:

```
<entry key="assumeRoleDurationInSeconds" value="3600"/>
```

assumeRoleSessionName

Default value: SailPointUser

An identifier for the assumed role session. Use the role session name to uniquely identify a session when the same role is assumed by different principals or for different reasons. In cross-account scenarios, the role session name is visible to, and can be logged by the AWS account that owns the role.

Set the value of the `assumeRoleSessionName` parameter as follows:

```
<entry key="assumeRoleSessionName" value="SailPointUser"/>
```

Additional Configuration Parameters for Throttling Support

The following parameters are used to manage the API throttling exceptions in AWS and to overcome the overload on the AWS managed system due to large data:

maxRetries

Maximum number of retry attempts. Default: 5

baseDelay

Delay in milliseconds after which a retry attempt is performed. The baseDelay will exponentially increase after every retry attempt. Default: 500

For example, if the defaultvalue is 10 seconds, for subsequent retry attempts baseDelay will be 20 seconds, and 40 secondsand so on.

throttledBaseDelay

Delay in milliseconds after which a retry attempt is performed. This delay is applicable to throttling errors. The throttledBaseDelay will exponentially increase after every retry attempt. Default: 1000

maxBackoffTime

Maximum backoff time in milliseconds. When the sleep time increases exponentially after each retry attempt, this value would be set to the maximum limit of the sleep time. Default: 20000

> **Note**
> The additional configuration parameters for throttling support are present out of the box for new and existing application with the default values mentioned.

> **Note**
> The connector uses the AWS SDK's retry mechanism, therefore the connector will only retry the errors that the AWS SDK is retrying.

Review and Test

Perform a test to confirm the connection to SailPoint.

1. Confirm that the entries in each field are correct.

   If you note any mistakes, return to the section and make corrections.

2. Select **Test Connection** to run the connection test.

Schema Objects and Attributes

The following schema objects are supported:

- Account
- Group (Primary)
- *AWS Managed Policy
- *Customer Managed Policy
- *Inline Policy
- *Service Control Policy
- *Roles
- *Organization Unit
- *AWS Accounts

> **Note**
> Schema objects with an asterisk (*) are only functional when you purchase Cloud Access Manager (CAM) or IdentityIQ Cloud Governance.

Tags Attribute

A new Tags Attribute is available for aggregation to identify and organize AWS resources. Tags are displayed as Key/Value pairs for the aggregated entities.

This multivalued, string attribute stores the Key/Value pair as a single string in K=V,K2=V2 format with tags separated by commas. For example:

```
[Key~Value, Costcenter~Austin, Department~QA, Location~Offshore ]
```

The form-data appears as follows:

- Key: `Costcenter, Department, Location`
- Value: `Austin, QA, Offshore`

By default the tilda(~) delimiter is between the key value pair. However, the separator can be configured as `tagKeyValueSeparator`.

```
<entry key="tagKeyValueSeparator"value="="/>
```

Identity Attributes

SailPoint requires certain attributes remain in your configuration. These attributes are referred to as Identity Attributes, and they must not be updated. If you update these attributes from their default values, the connector may fail. To resolve any issues caused by changing Identity Attributes, reconfigure them to their default values. The following table lists the Identity Attributes for this connector:

Identity Attribute	Schema Object Type
ARN	Account
ARN	Group
Id	InlinePolicy
ARN	AWSManagedPolicy
ARN	CustomerManagedPolicy
ARN	Role
ARN	SCP
ARN	AWSAccount
ARN	OrganizationUnit

Account Attributes

The following table describes the supported account attributes:

Attributes	Type	Description
UserName	String	Friendly name of the user.
UserId	String	Unique ID of the user.
Path	String	Path to the user.
ARN	String	Amazon Resource Name of the user. This is an Identity Attribute which must not be changed.
CreateDate	String	Creation date of the user.
ConsoleAccess	String	Password status of the user.
Groups	Group	Groups the user is a part of.
AWSManagedPolicies	AWSManagedPolicy	AWS Managed Policies directly assigned to the

Attributes	Type	Description
		user.
CustomerManagedPolicies	CustomerManagedPolicy	Customer Managed Policies directly assigned to the user.
InlinePolicies	InlinePolicy	Inline Policies directly assigned to the user.
Access Keys	String	Access keys associated with the user.
AWS CodeCommit HTTPS Credentials	String	AWS CodeCommit HTTPS Git credentials associated with the user.
AWS CodeCommit SSH Keys	String	AWS CodeCommit SSH public keys associated with the user.
Signing Certificates	String	Signing Certificates associated with the user.
Multi-Factor Authentication Device	String	Multi-Factor Authentication device associated with the user.
PasswordLastUsed	String	Password last used date of the user.
AccessKeyLastUsed	String	Access key last used details of the user.
Tags	String	Tag list in the format TagKey~TagValue pair

Group Attributes

The following table describes the supported attributes for the group schema:

Attributes	Type	Description
Object Type: Group		
GroupName	String	Friendly name of the group.
GroupId	String	Unique ID of the group.
Path	String	Path to the group.
ARN	String	Amazon Resource Name of the group. This is an Identity Attribute which must not be changed.
Create	String	Creation date of the group.
AWSManagedPolicies	AWSManagedPolicy	AWS Managed Policies directly assigned to the group.
CustomerManagedPolicies	CustomerManagedPolicy	Customer Managed Policies directly assigned to the group.
InlinePolicies	InlinePolicy	Inline Policies directly assigned to the group.
Object Type: AWSManagedPolicy		
PolicyName	String	The friendly name of the AWS managed policy.

Attributes	Type	Description
PolicyId	String	The unique ID of the AWS managed policy.
Description	String	A friendly description of the AWS managed policy.
ARN	String	Amazon Resource Name of the AWS managed policy. This is an Identity Attribute which must not be changed.
Path	String	The path to the AWS managed policy.
CreateDate	String	The creation date of the AWS managed policy.
UpdateDate	String	The last update date of the AWS managed policy.
DefaultVersionId	String	The currently enabled version ID of the AWS managed policy.
PolicyJSON	String	The JSON document for the AWS managed policy.
Object Type: Customer Managed Policy		
PolicyName	String	The friendly name of the customer managed policy.
PolicyId	String	The unique ID of the customer managed policy.
Description	String	A friendly description of the customer managed policy.
CreateDate	String	The creation date of the customer managed policy.
UpdateDate	String	The last update date of the customer managed policy.
ARN	String	Amazon Resource Name of the customer managed policy. This is an Identity Attribute which must not be changed.
Path	String	The path to the customer managed policy.
DefaultVersionId	String	The currently enabled version ID of the customer managed policy.
PolicyJSON	String	The JSON document for the customer managed policy.
PolicyGroups	String	Groups attached to the customer managed policy.
PolicyRoles	String	Roles attached to the customer managed policy.
Tags	String	Tag list in the format TatKey~TagValue pair
Object Type: InlinePolicy		
Name	String	The friendly name of the policy.
Id	String	The unique ID of the policy. This is an Identity Attribute which must not be changed.

Attributes	Type	Description
PolicyJSON	String	The JSON document for the policy.
Object Type: Role		
RoleName	String	The friendly name of the role.
RoleId	String	The unique ID of the role.
Path	String	Path to the Role.
ARN	String	Amazon Resource Name of the role. This is an Identity Attribute which must not be changed.
Description	String	Role Description.
CreateDate	String	Creation date of the role.
AWSManagedPolicies	AWSManagedPolicy	AWS Managed Policies directly assigned to the role.
CustomerManagedPolicies	CustomerManagedPolicy	Customer Managed Policies directly assigned to the role.
InlinePolicies	InlinePolicy	Inline Policies directly assigned to the role.
TrustPolicyJSON	String	Trust Relationship Policy JSON.
MaxSessionDuration	String	Maximum CLI/API session duration.
Tags	String	Tag list in the format TagKey~TagValue pair
Object Type: SCP		
SCPName	String	The friendly name of the Service Control Policy.
SCPId	String	The unique ID of the Service Control Policy.
ARN	String	Amazon Resource Name of the Service Control Policy. This is an Identity Attribute which must not be changed.
Description	String	A friendly description of the Service Control Policy.
AWSManaged	String	A boolean value that indicates whether the Service Control Policy is an AWS managed policy.
PolicyJSON	String	The JSON document for the Service Control Policy.
Tags	String	Tag list in the format TagKey~TagValue pair
Object Type: AWSAccount		
AWSAccountName	String	The friendly name of the AWS account.
AWSAccountId	String	The unique ID of the AWS account.
ARN	String	Amazon Resource Name of the AWS account.

Attributes	Type	Description
		This is an Identity Attribute which must not be changed.
Email	String	The email address associated with the AWS account.
Status	String	The status of the AWS account in the organization.
JoinedMethod	String	The method by which the AWS account joined the organization.
JoinedTimestamp	String	The date the AWS account became a part of the organization.
OrganizationUnit	OrganizationUnit	Organization unit holding the AWS Account.
Tags	String	Tag list in the format TagKey~TagValue pair
Object Type: OrganizationUnit		
OUName	String	The friendly name of the Organization Unit.
OUId	String	The unique ID of the Organization Unit.
ARN	String	Amazon Resource Name of the Organization Unit. This is an Identity Attribute which must not be changed.
ServiceControlPolicies	SCP	Service Control Policies attached to the Organization Unit.
Parent	OrganizationUnit	Parent Organization Unit.
AWSAccounts	AWSAccount	AWS Accounts attached to the Organization Unit.
Tags	String	Tag list in the format TagKey~TagValue pair

Provisioning Policy Attributes

The following default provisioning policies are defined for Account and Account-Group.

Account

Create

Attributes that are required for creating an account.

User Name*

Enter the user name for IAM user.

AWS Account*

Enter the Account ID or ARN of the AWS Account under which the IAM user is to be created.

Password

Enter the password for IAM user that allows users to sign-in to the AWS Management Console.

Require Password Reset

Users must create a new password at next sign-in. Users automatically receive the **IAMUserChangePassword** policy to allow them to change their own password.

Programmatic Access

Create an Access Key ID and Secret Access Key for Programmatic Access.

Path

Specify the path to the IAM User.

Enable

Attributes that are required for enabling an account.

Password

Enter the password for IAM user that allows users to sign-in to the AWS Management Console.

Access Keys

Enables the recent Access Key.

AWS CodeCommit SSH Keys

Enables the recent SSH Key.

AWS CodeCommit HTTPS Credentials

Enables the recent HTTPS Credential.

Account-Group

Create

Attributes that are required for creating a group and customer managed policy.

Group Name*

Enter the group name for IAM group.

AWS Account*

Enter the Account Id or ARN of the AWS account under which the IAM group is to be created.

Path

Specify the path to the IAM group.

CustomerManagedPolicy

Policy Name*

Enter the policy name.

AWS Account*

Enter the Account Id of the AWS account under which the IAM Policy is to be created.

Policy Description

Enter the policy description.

Policy JSON*

Enter the policy document as a JSON string.

Path

Specify the path to the policy.

Update

Attributes that are required for updating group and role.

Group Name

Enter the group name for the IAM group.

Path

Specify the path to the IAM group.

ARN

ARN of the group.

Creation Date

Creation date of the group.

AWS Managed Policies

Select the AWS managed policies name to be attached.

Customer Managed Policies

Select the Customer managed policies name to be attached.

Inline Policies

Associated inline policies.

UpdateRole

Role Name

Role name for the IAM role.

Path

Path to the IAM role.

ARN

ARN of the role.

Creation Date

Creation date of the role.

MaxSessionDuration

Duration in seconds for which this role can be assumed.

Trust Policy JSON

Trust policy JSON attached to the Role.

AWS Managed Policies

Select the AWS managed policies name to be attached.

Customer Managed Policies

Select the Customer managed policies name to be attached.

Inline Policies

Associated inline policies.

Multiple Group Objects Support

The AWS source supports multiple group objects. The details of the features are:

Feature	IAM Users
Aggregate	✓
Create, Update, Enable, Disable, Delete	✓
Group Entitlements (Read, Request, Revoke)	Groups, AWSManagedPolicies, CustomerManagedPolicies, InlinePolicies **Note** InlinePolicies can be read and revoked.

Group Entitlements

SailPoint provides the ability to aggregate additional details from the managed system through Group Entitlements. These objects have a separate schema defining list of attributes. The aggregation task fetches these as additional details when aggregation is run for that Group Entitlement type.

> The following group objects are only functional when you purchase Cloud Access Manager (CAM).
>
> - AWS Managed Polices
> - Customer Managed Policies
> - Inline Policy
> - Service Control Policy
> - Roles
> - Organization Unit
> - AWS Accounts

	Aggregation	Permissions	Read Heir-archy Group
Groups	✓	NA	NA
ASWManagedPolicies	✓	✓	NA

	Aggregation	Permissions	Read Hierarchy Group
CustomerManagedPolicies	✓	✓	NA
InlinePolicies	✓	✓	NA
Roles	✓	✓	NA
SCP	✓	✓	NA
AWSAccount	✓	NA	NA
OrganizationUnit	✓	NA	✓

- NA = Not Applicable.
- The objects such as, Groups, AWSManagedPolicies, CustomerManagedPolicies, InlinePolicies, and Roles are the IAM entities.
- The objects, such as Service Control Policies (SCP), AWSAccount and OrganizationUnit (OU) are the Organization entities.
- The AWS source supports JSON Policy for Permission Policy, and Trust Policy is represented as Permissions.
- The Permission Policy for the following AWS entities are represented as Permissions:
    - AWS Managed Policies
    - Customer Managed Policies
    - Inline Policies
    - Service Control Policies
- The Trust Policy for the following AWS entity is represented as Permissions:
    - Roles

Using Multiple Group Entitlements with a Pre-existing Source

To start using Multiple Group Entitlements with your current (pre-existing) AWS source(s), perform the following steps:

1. Use createSchema API to create new group schema for your source.

Example of API body content for adding Project Roles to an existing source

```
 AWS Managed Policies:
{
"name": "AWSManagedPolicy",
"nativeObjectType": "AWSManagedPolicy",
"identityAttribute": "ARN",
"displayAttribute": "PolicyName",
"hierarchyAttribute": null,
"includePermissions": false,
"features": [
"NO_GROUP_PERMISSIONS_PROVISIONING"
],
"configuration": {},
"attributes": [
{
"name": "PolicyName",
"type": "STRING",
"schema": null,
"description": "The friendly name of the AWS managed policy ",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "PolicyId",
"type": "STRING",
"schema": null,
"description": "The unique ID of the customer managed policy",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "Description",
"type": "STRING",
"schema": null,
"description": "A friendly description of the AWS managed policy",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},

{
"name": "ARN",
"type": "STRING",
"schema": null,
"description": "Amazon Resource Name of the AWS managed policy",
```

```
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "Path",
"type": "STRING",
"schema": null,
"description": "The path to the AWS managed policy",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "CreateDate",
"type": "STRING",
"schema": null,
"description": "The creation date of the AWS managed policy",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "UpdateDate",
"type": "STRING",
"schema": null,
"description": "The last update date of the AWS managed policy",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "DefaultVersionId ",
"type": "STRING",
"schema": null,
"description": "The currently enabled version ID of the AWS managed policy ",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "PolicyJSON",
"type": "STRING",
"schema": null,
"description": "The JSON document for the AWS managed policy",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
}
]
```

```
}

#########################################################

Customer Managed Policies:

{
"name": "CustomerManagedPolicy",
"nativeObjectType": "CustomerManagedPolicy",
"identityAttribute": "ARN",
"displayAttribute": "PolicyName",
"hierarchyAttribute": null,
"includePermissions": false,
"features": [
"PROVISIONING, NO_GROUP_PERMISSIONS_PROVISIONING"
],
"configuration": {},
"attributes": [
{
"name": "PolicyName",
"type": "STRING",
"schema": null,
"description": "The friendly name of the customer managed policy ",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "PolicyId",
"type": "STRING",
"schema": null,
"description": "The unique ID of the customer managed policy",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "Description",
"type": "STRING",
"schema": null,
"description": "A friendly description of the customer managed policy",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "CreateDate",
"type": "STRING",
"schema": null,
"description": "The creation date of the customer managed policy",
```

```
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "UpdateDate",
"type": "STRING",
"schema": null,
"description": "The last update date of the customer managed policy",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "ARN",
"type": "STRING",
"schema": null,
"description": "Amazon Resource Name of the customer managed policy",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "Path",
"type": "STRING",
"schema": null,
"description": "The path to the customer managed policy",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "DefaultVersionId ",
"type": "STRING",
"schema": null,
"description": "The currently enabled version ID of the customer managed
policy",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "PolicyJSON",
"type": "STRING",
"schema": null,
"description": "The JSON document for the customer managed policy",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
```

```
{
"name": "PolicyGroups",
"type": "STRING",
"schema": null,
"description": "Groups attached to the customer managed policy",
"isMulti": true,
"isEntitlement": true,
"isGroup": false
},
{
"name": "PolicyRoles",
"type": "STRING",
"schema": null,
"description": "Roles attached to the customer managed policy",
"isMulti": true,
"isEntitlement": true,
"isGroup": false
}
]
}

#####################################################

Inline Policies:

{
"name": "InlinePolicy",
"nativeObjectType": "InlinePolicy",
"identityAttribute": "Id",
"displayAttribute": "Name",
"hierarchyAttribute": null,
"includePermissions": false,
"features": [
"NO_GROUP_PERMISSIONS_PROVISIONING"
],
"configuration": {},
"attributes": [
{
"name": "Name",
"type": "STRING",
"schema": null,
"description": "The friendly name of the policy",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "Id",
"type": "STRING",
"schema": null,
```

```
"description": "The unique ID of the policy",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "PolicyJSON",
"type": "STRING",
"schema": null,
"description": "The JSON document for the policy",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
}
]
}


##################################################


Roles:

{
"name": "Role",
"nativeObjectType": "Role",
"identityAttribute": "ARN",
"displayAttribute": "RoleName",
"hierarchyAttribute": null,
"includePermissions": false,
"features": [
"PROVISIONING, NO_GROUP_PERMISSIONS_PROVISIONING"
],
"configuration": {},
"attributes": [
{
"name": "RoleName",
"type": "STRING",
"schema": null,
"description": "The friendly name of the role",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "RoleId",
"type": "STRING",
"schema": null,
```

```
"description": "The unique ID of the role",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "Path",
"type": "STRING",
"schema": null,
"description": "Path to the Role",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "ARN",
"type": "STRING",
"schema": null,
"description": "Amazon Resource Name of the role",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "Description",
"type": "STRING",
"schema": null,
"description": "Role Description",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "CreateDate",
"type": "STRING",
"schema": null,
"description": "Creation date of the role",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "AWSManagedPolicies",
"type": "STRING",
"schema": null,
"description": "AWS Managed Policies directly assigned to the role",
"isMulti": true,
"isEntitlement": true,
"isGroup": false
},
```

```
{
"name": "CustomerManagedPolicies",
"type": "STRING",
"schema": null,
"description": "Customer Managed Policies directly assigned to the role",
"isMulti": true,
"isEntitlement": true,
"isGroup": false
},
{
"name": "InlinePolicies",
"type": "STRING",
"schema": null,
"description": "Inline Policies directly assigned to the role",
"isMulti": true,
"isEntitlement": true,
"isGroup": false
},
{
"name": "TrustPolicyJSON",
"type": "STRING",
"schema": null,
"description": "Trust Relationship Policy JSON",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "MaxSessionDuration",
"type": "STRING",
"schema": null,
"description": "Maximum CLI/API session duration",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
}
]
}



##################################################
Service Control Policies:

{
"name": "SCP",
"nativeObjectType": "SCP",
"identityAttribute": "ARN",
"displayAttribute": "RoleName",
"hierarchyAttribute": null,
```

```
"includePermissions": false,
"features": [
"NO_GROUP_PERMISSIONS_PROVISIONING"
],
"configuration": {},
"attributes": [
{
"name": "SCPName",
"type": "STRING",
"schema": null,
"description": "The friendly name of the Service Control Policy",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "SCPId",
"type": "STRING",
"schema": null,
"description": "The unique ID of the Service Control Policy",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "ARN",
"type": "STRING",
"schema": null,
"description": "A friendly description of the Service Control Policy",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "Description",
"type": "STRING",
"schema": null,
"description": "A friendly description of the Service Control Policy",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "AWSManaged",
"type": "STRING",
"schema": null,
"description": "A boolean value that indicates whether the Service Control
Policy is an AWS managed policy",
"isMulti": false,
"isEntitlement": false,
```

```
"isGroup": false
},
{
"name": "PolicyJSON",
"type": "STRING",
"schema": null,
"description": "The JSON document for the Service Control Policy",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
}
]
}


              ######################################################
              AWS Accounts:
{
"name": "AWSAccount",
"nativeObjectType": "AWSAccount",
"identityAttribute": "ARN",
"displayAttribute": "AWSAccountName",
"hierarchyAttribute": null,
"includePermissions": false,
"features": [],
"configuration": {},
"attributes": [
{
"name": "AWSAccountName",
"type": "STRING",
"schema": null,
"description": "The friendly name of the AWS account.",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "AWSAccountId",
"type": "STRING",
"schema": null,
"description": "The unique ID of the AWS account.",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "ARN",
"type": "STRING",
"schema": null,
"description": "Amazon Resource Name of the AWS account.",
```

```
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "Email",
"type": "STRING",
"schema": null,
"description": "The email address associated with the AWS account.",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "Status",
"type": "STRING",
"schema": null,
"description": "The status of the AWS account in the organization.",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "CreateDate",
"type": "STRING",
"schema": null,
"description": "Creation date of the role",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "JoinedMethod",
"type": "STRING",
"schema": null,
"description": "The method by which the AWS account joined the organization.",
"isMulti": true,
"isEntitlement": true,
"isGroup": false
},
{
"name": "CustomerManagedPolicies",
"type": "STRING",
"schema": null,
"description": "Customer Managed Policies directly assigned to the role",
"isMulti": true,
"isEntitlement": true,
"isGroup": false
},
{
```

```
"name": "JoinedTimestamp",
"type": "STRING",
"schema": null,
"description": "The date the AWS account became a part of the organization.",
"isMulti": true,
"isEntitlement": true,
"isGroup": false
},
{
"name": "OrganizationUnit",
"type": "STRING",
"schema": null,
"description": "Organization unit holding the AWS Account",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
}
]
}

              ######################################################
               Organization Units:
{
"name": "OrganizationUnit",
"nativeObjectType": "OrganizationUnit",
"identityAttribute": "ARN",
"displayAttribute": "OUName",
"hierarchyAttribute": Parent,
"includePermissions": false,
"features": [],
"configuration": {},
"attributes": [
{
"name": "OUName",
"type": "STRING",
"schema": null,
"description": "The friendly name of the Organization Unit",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "OUId",
"type": "STRING",
"schema": null,
"description": "The unique ID of the Organization Unit",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
```

```
{
"name": "ARN",
"type": "STRING",
"schema": null,
"description": "Amazon Resource Name of the Organization Unit",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "ServiceControlPolicies",
"type": "STRING",
"schema": null,
"description": "Service Control Policies attached to the Organization Unit",
"isMulti": true,
"isEntitlement": true,
"isGroup": false
},
{
"name": "Parent",
"type": "STRING",
"schema": null,
"description": "Parent Organization Unit",
"isMulti": false,
"isEntitlement": false,
"isGroup": false
},
{
"name": "AWSAccounts",
"type": "STRING",
"schema": null,
"description": "AWS Accounts attached to the Organization Unit",
"isMulti": true,
"isEntitlement": true,
"isGroup": false
}
]
}
```

2. Update Account Schema with the following steps:

    a. Get account schema using getSchema API.

    b. Copy the schema to a file.

    c. Search for the attribute corresponding to the group schema that was added in step 1.

    d. Add/update the below two properties for the attribute found in the previous step (2.c.)

"isGroup": true,

"schema": {"type": "CONNECTOR_SCHEMA","id": "<Schema_ID_From_Step1>","name": "<Name_Of_Schema_Created_In_Step1>"}

e. Add the schema (modified in step 2.d.) to the payload to update the account schema in the source. Use replaceSchema API for this task.

> **Note**
> Remove `NO_PERMISSIONS_PROVISIONING` from the feature string so there won't be any work item while removing associated AWS Managed Policies, Customer Managed Policies, and Inline Policies from the user through certification.

Operation Specific Service IAM User permissions

This section lists the operation specific administrator permissions required for the following:

- IAM APIs
- Organization APIs

Identity and Access Management APIs

The following tables list the SailPoint operations along with the corresponding IAM API (Actions) used:

Operation	IAM API (Action)
Test Connection	GetUser
Account Update	CreateAccessKey
Reset Password	UpdateLoginProfile CreateLoginProfile
Group Create	CreateGroup
Group Update	UpdateGroup AttachGroupPolicy DetachGroupPolicy
Create Customer Managed Policy	CreatePolicy

Account Aggregation

Operation	IAM API (Action)
Summary/Attributes (UserName, UserId, Path, ARN, CreateDate, PasswordLastUsed)	ListUsers GetLoginProfile
ConsoleAccess	ListGroupsForUser
Groups	ListUserPolicies
AWSManagedPolicies and CustomerManagedPolicies	ListAttachedUserPolicies
InlinePolicies	ListAccessKeys
Access Keys	ListServiceSpecificCredentials
AWS CodeCommit HTTPS Credentials	ListSSHPublicKeys

Operation	IAM API (Action)
AWS CodeCommit SSH Keys Signing Certificates Multi-Factor Authentication (MFA) Device AccessKeyLastUsed	ListSigningCertificates ListMFADevices GetAccessKeyLastUsed

Account-Group Aggregation (Group)

Operation	IAM API (Action)
Summary/Attributes (GroupName, GroupId, Path, ARN, CreateDate) AWSManagedPolicies and CustomerManagedPolicies InlinePolicies	ListGroups ListAttachedGroupPolicies ListGroupPolicies

Account-Group Aggregation (AWSManagedPolicy and CustomerManagedPolicy)

Operation	IAM API (Action)
Summary/Attributes (PolicyName, PolicyId, ARN, Path, CreateDate, UpdateDate, DefaultVersionId) Description PolicyJSON (*Only for CustomerManagedPolicy*) PolicyGroups, PolicyRoles	ListPolicies GetPolicy GetPolicyVersion (*Only for CustomerManagedPolicy*) ListEntitiesForPolicy

Account-Group Aggregation (Role)

Operation	IAM API (Action)
Summary/Attributes (RoleName, RoleId, Path, ARN, Description, CreateDate, TrustPolicyJSON, MaxSessionDuration) AWSManagedPolicies and CustomerManagedPolicies InlinePolicies	ListRoles ListAttachedRolePolicies ListRolePolicies

Account-Group Aggregation (InlinePolicy)

Operation	IAM API (Action)
Id	No API is called for this attribute, it is formatted as: **ARN of the entity:InlinePolicy:InlinePolicyName**
Name	ListUserPolicies, ListGroupPolicies, ListRolePolicies
PolicyJSON	GetUserPolicies, GetGroupPolicies, GetRolePolicies

Account Refresh

Operation	IAM API (Action)
Summary/Attributes (UserName, UserId, Path, ARN, CreateDate)	GetUser
Groups	ListGroupsForUser
Access Keys	ListAccessKeys
Signing Certificates	ListSigningCertificates
Password	GetLoginProfile
MFA Device	ListMFADevices
AWS CodeCommit HTTPS Credentials and AWS CodeCommit SSH Keys: ListServiceSpecificCredentials	ListServiceSpecificCredentials

Refresh Operations

Operation	IAM API (Action)
Refresh Group	GetGroup
Refresh Role	GetRole
Refresh AWSManagedPolicy and CustomerManagedPolicy	GetPolicy
Refresh Inline Policy associated with User	GetUserPolicies
Refresh Inline Policy associated with Group	GetGroupPolicies
Refresh Inline Policy associated with Role	GetRolePolicies

Group Delete

Operation	IAM API (Action)
Read Accounts in the Group	DeleteGroup
Remove Accounts from the Group	GetGroup
Read Group Policies	RemoveUserFromGroup
Remove Group Policies	ListGroupPolicies
	DeleteGroupPolicy

Account Enable

Operation	IAM API (Action)
Set Password	UpdateLoginProfile
Activate Access Keys (Last created one)	UpdateAccessKey
Activate AWS CodeCommit HTTPS Credentials (Last created one)	UpdateServiceSpecificCredential
Activate AWS CodeCommit SSH Keys (Last created one)	UpdateSSHPublicKey

Account Delete

Operation	IAM API (Action)
Read Groups	ListGroupsForUser
Remove Groups	RemoveUserFromGroup
Read AWSManagedPolicy and CustomerManagedPolicy	ListAttachedUserPolicies
Remove AWSManagedPolicy and CustomerManagedPolicy	DetachUserPolicy
Read InlinePolicy	ListUserPolicies
Read Security Credentials	DeleteUserPolicy
• Access Keys	ListAccessKeys
• Signing Certificates	ListSigningCertificates
• Password	GetLoginProfile
• MFA Device	ListMFADevices
	ListServiceSpecificCredentials
	ListSSHPublicKeys

Operation	IAM API (Action)
• AWS CodeCommit HTTPS Credentials • AWS CodeCommit SSH Keys Remove Security Credentials • Access Keys • Signing Certificates • Password • MFA Device • AWS CodeCommit HTTPS Credentials • AWS CodeCommit SSH Keys	DeleteAccessKey DeleteSigningCertificate DeleteLoginProfile DeactivateMFADevice DeleteServiceSpecificCredential DeleteSSHPublicKey

Account Disable

Operation	IAM API (Action)
Delete Password Deactivate Access Keys (All) Deactivate AWS CodeCommit HTTPS Credentials (All) Deactivate AWS CodeCommit SSH Keys (All)	DeleteLoginProfile UpdateAccessKey UpdateServiceSpecificCredential UpdateSSHPublicKey

Request Entitlement (Group and Managed Policies for User)

Operation	IAM API (Action)
Add group to user Add AWSManagedPolicy and CustomerManagedPolicy to user	AddUserToGroup AttachUserPolicy

Remove Entitlement (Group, Managed Policies, Inline Policies from User)

Operation	IAM API (Action)
Remove group from user Remove AWSManagedPolicy and CustomerManagedPolicy from user Remove Inline Policy from user	RemoveUserFromGroup DetachUserPolicy DeleteUserPolicy

Remove Inline Policy

Operation	IAM API (Action)
Read from User Delete from User Read from Group Delete from Group Read Role Delete from Role	GetUserPolicies DeleteUserPolicy GetGroupPolicies DeleteGroupPolicy GetRolePolicies DeleteRolePolicy

Update Role

Operation	IAM API (Action)
Attach AWSManagedPolicy and CustomerManagedPolicy Remove AWSManagedPolicy and CustomerManagedPolicy	AttachRolePolicy DetachRolePolicy

Organization APIs

The following tables list the Operations along with the corresponding IAM APIs used for managing organizational entities:

Operations	Organizations API (Actions)
Test Connections	Role (Master Account): **organizations:ListAccounts**

Account-Group Aggregation (OrganizationUnit)

Operations	Organizations API (Actions)
Summary/Attributes (OUName, OUId, ARN, Parent) ServiceControlPolicies AWSAccounts	ListRoots, ListOrganizationalUnitsForParent ListPoliciesForTarget ListAccountsForParent

Account-Group Aggregation (SCP)

Operations	Organizations API (Actions)
• Summary/Attributes (SCPName, SCPId, ARN, Description, AWSManaged) • PolicyJSON	• ListPolicies • DescribePolicy

Account-Group Aggregation (AWSAccount)

Operations	Organizations API (Actions)
Summary/Attributes (AWSAccountName, AWSAccountId, ARN, EmailId, Status, JoinedType, JoinedTimestamp) OrganizationUnit	ListAccounts ListRoots, ListParents, DescribeOrganizationalUnit

Get Operations

Operations	Organizations API (Actions)
SCP	DescribePolicy
AWS Accounts	DescribeAccount, ListRoots, ListParents, DescribeOrganizationalUnit
Organizational Unit	DescribeOrganizationalUnit, ListRoots, ListParents, ListPoliciesForTarget, ListAccountsForParent

(*Optional*) Upgrade Consideration

When upgrading IdentityIQ:

- Enter the AWS Region on the configuration parameters page to save your application.
- To view the list of groups and roles that the customer managed policy is attached to, add the **PolicyGroups** and **PolicyRoles** attributes in schema of object type Customer Managed Policy.
- To get the information of the password last used date and access key last used details of IAM User, add the **PasswordLastUsed** and **AccessKeyLastUsed** attributes in the account schema.
- To get the tag information for IAM User, Role, Customer Managed Policy, Service Control Policy, Organization Unit and AWS Account, add the Tags attribute in the account schema as well as in the respective objects schemas with types: string and multivalued.

For more information on PasswordLastUsed and AccessKeyLastUsed attributes, see Account Schema.

Troubleshooting

If you encounter any of the following issues or errors, SailPoint recommends that you follow the guidance provided below to resolve the error before contacting SailPoint Support.

Test Connection Errors

Error

```
[ InvalidConfigurationException ] [ Possible suggestions ] Ensure that the AWS
Management Account ID is correctly configured. [ Error details ] Management Account ID
must be configured when "Manage All Accounts" is checked or AWS organization entities
needs to be managed.
```

For IAMRole Authentication, if **Manage All Accounts** is selected or if the included AWS Account IDs are mentioned, then the test connection fails.

Resolution: Provide 'Management Account ID' on the configuration settings page (Management Account ID is root AWS Account ID).

Error

```
[ InsufficientPermissionException ] [ Possible suggestions ] Service account must be
present in management account with the required permissions. [ Error details ] Test
Connection Failed: You don't have permissions to access this resource. (Service:
AWSOrganizations; Status Code: 400; Error Code: AccessDeniedException; Request ID:
<actual alpha-numerical request ID>)
```

If **Manage All Accounts** is selected, and the service account is in any of the member AWS accounts, test connection fails.

Resolution: Ensure the service account is created in the management AWS account with required permissions.

Error

```
[InsufficientPermissionException] [Possible suggestions] Service account must be present
in the management account with the required permissions. [Error details] Test Connection
Failed: You don't have permissions to access this resource. (Service: AWSOrganizations;
Status Code: 400; Error Code: AccessDeniedException; Request ID: 65fc15e5-7e90-11e8-
9d6a-6fc388fd2d28)
```

If Service user is in Member AWS account, Test Connection fails.

Resolution: Ensure that the service user is created in the management account with required permission to manage organization entities.
If you do not want to manage the Organization entities, remove them from schema.

Error

```
When configuring a new Amazon Web Services source, the Test Connection fails with the
following error message:
sailpoint.connector.ConnectionFailedException: [ ConnectionFailedException ] [ Error
details ] Your account is not a member of an organization. (Service: AWSOrganizations;
Status Code: 400; Error Code: AWSOrganizationsNotInUseException; Request ID: c8d77e54-
ec98-11e8-b722-bb0efb7fc919)
```

If Service user is in Member AWS account, Test Connection fails.

Resolution: Ensure that the AWS Account is a member of the AWS Organization which must be managed.

Error

For the upgraded sources, if multiple group objects are configured, work item(s) got created while revoking associated AWS Managed Policies, Customer Managed Policies, and Inline Policies from the user through **certification**.

Resolution: Remove `NO_PERMISSIONS_PROVISIONING` from the feature string in Source XML.

Role Not Created Error

Error

```
[ InvalidConfigurationException ] [ Possible suggestions ] Ensure that the required role
is created in the specified AWS accounts and the user has required permissions. [ Error
details ] Test connection failed for accounts [list of AWS account IDs] Failure
Reason=Access denied (Service: AWSSecurityTokenService)
```

If **Manage All Accounts** is selected, and the provided role is not present in any of the AWS accounts, then the test connection fails .

```
Exception during aggregation. Reason:openconnector.InvalidRequestException: Aggregation
is failed for following AWS Account Ids: [comma separated list of accounts]
```

Aggregation fails

Resolution: Ensure the role is created in all the AWS Accounts with the same name and having sufficient permissions.

AWS Account IDs Error

```
[ InvalidConfigurationException ] [ Error details ] "Include AWS Account IDs" must be
empty if "Manage All Accounts" is checked.
```

Include AWS Account IDs is populated and the **Manage All Accounts** checkbox is selected.

Resolution: The **Include AWS Account IDs** list must be empty if the **Manage All Accounts** checkbox is selected .

Exclude AWS Accounts Error

```
[ InvalidConfigurationException ] [ Error details ] Either "Manage All Accounts" is checked (with or without "Exclude AWS Account IDs") or "Include AWS Account IDs" must be populated.
```

Exclude AWS Account Ids is populated and **Manage All Accounts** checkbox is not selected.

Resolution: Select the **Manage All Accounts** checkbox if **Exclude AWS Account IDs** is populated.

Manage All Accounts Error

```
[ InvalidConfigurationException ] [ Error details ] Either "Manage All Accounts" is checked (with or without "Exclude AWS Account IDs") or "Include AWS Account IDs" must be populated.
```

Manage All Accounts is not selected and both the **Include AWS Account IDs** list and **Exclude AWS Account IDs** list are empty.

Resolution: Either **Manage All Accounts** (with or without **Exclude AWS Account IDs**) or **Include AWS Account IDs** must be populated.

Create Account or Add Entitlement Error

```
sailpoint.connector.ConnectorException: Invalid provisioning request. Attribute AWS Account does not match the entitlement requested : arn:aws:iam::<AWS Account ID>:group/<IAM Group Name>
```

If the IAM groups present in access profile do not belong to the AWS Account in which the IAM User needs to be created, then create account or add entitlement fails.

Resolution: Ensure that the access profile contains the IAM Groups as entitlements of the same AWS Account in which the IAM User needs to be created.

Aggregation time-out error

```
java.lang.RuntimeException: java.lang.InterruptedException: Timeout waiting for response
```

```
Exception during aggregation. Reason: java.lang.RuntimeException: An error occurred while aggregating Application <Source Name> [source-<Source ID>]
```

You may see this error while performing account aggregation or entitlement aggregation.

Resolution: Set the `aggregateTimeout` attribute using IdentityNow REST API. Enter the time-out value in milliseconds.

Create Account Error

```
sailpoint.connector.ConnectorException: Access denied (Service: AWSSecurityTokenService;
Status Code: 403; Error Code: AccessDenied; Request ID: f56b8ec5-1e7e-11e9-bab1-
d124100fa000)
```

Error while creating an account.

Resolution:Ensure that the Account ID or ARN of the AWS Account is correctly mentioned in the Account ID of the account attribute. For example:

```
arn:aws:organizations::441113549707:account/o-lqs5akk5dy/170915734915
```

Identity Attribute Error

```
sailpoint.connector.ConnectorException: Un-supported identity attribute for account
```

Resolution:The Account ID must be mapped with the ARN in the attribute schema.

Throttling Error

Aggregation fails with the following error:

```
openconnector.ConnectionFailedException: [ ConnectionFailedException ] [ Error details ]
Rate exceeded (Service: AmazonIdentityManagement; Status Code: 400; Error Code:
Throttling; Request ID: <id>)]
```

Resolution: Configure the throttling and set a higher value as per the requirement and allowed API limit.

Aggregation Error

```
Exception during aggregation of Object Type InlinePolicy on Application AWSDemo1
[source]. Reason: java.lang.RuntimeException: An error occurred while aggregating
Application 'ApplicationName' [source]
```

While performing Entitlement Aggregation when multiple group objects are supported.

Resolution: Set the `aggregate_timeout` attribute with a value in milliseconds (300, 1000) using IdentityNow REST API.

```
POST <url>/cc/api/source/update/<sourceID>
```

`<url>` : The URL for the customer's IdentityNow instance

`<sourceID`: The Source ID (number) obtained through the UI

In the body of the POST, use form-data as follows:

Key: `connector_aggregateTimeout`

Value: Enter the time-out value in milliseconds (300, 1000)

Confirmation: Search for the "aggregateTimeout" attribute using the endpoint

Miscellaneous Errors

Error

Tags are not aggregated for Role after upgrade.

Resolution: Ensure `updateRole` provisioning policy is configured for the application with the Tags attribute ReadOnly='True'

```
<Field displayName="con_prov_policy_AWS_Role_Tags" helpKey="help_con_form_AWS_Role_
Tags" name="Tags" reviewRequired="true" type="string">
        <Attributes>
            <Map>
              <entry key="readOnly" value="true"/>
            </Map>
          </Attributes>
        </Field>
```

When creating an inline policy for user an error displays

The user inline policy creation does not support large JSON formats. The system displays the following error:

```
An unexpected error occurred: org.hibernate.exception.DataException: could not execute
statement
```

Resolution: Complete one of the following:

- Split the policy JSON into smaller chunks of content.
- Create a group inline policy and attach that group to the user.