Generate JWT Token using **Okta**

To generate a **JWT token** using **Okta**, which can be used to authenticate API calls to a **Qualys application**, you need to follow the steps below.

This process involves registering a application in **Okta**, obtaining the necessary credentials, and then using the **OAuth 2.0 Client Credentials flow** to get the token.

Step 1: Set Up Your Okta Developer Account

1. **Create an Okta Developer Account**: If you haven't done so yet, sign up for a free Okta account at Okta Developer.
2. Once you sign in, you'll be taken to the **Okta Admin Dashboard**.

Step 2: Create an Okta Application

Since you need to create JWT token for machine-to-machine authentication, you will use **Client Credentials Flow**.

1. **Log into Okta Admin Console**.
2. From the **Admin Dashboard**, go to **Applications**.
3. Click **Add Application**.
4. Select **a Service** for machine-to-machine integration (or **Web** if Service is not available).
5. Choose **OAuth 2.0** as the method.
6. Click **Next**.

Step 3: Configure the Application

1. **Application Name**: Give your application a name (e.g., "3rd Party API Integration").
2. **Grant Types**: Enable **Client Credentials**.
3. **Redirect URI**: Since you won't need a callback, this is not necessary in this case.
4. **Scopes**: Define the appropriate scopes.
5. Click **Done**.

Step 4: Obtain the Client ID and Client Secret

1. After creating the app, go back to the **Applications** section in Okta.
2. Click on the app you just created.
3. Under the **General** tab, you'll see the **Client ID** and **Client Secret**.
   - **Client ID**: This is your application's public identifier.
   - **Client Secret**: This is a secret key that your application will use to authenticate itself.

Save these values; you'll need them later when requesting a JWT token.

Step 5: Set Up OAuth 2.0 Authorization Server (If Not Already Done)

1. Go to **Security > API > Authorization Servers** in the Okta Admin Console.
2. Okta provides a **default authorization server**. Click on it to check its details.
3. Note the **Issuer URI**. This will be used for generating tokens, such as:
   - https://{yourOktaDomain}/oauth2/default

Step 6: Request JWT Token Using Client Credentials Flow

Now that you have your **Client ID**, **Client Secret**, and **Issuer URI**, you can request a JWT access token from Okta using the **Client Credentials Flow**.

1. **Make a POST request** to Okta's token endpoint to get a JWT access token.

URL:

https://{yourOktaDomain}/oauth2/default/v1/token

Replace {yourOktaDomain} with your actual Okta domain (e.g., dev-123456.okta.com).

2. **Set the Headers**:
   - Content-Type: application/x-www-form-urlencoded
3. **Prepare the Request Body**: The body must be URL-encoded and include:
   - client_id: Your application's Client ID.
   - client_secret: Your application's Client Secret.
   - grant_type: client_credentials (for the Client Credentials Flow).

- scope: The scope(s) your API requires (e.g., api.read).

Example body:

client_id=your_client_id

client_secret=your_client_secret

grant_type=client_credentials

scope=api.read

4. **Make the POST Request**: Example using curl:
5. curl -X POST https://{yourOktaDomain}/oauth2/default/v1/token \
6.   -H "Content-Type: application/x-www-form-urlencoded" \
7.   -d "client_id=your_client_id&client_secret=your_client_secret&grant_type=client_credentials&scope=api.read"

Or using Postman, configure a POST request with the above URL, headers, and body parameters.

8. **Response**: If the request is successful, Okta will return a JSON response containing the JWT access token, like so:
9. {
10. "access_token": "eyJraWQiOiJ...",
11. "token_type": "bearer",
12. "expires_in": 3600,
13. "scope": "api.read"
14. }
    - **access_token**: This is the JWT token that you'll use to authenticate API requests.
    - **token_type**: Typically, this will be "bearer".
    - **expires_in**: The time in seconds until the token expires (e.g., 3600 seconds = 1 hour).

Save the access_token, as you will need to include it in the headers when making requests to the third-party API.

Step 7: Use the JWT Token to Access Third-Party APIs

With the JWT token in hand, you can now authenticate requests to the third-party API.

1. Include the JWT token in the Authorization header of your HTTP requests to the third-party API.

Example:

```
curl -X GET https://api.thirdparty.com/resource \

  -H "Authorization: Bearer {access_token}"
```

Replace {access_token} with the actual token you received from Okta.

Step 8: Handle Token Expiry and Refreshing

JWT tokens typically expire after a set period (e.g., 1 hour). If your token expires, you will need to request a new one by repeating Step 6.

This approach allows you to authenticate with Okta and obtain a JWT for use with third-party APIs without needing a callback URL or user interaction.

References:

Access Tokens in Okta:

https://developer.okta.com/docs/guides/implement-oauth-for-okta-serviceapp/main/

https://developer.okta.com/docs/guides/build-self-signed-jwt/java/main/