УДК 004

Защита приложения Spring Boot с помощью App ID

Еровлев Павел Андреевич
Приамурский государственный университет имени Шолом-Алейхема
Студент

Еровлева Регина Викторовна
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В данной статье рассматривается пример использования App ID Spring Boot Starter, который предоставляет разработчикам приложений Spring Boot простой способ использовать IBM Cloud App ID для защиты в приложениях. В статье будет использоваться язык программирования Java и фреймворк SpringBoot.
Ключевые слова: Java, Spring, IBM

Securing Spring Boot Applications with App ID

Erovlev Pavel Andreevich
Sholom-Aleichem Priamursky State University
Student

Erovleva Regina Victorovna
Sholom-Aleichem Priamursky State University
Student

Abstract
This article provides an example of using the Spring Boot Starter App ID, which provides Spring Boot application developers with an easy way to use the IBM Cloud App ID to secure applications. The article will use the Java programming language and the SpringBoot framework.
Keywords: Java, Spring, IBM

1 Введение

1.1 Актуальность

IBM Cloud App ID — это облачная служба, которая позволяет разработчикам добавлять в приложения возможности аутентификации и авторизации, в то время как все операционные аспекты службы обрабатываются в IBM Cloud Platform. Идентификатор приложения предназначен для разработчиков, которые хотят защитить приложения, но не имеют ресурсов для реализации всех протоколов безопасности. Сервис

предоставляет такие возможности, как облачный каталог, федерацию корпоративных удостоверений, вход через социальные сети, систему единого входа, настраиваемый виджет входа в систему, гибкие элементы управления доступом и профили пользователей, многофакторную аутентификацию, набор SDK для удобной настройки приложений.

1.2 Обзор исследований

В.Л. Радишевский и А.Д. Кульневич описали принцип работы распределенного брокера сообщений Kafka для высокоскоростной передачи и агрегации данных [1]. Ю.А. Флёров и Л.Л. Вышинский разработали программу для организации взаимодействия с Web-клиентами в программных комплексах [2]. А.И. Куреленко разработал программу предназначенную для генерации, приема и передачи типизированных сообщений для программного брокера Apache Kafka [3]. М.А. Потовиченко, М.В. Привалов и С.В. Корнев рассмотрели разработку программного продукта, который обеспечивает учет данных посещения занятий студентов, а также защиту их работ [4]. В.А. Сухомлин описал кратко в своей статье принцип работа комплексной программы дополнительного образования, ориентированную на подготовку разработчиков Java enterprise приложений [5].

1.3 Цель исследования

Цель исследования – использовать App ID Spring Boot Starter для защиты приложения Spring Boot с помощью App ID.

2 Материалы и методы

Для реализации будет использоваться язык программирования Java и фреймворк SpringBoot.

3 Результаты и обсуждения

На странице «Spring Initializr» сгенерируем «Maven Project» со следующими спецификациями (рис.1).

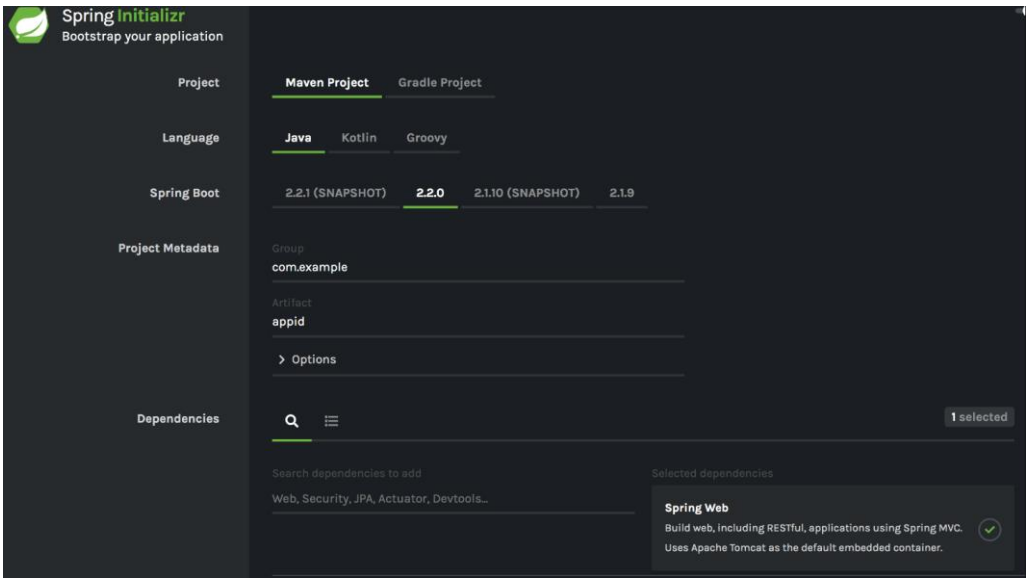

Рисунок 1 – Создание проекта

Далее в каталоге IBM Cloud в категории «Безопасность и идентификация» выберем службу App ID (рис.2).

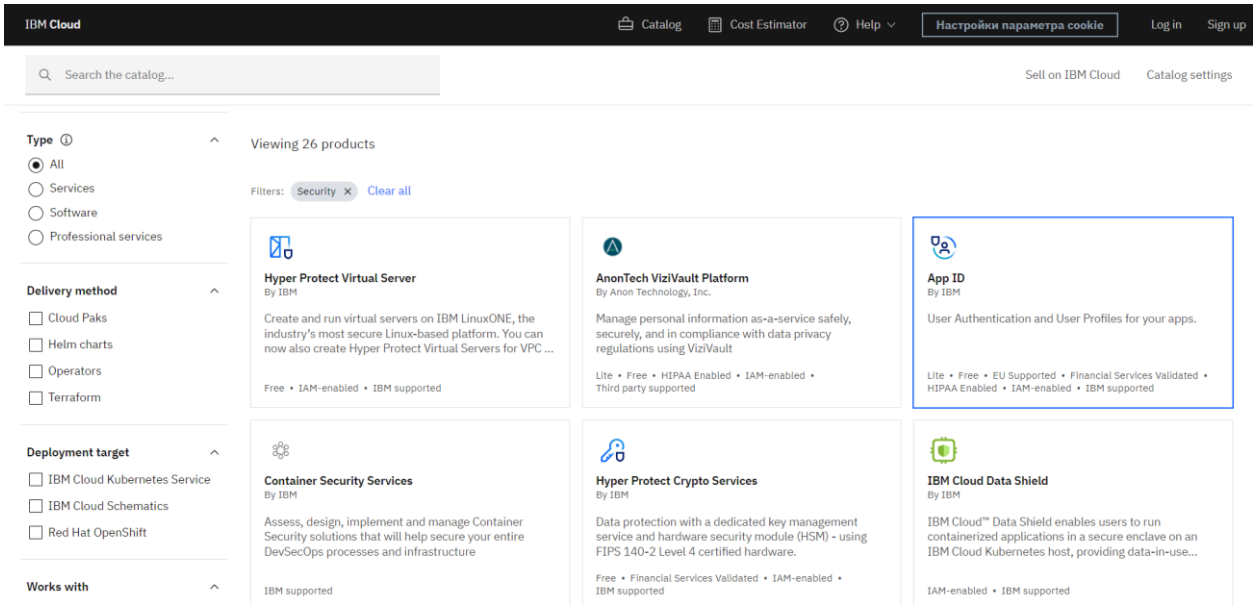

Рисунок 2 – Выбор службы на сайте IBM

Далее отредактируем разархивированный проект «Spring Initializr» и добавим конфигурацию в «application.yml» файл со следующими именами свойств (рис.3)

```yaml
spring:
  security:
    oauth2:
      client:
        registration:
          appid:
            clientId: <<clientId>>
            clientSecret: <<clientSecret>>
            region: <<region>>
            tenantId: <<tenantId>>
```

Рисунок 3 – Файл application.yml

Теперь добавим url адрес перенаправления. Для этого находясь на панели мониторинга службы App ID, перейдем в раздел «Управление проверкой подлинности», далее «Управление поставщиками удостоверений».

Добавим «http://localhost:8080/login/oauth2/code/appid» в качестве URL-адреса веб-перенаправления.

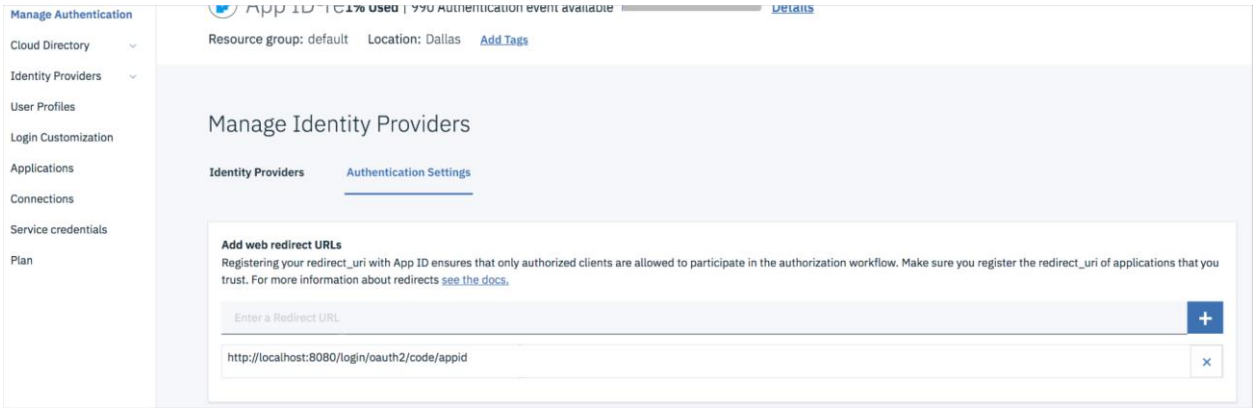

Рисунок 4 – Добавление адреса переадресации

После того как «App ID» завершает процесс OAuth2, он перенаправляется на указанный URL-адрес.

Также добавим «App ID Spring Boot Starter» и другие необходимые зависимости в pom.xml (рис.5).

```xml
<dependencies>
    <dependency>
        <groupId>com.ibm.cloud.appid</groupId>
        <artifactId>appid-spring-boot-starter</artifactId>
        <version>0.0.5</version>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
    </dependency>
    <dependency>
        <groupId>org.webjars</groupId>
        <artifactId>jquery</artifactId>
        <version>2.1.1</version>
    </dependency>
    <dependency>
        <groupId>org.webjars</groupId>
        <artifactId>webjars-locator-core</artifactId>
    </dependency>
    <dependency>
        <groupId>org.webjars</groupId>
        <artifactId>js-cookie</artifactId>
        <version>2.1.0</version>
    </dependency>
    <dependency>
        <groupId>org.webjars</groupId>
        <artifactId>bootstrap</artifactId>
        <version>3.2.0</version>
    </dependency>
</dependencies>
```

Рисунок 5 – Файл Pom.xml

Создадим класс «SecurityConfiguration.java» и добавим настройки безопасности (рис.6)

```java
@Configuration
@EnableWebSecurity
public class SecurityConfiguration extends WebSecurityConfigurerAdapter {

    @Override
    protected void configure(HttpSecurity http) throws Exception {
        http
        .authorizeRequests()
        .antMatchers("/login**", "/user", "/userInfo").authenticated()
        .and()
        .oauth2Login();
    }
}
```

Рисунок 6 – Класс SecurityConfiguration

Здесь происходит настройка безопасности при расширении «WebSecurityConfigurerAdapter» класса и переопределении метода «configure».

«/user» и «/userInfo» конечные точки доступны только аутентифицированному пользователю.

Чтобы добавить конечные точки REST, создадим класс «UserController.java» и добавим их (рис.7).

```
@RestController
public class UserController {

    @RequestMapping("/user")
    public Principal user(@AuthenticationPrincipal Principal principal) {
        return principal;
    }

    @RequestMapping("/userInfo")
    public String userInfo(@AuthenticationPrincipal Principal principal) {
        return String.valueOf(principal);
    }
}
```

Рисунок 7 – Класс UserController

Конечная точка «/user» предоставляет вошедшему в систему пользовательский объект, а конечная точка «/userInfo» возвращает строку с данными пользователя.

Создадим файл в формате «index.html», и добавьте следующий код, который показывает информацию о вошедшем в систему пользователе (рис.8)

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Spring Boot App ID Sample</title>
        <link type="text/css" href="css/style.css" rel="stylesheet" />
        <script type="text/javascript" src="/webjars/jquery/jquery.min.js"></script>
        <script type="text/javascript" src="/webjars/js-cookie/js.cookie.js"></script>
        <script type="text/javascript">
            $.ajaxSetup({
                beforeSend : function(xhr, settings) {
                    if (settings.type == 'POST' || settings.type == 'PUT' || settings.type == 'DELETE' || settings.type == 'GET') {
                        if (!(/^http:.*/.test(settings.url) || /^https:.*/.test(settings.url))) {
                            xhr.setRequestHeader("X-XSRF-TOKEN", Cookies.get('XSRF-TOKEN'));
                        }
                        xhr.setRequestHeader("X-XSRF-TOKEN", Cookies.get('XSRF-TOKEN'));
                    }
                }
            });
            $.get("/user", function(data) {
                if (data.principal != null) {
                    $("#user").html(data.principal.attributes.name);
                    $("#userSub").html(data.principal.attributes.sub);
                    $("#userEmail").html(data.principal.attributes.email);
                    $("#provider").html(data.principal.attributes.identities[0].provider);
                    $(".unauthenticated").hide();
                    $(".authenticated").show();
                } else {
                    $(".unauthenticated").show();
                    $(".authenticated").hide();
                }
            }).fail(function() {
                $(".unauthenticated").show();
                $(".authenticated").hide();
            });
            $.get("/userInfo", function(data) {
                if (data.includes("Principal")) {
                    $("#userInfoString").html(data);
                    $(".unauthenticated").hide();
                    $(".authenticated").show();
                } else {
                    $(".unauthenticated").show();
                    $(".authenticated").hide();
                }
            }).fail(function() {
                $(".unauthenticated").show();
                $(".authenticated").hide();
            });
        </script>
    </head>
    <body>
        <div class="container unauthenticated" style="text-align: center;">
            <a href="/login">Login</a>
        </div>
        <div class="container authenticated" style="text-align: center;" >
            <strong>Logged in as: <span id="user"></span></strong>
            <br>
            <br>
            <strong>Sub: </strong><span id="userSub"></span>
            <br>
            <strong>Email: </strong><span id="userEmail"></span>
            <br>
            <strong>Provider: </strong><span id="provider"></span>
            <br>
            <br>
            <strong>User Profile Information: </strong>
            <br>
            <span id="userInfoString"></span>
            <br>
            <br>
        </div>
```

Рисунок 8 – Файл index.html

Теперь можно запускать проект и проверять его работу (рис.9).

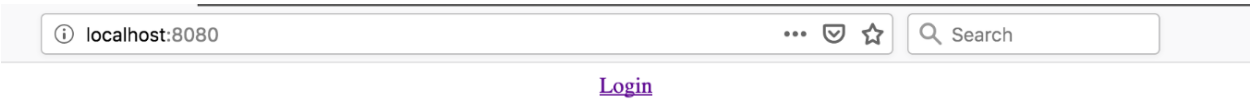

Рисунок 9 – Главная страница

Теперь нажмем на кнопгу авторизации и убедимся, что авторизация происходит на стороне App ID (рис.10).

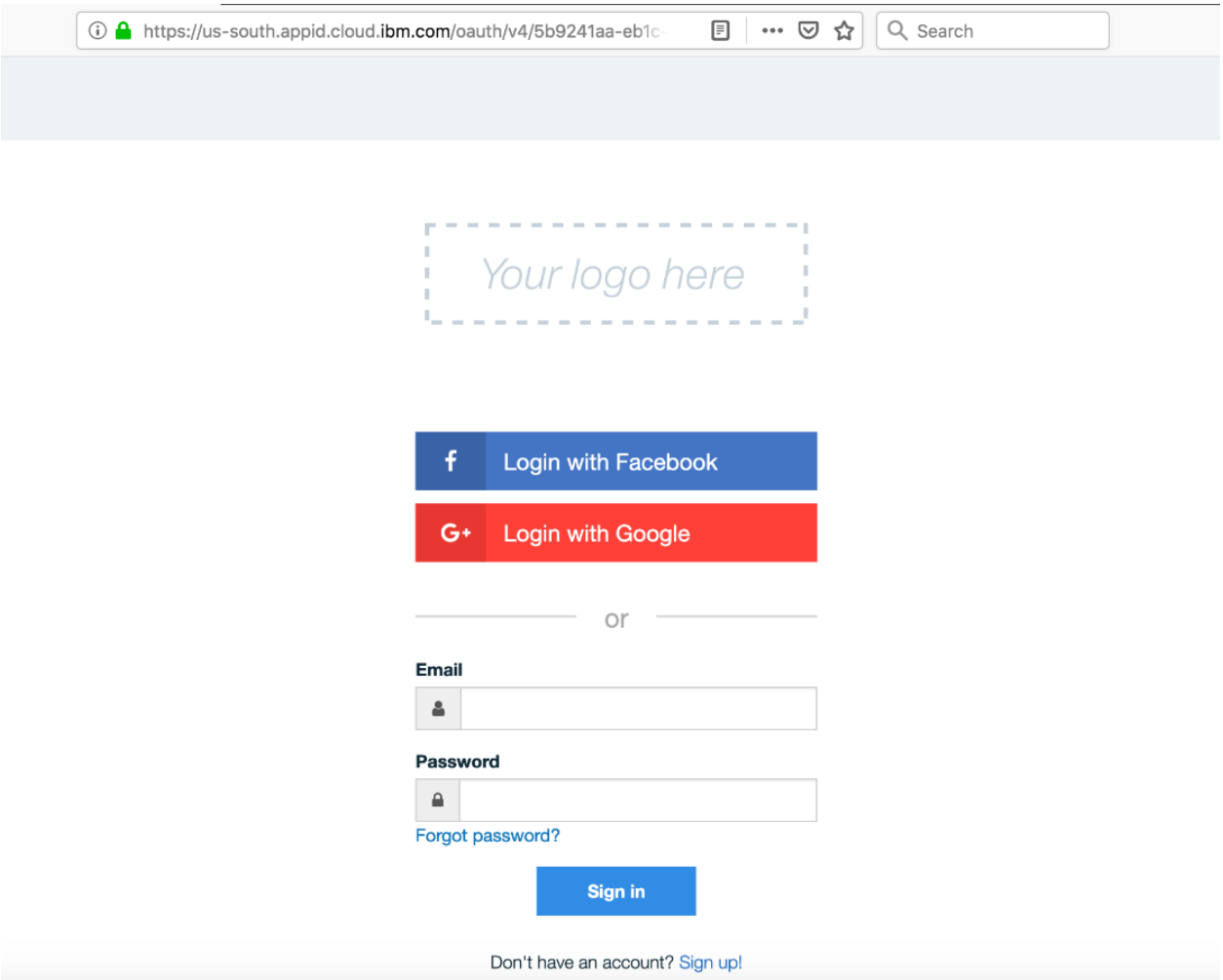

Рисунок 10 – Переадресация для авторизации

После успешного входа будете выполнено перенаправление обратно в образец приложения, который был создан с информацией о пользователе.

Выводы

App ID Spring Boot Starter упрощает настройку и использование приложения в стиле Spring Security с использованием автоматической настройки и свойств Spring Boot.

Библиографический список

1. Радишевский В.Л., Кульневич А.Д. Распределенный брокер сообщений kafka для высокоскоростной передачи и агрегации данных // Молодёжь и современные информационные технологии. 2018. № 1. С. 284-285.
2. Флёров Ю.А. и Вышинский Л.Л. Программа организации интерфейса web-серверов с java-приложениями // Аллея науки. 2016. № 7. С. 58-61.
3. Куреленко А.И. Генератор сообщений для программного брокера сообщений apache kafka // Тихоокеанский государственный университет. 2017. С. 171-176.
4. Потовиченко М.А., Привалов М.В., Корнев С.В. Компьютеризированная

подсистема учета текущей успеваемости студента в условиях вуза // Информатика, управляющие системы, математическое и компьютерное моделирование. 2019. № 2. С. 71-75.

5. Сухомлин В.А. Подготовка разработчиков корпоративных java-приложений в режиме дистанционного обучения // Информатика, управляющие системы, математическое и компьютерное моделирование. 2013. № 9. С. 175-180.