Kerberos OAuth2 Grant

Kerberos is a security protocol that has support built into various operating systems and open-source distributions (e.g., Ubuntu, Windows, RedHat, OpenSolaris, etc). In addition, a majority of browsers support some Kerberos functions as well. As WSO2 API Manager uses the OAuth 2.0 protocol, the Kerberos OAuth2 grant type allows organizations to exchange a Kerberos ticket for an OAuth 2.0 token. Thereby, allowing organizations to re-use their existing Kerberos infrastructure, while easier adopting OAuth 2.0 within these organizations.

- Kerberos OAuth2 grant flow
- Configuring Kerberos Grant with API Manager

Kerberos OAuth2 grant flow

The following section describes the flow involved in exchanging a Kerberos ticket for an OAuth2 token.

1. The Kerberos client requests the Kerberos Service Ticket from the Kerberos Key Distribution Center (KDC) to invoke the service. The Kerberos Key Distribution Center can be any Kerberos Server.
2. The Kerberos Key Distribution Center sends a response with the Kerberos Service Ticket. If the client and the requested service is valid, the Key Distribution Center (KDC) sends a Kerberos ticket encrypted with the service owner's private key. The API handles the exchanging of the Ticket Granting Ticket (TGT), Service Granting Ticket (SGT), and all other low-level Kerberos details.
3. The Kerberos client requests the OAuth2 token. The message format of the OAuth2 token request should be as follows:
   - cURL Request Format
     You can use one of the following two cURL commands to request for the OAuth2 token.

![Kerberos OAuth2 Grant Diagram](image-url)
curl -v -X POST -H "Authorization: Basic <base64-encoded-client-id>:
<client-secret-value>" -k -d
"grant_type=kerberos&kerberos_realm=<kerberos-
realm>&kerberos_token=<kerberos-token>&scope=<scope>" -H "Content-Type:
apPLICATION/x-www-form-urlencoded" https://localhost:8243/token

curl -u <client-id>:<client-secret> -k -d
"grant_type=kerberos&kerberos_realm=<kerberos-
realm>&kerberos_token=<kerberos-token>&scope=<scope>" -H "Content-Type:
apPLICATION/x-www-form-urlencoded" https://localhost:8243/token

The "scope=my_scope" is an optional parameter that you can add to the string in the token request body.

Example

grant_type=kerberos&scope=my_scope&kerberos_realm=example.com&kerberos_token=YII1...

cURL Response

Example

POST /token HTTP/1.1
Host: idp.example.com:8243
Content-Type: application/x-www-form-urlencoded
Authorization: Basic
MW91TDJmTzZTeGxmRDJMRhcxMjVjVG8wd1FrYTp1UVV0bTg5dFk2UVp1YtvTcVpmTDkyQ
kRGJUFh
grant_type=kerberos&kerberos_realm=example.com&kerberos_token=YII1...

4. The Kerberos client receives the OAuth2 token. The Kerberos Grant validates the received token with the provided Identity Provider (IDP) credentials and if it is a valid token, it issues an OAuth2 token to the client.

Example

{"access_token":"636ce45f-c7f6-3a95-907f-d1f8aca28403",
"refresh_token":"831271d9-16ba-3bad-af18-b9f6592a8677",
"scope":"my_scope",
"token_type":"Bearer",
"expires_in":521}

Configuring Kerberos Grant with API Manager

Follow the instructions below to configure Kerberos Grant with WSO2 API Manager:

⚠️ Download the kerberos_grant_1.0.0.jar from here. Copy it to the <API-M_HOME>/repository/components/lib folder.
1. Add following entry under `<SupportedGrantTypes>` in the `<API-M_HOME>/repository/conf/identity/identity.xml` file.

```
<SupportedGrantType>
  <GrantTypeName>kerberos</GrantTypeName>
  <GrantTypeHandlerImplClass>org.wso2.carbon.identity.oauth2.grant.kerberos.ExtendedKerberosGrant</GrantTypeHandlerImplClass>
  <GrantTypeValidatorImplClass>org.wso2.carbon.identity.oauth2.grant.kerberos.KerberosGrantValidator</GrantTypeValidatorImplClass>
</SupportedGrantType>
```

2. Create a file named `jaas.conf` in the `<API-M_HOME>/repository/conf/identity` directory with the following content.

```
Server {
  com.sun.security.auth.module.Krb5LoginModule required
  useKeyTab=false
  storeKey=true
  useTicketCache=false
  isInitiator=false;
};
Client {
  com.sun.security.auth.module.Krb5LoginModule required
  useTicketCache=false;
};
```

3. Copy the following JARs into the `<API-M_HOME>/repository/components/dropins` directory.

- `org.wso2.carbon.identity.application.authenticator.iwa-5.3.0.jar`
- `org.wso2.carbon.identity.idp.metadata.saml2_1.0.1.jar`

4. Configure OAuth2 for your client application with the Kerberos grant type.

   a. Start the WSO2 API-M server by navigating to the `<API-M_HOME>/bin` directory in your console and running one of the following scripts based on your OS.

      - On Windows: `wso2server.bat --run`
      - On Linux/Mac OS: `sh wso2server.sh`

   b. Sign in to the API Store.
      `https://<hostname>:9443/store`
   c. Click `Applications` and click on the name of the application that you want to configure the OAuth2 with the Kerberos grant type.
   d. Generate the Production Keys.

      i. Click `Production Keys`.
      ii. Click on the `Kerberos` checkbox as shown in the screenshot.

      ![Screenshot of Production Keys with Kerberos checkbox](image)

   e. Generate the Sandbox Keys.

      i. Click `Sandbox Keys`.
      ii. Click on the `Kerberos` checkbox.
5. Configure the Service Principal Name (SPNName) and Service Principal Password (SPNPassword).

- **Identity Provider Name**: example.com
- **Alias**: https://192.168.53.12:9443/oauth2/token
- **Server Principal Name**: HTTP/idp.example.com@EXAMPLE.COM

6. Generate the Kerberos token.

Let's follow the instructions below to generate the Kerberos token using the KerbClientProject sample client.

6a. **Git clone the KerbClientProject.**

```bash
git clone https://github.com/erandacr/KerbClientProject
```

6b. **Run KerbClient.cs using an IDE.**

You can run it using Visual Studio by downloading and installing the following required libraries and programs.

- **Visual Studio sdk (.NET Core 2.1)**
- **Microsoft Visual Studio (Professional Edition)**
- **Install System.Net.Http.dll and define the path in the KerbClientProject.csproj file.**

6c. **Configure the following parameters in the project according to your setup.**
d. Run the project.
   Select the **Start without Debugging** option in the Visual Studio editor to run the project.

   The latter mentioned action will also invoke the token endpoint using the message format that was discussed in **step 3**.

   This project generates a Kerberos ticket. Thereafter it generates the Kerberos token using the Kerberos ticket, and finally, it passes the Kerberos token to generate the OAuth token.

   For the users to be counted in the **Registered Application Users statistics**, which considers the number of users that are shared across each application, they should generate access tokens using the **Password Grant** type.