Non-Employee Identity Suite - Integration Guide

Version Number 4.6.0-ni201.1

Contents

Copyright

Chapter 1

Overview

This integration guide provides detailed information about integrating an existing identity management system or Identity Provider (IdP), with Target System via SCIM 2.0 APIs.

This integration guide covers -

- Authentication via OAuth 2.0 - how to hook IAM application up to OAuth 2.0 server for authentication.
- Integration via SCIM 2.0 - how to hook IAM application up to SCIM 2.0 server.

Chapter 2

Authentication via OAuth 2.0

Topics:

Overview - Authentication Service

OAuth 2.0 + OpenID Connect is the strategy for providing AuthN/AuthZ as a service component that web clients and services can use to authenticate and authorize access to APIs and resources. The Authentication service exposes this capability.

It includes the ability to call out to the LDAP gateway for authentication as well as establishment of the claims associated with the user (identification of whether the user is a sysadmin, resource manager, group manager, primary contract, or normal user).

The Account Management service (SCIM 2.0) APIs uses the Authentication service (OAuth 2.0 protocol) for authorization.

Authentication service - Endpoints

- The base URL for Authentication service is

  https://server.example:51000/
- To get all endpoint URLs like authorization endpoint, token endpoint, introspection endpoint and the other necessary configuration data of Authentication service like issuer, supported response types etc, please use below URL.

  https://server.example:51000/.well-known/openid-configuration

Authentication Service

Grant Types

The OpenID Connect and OAuth 2.0 specifications define grant types (often also called flows - or protocol flows). Grant types specify how a client can interact with the token service.

The Authentication Service for NEIS supports the following OAuth2 Grant Types OOTB:

- Resource Owner Password - used by NEIS for inter-server communication
- Hybrid (OpenID Connect) - used by NEIS for the user to login to the portal. This uses a response type of `code id_token` to add an additional identity token to the response. This token is signed and protected against substitution.
- Client Credentials - used by external applications for server-to-server communication. Tokens are always requested on behalf of a client, no interactive user is present. In this scenario, you send a token request to the token endpoint using the client credentials grant type. The client typically has to authenticate with the token endpoint using its client ID and secret.
- Authorization Code - used for browser-based authentication. This is the most common type of client scenario: web applications, SPAs or native/mobile apps with interactive users.
- Implicit - used for JS-based authentication (when a Client Secret cannot be protected)

Use existing registered client

1. Navigate to the `<NEIS folder>\authentication\` folder and open `appsettings.json` file.
2. Locate the RegisteredClients attribute and obtain client information from Clients attribute values.
3. To use client credentials flow, use `serviceclient` client.
   - The allowed scope for this client is `idportalscope`.
4. To use authorization code flow, use `webclient` client.
   - The allowed scope for this client are `offline_access`, `idportalscope` and `openid`.
   - Modify the `RedirectUris` with appropriate value.

5. To use hybrid flow, use `idportal` client.
   - The allowed scope for this client are `offline_access`, `idportalscope`, `openid` and `profile`.
   - By default the `RequireConsent` is `false`.
   - The `RedirectUris` and `PostLogoutRedirectUris` are already set to default values.
6. To use resource owner password flow, use `coreapi` client.
   - The allowed scope for this client are `offline_access`, `idportalscope`.
7. To use implicit flow, use `jsclient` client.
   - The allowed scope for this client are `offline_access`, `idportalscope` and `openid`.
   - Modify the `RedirectUris` with appropriate value.
   - By default the `AllowAccessTokensViaBrowser` is `true`.

Note:

- The default secret for all the clients is `idportalpass`. The secrets must be a encrypted value. To encrypt the secrets, please follow *Encryption Utility* section in `<NEIS installation folder>\docs \NEIS_user_guide.pdf`.
- By default, an Access Token for all clients is valid for 3600 seconds (60 minutes). We recommend to set the validity period of the access token based on your security requirements.
- You can also register a new client instead of using existing client for a particular flow. To register a new client, copy the specific section of a grant type / flow from the existing Clients array and add that entry at the end to Clients array and set it's properties.

Chapter 3

Integration via SCIM 2.0

Topics:

Overview - Integration via SCIM 2.0

This integration provides information about integrating an existing identity management system or Identity Provider (IdP), with Target System via SCIM 2.0 API.

Below NEIS services are required for this integration.

- Authentication service (Auth Server)
- Configuration service (Config Server)
- Account management service (SCIM 2.0 Server)

Note:

- Please refer to `<NEIS installation folder>\docs\NEIS_user_guide.pdf` for installation, configurations of NEIS services.
- The Account Management service has two jobs (End Date Scheduler and Purge Job) that runs daily. These are not part of SCIM 2.0 and needs to be disabled. To disable these jobs -

  1. Navigate to the `<NEIS folder>\acctmgmt\` folder and open `appsettings.json` file.
  2. **End Date Scheduler** - Locate the SchedulerConfig and set the EndAt property to any date previous to today's date.
  3. **Purge Job** - Locate the AccountPurgeJobConfig and set the Enabled property to `false`.

SCIM 2.0

The SCIM protocol is an application-level REST protocol for provisioning and managing identity data on the web. The protocol supports creation, discovery, retrieval, and modification of core identity resources.

References: IDMWORKS implements SCIM 2.0 as specified in the RFC documents.

- https://tools.ietf.org/html/rfc7642 : Definitions, Overview, Concepts, and Requirements
- https://tools.ietf.org/html/rfc7643 : Core Schema
- https://tools.ietf.org/html/rfc7644 : Protocol

Account Management Service

- The base URL for Account Management service (SCIM 2.0) is

  https://server.example:51003/scim/v2

Token-based Authentication

OAuth 2.0 enables clients to access protected resources by obtaining an access token, which is defined in "The OAuth 2.0 Authorization Framework" [RFC6749] as "a string representing an access authorization issued to the client", rather than using the resource owner's credentials directly.

The access tokens generated are JSON Web Token (JWT) Access Tokens conform to the JSON Web Token standard and contain information about an entity in the form of claims. Please refer section Authentication Service on page 9 for the Auth server details and different OAuth 2.0 flows to get access tokens.

The Account Management service (SCIM 2.0) uses token-based authentication. Access Tokens issued by Auth server are used to allow an application to access the Account Management APIs.

To access the Account Management (SCIM 2.0) APIs, the client / application should send the JWT, in the Authorization header using the Bearer schema. The content of the header should look like the following:

```
Authorization: Bearer <token>
```