[spring security CSRF 问题 Invalid CSRF Token 'null' was found on ......](http://blog.csdn.net/u012373815/article/details/55047285)

标签： [spring security](http://www.csdn.net/tag/spring%20security)[csrf](http://www.csdn.net/tag/csrf)

2017-02-13 13:08 740人阅读 [评论](http://blog.csdn.net/u012373815/article/details/55047285#comments)(0) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/u012373815/article/details/55047285#report)

http://static.blog.csdn.net/images/category_icon.jpg 分类：

-->2.5 springboot（23） http://static.blog.csdn.net/images/arrow_triangle%20_down.jpg

版权声明：本文为博主原创文章，未经博主允许不得转载。

目录[(?)[-]](http://blog.csdn.net/u012373815/article/details/55047285)

1. [问题](http://blog.csdn.net/u012373815/article/details/55047285#t0)
   1. [比较关键的一点是](http://blog.csdn.net/u012373815/article/details/55047285#t1)
2. [解决](http://blog.csdn.net/u012373815/article/details/55047285#t2)

**1. 问题**

前面几篇博客 [**spring**](http://lib.csdn.net/base/javaee) security在集成spring boot的[**微服务**](http://lib.csdn.net/base/microservice)框架后，实现了cas认证和权限控制。但是在使用 postman 进行调用的时候出现这个问题

HTTP Status 403－Invalid CSRF Token 'null' was found on the request parameter '\_csrf' or header 'X-CSRF-TOKEN'.

* 1
* 1

然后我上网查找了这个CSRF 资料整理如下：

Spring Security 4.0之后，引入了CSRF，默认是开启。不得不说，CSRF和RESTful技术有冲突。CSRF默认支持的方法： GET|HEAD|TRACE|OPTIONS，不支持POST。

科普一下，什么是csrf,这是一个web应用安全的问题，CSRF（Cross-site request forgery跨站请求伪造，也被称为“One Click Attack” 或者Session Riding，攻击方通过伪造用户请求访问受信任站点。

我们知道，客户端与服务端在基于http协议在交互的数据的时候，由于http协议本身是无状态协议，后来引进了cookie的 方式进行记录服务端和客户端的之间交互的状态和标记。cookie里面一般会放置服务端生成的session id（会话ID）用来识别客户端访问服务端过 程中的客户端的身份标记。

在跨域 (科普一下：同一个ip、同一个网络协议、同一个端口，三者都满足就是同一个域，否则就有跨域问题) 的情况下， session id可能会被恶意第三方劫持，此时劫持这个session id的第三方会根据这个session id向服务器发起请求，此时服务器收到这个请求会 认为这是合法的请求，并返回根据请求完成相应的服务端更新。

比较关键的一点是，

如果这个http请求是get方式发起的请求，意味着它只是访问服务器 的资源，仅仅只是查询，没有更新服务器的资源，所以对于这类请求，spring security的防御策略是允许的，

如果这个请求是通过post请求发起的， 那么spring security是默认拦截这类请求的，因为这类请求是带有更新服务器资源的危险操作，如果恶意第三方可以通过劫持session id来更新 服务器资源，那会造成服务器数据被非法的篡改，所以这类请求是会被Spring security拦截的，在默认的情况下，spring security是启用csrf 拦截功能的，这会造成，在跨域的情况下，post方式提交的请求都会被拦截无法被处理（包括合理的post请求），前端发起的post请求后端无法正常 处理，虽然保证了跨域的安全性，但影响了正常的使用，如果关闭csrf防护功能，虽然可以正常处理post请求，但是无法防范通过劫持session id的非法的post请求，所以spring security为了正确的区别合法的post请求，采用了token的机制。

在跨域的场景下，客户端访问服务端会首先发起get请求，这个get请求在到达服务端的时候，服务端的Spring security会有一个过滤 器 CsrfFilter去检查这个请求，如果这个request请求的http header里面的X-CSRF-COOKIE的token值为空的时候，服务端就好自动生成一个 token值放进这个X-CSRF-COOKIE值里面，客户端在get请求的header里面获取到这个值，如果客户端有表单提交的post请求，则要求客户端要 携带这个token值给服务端，在post请求的header里面设置\_csrf属性的token值，提交的方式可以是ajax也可以是放在form里面设置hidden 属性的标签里面提交给服务端，服务端就会根据post请求里面携带的token值进行校验，如果跟服务端发送给合法客户端的token值是一样的，那么 这个post请求就可以受理和处理，如果不一样或者为空，就会被拦截。由于恶意第三方可以劫持session id，而很难获取token值，所以起到了 安全的防护作用。

**2. 解决**

原因找到了：spring Security 3默认关闭csrf，Spring Security 4默认启动了csrf。    
解决方案：   
如果不采用csrf，可禁用security的csrf

[**Java**](http://lib.csdn.net/base/javase)注解方式配置：

加上 .csrf().disable()即可。

修改前WebSecurityConfig.java

@Override

protected void configure(HttpSecurity http) throws Exception {

http.authorizeRequests()

.antMatchers("/", "/home").permitAll()

.and()

.formLogin()

.loginPage("/login").permitAll()

.and()

.logout().logoutUrl("/logout")

.logoutSuccessUrl("/hello")

.permitAll();

http.addFilterBefore(customizeFilterSecurityInterceptor, FilterSecurityInterceptor.class)

.csrf().disable();

}

参考资料：   
<http://blog.csdn.net/icanactnow2/article/details/53515844>   
<http://group.jobbole.com/24128/>   
<http://www.cnblogs.com/yjmyzz/p/customize-CsrfFilter-to-ignore-certain-post-http-request.html>