restful安全之JWT

传统的 cookie-session 机制可以保证的接口安全，在没有通过认证的情况下会跳转至登入界面或者调用失败。

在如今 RESTful 化的 API 接口下，cookie-session 已经不能很好发挥其余热保护好你的 API 。

更多的形式下采用的基于 Token 的验证机制，JWT 本质的也是一种 Token，但是其中又有些许不同。

## 什么是 JWT？

JWT 及时 JSON Web Token，它是基于 [RFC 7519](https://tools.ietf.org/html/rfc7519) 所定义的一种在各个系统中传递紧凑和自包含的 JSON 数据形式。

* 紧凑（Compact） ：由于传送的数据小，JWT可以通过GET、POST和放在 HTTP的header中，同时也是因为小也能传送的更快。
* 自包含（self-contained） : Payload 中能够包含用户的信息，避免数据库的查询。

JSON Web Token 由三部分组成使用 . 分割开：Header. Payload. Signature。

一个 JWT 形式上类似于下面的样子：xxxxx.yyyy.zzzz。

## Header

一般由两个部分组成：alg、typ。

alg是使用的hash算法，例如HMAC SHA256或RSA，typ是Token的类型自然就是 JWT。

|  |
| --- |
| {  "alg": "HS256",  "typ": "JWT"  } |

然后使用Base64Url编码成第一部分。例如：

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.<second part>.<third part>

## Payload

这一部分是 JWT 主要的信息存储部分，其中包含了许多种的声明（claims）。

Claims 的实体一般包含用户和一些元数据，这些 claims 分成三种类型：reserved, public, 和 private claims。

* （保留声明）reserved claims ：预定义的 [一些声明](http://www.iana.org/assignments/jwt/jwt.xhtml)，并不是强制的但是推荐，它们包括 iss (issuer), exp (expiration time), sub (subject),aud(audience) 等。这里都使用三个字母的原因是保证 JWT 的紧凑。
* （公有声明）public claims : 这个部分可以随便定义，但是要注意和 [IANA JSON Web Token](http://www.iana.org/assignments/jwt/jwt.xhtml) 冲突。
* （私有声明）private claims : 这个部分是共享被认定信息中自定义部分。

一个 Pyload 可以是这样子的：

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

这部分同样使用 Base64Url 编码成第二部分：

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiYWRtaW4iOnRydWV9.<third part>

## Signature

在创建该部分时候你应该已经有了 编码后的Header和Payload 还需要一个一个秘钥，这个加密的算法应该 Header 中指定。

一个使用 HMAC SHA256 的例子如下:

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

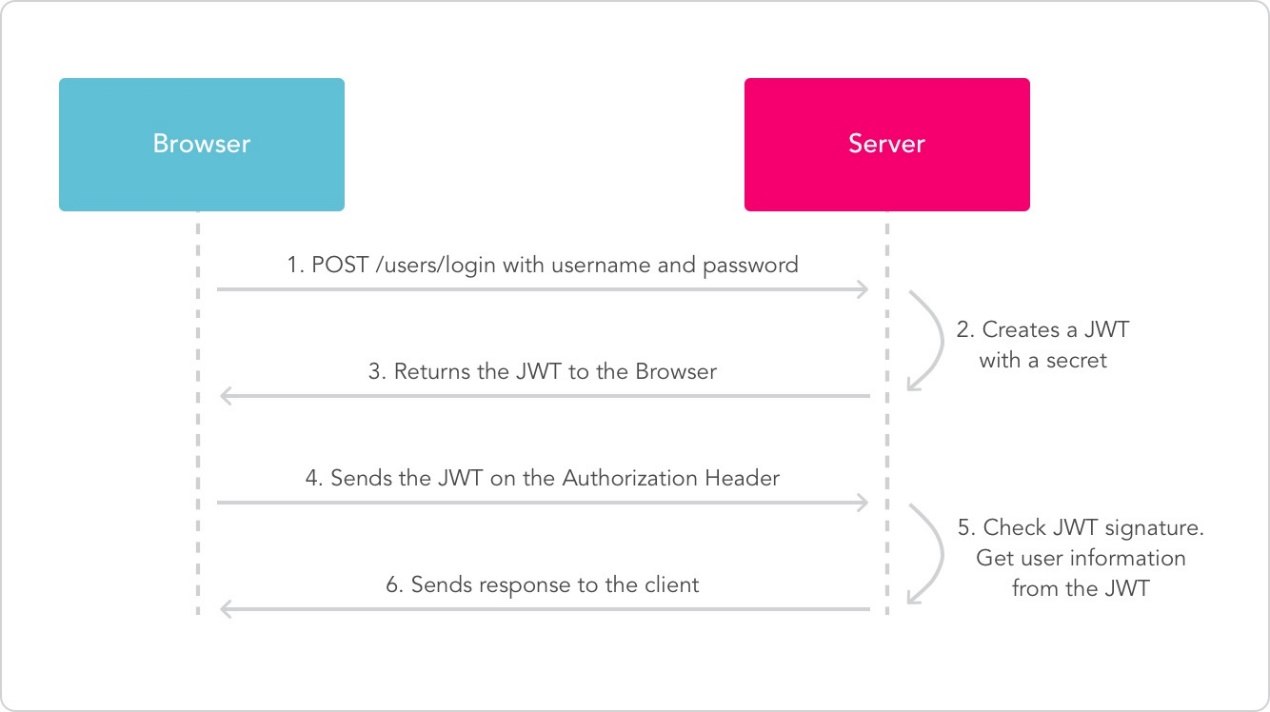
这个 signature 是用来验证发送者的 JWT 的同时也能确保在期间不被篡改。

所以，做后你的一个完整的 JWT 应该是如下形式：

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiYWRtaW4iOnRydWV9.TJVA95OrM7E2cBab30RMHrHDcEfxjoYZgeFONFh7HgQ

## 工作流程

在用户使用证书或者账号密码登入的时候一个JSON Web Token将会返回，同时可以把这个JWT存储在local storage、或者cookie中，用来替代传统的在服务器端创建一个 session返回一个cookie。



当用户想要使用受保护的路由时候，应该要在请求得时候带上 JWT ，一般的是在 header 的 Authorization 使用 Bearer 的形式，一个包含的 JWT 的请求头的 Authorization 如下：

Authorization: Bearer <token>

这是一中无状态的认证机制，用户的状态从来不会存在服务端，在访问受保护的路由时候回校验 HTTP header 中 Authorization 的 JWT，同时 JWT 是会带上一些必要的信息，不需要多次的查询数据库。

这种无状态的操作可以充分的使用数据的 APIs，甚至是在下游服务上使用，这些 APIs 和哪服务器没有关系，因此，由于没有 cookie 的存在，所以在不存在跨域（CORS, Cross-Origin Resource Sharing）的问题。

## 注意

* Playload不要存放敏感信息，因为该部分用base64编码，在客户端是可解的；
* 服务端保护好secre私钥，一旦客户端得带私匙就可以自己签发JWT了；
* 在网络层面上token明文传输的话会非常的危险，所以建议一定要使用HTTPS，并且把token放在post body里。

## 其他

restful安全策略还可以选择oauth2.0，只是oauth2.0属于第三方认证，不在本篇的讨论范畴之内，可以阅读<https://blog.csdn.net/seccloud/article/details/8192707>

## 附录

<https://blog.csdn.net/liuwenbiao1203/article/details/52351772>

<https://www.cnblogs.com/panxuejun/p/5866150.html>

<https://blog.csdn.net/wangcantian/article/details/74199762>

<https://blog.csdn.net/wangcantian/article/details/74330458>

https://blog.csdn.net/zhengholien/article/details/77777749