Excercise for OAuth2

Andreas Falk

Table of Contents

Chapter 1. What we will build

We will extend the existing two microservices to use single sign authentication based on OAuth2.

- OAuth2 Authorization Server: This is the new microservice for single sign on which holds all users with their credentials
- OAuth2 Resource Server (Product Backend): The microservice providing product data maps to a resource server
- OAuth2 Client (UI Microservice): The thymeleaf UI microservice consuming the products maps to an OAuth2 client

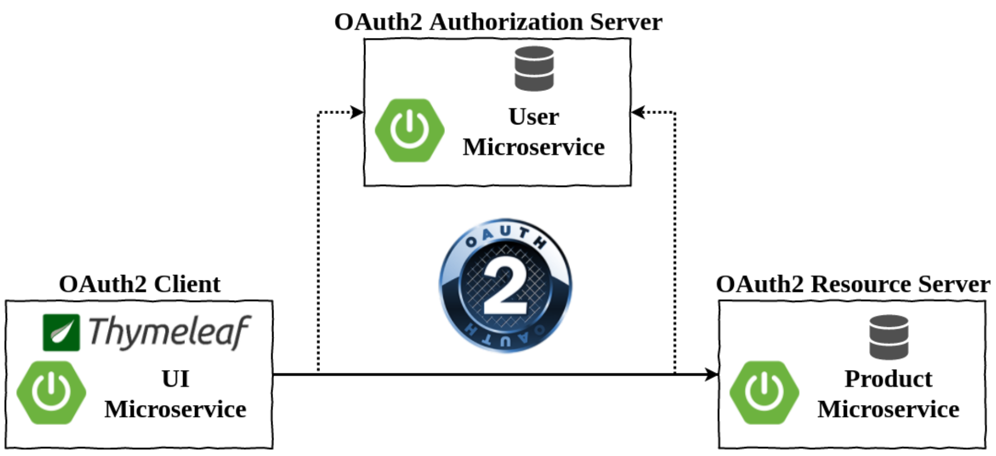

Table 1. Microservice URL Adresses

Microservice	URL
Authorization Server	http://localhost:9999/users
Client (UI)	http://localhost:8081
Resource Server (Products)	http://localhost:8080

Chapter 2. Basic OAuth2 Components

2.1. Authorization Server

You may look into the spring boot reference documentation Spring Boot Reference Documentation on how to implement an authorization server.

To prevent conflicts with different JSESSION cookies the authorization server must run on a separate context path (not '/'). In our example please use '/users' as context path. In spring boot this can be achieved by the *server.context* property

To ensure OAuth2 authorization code grant works correctly with the other components the end points of the authorization server must be as follows:

Table 2. Authorization Server Endpoints

Endpoint	Description	Caller
/oauth/authorize	Authorization endpoint (for login and client authorization)	Client
/oauth/token	Token endpoint (exchanges given authorization code for access token)	Client
/oauth/check_token	Check token endpoint (returns internal contents for access token)	Resource Server

2.1.1. Maven dependencies

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>com.example</groupId>
    <artifactId>authorizationserver</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <packaging>jar</packaging>

    <name>authorizationserver</name>
    <description>Demo project for Spring Boot</description>

    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>1.5.3.RELEASE</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>

    <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
        <java.version>1.8</java.version>
    </properties>

    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-actuator</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-security</artifactId> ①
        </dependency>
        <dependency>
            <groupId>org.springframework.security.oauth</groupId>
            <artifactId>spring-security-oauth2</artifactId> ②
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-data-jpa</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-hateoas</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>

        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-devtools</artifactId>
            <scope>runtime</scope>
        </dependency>
        <dependency>
            <groupId>com.h2database</groupId>
            <artifactId>h2</artifactId>
            <scope>runtime</scope>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
        </dependency>
    </dependencies>

    <build>
        <plugins>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
            </plugin>
        </plugins>
    </build>

</project>
```

① Dependency for base security filters (e.g. basic authentication)

② Dependency for OAuth2 support

2.1.2. Java Implementation

```
@EnableAuthorizationServer ①
@SpringBootApplication
public class AuthorizationServerApplication {

    public static void main(String[] args) {
        SpringApplication.run(AuthorizationServerApplication.class, args);
    }

}
```

① Annotation to enable auto configuration of an Authorization Server

2.1.3. Configuration

```
server.context-path=/users
server.port=9999
security.oauth2.client.client-id=productclient ①
security.oauth2.client.client-secret=secretkey ②
security.oauth2.client.scope=read-products ③
security.oauth2.authorization.check-token-access=isAuthenticated() ④
security.user.name=user ⑤
security.user.password=secret ⑥
```

2.2. Resource Server (Products)

2.2.1. Maven dependencies

```
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-security</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.security.oauth</groupId>
        <artifactId>spring-security-oauth2</artifactId>
    </dependency>
```

2.2.2. Java Implementation

```
@EnableResourceServer ①
@SpringBootApplication
public class ProductApplication {

    ...

    public static void main(String[] args) {
        SpringApplication.run(ProductApplication.class, args);
    }
}
```

2.3. OAuth2 Client (Thymeleaf UI)

2.3.1. Maven dependencies

```
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-security</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.security.oauth</groupId>
        <artifactId>spring-security-oauth2</artifactId>
    </dependency>
```

2.3.2. Java Implementation

```
@EnableOAuth2Sso ①
@SpringBootApplication
public class UiApplication {

    public static void main(String[] args) {
        SpringApplication.run(UiApplication.class, args);
    }

    @Bean
    public OAuth2RestTemplate oauth2RestTemplate(OAuth2ClientContext
oauth2ClientContext, ②
                                                 OAuth2ProtectedResourceDetails
details) {
        return new OAuth2RestTemplate(details, oauth2ClientContext);
    }
}
```

Chapter 3. Advanced Level

To make the sample application even more secure we will enhance the authorization server to…

- …use a persistent store for users
- …encrypt the passwords
- …enable login using a form login page

3.1. Use persistent store

3.2. Encrypt the passwords

3.3. Provide form based login