Spring Authorization Server入门 (九) Spring Boot引入Resource Server对接认证服务

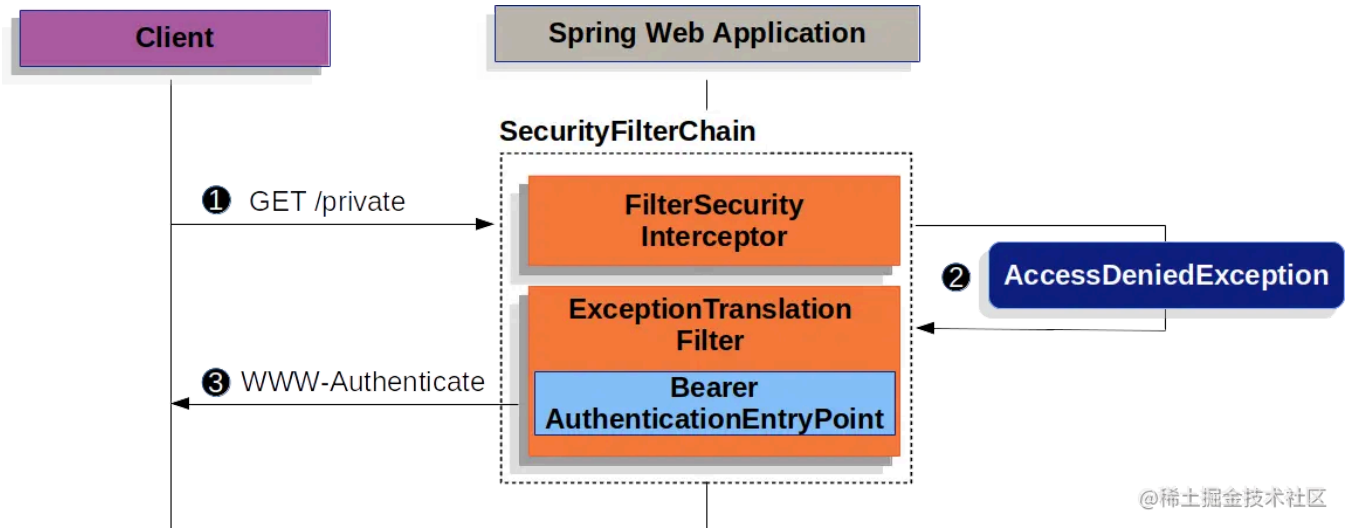
叹雪飞花 2023-06-13 1,840 阅读5分钟

前言

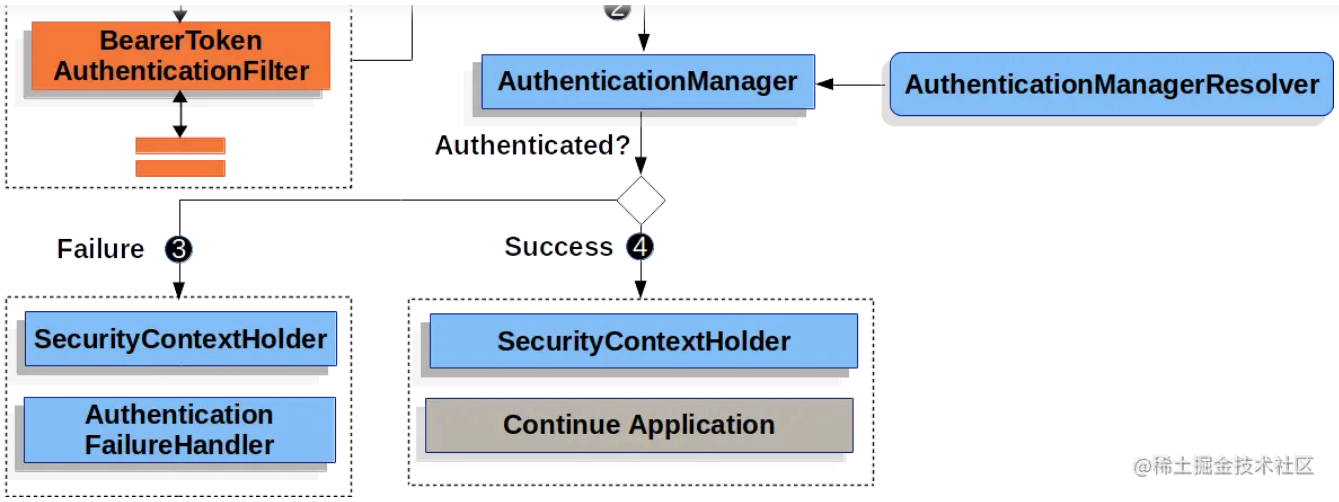
书接上文，本次来对接一下资源服务，在本篇文章中会带领大家去构建一个资源服务器，通过注解校验token中的权限，怎么放行一个接口，使其不需要认证也可访问。

流程解析

文档



没有携带token访问认证信息会抛出 `AccessDeniedException` 异常，并且会调用 `BearerAuthenticationEntryPoint` 去处理。



请求携带token到达资源服务器后会使用 `BearerTokenAuthenticationFilter` 去解析和校验token，成功会将认证信息存入 `SecurityContextHolder` 中并继续往下执行，失败则调用 `AuthenticationEntryPoint` 返回异常信息。

以上两种异常处理默认都是在响应头中添加，响应头是WWW-Authenticate，值就是具体的异常信息。

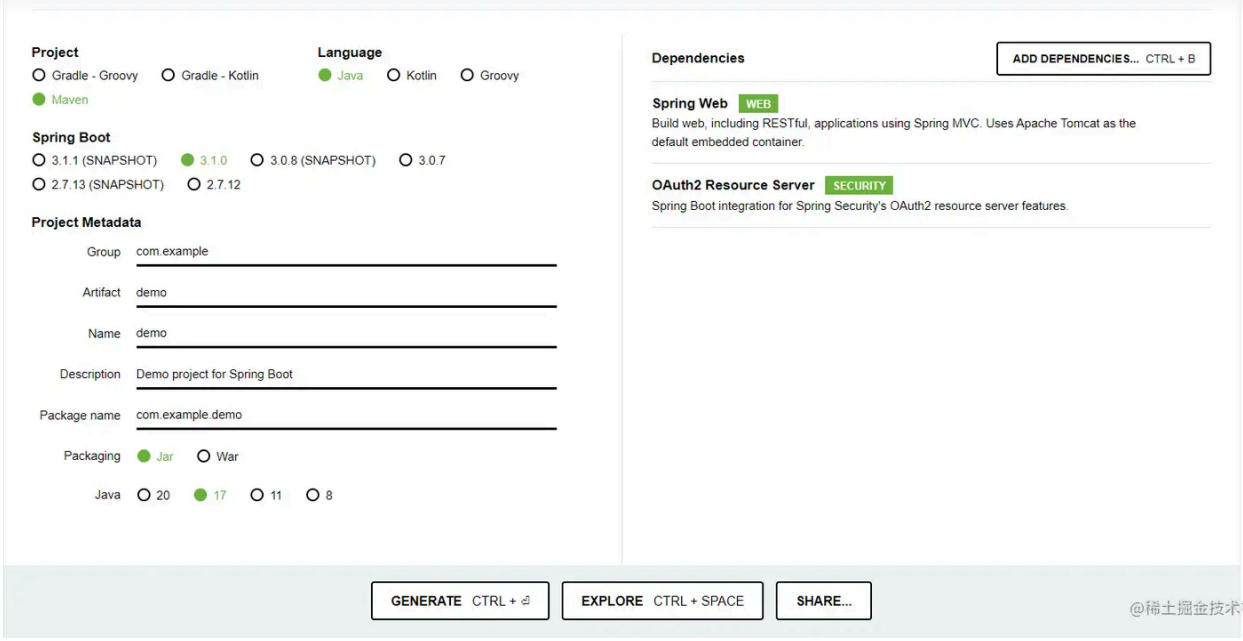
整合过程

1. 创建项目
2. 添加resource server和web依赖
3. 添加yml配置

resource server和oauth2 client一样，是一个单独的服务，不需要跟认证服务器的版本保持一致，读者可自选springboot版本，它们之间通过oauth2的协议可以无缝对接。

创建项目

通过IDEA或Spring Initializr创建一个项目，同时选择web和OAuth2 Resource Server依赖。



pom.xml如下

xml 复制代码

```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-i
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/mave
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>3.1.0</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>
    <groupId>com.example</groupId>
    <artifactId>resource-server-example</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>resource-server-example</name>
    <description>resource-server-example</description>
    <properties>
        <java.version>17</java.version>
    </properties>
    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-oauth2-resource-server</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
```

```
            <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>

```

添加yml配置

资源服务器配置文档

yaml 复制代码

```yaml
server:
  # 设置资源服务器端口
  port: 8100

spring:
  security:
    oauth2:
      # 资源服务器配置
      resourceserver:
        jwt:
          # Jwt中claims的iss属性，也就是jwt的签发地址，即认证服务器的根路径
          # 资源服务器会进一步的配置，通过该地址获取公钥以解析jwt
          issuer-uri: http://192.168.120.33:8080
```

至此，一个简易的资源服务就搭建完毕了，资源服务比客户端服务的逻辑稍微简单些，就是从认证服务获取公钥，然后解析jwt类型的token。

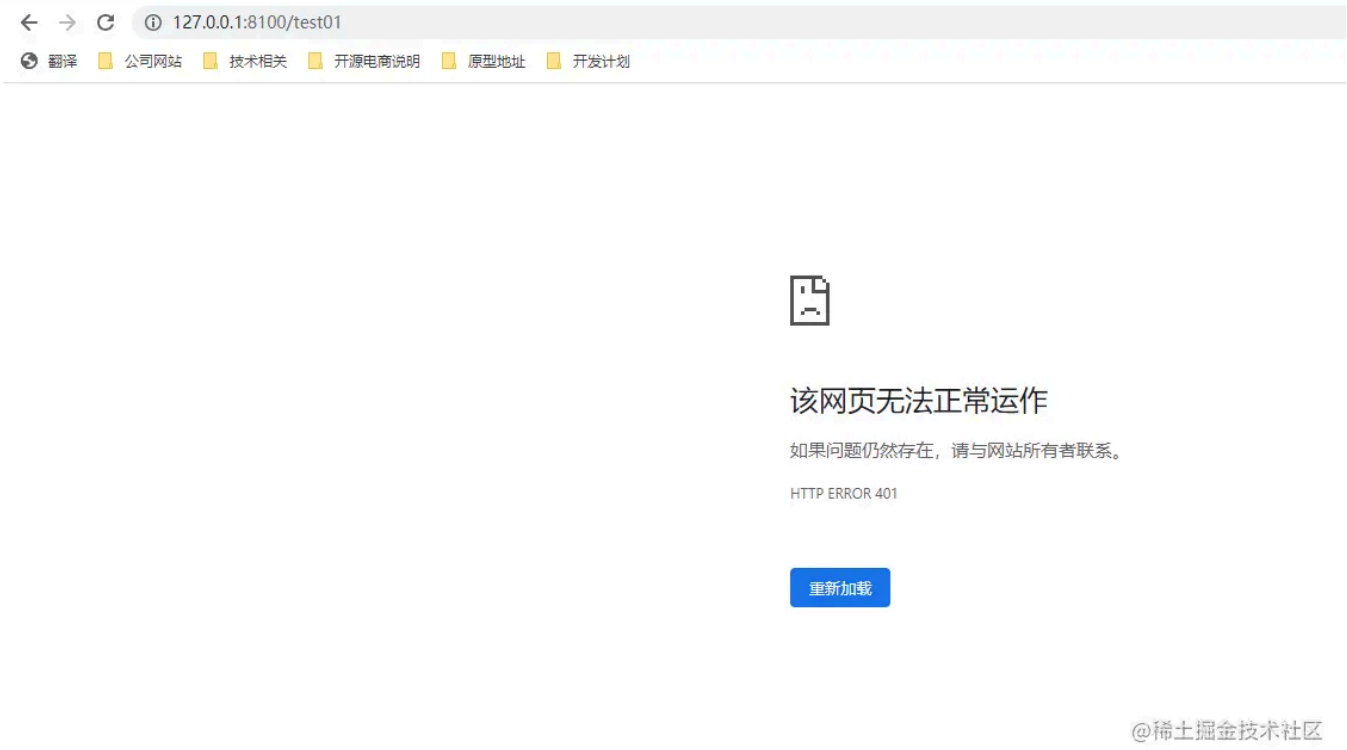
添加测试接口

照例，从认证服务copy一下TestController

java 复制代码

```java
package com.example.controller;

import org.springframework.security.access.prepost.PreAuthorize;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

/**
 * 测试接口
 *
 * @author vains
 */
@RestController
public class TestController {

    @GetMapping("/test01")
    @PreAuthorize("hasAuthority('SCOPE_message.read')")
    public String test01() {
        return "test01";
    }

    @GetMapping("/test02")
    @PreAuthorize("hasAuthority('SCOPE_message.write')")
    public String test02() {
        return "test02";
    }

    @GetMapping("/app")
    @PreAuthorize("hasAuthority('app')")
    public String app() {
        return "app";
    }

}
```

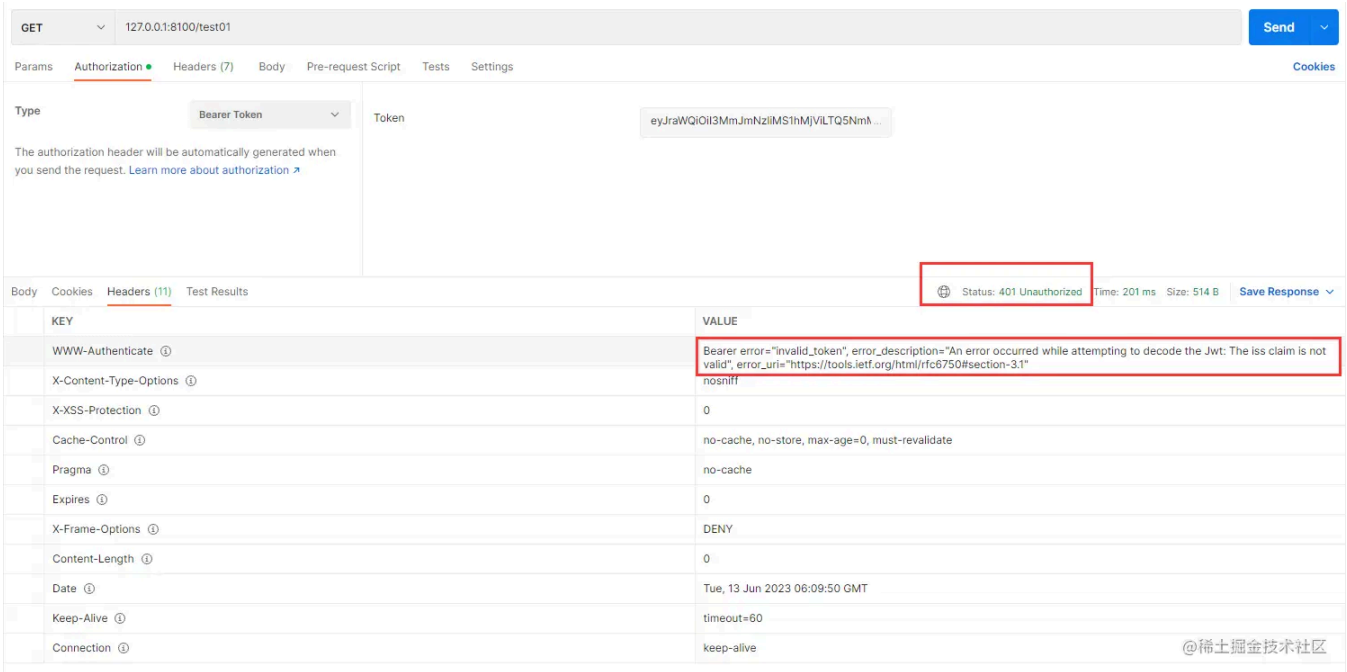
添加完成后启动认证服务和资源服务



http状态码401代表尚未认证，可见已经生效，需要携带token访问接口。

Postman访问test01

从认证服务器获取一个access_token，放入请求头中。



携带token却依然响应了401，通过异常描述发现是iss属性与代码中配置的不一致；使用在线解析jwt的网站解析一下access token查看一下iss属性

jwt中的iss是127.0.0.1:8080，但是资源服务器中配置的是192.168.120.33:8080，如果只是开发阶段则spring.security.oauth2.resourceserver.jwt.issuer-uri配置为127.0.0.1:8080的也没什么问题，但如果认证服务要部署至公网或者其它测试机中，这时候就要设置认证服务的iss属性了。

设置认证服务生成的jwt中的iss属性

更改认证服务中的AuthorizationConfig.java，修改AuthorizationServerSettings的配置，设置jwt签发地址

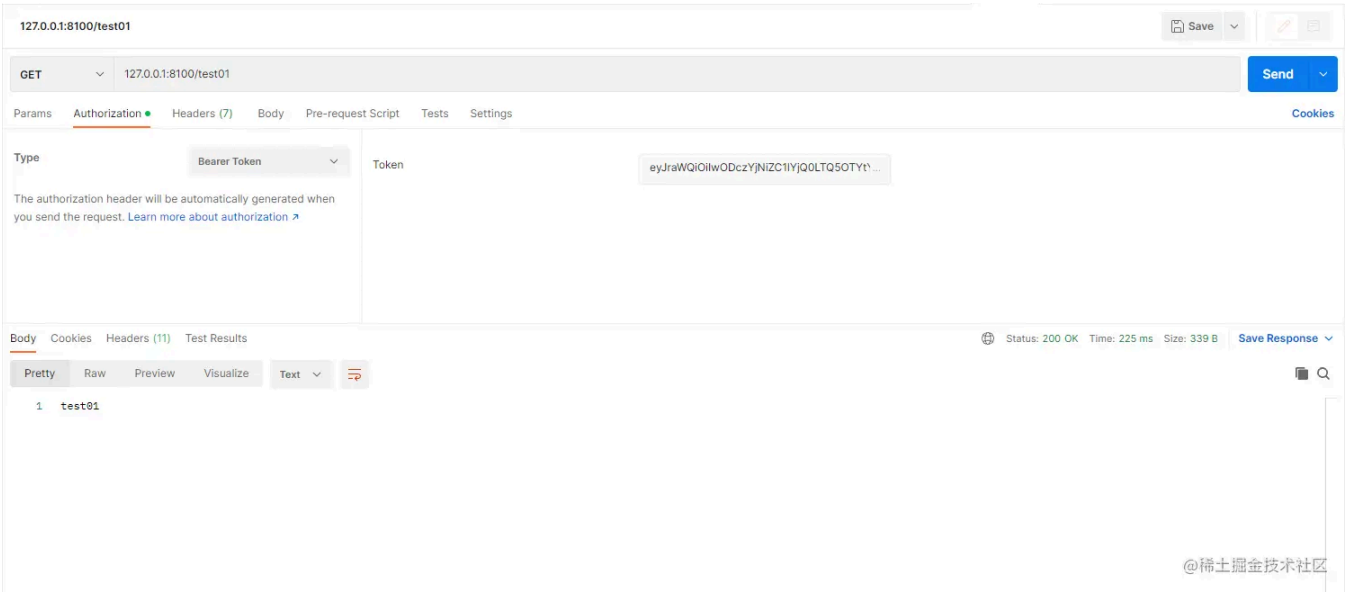
java 复制代码

```java
/**
 * 添加认证服务器配置，设置jwt签发者、默认端点请求地址等
 *
 * @return AuthorizationServerSettings
 */
@Bean
public AuthorizationServerSettings authorizationServerSettings() {
    return AuthorizationServerSettings.builder()
            /*
                设置token签发地址(http(s)://{ip}:{port}/context-path, http(s)://domain.com/contex
                如果需要通过ip访问这里就是ip，如果是有域名映射就填域名，通过什么方式访问该服务这里就填什
             */
            .issuer("http://192.168.120.33:8080")
```

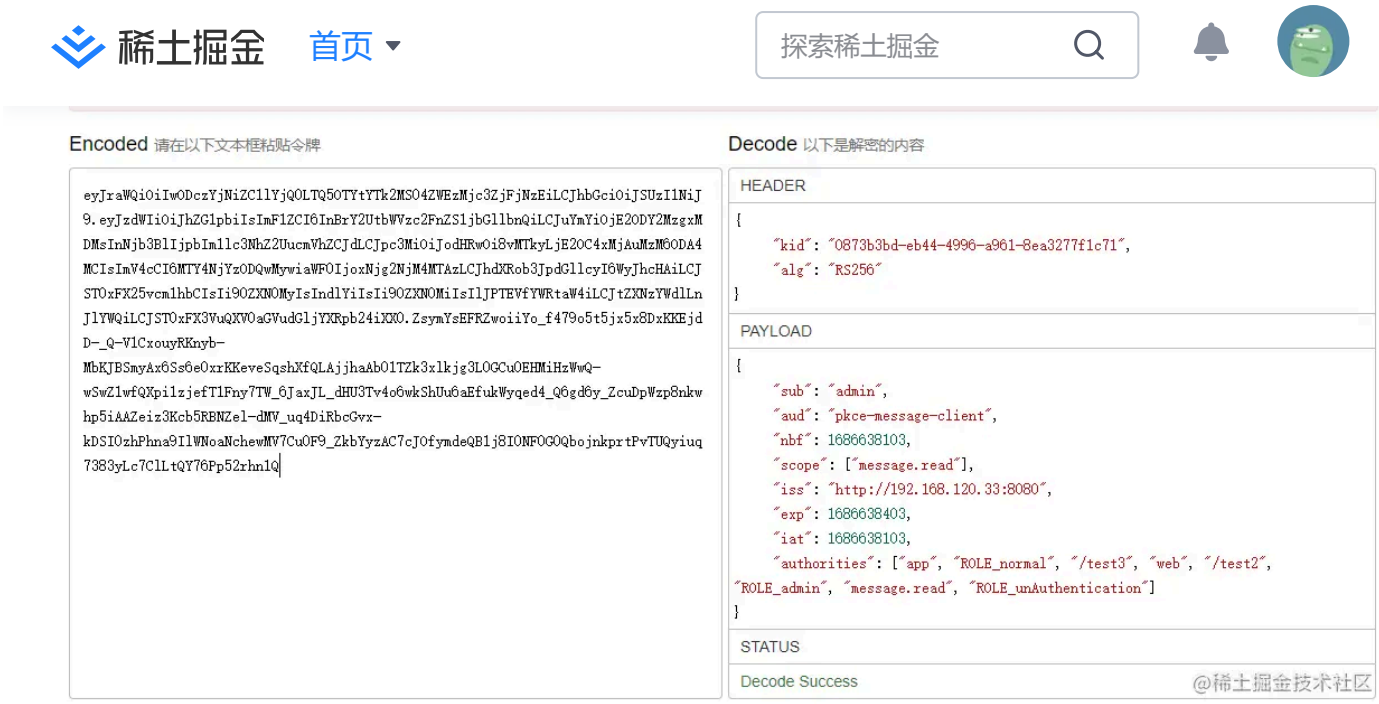
重启认证服务与资源服务

重新测试

从认证服务器申请一个token，通过postman对资源服务发起请求。

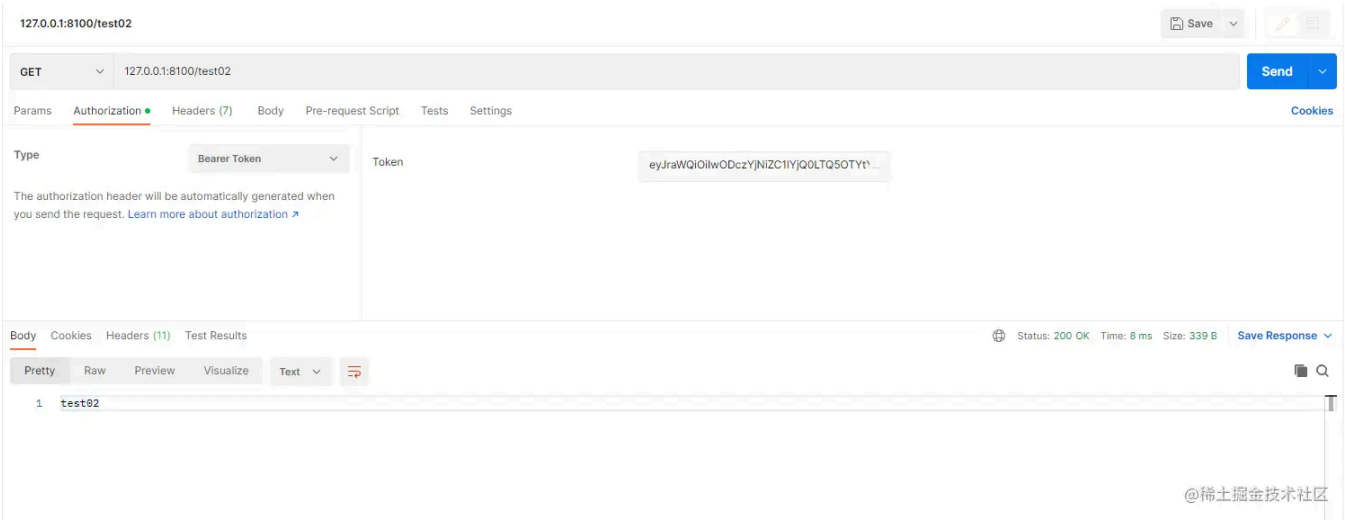


这时候就没有什么问题，说明资源服务可以正确解析token了，并且提取出了认证信息，来解析一下token看看



大家可以看到这里的iss属性是之前在认证服务中配置的地址。

接下来测试下test02接口



也请求成功了

添加注解使鉴权注解生效

大家应该也注意到了，测试接口上是有权限校验注解的，请求test01时token中message.read的scope，但是请求test02时token中并没有message.write的scope，请求成功说明注解尚未生效，这时候就需要添加注解了，也就是该系列文章的第三篇中提到的 `@EnableWebSecurity` 和 `@EnableMethodSecurity`

在上一篇文章中的AuthorizationConfig.java配置类中，类上有三个注解，分别是@Configuration、@EnableWebSecurity和@EnableMethodSecurity注解，虽然在类中有注释，但是这里在细讲一下，同时放一下官网的说明

1. @EnableWebSecurity

> 1. 加载了WebSecurityConfiguration配置类, 配置安全认证策略。
> 2. 加载了AuthenticationConfiguration, 配置了认证信息。

2. @EnableMethodSecurity 官网文档说明

> 默认启用方法级别的安全校验
>
> 1. 设置注解属性 jsr250Enabled = true 是为了启用JSR250注解支持，例如@RolesAllowed、@PermitAll和@DenyAll注解
> 2. 设置属性securedEnabled = true 是为了启用@Secured注解支持，不设置属性则添加Secured注解无效

3. @Configuration 文档说明地址 在Spring security 6.0 版本及之后版本中将@Configuration注解从@EnableWebSecurity, @EnableMethodSecurity, @EnableGlobalMethodSecurity 和 @EnableGlobalAuthentication 中移除，使用这些注解需手动添加 @Configuration 注解

在config包下创建ResourceServerConfig类

java 复制代码

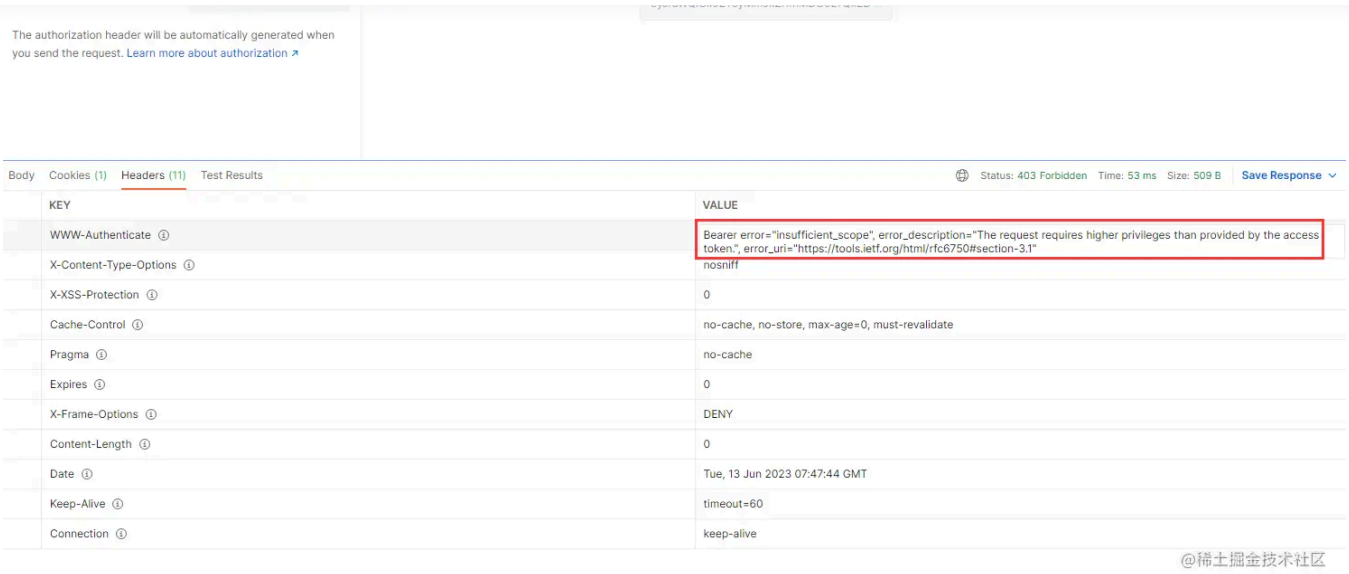
```java
package com.example.config;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.security.config.Customizer;
import org.springframework.security.config.annotation.method.configuration.EnableMethodSecurity;
import org.springframework.security.config.annotation.web.builders.HttpSecurity;
import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
import org.springframework.security.web.SecurityFilterChain;

/**
 * OAuth2 Resource Server
 *
 * @author Yu jin xiang 2023/6/13
```

```java
@EnableMethodSecurity(jsr250Enabled = true, securedEnabled = true)
public class ResourceServerConfig {

    @Bean
    public SecurityFilterChain defaultSecurityFilterChain(HttpSecurity http) throws Exception {
        http.authorizeHttpRequests(authorize -> authorize
                        // 下边一行是放行接口的配置，被放行的接口上不能有权限注解，e.g. @PreAuthorize
                        // .requestMatchers("/test02").permitAll()
                        .anyRequest().authenticated()
                )
                .oauth2ResourceServer(oauth2 -> oauth2
                        // 可在此处添加自定义解析设置
                        .jwt(Customizer.withDefaults())
                        // 添加未携带token和权限不足异常处理(已在第五篇文章中说过)
//                        .accessDeniedHandler(SecurityUtils::exceptionHandler)
//                        .authenticationEntryPoint(SecurityUtils::exceptionHandler)
                );
        return http.build();
    }

    // 添加自定义解析token配置，注入一个JwtAuthenticationConverter
    // (已在第六章中说过，这里就不重复实现了)

}
```

资源服务的异常处理已在第五章讲过，这一章就不详细说了，按照代码中的位置配置即可，第六章中也讲解过如何配置一个JwtAuthenticationConverter去自定义解析token。

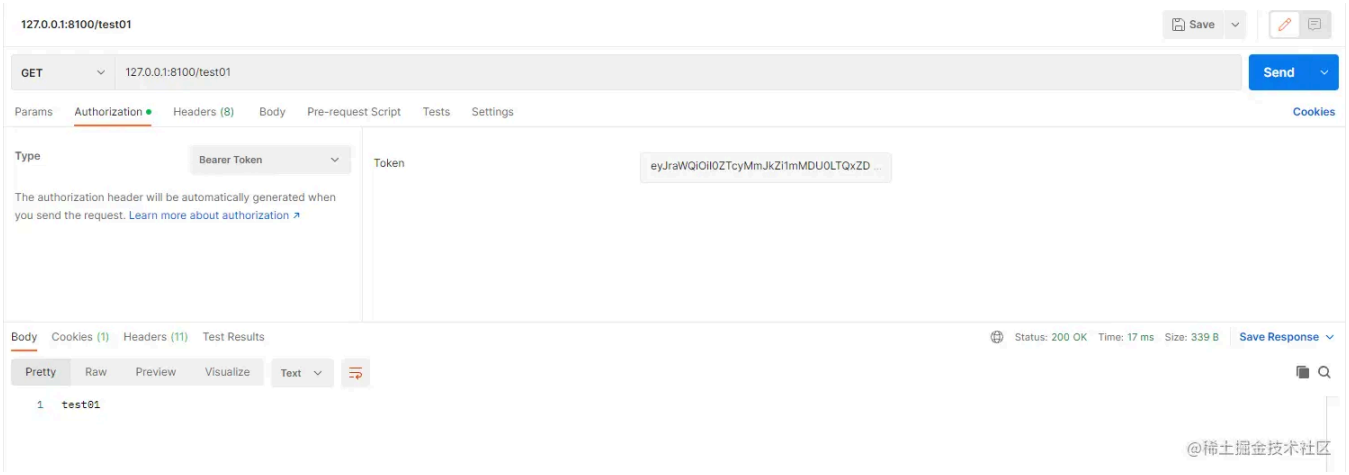
测试

申请一个token访问接口test02接口



权限不足

访问test接口



这样基本就已经完成了，如果有啥问题请在评论区留言。

代码已提交至Gitee：gitee.com/vains-Sofia...

标签： Spring Spring Boot 话题： 我的技术写作成长之路

本文收录于以下专栏

1 / 2

上一篇 Spring Authorization Server... 下一篇 Spring Authorization Server...

评论 8

平等表达，友善交流

0 / 1000 发送

最热 最新

当我遇上你 LV.4 Java

大佬，请问oauth2.1生产中，操作权限可能很多，放jwt-token中可能很大，一般如何做资源权限校验?

4月前 点赞 1

叹雪飞花 作者：使用匿名token，或者生成jwt时将权限放入redis，解析时从redis取权限

4月前 点赞 回复

Yooooum

大佬能讲讲怎么自定义认证模式，比如短信登录。我模仿pig项目中写了但是跑不了

9月前 点赞 5

叹雪飞花 作者：我需要研究下，这几天比较忙，等忙完我再来写一下

9月前 点赞 回复

叹雪飞花 作者：我实现了一种短信登录的方式：juejin.cn，你看下是不是你需要的那种，还有另一种是自定义grant_type的方式，见附图，不过附图的这种方式的文章还需要在琢磨一下怎么写，代码还需要优化优化

9月前 点赞 回复 ...

查看全部 5 条回复

目录

收起

相关推荐

Spring Authorization Server优化篇：自定义UserDetailsService实现从数据库获取用户信息
1.1k阅读 · 5点赞

Spring Authorization Server优化篇：持久化JWKSource，解决重启后无法解析AccessToken问题
1.0k阅读 · 5点赞

Spring Authorization Server入门 (十四) 联合身份认证添加微信登录

207阅读 · 3点赞

Spring Authorization Server入门 (五) 自定义异常响应配置
1.8k阅读 · 4点赞

精选内容

从零开始写 Docker(八)---实现 mydocker run -d 支持后台运行容器
探索云原生 · 66阅读 · 0点赞

Nginx实现一个端口开启https和http并全站https
Hamm · 204阅读 · 2点赞

为什么操作系统需要虚拟内存
写bug写bug · 84阅读 · 1点赞

Go 团队近两年在做什么，AI 方面如何发力?
煎鱼eddycjy · 482阅读 · 6点赞

【可套用】15个应用场景的算法实现
威哥爱编程 · 69阅读 · 1点赞

为你推荐

Spring Authorization Server入门 (二) Spring Boot整合Spring Authorization Server
叹雪飞花 9月前 6.8k 31 82 Java

Spring Authorization Server入门 (十二) 实现授权码模式使用前后端分离的登录页面
叹雪飞花 8月前 4.7k 24 65 后端 Spring ... Spring

Spring Authorization Server入门 (十) 添加短信验证码方式登录
叹雪飞花 9月前 3.4k 20 9 Spring Spring ...

Spring Authorization Server入门 (八) Spring Boot引入Security OAuth2 Client对接认...
叹雪飞花 9月前 2.8k 13 43 Spring ... Spring

Spring Authorization Server入门 (十六) Spring Cloud Gateway对接认证服务
叹雪飞花 6月前 2.5k 17 44 Spring ... Spring ... 安全

Spring Authorization Server入门 (十三) 实现联合身份认证，集成Github与Gitee的OAu...

Spring Authorization Server入门 (十一) 自定义grant_type(短信认证登录)获取token

叹雪飞花 9月前 2.5k 15 39 Spring Spring ... 安全

Spring Authorization Server入门 (七) 登录添加图形验证码

叹雪飞花 9月前 2.9k 18 4 Spring ...

SpringBoot3.x最简集成SpringDoc-OpenApi

叹雪飞花 4月前 2.3k 16 评论 后端 Spring ... Java

Spring Authorization Server优化篇：添加Redis缓存支持和统一响应类

叹雪飞花 8月前 1.7k 6 12 Spring Spring ... 安全

Spring Authorization Server入门 (十五) 分离授权确认与设备码校验页面

叹雪飞花 7月前 1.6k 14 8 Spring ... Spring Vue.js

Spring Authorization Server入门 (十九) 基于Redis的Token、客户端信息和授权确认信...

叹雪飞花 4月前 1.2k 8 19 Spring ... 后端 Redis

Spring Authorization Server入门 (二十) 实现二维码扫码登录

叹雪飞花 1月前 907 16 6 Spring ... Spring Java

Spring Authorization Server入门 (十七) Vue项目使用授权码模式对接认证服务

叹雪飞花 6月前 756 9 12 Vue.js 安全 Spring ...

Spring Cloud Gateway集成SpringDoc，集中管理微服务API

叹雪飞花 3月前 747 7 评论 Spring ... Spring ... Java