JWT 简介

时间 2016-05-03 10:00:39 SegmentFault (/sites/3uEjYv)

原文
https://segmentfault.com/a/1190000005047525 (https://segmentfault.com/a/1190000005047525?utm_source=tuicool&utm_medium=referral)

主题 XML（/topics/11000094）加密解密（/topics/11100078）

本文翻译自JWT官方网站对JWT是什么以及能做什么的简介。

JWT是一种用于双方之间传递安全信息的简洁的、URL安全的表述性声明规范。JWT作为一个开放的标准（ RFC 7519 (https://tools.ietf.org/html/rfc7519) ），定义了一种简洁的，自包含的方法用于通信双方之间以Json对象的形式安全的传递信息。因为数字签名的存在，这些信息是可信的，JWT可以使用HMAC算法或者是RSA的公私秘钥对进行签名。

- 简洁(Compact): 可以通过URL，POST参数或者在HTTP header发送，因为数据量小，传输速度也很快
- 自包含(Self-contained)：负载中包含了所有用户所需要的信息，避免了多次查询数据库

JWT的主要应用场景

- 身份认证

  在这种场景下，一旦用户完成了登陆，在接下来的每个请求中包含JWT，可以用来验证用户身份以及对路由，服务和资源的访问权限进行验证。由于它的开销非常小，可以轻松的在不同域名的系统中传递，所有目前在单点登录（SSO）中比较广泛的使用了该技术。

- 信息交换

  在通信的双方之间使用JWT对数据进行编码是一种非常安全的方式，由于它的信息是经过签名的，可以确保发送者发送的信息是没有经过伪造的。

JWT的结构

JWT包含了使用 `.` 分隔的三部分：

- Header 头部
- Payload 负载
- Signature 签名

其结构看起来是这样的

```
xxxxx.yyyyy.zzzzz
```

Header

在header中通常包含了两部分：token类型和采用的加密算法。

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

接下来对这部分内容使用 **Base64Url** 编码组成了JWT结构的第一部分。

Payload

Token的第二部分是负载，它包含了claim， Claim是一些实体（通常指的用户）的状态和额外的元数据，有三种类型的claim：*reserved* , *public* 和 *private* .

- Reserved claims: 这些claim是JWT预先定义的，在JWT中并不会强制使用它们，而是推荐使用，常用的有 `iss（签发者）` , `exp（过期时间戳）` , `sub（面向的用户）` , `aud（接收方）` , `iat（签发时间）` 。
- Public claims：根据需要定义自己的字段，注意应该避免冲突
- Private claims：这些是自定义的字段，可以用来在双方之间交换信息

负载使用的例子：

```
{
  "sub": "1234567890",
  "name": "John Doe",
  "admin": true
}
```

上述的负载需要经过 **Base64Url** 编码后作为JWT结构的第二部分。

Signature

创建签名需要使用编码后的header和payload以及一个秘钥，使用header中指定签名算法进行签名。例如如果希望使用HMAC SHA256算法，那么签名应该使用下列方式创建：

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

签名用于验证消息的发送者以及消息是没有经过篡改的。

完整的JWT

JWT格式的输出是以 . 分隔的三段Base64编码，与SAML等基于XML的标准相比，JWT在HTTP和HTML环境中更容易传递

推酷 (http://www.tuicool.com/)
文章 (http://www.tuicool.com/ah)　站点 (http://www.tuicool.com/sites/hot)
主题 (http://www.tuicool.com/topics)　活动 (http://huodong.tuicool.com/)
APP 荐 (http://www.tuicool.com/mobile)　周刊　更多

下列的JWT展示了一个完整的JWT格式，它拼接了之前的Header， Payload以及秘钥签名：

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4
gRG9lIiwiaXNTb2NpYWwiOnRydWV9.
4pcPyMD09olPSyXnrXCjTwXyr4BsezdI1AVTmud2fU4
```

如何使用JWT?

在身份鉴定的实现中，传统方法是在服务端存储一个session，给客户端返回一个cookie，而使用JWT之后，当用户使用它的认证信息登陆系统之后，会返回给用户一个JWT，用户只需要本地保存该token（通常使用local storage，也可以使用cookie）即可。

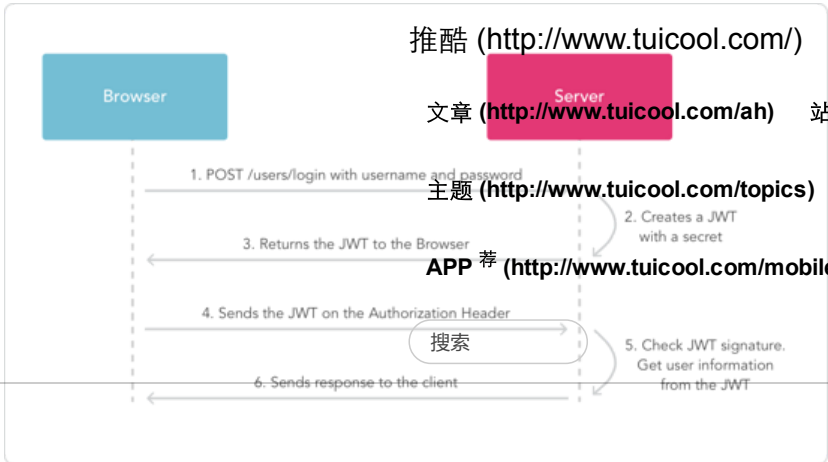
当用户希望访问一个受保护的路由或者资源的时候，通常应该在 `Authorization` 头部使用 `Bearer` 模式添加JWT，其内容看起来是下面这样：

```
Authorization: Bearer <token>
```

因为用户的状态在服务端的内存中是不存储的，所以这是一种 **无状态** 的认证机制。服务端的保护路由将会检查请求头 `Authorization` 中的JWT信息，如果合法，则允许用户的行为。由于JWT是自包含的，因此减少了需要查询数据库的需要。

JWT的这些特性使得我们可以完全依赖其无状态的特性提供数据API服务，甚至是创建一个下载流服务。因为JWT并不使用Cookie的，所以你可以使用任何域名提供你的API服务而不需要担心跨域资源共享问题（CORS）。

下面的序列图展示了该过程：



为什么要使用JWT?

相比XML格式，JSON更加简洁，编码之后更小，这使得JWT比SAML更加简洁，更加适合在HTML和HTTP环境中传递。

在安全性方面，SWT只能够使用HMAC算法和共享的对称秘钥进行签名，而JWT和SAML token则可以使用X.509认证的公私秘钥对进行签名。与简单的JSON相比，XML和XML数字签名会引入复杂的安全漏洞。

因为JSON可以直接映射为对象，在大多数编程语言中都提供了JSON解析器，而XML则没有这么自然的文档-对象映射关系，这就使得使用JWT比SAML更方便。

原文： Introduction to JSON Web Tokens (http://jwt.io/introduction/)

推荐文章

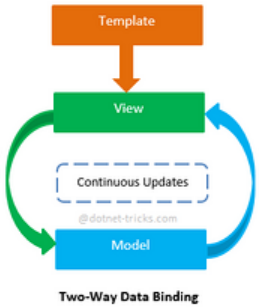
- 1. 就是要你懂 TCP | 最经典的TCP性能问题 (/articles/aEnqMbm)
- 2. HTTP Session 的工作原理以及几个思维扩展 (/articles/2Az2MzQ)
- 3. 为什么我使用 superagent 取替 request (/articles/36Fbaml)
- 4. Qzone 高性能 HTTPS 实践 (/articles/vuEbmey)
- 5. 浏览器缓存机制浅析--HTTP缓存 (/articles/bQ7j6rF)
- 6. 如何通过 WebP 兼容减少图片资源大小 (/articles/…)

相关推刊

- by 冰城 (/kans/603588574) 《默认推刊》 (/kans/603588574) *1*

- 

《匿名收藏》 4

- by irui 

View odel Chang updat (/kans/3378171438) 《默认推刊》 (/kans/3378171438) *4*

我来评几句

请输入评论内容...

登录后评论

已发表评论数(0)

推酷 (http://www.tuicool.com/)

文章 **(http://www.tuicool.com/ah)** 站点 **(http://www.tuicool.com/sites/hot)**

主题 **(http://www.tuicool.com/topics)** 活动 **(http://huodong.tuicool.com/)**

APP 荐 **(http://www.tuicool.com/mobile)** 周刊 更多