

## 第八章 SpringCloud OAuth2认证中心-基于JWT认证

本章完整源码地址: <https://github.com/kwang2003/springcloud-study-ch08.git>

### 1.项目概要

这一章节的内容以第七章的代码为基础改造而成<https://github.com/kwang2003/springcloud-study-ch07.git>。

传统的web应用中,我们通常通过cookie+session机制来保证调用的安全,在没有认证的情况下自动重定向到登录页面或者调用失败页面,而现在整个架构编程微服务模式了,cookie和session机制已经不能很好的满足保护API的需求了,更多的情况下采用token的验证机制,JWT的本质也是一种token。

JWT: JSON Web Token,是JSON风格的轻量级授权和认证规范,可以实现无状态,分布式的web应用授权。JWT的内容由三部分组成,分别是Header,Payload,Signature,三个部分之间通过.分割,举例

xxxxx.yyyyyy.zzzzz

#### Header

头部Header一般由2个部分组成alg和typ,alg是加密算法,如HMAC或SHA256,typ是token类型,取值为jwt,一个Header的例子

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

然后对Header部分进行Base64编码,得到第一部分的值

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ0b290IjoiInR5cCI6IkpXVCJ9.eyJ0b290IjoiInR5cCI6IkpXVCJ9

#### Payload

内容部分Payload是JWT存储信息的主体,包含三类内容

- 标准中注册的声明
- 公共的声明
- 私有的声明

标准中注册的声明

- iss:jwt签发者
- sub:jwt所面向的用户
- aud:接收jwt的一方
- exp:jwt的过期时间
- nbf:定义在什么时间之前该jwt是不可用的
- iat:jwt的签发时间
- jti:jwt的唯一标识,主要用作一次性token,避免重放攻击

公共的声明:

可以存放任何信息,根据业务实际需要添加,如用户id,名称等,但不要存放敏感信息

私有的声明:

私有声明是提供者和消费者所共同定义的声明,不建议存放敏感信息

举例:定义一个payload:

{ "sub": "1234567890", "name": "John Doe", "admin": true } 对其进行Base64编码,得到第二部分

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ0b290IjoiInR5cCI6IkpXVCJ9.eyJ0b290IjoiInR5cCI6IkpXVCJ9

{SIGNATURE}

#### Signature

token的签名部分由三部分组成256签名

```
var encodedString = base64UrlEncode(header) + '.' + base64UrlEncode(payload);
```

```
var signature = HMACSHA256(encodedString, 'secret');
```

得到最终的token串

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ0b290IjoiInR5cCI6IkpXVCJ9.eyJ0b290IjoiInR5cCI6IkpXVCJ9.TJVA95O  
rM7E2cBab30RMHrHDcEfxjoYZgeFONFh7HgQ

### 2.Oauth2-Server中生成JWT Token

a)通过keytool生成证书

```
keytool -genkeypair -alias kevin_key -keyalg RSA -keypass 123456 -keystore kevin_key.jks -storepass 123456
```

```
C:\Users\Administrator>keytool -genkeypair -alias kevin_key -keyalg RSA -keypass 123456 -keystore kevin_key.jks -storepass 123456
您的名字与姓氏是什么?
[Unknown]: 王旭政
您的组织单位名称是什么?
[Unknown]: 海尔
您的组织名称是什么?
[Unknown]: 海尔
您所在的城市或区域名称是什么?
[Unknown]: 青岛
您所在的省/市/自治区名称是什么?
[Unknown]: 山东省青岛市
该单位的双字母国家/地区代码是什么?
[Unknown]: CN
CN=" 王旭政", OU=海尔, O=海尔, L=青岛, ST=山东省青岛市, C=CN是否正确?
[否]: 是
```

C:\Users\Administrator>

查看证书信息:

keytool -list -v -keystore kevin\_key.jks -storepass 123456

```
C:\Users\Administrator>keytool -list -v -keystore kevin_key.jks -storepass 123456

密钥库类型: JKS
密钥库提供方: SUN

您的密钥库包含 1 个条目

别名: kevin_key
创建日期: 2017-11-8
条目类型: PrivateKeyEntry
证书链长度: 1
证书 [1]:
所有者: CN=" 王旭政", OU=海尔, O=海尔, L=青岛, ST=山东省青岛市, C=CN
发布者: CN=" 王旭政", OU=海尔, O=海尔, L=青岛, ST=山东省青岛市, C=CN
序列号: e9752b8
有效期开始日期: Wed Nov 08 16:45:33 CST 2017, 截止日期: Tue Feb 06 16:45:33 CST 2018
证书指纹:
    MD5: BC:76:91:BC:37:E3:F1:4D:25:FA:91:68:05:B9:97:C1
    SHA1: B4:4A:78:BD:03:47:23:F8:AD:2A:DA:AF:56:48:09:9D:40:2B:2C:84
    SHA256: 50:EB:4A:77:50:B2:37:92:BA:2C:48:26:DA:D7:C3:F0:DF:82:00:56:8B:45:10:95:26:F7:0A:81:7B:99:AE:52
    签名算法名称: SHA256withRSA
    版本: 3

扩展:

#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: D6 7E A2 29 8A F1 99 F2    65 02 9A B1 66 B0 CA 14    ...>....e...f...
0010: 70 C6 C8 74                                p..t
]
]

*****
*****

C:\Users\Administrator>
```

查看公钥信息

keytool -list -rfc -keystore kevin\_key.jks -storepass 123456

```
C:\Users\Administrator>keytool -list -rfc -keystore kevin_key.jks -storepass 123456

密钥库类型: JKS
密钥库提供方: SUN

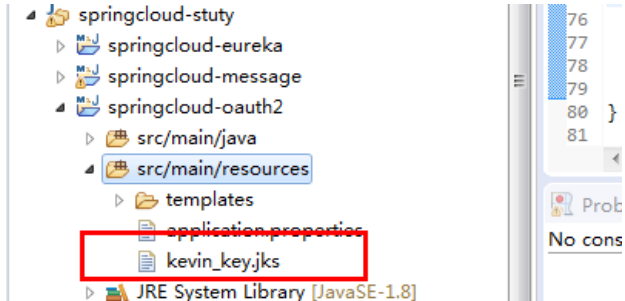
您的密钥库包含 1 个条目

别名: kevin_key
创建日期: 2017-11-8
条目类型: PrivateKeyEntry
证书链长度: 1
证书[1]:
-----BEGIN CERTIFICATE-----
MIIDhTCCAm2gAwIBAgIEJDsUDANBgkqhkiG9w0BAQsFAADBzMQswCQYDVQQGEwJD
TjEhMBkGA1UECAwS5bGx5Lic55yB6Z2S5bKb5biCMQ8wDQYDVQQHDAhbnZLlsp
DzANBgNUBA0MBua1t+Ww1DEPMA0GA1UECwwG5rW35bCUMRQwEgYDVQQDDAsGIOeO
i+aXreaUvzAeFw0xNzExMDgwODQ1MzNaFw0xODAwMDYwODQ1MzNaMHMxCzAJBgNU
BAYTAKNOMRswGQYDVUQIDBLlsbHkuJznnIHpnZLlspvluIIXDzANBgNUBAcMBumd
kuWymzEPMA0GA1UECwwG5rW35bCUMQ8wDQYDVUQQLDAbmtbf1sJQxPDASBgNUBAMM
CyAg546L5pet5pS/MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEa0ZQf
HaHQEzGuWBESlrc0m9DMiUM/fHEjLG7Uk3COZ1xuM49P4pC+pNY3nm+wL/Wm8QQd
VNjPetzCiltLdW60rQa/+osW599SkmUSGF7rYXI9y4n1N4h0k9jLBdZ9n2/5miWW
PDwrbsIGgLKp/NIsh4T2Gj61UZuMj4cskIjU8P12S5TJEQ3N+PGYBY+G8zWzYB1d
r3LssqAT0qv1/XH+kPEesAtaaxJsfa/SWpaxDZnM5JGSjQ1/FEjvF2e0yAKbU/Nq
HnIqgnucr/StFbn/sGloEs1DAj8nIMtoiklqeJqJvJCSdOC75f2N3iK72DgAUZkS
NtfEdsUyghio0tBtWQIDAQABoyEwHzAdBgNUHQ4EFgQU1n6iKYrxmfJlApqxZrDK
FHDGyHQwDQYJKoZIhvcNAQELBQADggEBAABGL4GiFo3o5eoh7/F2hG9M8sIRo6oAS
AM28u1zZoJbNWQpb1j1F0d+mNjWAmOG9BKK4rqUyduyFNU1UPw1XfMvY8Zwv08sU
kUywoovUrgLHUOLQ2PW29JdLf5ypQmUpit4s+NxN31GLUXwBLfLe/1e54tJD6WpX
DMNPoSWhihjqEW2ua7Jya9Tds07CrFrPNKYCEn1CjgURGH3FxaFzaxZXRowAR5z
5Ko8/0Gmndt9LxsZGJ01SNPm4jtnguhY69IBK0mH/Kb5QW++F1VZTtL2Dd5YadoY
zqJXg10nnSyUUONB/x4W2cxAAATC0bzMEZ/mIrbMGbcZmLWhLC1sidvo=
-----END CERTIFICATE-----

*****
*****

C:\Users\Administrator>
```

b)将生成的 kevin\_key.jks 文件放到 oauth2-server 工程的 src/main/resources 目录下



c)添加 jwt 相关 jar 包依赖

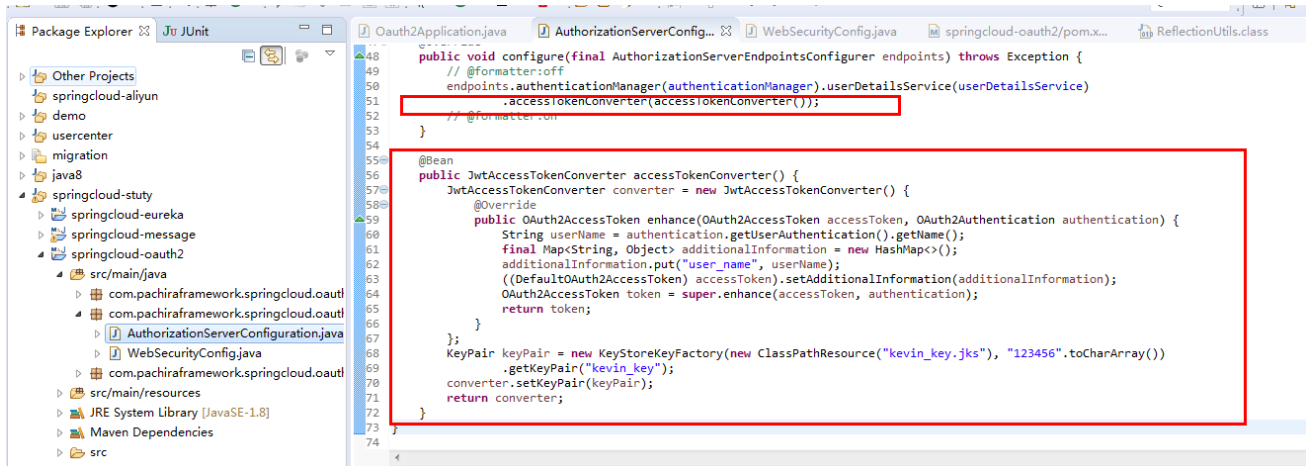
```
<dependency>
```

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

```
<artifactId>spring-security-jwt</artifactId>
```

```
</dependency>
```

c)在 OAuth2 服务器端配置核心类 AuthorizationServerConfiguration 中增加 jwt token 相关配置



### 3.测试Oauth2服务

[http://localhost:8888/oauth/authorize?response\\_type=code&client\\_id=client&redirect\\_uri=http://baidu.com&state=123](http://localhost:8888/oauth/authorize?response_type=code&client_id=client&redirect_uri=http://baidu.com&state=123)

出现登录页面,输入用户名: admin 密码: 123456

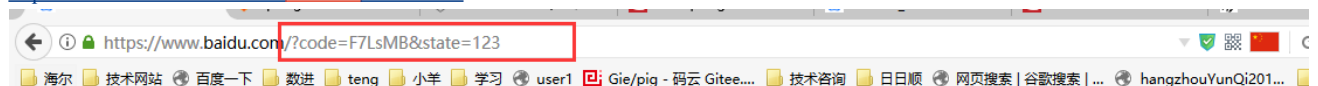


点击Submit按钮,进入用户授权确认页面



点击Approve,跳转到baidu页面,后面携带了code和state参数

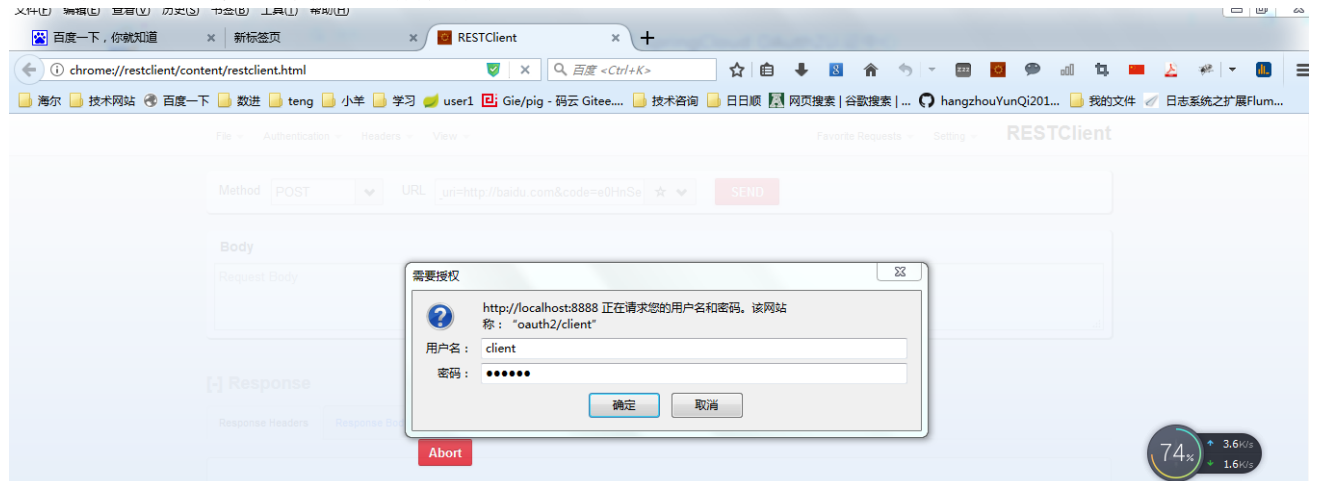
<https://www.baidu.com/?code=F7LsMB&state=123>



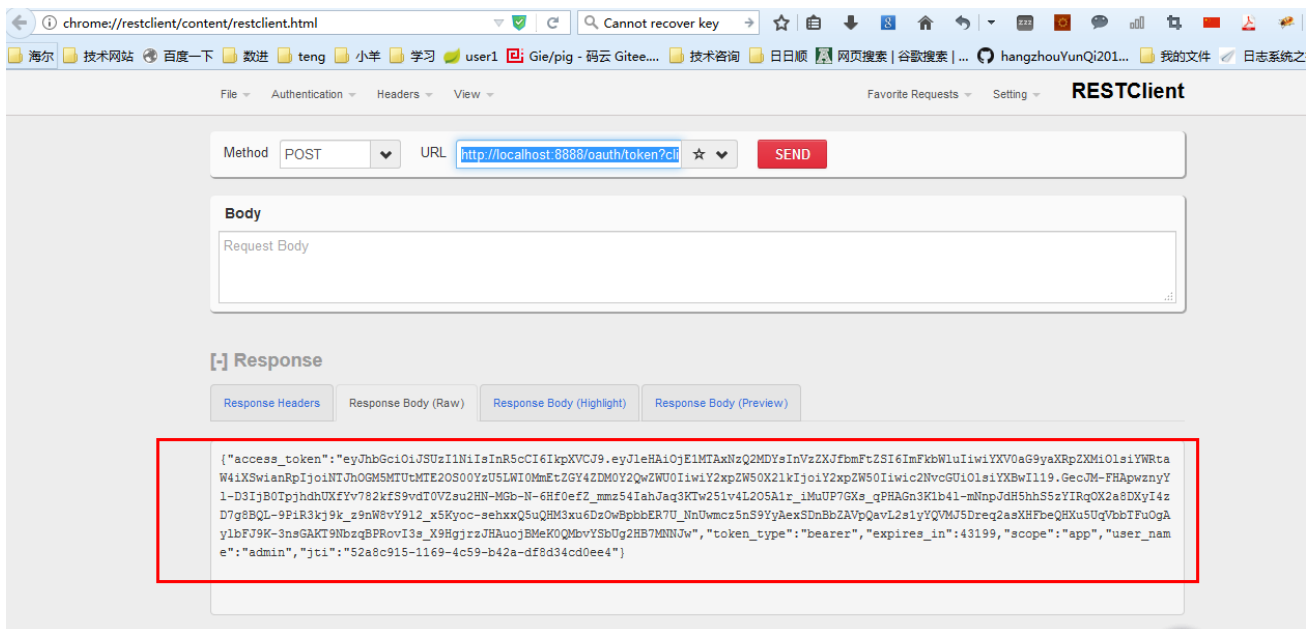
根据code换取access\_code, 注意使用post方法

[http://localhost:8888/oauth/token?client\\_id=client&grant\\_type=authorization\\_code&redirect\\_uri=http://baidu.com&code=F7LsMB](http://localhost:8888/oauth/token?client_id=client&grant_type=authorization_code&redirect_uri=http://baidu.com&code=F7LsMB)

注意这个code要和上个步骤中获得的code保持一致

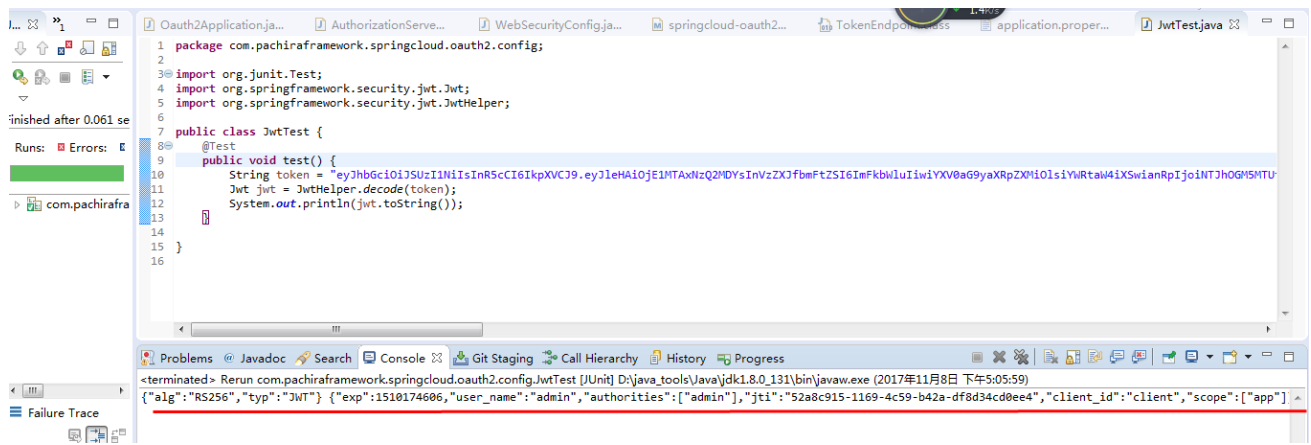


用户名输入client, 密码是secret,点击确定,可以看到access\_token已经是jwt格式的字符了



## 4.access\_token信息解析

```
import org.junit.Test;
import org.springframework.security.jwt.Jwt;
import org.springframework.security.jwt.JwtHelper;
```



通过这个测试我们可以看出来，`token`中已经包含了当前用户的信息了，包括我们在`accessTokenConvertor()`方法中给`token`中添加的额外信息`user_name`