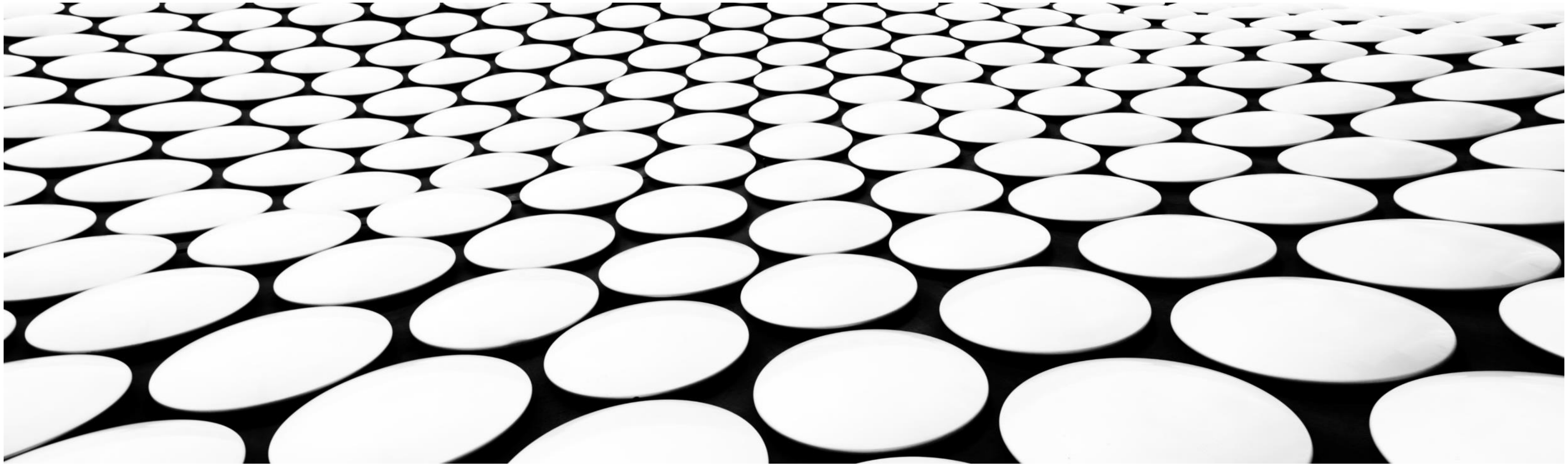


服务端开发-OAuth2

陶召胜

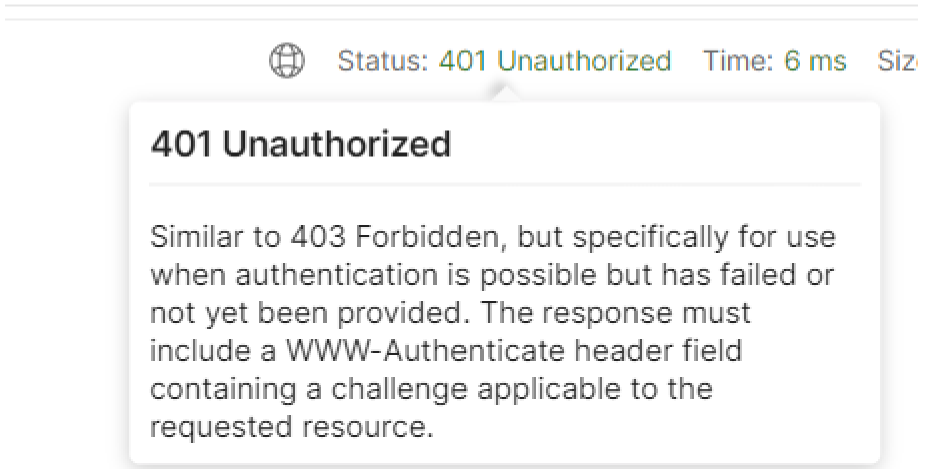
内容

主Taco Cloud增加REST API

- 控制器实现：IngredientController

增加权限控制

- POST：http://tacocloud:8080/api/ingredients
- DELETE ：http://tacocloud:8080 /api/ingredients/*
- 两种方式
  - ✓ .antMatchers(HttpMethod.POST, "/api/ingredients").hasAuthority("ROLE_USER")
  - ✓ @PreAuthorize("hasAuthority('ROLE_USER')")



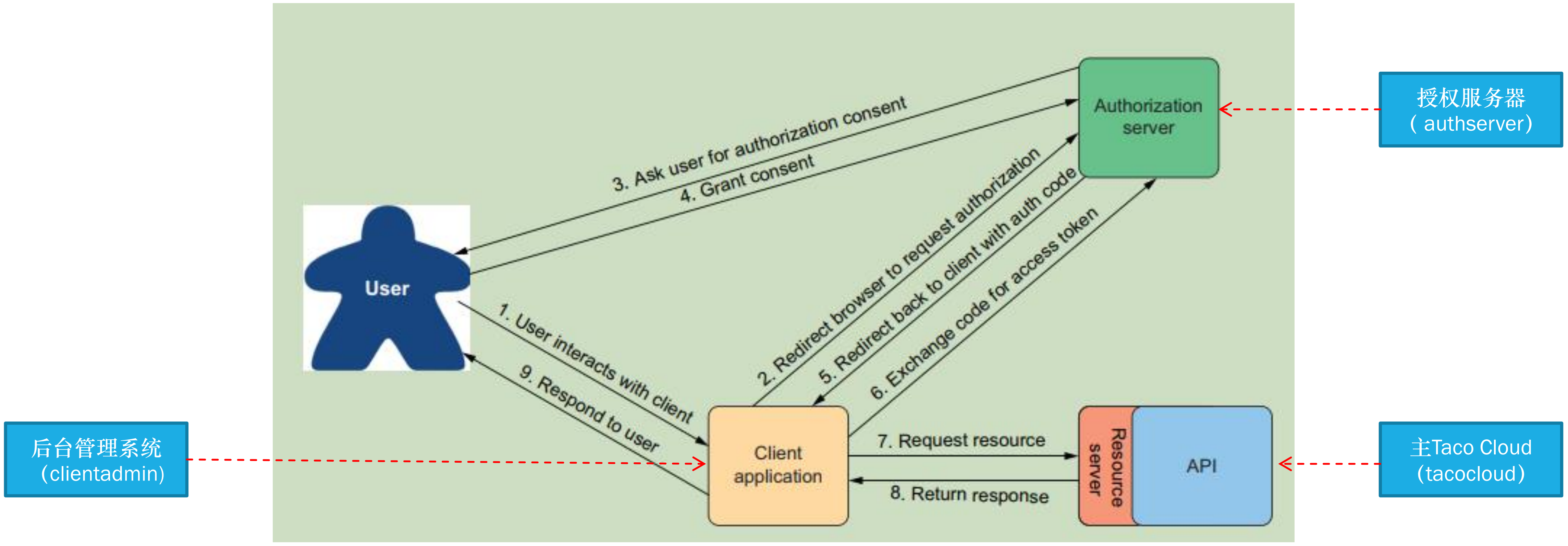
Status: 401 Unauthorized Time: 6 ms Siz

401 Unauthorized

Similar to 403 Forbidden, but specifically for use when authentication is possible but has failed or not yet been provided. The response must include a WWW-Authenticate header field containing a challenge applicable to the requested resource.

OAuth 2是一个安全规范

- 授权码授权（authorization code grant）模式



其他模式

- 隐式授权（implicit grant），直接返回访问令牌，而不是授仅码
- 用户凭证（或密码）授权（user credentials (password) grant）：用户凭证直接换取访问令牌，不经过浏览器登录
- 客户端凭证授权（client credentials grant）：客户端交换自己的凭证以获取访问令牌

配置本地域名

- Windows：C:\Windows\System32\drivers\etc\hosts
- Linux：/etc/hosts

```
127.0.0.1 tacocloud
127.0.0.1 authserver
127.0.0.1 clientadmin
```

JWT

- JSON web token
- "access_token": "eyJraWQiOiI3NjU5OGI1ZS03NDExLTQ3OWEtYTJmZi0wYjZmZmRhMGU3NmEiLCJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdWIiOiJoYWJ1bWEiLCJhdWQiOiJ0YWNvLWFkbWluLWNsaWVudCIsIm5iZiI6MTY5OTA1OTk4NCwic2NvcGUiOlsiZGVsZXRlSW5ncmVkaWVudHMiLCJ3cml0ZUluZ3JlZGllbnRzIl0sImlzcyI6Imh0dHA6XC9cL2F1dGhzZXJ2ZXI6OTAwMCIsImV4cCI6MTY5OTA2MDI4NCwiaWF0IjoxNjk5MDU5OTg0LCJqdGkiOiIzY2NhNjQxZS1hOGFjLTQzMzUtODBmNy0wZjc0ZjlhOTA1ZDIifQ.NwFJ77QfxDb3PQNPPOYIjBoSLCQzKR_1YaWNQlAL_e-zmjhGQeQF9q6hmnB1Rrk8qfbCAh4qkF56_mtsoN8nJlApOWqnpNSaae5j7qM0m3J6o8ml07R_saK-Mgb5aCSRa3kuti5z38GyE4LRhTlxxixq24hDR8mgGIn04f8DGXihjBUgJrjarcZC8W2DNyFHtIvypwdRwYLKWiZjv-3lA7H-YKiNrfQmNeEeUi0FfaNlYumKr2SQw2Ro6wIasnhTp5W32ZKH0dGKf0MQ2kE2G3hDg4PvZkmxXejL6PfuLLGnb0O74ejLzNRhG1ZI6WkFaOBLHyDDiyefhIcfcjDB3Q"

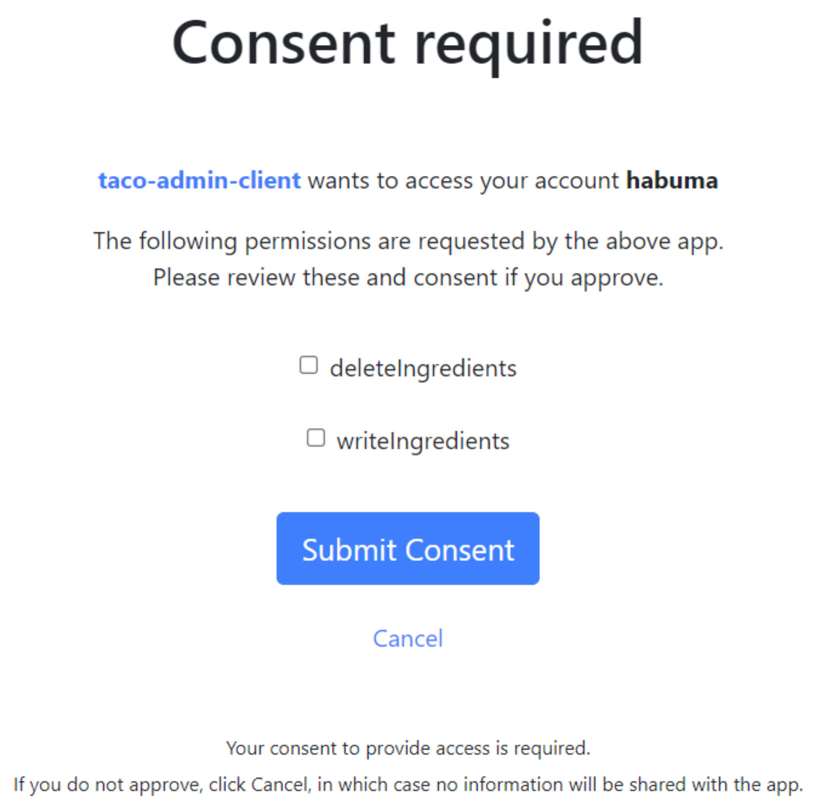
创建授权服务器

- Spring Authorization Server
- OAuth2AuthorizationServerConfiguration默认配置
- 客户端存储库，RegisteredClientRepository（接口），RegisteredClient，InMemoryRegisteredClientRepository
- OAuth 2 scope：writeIngredients、deleteIngredients

重定向的例子

- http://clientadmin:9090/login/oauth2/code/taco-admin-client?code=MGrX0ZgOoCpICZuGnhRGvsxBp3tV2DFlj9ScBIg1UGcla81haYuUjrpfGuZfsV3Od5a_gk7T6TZWspjCdEAkins4gE-fvWL_bhYmZO3D7KJ5NXHqNHNG1vMreI1p5-gz

clientSettings.requireUserConsent(true)



Consent required

taco-admin-client wants to access your account **habuma**

The following permissions are requested by the above app.
Please review these and consent if you approve.

☐ deleteIngredients

☐ writeIngredients

Submit Consent

Cancel

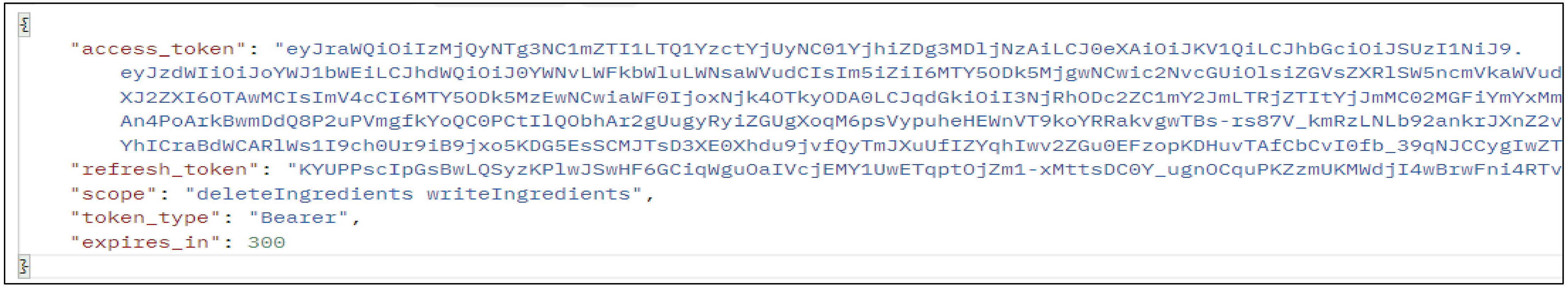
Your consent to provide access is required.
If you do not approve, click Cancel, in which case no information will be shared with the app.

生成JWK

- JSON web key，RSA密钥对（公钥、私钥），用于对令牌签名，令牌会用私钥签名，资源服务器会通过从授权服务器获取到的公钥验证请求中收到的令牌是否有效
- interface JWKSource
- interface JwtDecoder

运行授权服务器

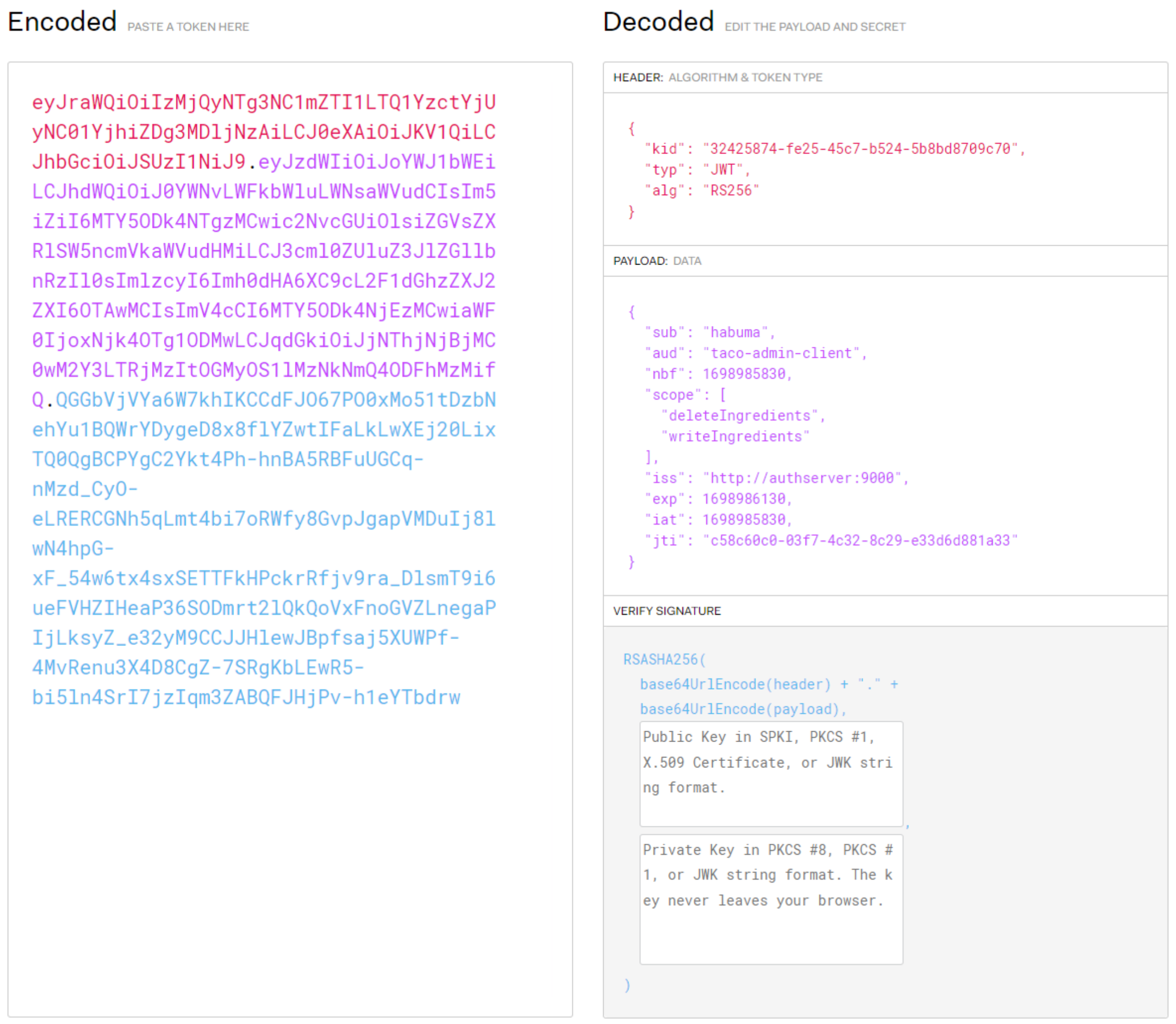
- 获得授权码（浏览器访问）：http://authserver:9000/oauth2/authorize?response_type=code&client_id=taco-admin-client&scope=writeIngredients+deleteIngredients&redirect_uri=http://clientadmin:9090/login/oauth2/code/taco-admin-client
- 获得token（postman访问） POST： http://authserver:9000/oauth2/token
- 刷新token（postman访问）： http://authserver:9000/oauth2/token



```
{
    "access_token": "eyJraWQiOiIzMjQyNTg3NC1mZTI1LTQ1YzctYjUyNC01YjhiZDg3MDljNzAiLCJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.
        eyJzdWIiOiJoYWJ1bWEiLCJhdWQiOiJ0YWNvLWFkbWluLWNsaWVudCIsIm5iZiI6MTY5ODk5MjgwNCwic2NvcGUiOlsiZGVsZXRlSW5ncmVkaWVud
        XJ2ZXI6OTAwMCIsImV4cCI6MTY5ODk5MzEwNCwiaWF0IjoxNjk4OTkyODA0LCJqdGkiOiI3NjRhODc2ZC1mY2JmLTRjZTItYjJmMC02MGFiYmYxMm
        An4PoArkBwmDdQ8P2uPVmgfkYoQC0PCtIlQObhAr2gUugyRyiZGUgXoqM6psVypuheHEWnVT9koYRRakvgwTBs-rs87V_kmRzLNLb92ankrJXnZ2v
        YhICraBdWCARlWs1I9ch0Ur9iB9jxo5KDG5EsSCMJTsD3XE0Xhdu9jvfQyTmJXuUfIZYqhIwv2ZGu0EFzopKDHuvTAfCbCvI0fb_39qNJCCygIwZT
    "refresh_token": "KYUPPscIpGsBwLQSyzKPlwJSwHF6GCiqWguOaIVcjEMY1UwETqptOjZm1-xMttsDC0Y_ugnOCquPKZzmUKMWdjI4wBrwFni4RTv
    "scope": "deleteIngredients writeIngredients",
    "token_type": "Bearer",
    "expires_in": 300
}
```

jwt解码

- https://jwt.io/



Encoded PASTE A TOKEN HERE

```
eyJraWQiOiIzMjQyNTg3NC1mZTI1LTQ1YzctYjU
yNC01YjhiZDg3MDljNzAiLCJ0eXAiOiJKV1QiLC
JhbGciOiJSUzI1NiJ9.eyJzdWIiOiJoYWJ1bWEi
LCJhdWQiOiJ0YWNvLWFkbWluLWNsaWVudCIsIm5
iZiI6MTY5ODk4NTgzMCwic2NvcGUiOlsiZGVsZX
RlSW5ncmVkaWVudHMiLCJ3cml0ZUluZ3JlZGllb
nRzIl0sImlzcyI6Imh0dHA6XC9cL2F1dGhzZXJ2
ZXI6OTAwMCIsImV4cCI6MTY5ODk4NjEzMCwiaWF
0IjoxNjk4OTg1ODMwLCJqdGkiOiJjNThjNjBjMC
0wM2Y3LTRjMzItOGMyOS1lMzNkNmQ4ODFhMzMif
Q.QGGbVjVYa6W7khIKCCdFJO67PO0xMo51tDzbN
ehYu1BQWrYDygeD8x8flYZwtIFaLkLwXEj20Lix
TQ0QgBCPYgC2Ykt4Ph-hnBA5RBFuUGCq-
nMzd_CyO-
eLRERCGNh5qLmt4bi7oRWfy8GvpJgapVMDuIj8l
wN4hpG-
xF_54w6tx4sxSETTFkHPckrRfjv9ra_DlsmT9i6
ueFVHZIHeaP36SODmrt2lQkQoVxFnoGVZLnegaP
IjLksyZ_e32yM9CCJJHlewJBpfsaj5XUWPf-
4MvRenu3X4D8CgZ-7SRgKbLEwR5-
bi5ln4SrI7jzIqm3ZABQFJHjPv-h1eYTbdrw
```

Decoded EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "kid": "32425874-fe25-45c7-b524-5b8bd8709c70",
  "typ": "JWT",
  "alg": "RS256"
}
```

PAYLOAD: DATA

```
{
  "sub": "habuma",
  "aud": "taco-admin-client",
  "nbf": 1698985830,
  "scope": [
    "deleteIngredients",
    "writeIngredients"
  ],
  "iss": "http://authserver:9000",
  "exp": 1698986130,
  "iat": 1698985830,
  "jti": "c58c60c0-03f7-4c32-8c29-e33d6d881a33"
}
```

VERIFY SIGNATURE

```
RSASHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  Public Key in SPKI, PKCS #1,
  X.509 Certificate, or JWK stri
  ng format.
  ,
  Private Key in PKCS #8, PKCS #
  1, or JWK string format. The k
  ey never leaves your browser.
)
```

内容

1. 创建授权服务器
2. 创建资源服务器
3. 开发客户端应用

创建资源服务器

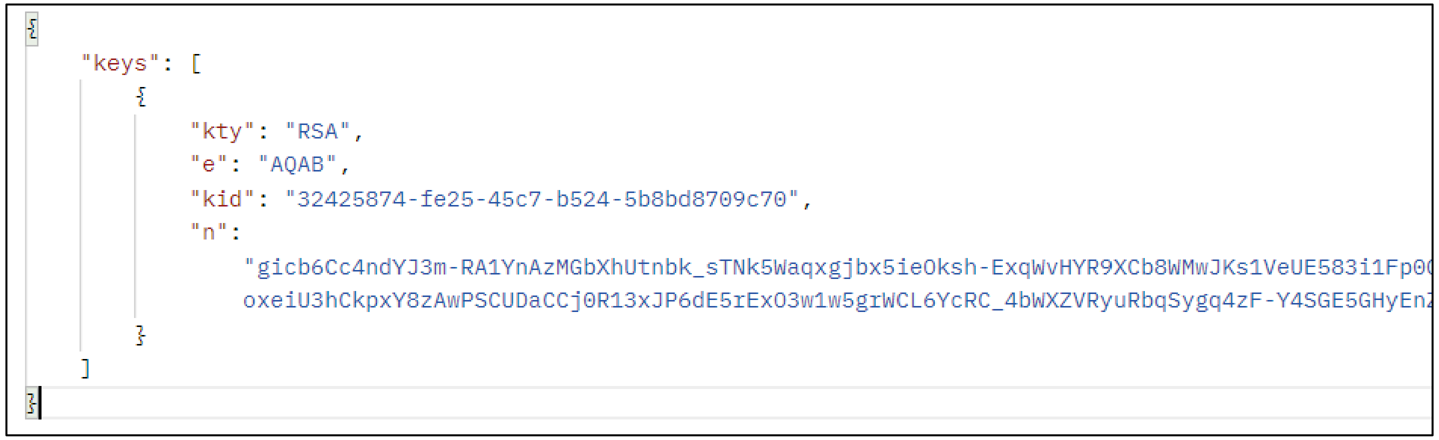
```
<dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-oauth2-resource-server</artifactId>
   </dependency>

.antMatchers(HttpMethod.POST, "/api/ingredients").hasAuthority("SCOPE_writeIngredients")
.antMatchers(HttpMethod.DELETE, "/api/ingredients/*").hasAuthority("SCOPE_deleteIngredients")
```

- 开启调用API前的过滤器：

```
.and()
        .oauth2ResourceServer(oauth2 -> oauth2.jwt())
```

配置资源服务器从何处获取公钥

```yaml
spring:
  security:
    oauth2:
      resourceserver:
        jwt:
          jwk-set-uri: http://tacocloud:9000/oauth2/jwks
```



```json
{
    "keys": [
        {
            "kty": "RSA",
            "e": "AQAB",
            "kid": "32425874-fe25-45c7-b524-5b8bd8709c70",
            "n":
                "gicb6Cc4ndYJ3m-RA1YnAzMGbXhUtnbk_sTNk5Waqxgjbx5ieOksh-ExqWvHYR9XCb8WMwJKs1VeUE583i1Fp0
                oxeiU3hCkpxY8zAwPSCUDaCCj0R13xJP6dE5rEx03w1w5grWCL6YcRC_4bWXZVRyuRbqSygq4zF-Y4SGE5GHyEn
        }
    ]
}
```

从POSTMAN访问被保护资源

- 带着token（Authorization属性）访问POST：http://tacocloud:8080/api/ingredients
- 带着token（Authorization属性）访问DELETE：http://tacocloud:8080/api/ingredients/*

内容

开发客户端应用

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

代码配置

```
@Bean
 SecurityFilterChain defaultSecurityFilterChain(HttpSecurity http) throws Exception {
   http
     .authorizeRequests(
         authorizeRequests -> authorizeRequests.anyRequest().authenticated()
     )
     .oauth2Login(
       oauth2Login ->
       oauth2Login.loginPage("/oauth2/authorization/taco-admin-client"))
     .oauth2Client(withDefaults());
   return http.build();
 }
```

属性配置

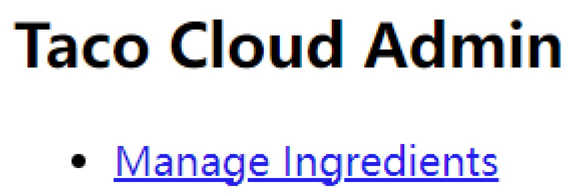
```
spring:
  security:
    oauth2:
      client:
        registration:
          taco-admin-client:
            provider: tacocloud
            client-id: taco-admin-client
            client-secret: secret
            authorization-grant-type: authorization_code
            redirect-uri: "http://clientadmin:9090/login/oauth2/code/{registrationId}"
            scope: writeIngredients,deleteIngredients,openid
        provider:
          tacocloud:
            issuer-uri: http://authserver:9000
```

实现请求拦截器，

- 请求头增加属性：Authorization

访问

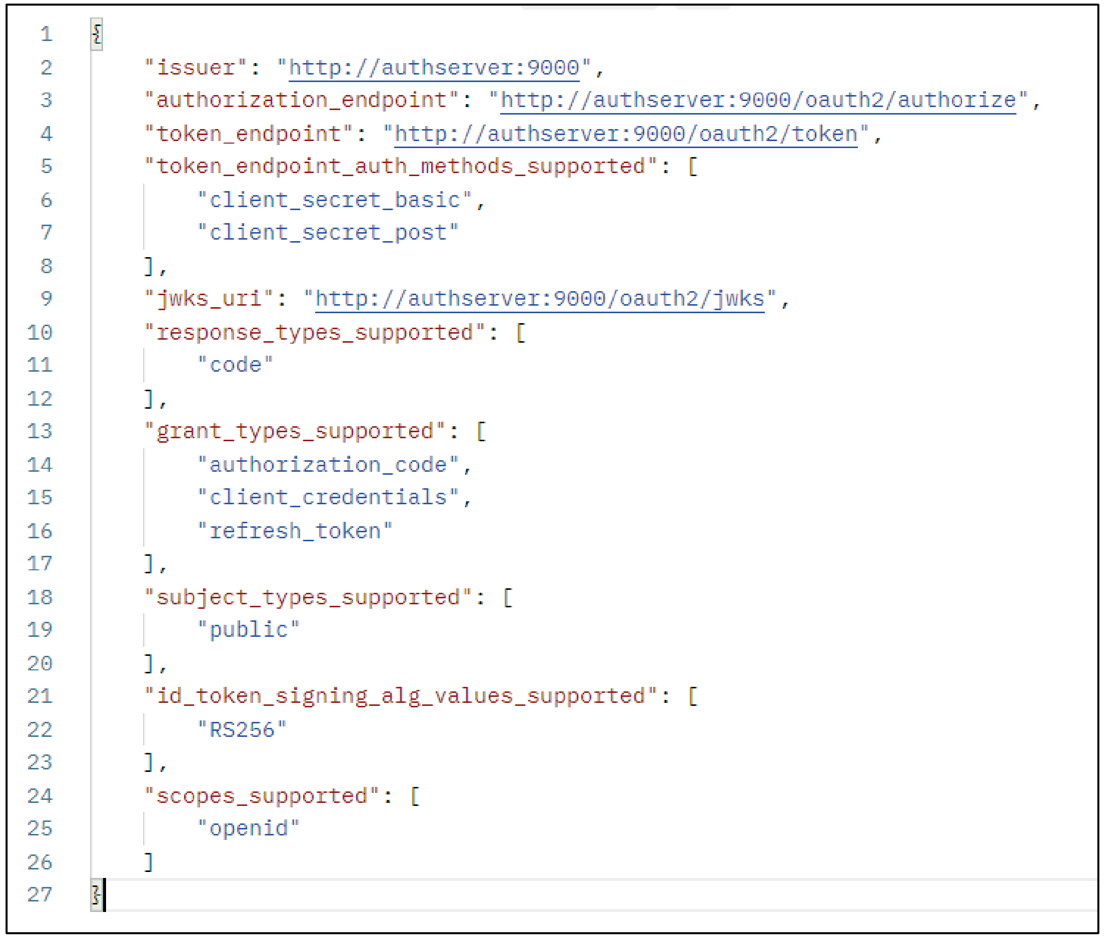
- http://clientadmin:9090



Taco Cloud Admin

- Manage Ingredients

http://authserver:9000/.well-known/openid-configuration



```json
{
    "issuer": "http://authserver:9000",
    "authorization_endpoint": "http://authserver:9000/oauth2/authorize",
    "token_endpoint": "http://authserver:9000/oauth2/token",
    "token_endpoint_auth_methods_supported": [
        "client_secret_basic",
        "client_secret_post"
    ],
    "jwks_uri": "http://authserver:9000/oauth2/jwks",
    "response_types_supported": [
        "code"
    ],
    "grant_types_supported": [
        "authorization_code",
        "client_credentials",
        "refresh_token"
    ],
    "subject_types_supported": [
        "public"
    ],
    "id_token_signing_alg_values_supported": [
        "RS256"
    ],
    "scopes_supported": [
        "openid"
    ]
}
```

谢谢观看！