Enabling Identity Federation in Keycloak

Identity federation allows you to delegate authentication to external identity providers (IdPs) like Google, GitHub, Facebook, or enterprise systems such as LDAP and Active Directory. This guide explains how to integrate identity providers using the Keycloak Admin Console, REST API, and Docker CLI (kcadm.sh). It includes configuration examples, permission requirements, best practices, and common issues.

Adding Identity Providers via Keycloak Admin Console

This method supports most popular providers like Google, GitHub, Facebook, and SAML/LDAP.

Access the Admin Console

Log in to your Keycloak Admin Console:

```
http://<your-keycloak-domain>/admin/
```

Select the realm where you want to add the identity provider.

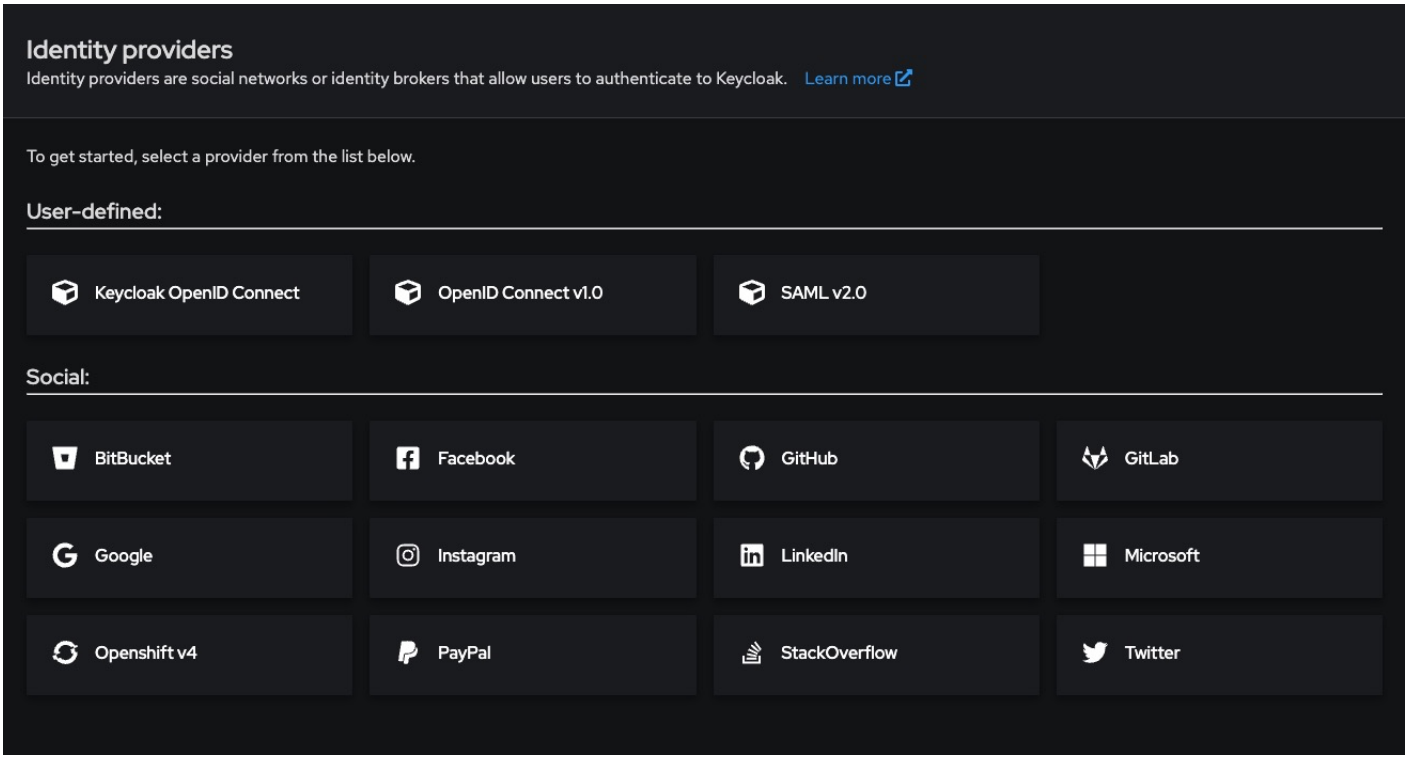
Add an Identity Provider (OIDC-based)

1. Go to **Identity Providers > Add Provider**
2. Choose an option like Google, GitHub, or OpenID Connect v1.0
3. Fill in the following:

    - **Alias**: A unique name like google or github
    - **Client ID**: From the external IdP
    - **Client Secret**: From the external IdP
    - **Authorization URL**, **Token URL**, **User Info URL**: Auto-filled for well-known providers

4. Set **Sync Mode** (e.g., IMPORT, FORCE, or LEGACY)

5. Enable **Store Tokens** if you want offline access
6. Click **Save**



Test the Identity Provider

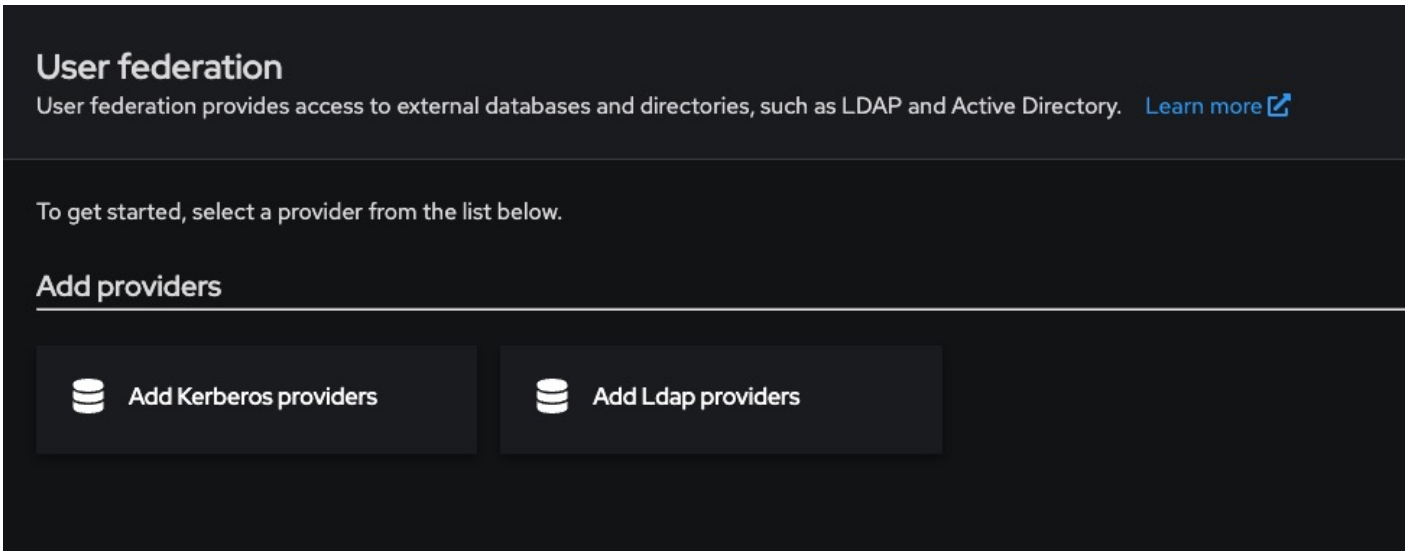
1. Go to the realm login page
2. You'll now see a **"Login with Google"** or equivalent option

Adding LDAP or Active Directory

1. Go to **User Federation > Add Provider → LDAP**
2. Fill in connection details:

Field	Example
Connection URL	ldap://ldap.mycompany.com
Users DN	ou=users,dc=mycompany,dc=com
Bind DN	cn=admin,dc=mycompany,dc=com
Bind Credential	Your LDAP password
Vendor	Active Directory, Other, etc.

2. Choose **Edit Mode**: READ_ONLY, WRITABLE, or UNSYNCED
3. Enable **Periodic Sync** if needed
4. Save and test the connection



Adding Identity Providers via REST API

Get Access Token

```
curl -X POST "https://<keycloak-domain>/realms/master/protocol/openid-connect/token" \
  -H "Content-Type: application/x-www-form-urlencoded" \
  -d "username=admin" \
  -d "password=admin-password" \
  -d "grant_type=password" \
  -d "client_id=admin-cli"
```

Save the access_token.

Add OIDC Identity Provider

```
curl -X POST "https://<keycloak-domain>/admin/realms/<realm>/identity-provider/instances" \
  -H "Authorization: Bearer <access_token>" \
  -H "Content-Type: application/json" \
```

```
  -d '{
    "alias": "google",
    "providerId": "google",
    "enabled": true,
    "trustEmail": true,
    "storeToken": false,
    "addReadTokenRoleOnCreate": false,
    "firstBrokerLoginFlowAlias": "first broker login",
    "config": {
      "clientId": "GOOGLE_CLIENT_ID",
      "clientSecret": "GOOGLE_CLIENT_SECRET"
    }
  }'
```

Adding Identity Providers via Docker CLI

Access the Container

```
docker exec -it keycloak bash
```

Add Provider

```
/opt/keycloak/bin/kcadm.sh config credentials \
  --server http://localhost:8080 \
  --realm master --user admin --password admin

/opt/keycloak/bin/kcadm.sh create identity-provider/instances -r <realm> \
  -s alias=github -s providerId=github \
  -s enabled=true \
  -s config.clientId=GITHUB_CLIENT_ID \
  -s config.clientSecret=GITHUB_CLIENT_SECRET
```

Required Permissions for Identity Federation

- Requires manage-identity-providers or admin role in the target realm
- REST tokens must come from a user with these privileges

To assign via Admin Console:

```
Users > [admin-user] > Role Mappings > Realm Roles > Add 'manage-identity-providers'
```

Best Practices for Identity Federation

- **Use Standard Broker Flows:** Leverage **First Broker Login** flow to prompt for email verification or account linking.
- **Map External Claims to Roles:** Use **Identity Provider Mappers** to assign roles or sync attributes (like email, groups, org) automatically.
- **Avoid Using Public Client IDs in Backend:** Always use confidential clients when configuring from the backend or REST API.
- **Enable Logging During Setup:** Use Keycloak's **Events > Settings** to track login attempts and errors for debugging.
- **Test With Separate Test Realm First:** Validate your configuration in a dev/test realm before enabling in production.

Common Issues and Troubleshooting

Issue	Possible Cause	Solution
Login button not showing on login page	Provider not enabled	Ensure enabled=true and alias is correct
Invalid client_id error	Client ID mismatch	Verify credentials from the IdP provider dashboard

Issue	Possible Cause	Solution
User not found after login	No email or username claim returned	Check mappers and ensure email or preferred_username is mapped
LDAP users not visible in UI	Wrong base DN or invalid bind credentials	Test connection under **User Federation** settings
403 Forbidden on REST call	Missing role or token scope	Ensure token has manage-identity-providers

---

Revision #1
Created 2025-06-17 12:25:59 UTC
Updated 2025-06-17 13:54:53 UTC