Configure Users and Groups

Contents:

The Trifacta® platform can be configured to support the use of groups for users.

> **NOTE:** This feature is in Beta release.

Limitations:

- Group definitions must be pulled in from LDAP through a supported SSO integration.
  - You cannot create and manage groups from within the product.
  - You cannot import groups from other identity providers.
- Supported SSO integrations:
  - *Configure SSO for AD-LDAP* - platform native method
- Untested SSO integrations:
  - *Configure SSO for AD-LDAP* - reverse proxy method
  - *Configure SSO for SAML*
  - *Configure SSO for Azure AD*
- In this release, groups apply only to the sharing of connections and flows.

Enable

Enable and configure SSO

You must enable and configure one of the supported SSO integration methods.

Configure platform

Please review and set the following platform settings.

Steps:

1. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.
2. Locate the following settings and apply values as needed:

Setting	Description
"feature.groups.enabled"	Set this value to `true` to enable use of LDAP groups in the platform.

"feature.groups.mapping.groupName"	Set this value to the LDAP search result parameter containing the value to be used as the name of a group in the Trifacta application. This value must have unique values, since groups in the Trifacta platform must have unique names. **Tip:** `"cn"` is a good choice.
"feature.groups.ldapServers"	(optional) An array of parameters, listing LDAP servers to use for synching of groups. If this parameter is not specified, then the LDAP server specified in the parameter `webapp.ldap.server` is used for synching.

"feature.groups.defaultGroupFilters"	(optional) You must provide at least one search filter string to use to query the LDAP servers for groups. The following example searches for all groups named `foo` and `bar`. In the UI:
	
	`(ou=foo),(ou=bar)`
	
	If editing this parameter through
	
	`trifacta-conf.json`
	
	, this value must be stored as an array with appropriate syntax:
	
	`["(ou=foo)","(ou=bar)"]`
	
	Notes:
	
	A search filter doesn't need to use the `ou` parameter. Any valid LDAP search filter can be used.
	
	Each search filter must include parentheses at the beginning and the end.
	
	Each filter string is expected to return a single item. If the search results include multiple items, only the first item is used.
	
	If this value is empty, no groups are synched.

3. Save your changes and restart the platform.

Create users

All users must be created in the Trifacta platform.

NOTE: The email address for the user in the Trifacta platform must match the LDAP email attribute.

- See *Manage Users*.
- For more information on creating users via API, see *API People Create v4*.

Synching:

After the platform users and groups have been synched with the LDAP identity provider:

- Any objects shared to a group are shared to individual users of the group as collaborators.
- If an LDAP user has no corresponding Trifacta platform user at the time of synching, the platform user is automatically added to the group and inherits the group's permissions when the account is created.

NOTE: If a Trifacta platform user is removed from an LDAP group, the user remains a member of the platform group until groups are synched again. When groups are synched, the user is removed from the group and loses access to any objects shared with the group.

Sync Users and Groups via API

You can use the following endpoint to sync the platform with the configured LDAP servers for their groups.

> **NOTE:** This endpoint must be triggered using an admin account.

Endpoint	`http://www.example.com:3005/v4/groups/syncGroups`
Authentication	Required
Method	POST
Request Body	Empty.
Response Status Code	200 - OK
Response Body	The response body contains the list of groups that have been added or removed based on the synching: `{ "data": [{ "ldap": "LDAP://www.ldap.example.com", "updatedGroups": [{ "id": 55, "members": [{ "id": 94, "email": "guest1@example.com", "name": "Guest Number One" }, { "id": 95, "email": "guest2@example.com", "name": "guest2 NumberTwo" }], "name": "deptGRP1" }], "deletedGroups": [{ "id": 54, "name": "deptGRP2" }] }] }`

cURL example:

- The backslash \ character indicates that the line continues on the following line.
- The following example references the use of an API token generated for the admin user. For more information, see *Manage API Access Tokens*.

```
curl -X POST \
  http://www.example.com:3005/v4/groups/syncGroups \
  -H 'authorization: Basic <auth_token>' \
  -H 'cache-control: no-cache'
```

Testing - Share Flows and Connections

- **Flows:** You can share a flow to an imported group like you share with individual users. For more information, see *Share Flow Dialog*.
- **Connections:** You can share your connection to an imported group. For more information, see *Share Connection Window*.