★ (/bin/view/Main/) ▼ / 云平台部 (/bin/view/%E4%BA%91%E5%B9%B3%E5%8F%B0%E9%83%A8/) ▼ / web后台 (/bin/view/%E4%BA%91%E5%B9%B3%E5%8F%B0%E9%83%A8/web%E5%90%8E%E5%8F%B0/) ▼ / OAuth2和JWT快速入门 (/bin/view/%E4%BA%91%E5%B9%B3%E5%8F%B0%E9%83%A8/web%E5%90%8E%E5%8F%B0/OAuth2%E5%92%8CJWT%E5%BF%AB%E9%80%9F%E5%85%A5%E9%97%A8/) ▼

OAuth2和JWT快速入门

由 qkup (/bin/view/XWiki/qkup)于 2021/09/16 15:36:20 最后修改

传统Session认证方式

HTTP是无状态的,在用户登录传统认证过程中,服务器为每个用户存储一份登录信息,通过session id唯一标识,该session id在服务器响应前端请求时返回给前端,前端将 其保存在cookie中,后面每一次请求都会携带该session id,服务器通过前端请求携带的session id就可以识别这个请求来自哪个用户,这就是传统的session认证。

传统基于session登录认证方式有很多缺点: 通常session保存在服务器内存中,不能实现跨实例共享;而且用户量越多,session开销越大;session依赖于浏览器的cookie机制,而移动端采用token机制;客户端cookie不安全,不方便做签名认证。

OAuth2

• 什么是OAuth2?

OAuth是一个关于授权(authorization)的开放网络标准,允许用户让第三方应用以安全且标准的方式获取该用户在某一网站、应用上存储的私密资源(个人信息、照片、视频等),而无需将用户名和密码提供给第三方应用。简单说,OAuth是一种授权机制, 资源所有者,同意授权第三方应用进入系统获取数据,系统产生一个短期的进入令牌(token),用来代替密码,供第三方应用使用。 该标准已在全世界得到广泛应用,目前的版本是2.0。

**客户端/Client: **第三方应用小程序

**用户: **Resource Owner

**认证服务器: **Authorization Server

- (A) 用户打开客户端以后, 客户端要求用户授权。
- (B) 用户同意给客户端授权。
- (C) 客户端使用上一步获得的授权, 向认证服务器申请令牌。
- (D) 认证服务器对客户端进行认证以后,确认无误,同意发放令牌。
- (E) 客户端使用令牌,向资源服务器申请获取资源。
- (F) 资源服务器确认令牌无误,同意向客户端开放资源。
 - 客户端授权4种模式
- a.授权码模式:authorization-code(先申请一个授权码,然后再用该码获取令牌)
- b.简化模式:implicit(无需授权码,很不安全,令牌有效期非常短,仅会话期有效)
- c.密码模式:password (用户高度信任某应用,直接向应用提供用户名和密码)
- d.客户端模式:client credentials(客户端以自己名义,非以用户名义,向"服务提供商"进行认证)

不管哪一种授权方式,第三方申请令牌之前,都必须先到系统备案,说明自己的身份,然后会拿到两个身份识别码:客户端 ID(client ID)和客户端密钥(client secret)。 这是为了防止令牌被滥用,没有备案过的第三方应用,是不会拿到令牌的。

微信用户使用微信身份安全登录第三方应用或网站(安电U):

• 用户同意授权,获取code

(https://open.weixin.qq.com/connect/oauth2/authorize?appid=APPID&redirect_uri=REDIRECT_URI&response_type=code&scope=SCOPE&state=STATE#wechat_redirect (https://open.weixin.qq.com/connect/oauth2/authorize?appid=APPID&redirect_uri=REDIRECT_URI&response_type=code&scope=SCOPE&state=STATE#wechat_redirect))

• 通过code获取网页授权access_token

```
(https://api.weixin.qq.com/sns/oauth2/access_token?
```

 $appid=settings. WECHAT_OPEN_APP_ID\&secret=settings. WECHAT_OPEN_SECRET_KEY\&code=params['code']\&grant_type=authorization_code\\ (https://api.weixin.qq.com/sns/oauth2/access_token?appid=settings. WECHAT_OPEN_APP_ID\&secret=settings. WECHAT_OPEN_SECRET_KEY\&code=params[))$

```
返回案例:
```

{

```
"access_token":"ACCESS_TOKEN",
"expires_in":7200,
"refresh_token":"REFRESH_TOKEN",
"openid":"OPENID",
"scope":"SCOPE"
```

}

• 更新access_token

access_token拥有较短有效期,一般2小时,当access_token超时后,可以使用refresh_token进行刷新,refresh_token拥有较长的有效期(7天、30天、60天、90天),当 refresh_token失效的后,需要用户重新授权。

(https://api.weixin.qq.com/sns/oauth2/refresh_token?appid= (https://api.weixin.qq.com/sns/oauth2/refresh_token?

appid=APPID&grant_type=refresh_token&refresh_token=REFRESH_TOKEN) settings.WECHAT_OPEN_APP_ID (https://api.weixin.qq.com/sns/oauth2/access_token? appid=settings.WECHAT_OPEN_APP_ID&secret=settings.WECHAT_OPEN_SECRET_KEY&code=params[) &grant_type=refresh_token&refresh_token=REFRESH_TOKEN (https://api.weixin.qq.com/sns/oauth2/refresh_token?appid=APPID&grant_type=refresh_token&refresh_token=REFRESH_TOKEN))

• 拉取用户信息

如果授权作用域为snsapi_userinfo,开发者可以通过access_token和openid拉取用户信息

(https://api.weixin.qq.com/sns/userinfo?access_token=ACCESS_TOKEN&openid=OPENID&lang=zh_CN (https://api.weixin.qq.com/sns/userinfo?acce

JWT

- 概述: JWT(Json Web Token)是实现token技术的一种解决方案,是目前最流行的跨域身份验证解决方案,服务器不保存任何会话数据,即服务器变为无状态,使其更容易扩展,下面是基于token身份验证大致流程:
- JWT结构

JW对象是一个很长的字符串,由三个部分组成,字符之间通过":"分隔符分为三个子串: JWT头、有效荷载、签名

JWT头:

是一个描述JWT元数据的JSON对象:

alg属性表示签名使用的算法,默认为HMAC SHA256

typ属性表示令牌的类型,JWT令牌统一写为JWT

有效荷载:

是JWT的主体内容部分,也是一个JSON对象,包含七个默认字段供选择,还可以自定义私有字段

签名:

签名哈希是对上面两部分数据进行签名,通过指定的算法生成哈希,以确保数据不会被篡改:

HMACSHA256(base64UrlEncode(header) + "." + base64UrlEncode(payload), secret)

计算得到上面三部分内容后,用.连接起来就是一个完整的JWT TOKEN,秘钥是保存在服务器上的一个私有密钥

JWT用法

客户端将收到的JWT存储在Cookie或localStorage中。 此后,客户端与服务器交互中都会带JWT。 如果将它存储在Cookie中,可以自动发送,但是不会跨域,因此一般是将 它放入HTTP请求的Header Authorization字段中。Authorization: Bearer

当跨域时,也可以将JWT被放置于POST请求的数据主体中。

- JWT实现库-python和SDK(hydra-sdk 1.0.2)
- Pyjwt库的Python简单案例:

参考资料:

http://www.ruanyifeng.com/blog/2019/04/oauth-grant-types.html (http://www.ruanyifeng.com/blog/2019/04/oauth-grant-types.html)

http://www.ruanyifeng.com/blog/2014/05/oauth_2_0.html (http://www.ruanyifeng.com/blog/2014/05/oauth_2_0.html)

https://jwt.io/ (https://jwt.io/)

https://github.com/ory/hydra (https://github.com/ory/hydra)

https://www.cnpython.com/pypi/hydra-sdk (https://www.cnpython.com/pypi/hydra-sdk)

https://blog.csdn.net/u011277123/article/details/78918390 (https://blog.csdn.net/u011277123/article/details/78918390)

https://blog.csdn.net/qq_15766181/article/details/80707923 (https://blog.csdn.net/qq_15766181/article/details/80707923)

♥0

标签:

由 qkup (/bin/view/XWiki/qkup) 在 2021/09/16 15:32:23 创建

该文档没有评论

●评论