**Okta OIDC Client Creation and Configuration for Spring Boot**

## **1. Creating an OIDC Client in Okta**

1. Log in to your Okta Developer Console.
2. Navigate to **Applications** > **Applications**.
3. Click on **Create App Integration**.
4. Select **OIDC - OpenID Connect** as the Sign-in method.
5. Choose **Web Application** and click **Next**.
6. Fill in the details:
   * App Name: SpringBootApp
   * Sign-in redirect URIs: http://localhost:8080/callback
   * Sign-out redirect URIs: http://localhost:8080
7. Assign users or groups as needed.
8. Click **Save** and copy the **Client ID** and **Client Secret**.

## **2. Configuring Default Scopes and Policies**

1. Navigate to **Security** > **API** > **Authorization Servers**.
2. Select the **default** authorization server.
3. Go to the **Scopes** tab and add the following scopes if not present:  
   * openid
   * profile
   * email
4. Go to the **Access Policies** tab:  
   * Create a new policy if needed.
   * Add a rule to allow access to your OIDC client.

## **3. Spring Boot Integration**

### **Dependencies**

Add the following dependencies to pom.xml:

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-oauth2-client</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-security</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.security</groupId>

<artifactId>spring-security-oauth2-jose</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.security</groupId>

<artifactId>spring-security-oauth2-resource-server</artifactId>

</dependency>

</dependencies>

### **Spring Security Filter Configuration**

Create a security configuration class:

import org.springframework.context.annotation.Bean;

import org.springframework.security.config.annotation.web.builders.HttpSecurity;

import org.springframework.security.web.SecurityFilterChain;

import org.springframework.beans.factory.annotation.Value;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.security.config.Customizer;

import org.springframework.security.config.annotation.web.builders.HttpSecurity;

import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;

import org.springframework.security.oauth2.jwt.JwtDecoder;

import org.springframework.security.oauth2.jwt.JwtDecoders;

import org.springframework.security.web.SecurityFilterChain;

@Configuration

@EnableWebSecurity

public class SecurityConfig {

@Value("${spring.security.oauth2.resourceserver.jwt.issuer-uri}")

private String issuerUri;

@Bean

public JwtDecoder jwtDecoder() {

// Replace this with your actual JWT issuer URL or secret for JWT signing key

return JwtDecoders.fromIssuerLocation(issuerUri);

}

@Bean

public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {

http

.authorizeHttpRequests((authorize) -> authorize

.requestMatchers("/h2/\*").permitAll()

.requestMatchers("/h2/login.do").permitAll()

.anyRequest().authenticated()

)

.oauth2Client(Customizer.withDefaults())

.oauth2ResourceServer((oauth2) -> oauth2.jwt(Customizer.withDefaults()));

return http.build();

}

}

### **Application Properties Configuration**

Add the following to application.properties:

spring.security.oauth2.resourceserver.jwt.issuer-uri=https://dev-28365606.okta.com/oauth2/default

spring.security.oauth2.client.registration.okta.client-id=0oancohml3daiDbi45d7

spring.security.oauth2.client.provider.okta.authorization-uri=https://dev-28365606.okta.com/oauth2/default/v1/authorize

spring.security.oauth2.client.provider.okta.token-uri=https://dev-28365606.okta.com/oauth2/default/v1/token

## **4. Running and Testing**

1. Start the Spring Boot application.
2. Access http://localhost:8080, and you should be redirected to Okta for authentication.
3. Upon successful login, Okta redirects back to the application with user authentication.

This setup ensures secure authentication and authorization for your Spring Boot application using Okta OIDC.