

0x01 前言几句

平时使用burp进行简单的测试，遇到各种各样的问题，使用各种各样的方法，我们可以抓包、改包、重放、爆破，同时有很多BApp Store有很多开源的扩展工具，今天我想要分享的是巧用一个名为“Extractor”的扩展插件和Burpsuite Marco的结合用法，获取他在你解决某些场景下的anti-token的问题时候可以有所帮助。

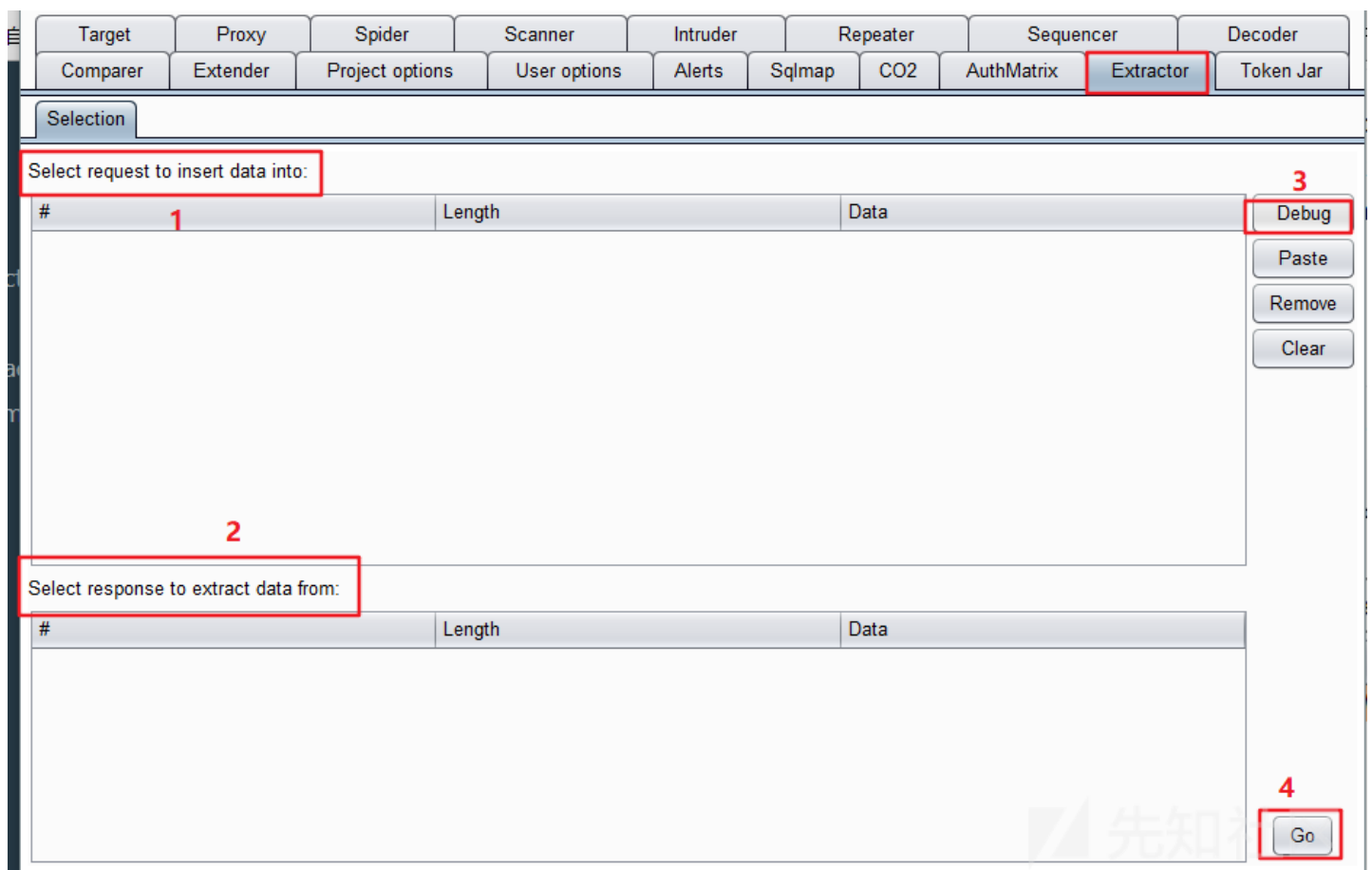
0x02 Extractor

1. 是什么?

A Burp extension for generic extraction and reuse of data within HTTP requests and responses.

用于在HTTP请求和响应中进行提取和重用数据的Burp扩展。

[Extractor](#)的功能用上面的一句话完全概括了，请求响应数据的提取和请求中数据的替换重放使用，那么这个工具的功能到底有多强大，多复杂呢？



可以看到，扩展界面大致分为三大部分：

- 1：选择要插入替换数据的请求；
- 2：选择要提取数据的响应；
- 3：开启Debug；
- 4：Go！！！！

2. 怎么用?

[Extractor](#)使用方法就是从请求的相应中提取数据，再另一个http请求中重用，诸如：CSRF token、timestamps(时间戳)、Auth Bearer token等某些场景中，通过使用正则提取数据，并且在burp发送指定请求时将提取出的值替换掉请求中的正则匹配到的值。

0x03 Marco

burpsuite自带的功能，宏这个功能应该都不陌生，之前已经有相关文章介绍Marco，如：[“Burpsuite中宏的使用”](#)一文。文章中详细的介绍了一般宏的录制以及使用，本文

0x04 实例分析

1. 发现问题

之前在测试某个接口的时候，发现有这样一个请求：

```
POST /WechatApp/public/zzbcd/P8009.do HTTP/1.1
Host: www.xxx.com

{"No": "123456"}
```

请求的返回中包含了如下信息：

```
{ "STATUS": "1", "No": "123456", "tranDate": "2018-8-9", "ID": "411103199206121819", "Name": " ", "Phone": "13333333333", "idType": "10" }
```

这里传入的参数是一个规律递增的ID值，通过重放的方式，可以批量获取大量个人敏感的身份信息。问题不算太大，但绝不是可以忽略掉的问题。

2. 修复问题

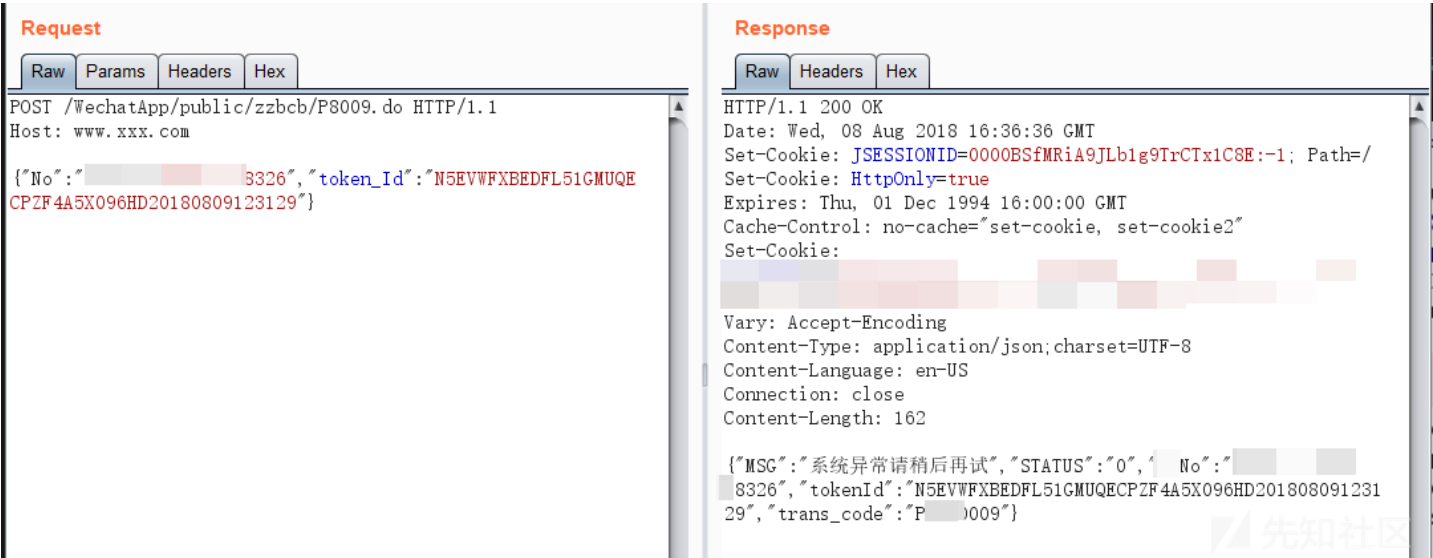
修复方案是建议是防止重放攻击，并且模糊处理返回的敏感信息。几日后，收到复测的安排，遂进行复测。

修复后的接口请求如下，其添加了一个Token_Id值，用来防止重放攻击？

```
POST /WechatApp/public/zzbcd/P8009.do HTTP/1.1
Host: www.xxx.com

{"No": "123456", "Token_Id": "N5EVWFXBEDFL51GMUQECFZF4A5X096HD20180809123129"}
```

重放之后提示“系统异常请稍后重试”，通过修改cookie参数中的值、refrer参数等方法，均无法绕过token_Id的检验。



重新查看proxy中的请求记录，发现在每次请求之前，会先发送一个请求，该请求返回包中，只有一个参数，即tokenId。如下所示：

361	https://www.xxx.com	POST	/WechatApp/public/zzbcd/P8009.do	200	405	JSON	data
358	https://www.xxx.com	POST	/WechatApp/public/zzbcd/P8009.do	200	641	JSON	data
355	https://www.xxx.com	POST	/WechatApp/public/check/setTokenId.do	200	405	JSON	data
354	https://www.xxx.com	GET	/WechatApp/public/zzbcd/P8009.do	200	3337	HTML	html
353	https://www.xxx.com	GET	/WechatApp/public/zzbcd/P8009.do	200	11218	JSON	data

Request

RawParamsHeadersHex

POST /public/check/setTokenId.do HTTP/1.1
Host: www.xxx.com

{}

Response

RawHeadersHex

HTTP/1.1 200 OK
Date: Wed, 08 Aug 2018 16:38:50 GMT
Set-Cookie: JSESSIONID=0000x1jXdX-7AXg11CQ5qBQf3zE:-1;
Path=/
Set-Cookie: HttpOnly=true
Expires: Thu, 01 Dec 1994 16:00:00 GMT
Cache-Control: no-cache="set-cookie, set-cookie2"
Vary: Accept-Encoding
Content-Type: application/json; charset=UTF-8
Content-Language: en-US
Connection: close
Content-Length: 98

{"MSG":"token获取成功!", "STATUS":"1", "token":"PUMJID4HKWGVMMXR
LD7UY48SLMFUXU5Z20180809124138"}

到此，脑子里首先想到的就是刚刚提到的“Burpsuite中宏的使用”，于是仔细阅读了该篇文章的用法，并仔细的对本地复测的情况录制了宏，遗憾的是最终并未能够成功，

- 自己配置/使用不正确(自认为概率很小，因为请教了不同的大佬，并进行了多次尝试)；
- 文章中示例使用DVWA搭建，具体到请求和场景和真实情况有所出入；
- 自己写了等效的脚本，一个函数请求并获取tokenId，另一个替换tokenId后发送请求获取信息，均已失败告终看来，此处的问题真的是修复了吧.....

3. 再次尝试绕过

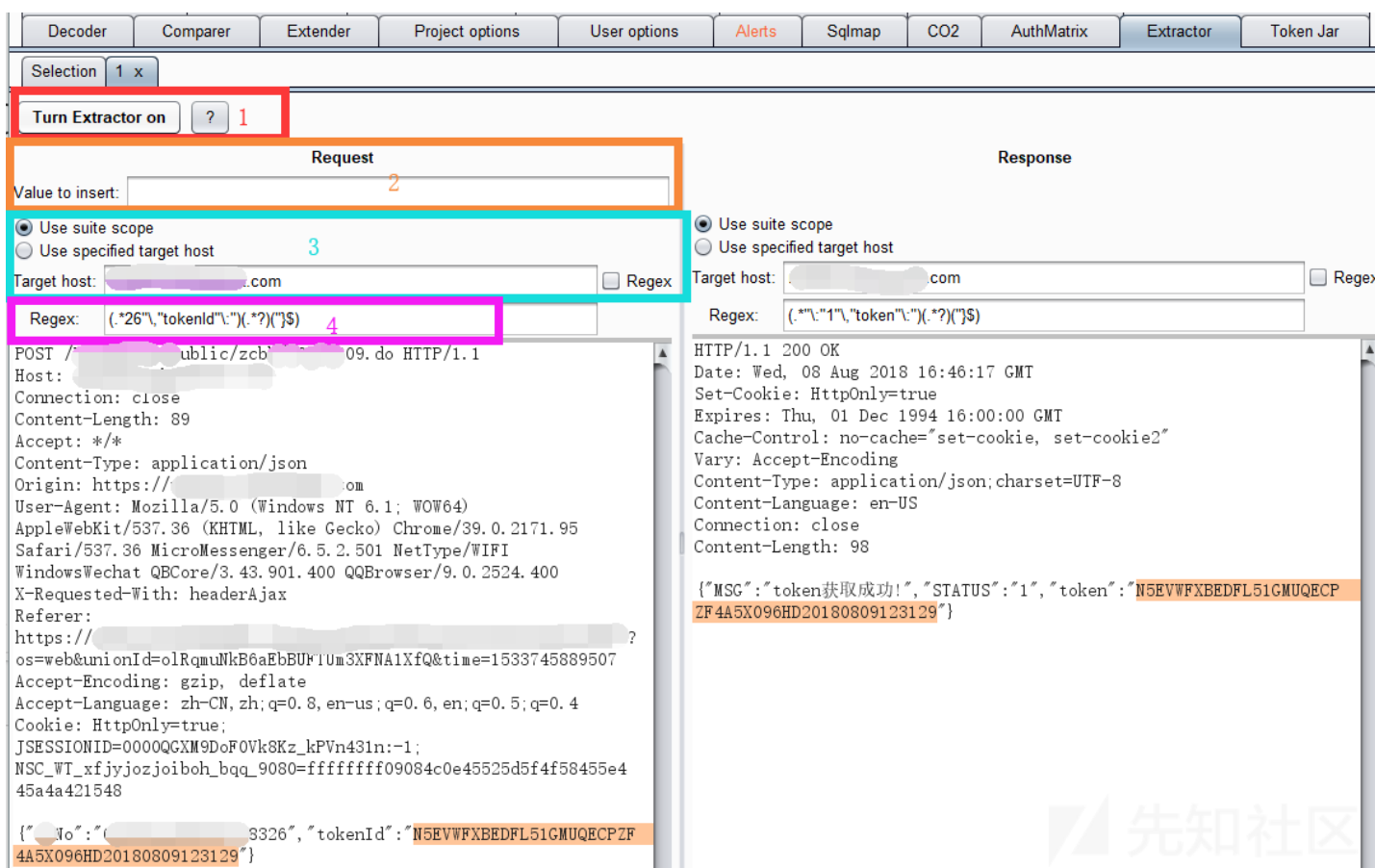
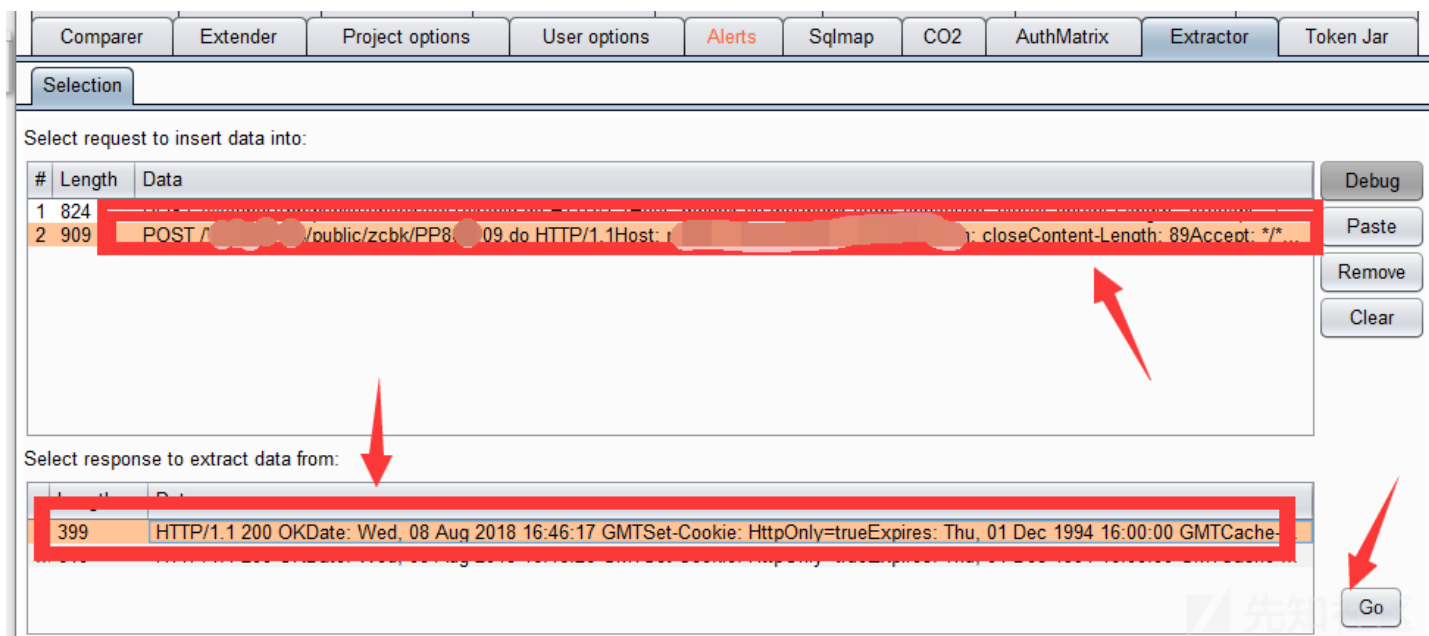
当看到Extractor插件的时候，我决定重新在生产上复测这个当时存疑的接口。

首先，分别将两个接口发送到Extractor扩展。

Method	URL	Params	Edited	Status	Length	MIME t...	Extension	Title
POST				200	405	JSON	do	
POST	pp/public/zcbk/PP800009.do		✓	200	641	JSON	do	
POST	pp/public/check/setTokenId.do							
GET								
GET								

Add to scope
Remove from scope
Spider from here
Do an active scan
Do a passive scan
Send to Comparer (requests)
Send to Comparer (responses)
send to Sqlmap
Send to SQLMapper
Send to CeWler
Send to Laudanum
Send request(s) to AuthMatrix
Send to Extractor
Show new history window

可以在Extractor看到两个请求，上工作区选择“使用tokenId”的请求，下工作区选择“获取tokenId”的响应包，点击Go，Extractor就准备好了。

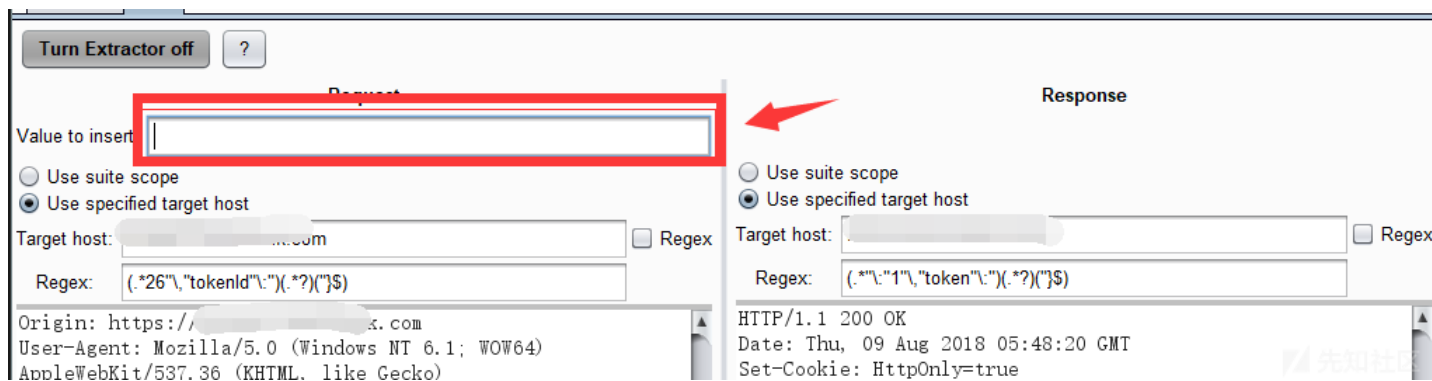


上图的中标注了四个点，解释一下：

- 1：开启Extractor；
- 2：最新正则匹配到的tokenId显示区域
- 3：设置host，可以直接使用burp scope中的host，也可以自定义host
- 4：左右两边，分别设置需要正则匹配的内容(此处左侧正则匹配要带如请求的tokenId，右侧表示正则匹配获取到的tokenId)

当“Turn Extractor

on”被开启后，Extractor即开始起作用。此时去Repeater进行重放，看是否Extractor已经生效，每次请求替换最新获取匹配的tokenId值，但是你会发现，可能会失败了

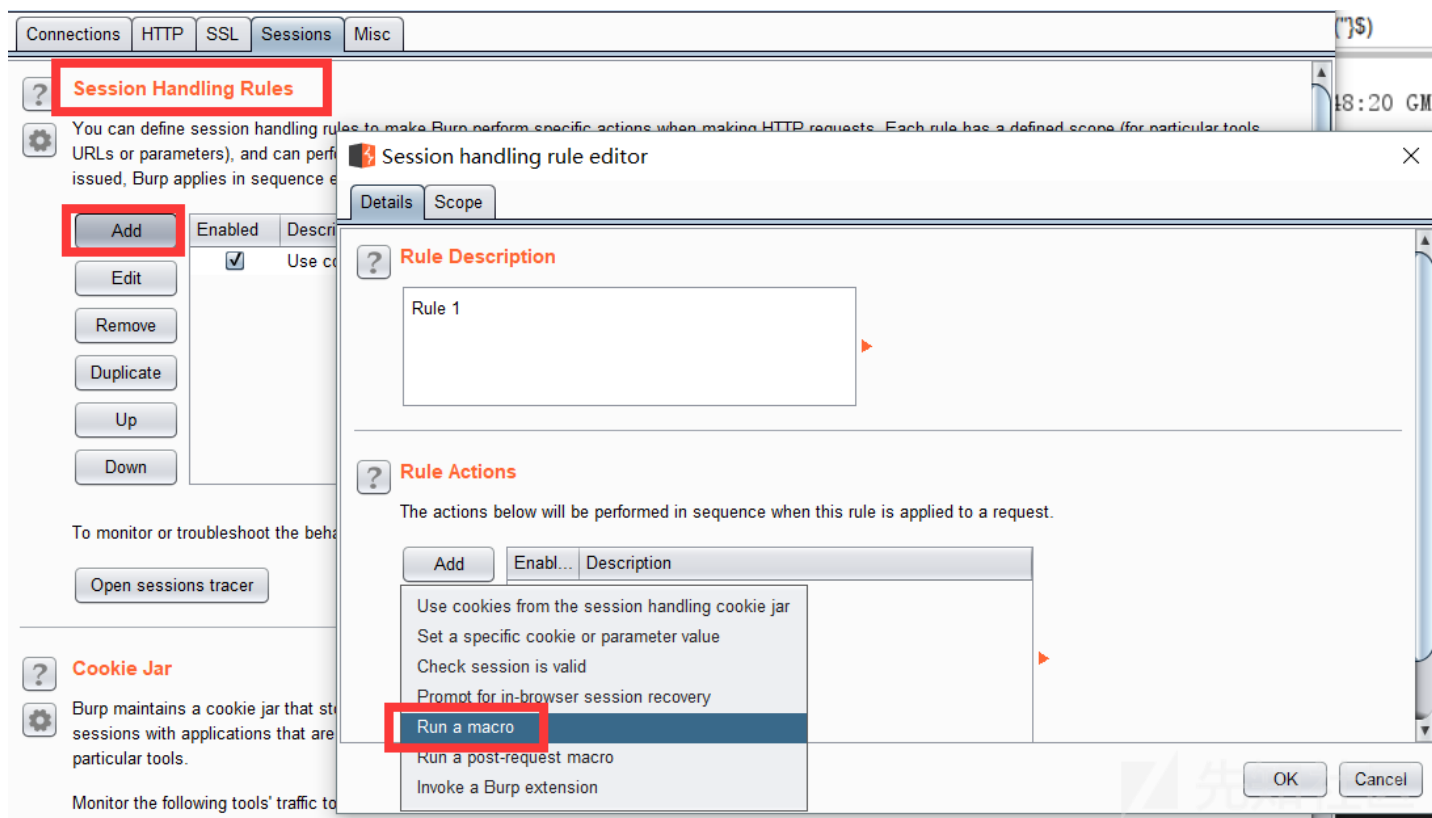


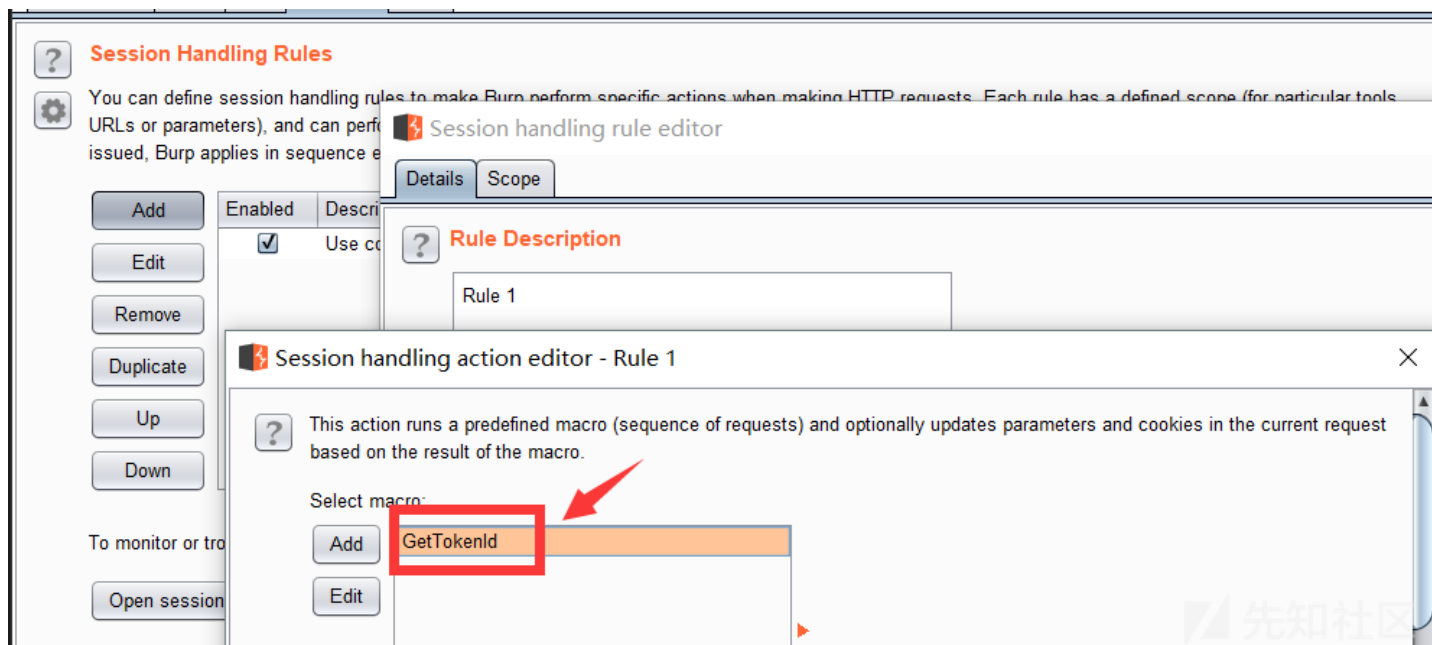
“Value to

insert”值不为空才可以正常使用，需要先触发“获取tokenId”请求，之后Extractor才可以从相应中正则匹配到tokenId值，[Extractor项目](#)中的gif动画因为是简单的演示了token。

所以此处我们如果想要进行爆破或者枚举的操作，必须每次请求前先请求一次“获取tokenId”，这样才可以达到枚举爆破的效果，So，是时候用Marco了。

这里的思路倒也简单，可以帮助理解burp在使用Marco(宏)的时候是怎样的一个逻辑，录制一个宏，什么条件都不需要附加，只需要保证在请求之前先把宏中设置的“获取to



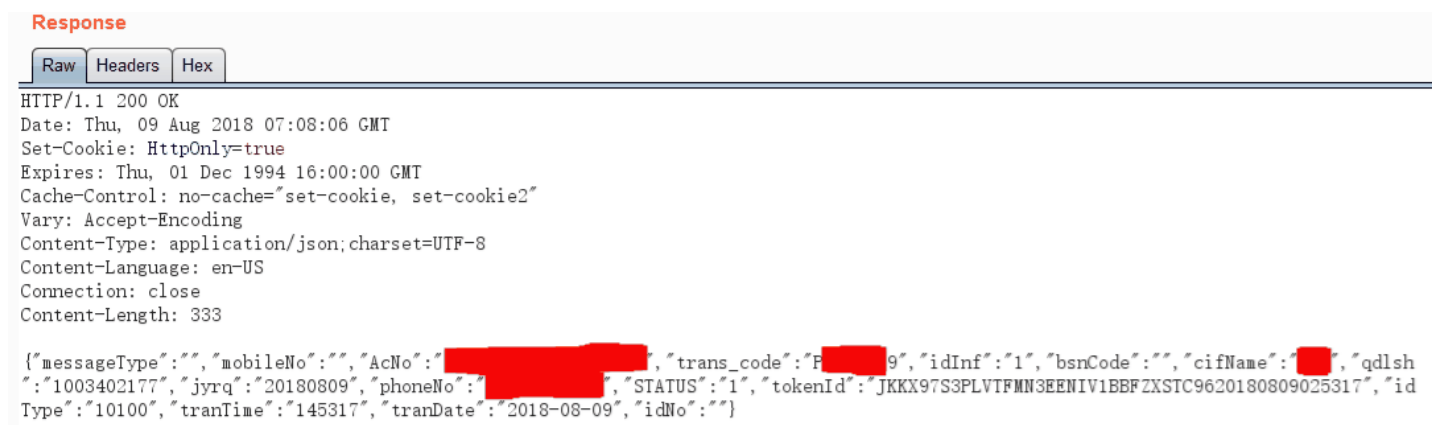


注(这里的几点说明)：

- Add添加一个规则Rule；
- 规则内容为“在请求之前，先运行一个Marco(宏)”；
- “Session Handling Rules”和“Marco”配合来用，前者配置一个规则，调用后者的内容；
- 通过在“Scope”中配置在哪个特殊请求或者全部请求需要触发宏，并且选择在哪些功能区触发之后，可以进行验证了。

4. 最终验证

下面是在Repeater中连续点击多次之后的返回，如下所示：

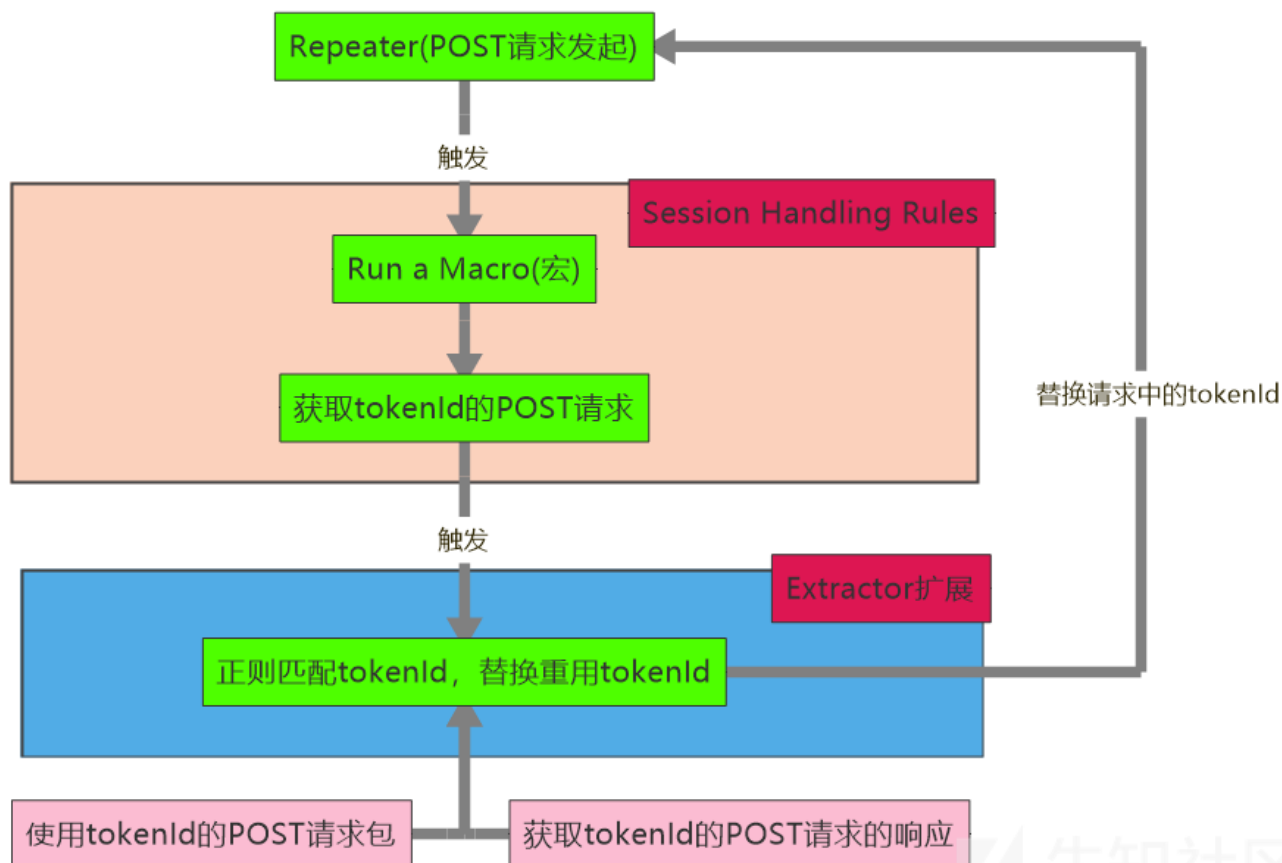


之后可以Intruder可以进行爆破咯~，结果就是问题真的修复了，虽然解决了tokenId的问题，但是修复再其他位置也做了校验，所以突破了tokenId的限制，但是无法重

PS: 一时手边没有真实有效的例子，之后再做补充真实过程。

0x06 绕过流程

整个过程中的逻辑流程，帮助理解整个绕过token校验的流程：



0x05 再来一发

1. 发现问题

某微信公众号的注册接口

4.经营状况

从业人数

营业收入(万)

资产总额(万)

☒ 本人已阅读并充分理解 《

结算账户管理协议

》

提交

填写资料后提交请求，会发送短信通知返回结果。

POST /mina/saveCustomerInfo?applyId=2018080916437317 HTTP/1.1
Host: www.xxx.com


```
POST /mina/saveCustomerInfo applyId=2018080916437317 HTTP/1.1
Host: 
Connection: close
Content-Length: 786
Accept: application/json
Origin: 
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (Linux; Android 4.4.2; SM-G955N Build/NRD90M)
AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/30.0.0.0
Mobile Safari/537.36 MicroMessenger/6.6.7.1321(0x26060739) NetType/WIFI
Language/zh_CN
Content-Type: application/x-www-form-urlencoded
Referer: 

Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,en-US;q=0.8
Cookie: JSESSIONID=nv8lqYX6aMG0SpWvioX7JKBJIzLWGg4YlBNFltz

depositorName=&accntType=yiban&depositorType=01&registeredCapital=100&zipCode=010111&workProvince=110000&workCity=110100&workArea=110108&workAddress=E5%&A3%E7%94%B%EA2&businessScope=&9A&legalCardType=15&cardDue=9999-12-31&legalTelephone=&leType=00&fileNo=11A1111111112312&fileDue=9999-12-31&empNumber=&sales=&totalAssets=
```

```
HTTP/1.1 200 OK
Server: nginx/1.12.2
Date: Thu, 09 Aug 2018 08:46:30 GMT
Content-Type: application/json;charset=UTF-8
Connection: close
Vary: Accept-Encoding
X-Application-Context: weixin-admin:9701
Content-Length: 100

{"rel":true,"encrypt":false,"result":{"applyId":"2018080916437317","internalId":"YTD2018080914675"}}
```



Raw Params Headers Hex

POST /mina/saveCustomerInfo?applyId=2018080916437317 HTTP/1.1

Host:

Connection: close

Content-Length: 786

Accept: application/json

Origin:

X-Requested-With: XMLHttpRequest

User-Agent: Mozilla/5.0 (Linux; Android 4.4.2; SM-G955N Build/NRD90M)

AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/30.0.0.0

Mobile Safari/537.36 MicroMessenger/6.6.7.1321 (0x26060739) NetType/WIFI

Language/zh_CN

Content-Type: application/x-www-form-urlencoded

Referer:

Accept-Encoding: gzip, deflate

Accept-Language: zh-CN, en-US; q=0.8

Cookie: JSESSIONID=nv8kpYX6aMG0SpWVioX7JKBJ1zLWGg4Y1BNFltz

depor: ...

torType=00&inausType=...

...®isteredCapital=100&zipCode=...

...legal...

cardType=15&leg...no=11111111111111111111&...IdcardDue=9999-12-31&

legalTelephone=...legalMobil...fileType=00&fileNo=11

...9999-12-31&...s=&totalAssets=

Raw	Headers	Hex
HTTP/1.1 200 OK Server: nginx/1.12.2 Date: Thu, 09 Aug 2018 08:49:09 GMT Content-Type: application/json;charset=UTF-8 Connection: close Vary: Accept-Encoding X-Application-Context: weixin-admin:9701 Content-Length: 86 {"rel":false,"encrypt":false,"msg":"异常信息提示: 异常操作导致的错误"}		

```
{"rel":false,"encrypt":false,"msg":"异常信息提示: 异常操作导致的错误"}
```

```
UserData(■■■■■■)
```


Request

RawParamsHeadersHex


POST /mina/apply HTTP/1.1
Host: [REDACTED]
Connection: close
Content-Length: 376
Accept: application/json
Origin: [REDACTED]
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (Linux; Android 4.4.2; SM-G955N Build/NRD90M)
AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/30.0.0.0
Mobile Safari/537.36 MicroMessenger/6.6.7.1321(0x26060739) NetType/WIFI
Language/zh_CN
Content-Type: application/x-www-form-urlencoded
Referer: [REDACTED]
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN, en-US; q=0.8
Cookie: JSESSIONID=nv8kpYX6aMG0SpWVioX7JKBJ1IzLWGg4YlBNF1tz
[REDACTED]
[REDACTED]&name=%E6%9E%9C%E5%AE%9E%7%A7%91%E6%8A%80&companyType=yiban&selectBankName=313301008887&province=110000&city=110100&district=110108&openBank=BASEROOTNODE-313301008887-e0F2G0WLOIs4-WIPbc3aXc2r&applyTime=2018-08-09&contactsName=%E7%8E%8B%E6%AF%9B&mobile=18[REDACTED]669&bankC[REDACTED]&companyType=[REDACTED]

Response

RawHeadersHex

HTTP/1.1 200 OK
Server: nginx/1.12.2
Date: Thu, 09 Aug 2018 08:43:43 GMT
Content-Type: application/json; charset=UTF-8
Connection: close
Vary: Accept-Encoding
X-Application-Context: weixin-admin:9701
Content-Length: 56

{ "rel": true, "encrypt": false, "result": "2018080916437317" }



先知社区

2. 绕过token限制

绕过的思路和操作已经很清晰，遵循上一个实例的套路：

- 添加两个请求到Extractor，分别选择“请求”和“响应”；
- 录制一个“获取响应”的宏，并且定制好Rule；
- 可以进行短信炸弹了，嗯，没有错，是短信炸弹，送给那个上班时间扣手机的同事，Intruder Null Payload来一百条为敬。

Selection 4 x 5 x

Turn Extractor on ?

Request

Value to insert: 2018080917241029
☐ Use suite scope
☒ Use specified target host
Target host: [REDACTED] ☐ Regex
Regex: (.?info?applyId=)(.*) HTTP/1.1\nHost.*
POST /mina/saveCustomerInfo?applyId=2018080916437304 HTTP/1.1
Host: [REDACTED]
Connection: close
Content-Length: 786
Accept: application/json
Origin: [REDACTED]
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (Linux; Android 4.4.2; SM-G955N Build/NRD90M)
AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/30.0.0.0
Mobile Safari/537.36 MicroMessenger/6.6.7.1321(0x26060739)
NetType/WIFI Language/zh_CN
Content-Type: application/x-www-form-urlencoded

Response

☐ Use suite scope
☒ Use specified target host
Target host: [REDACTED] ☐ Regex
Regex: (.?else?,result":)(.*)(\$)
HTTP/1.1 200 OK
Server: nginx/1.12.2
Date: Thu, 09 Aug 2018 08:43:47 GMT
Content-Type: application/json; charset=UTF-8
Connection: close
Vary: Accept-Encoding
X-Application-Context: weixin-admin:9701
Content-Length: 56

{ "rel": true, "encrypt": false, "result": "2018080916437304" }

先知社区

Intruder attack 4						
Attack Save Columns						
Results Target Positions Payloads Options						
Filter: Showing all items						
Request	Payload	Status	Error	Timeout	Length	Comment
1	null	200	<input type="checkbox"/>	<input type="checkbox"/>	323	
2	null	200	<input type="checkbox"/>	<input type="checkbox"/>	323	
3	null	200	<input type="checkbox"/>	<input type="checkbox"/>	323	
4	null	200	<input type="checkbox"/>	<input type="checkbox"/>	323	
5	null	200	<input type="checkbox"/>	<input type="checkbox"/>	323	
6	null	200	<input type="checkbox"/>	<input type="checkbox"/>	323	
7	null	200	<input type="checkbox"/>	<input type="checkbox"/>	323	
8	null	200	<input type="checkbox"/>	<input type="checkbox"/>	323	
9	null	200	<input type="checkbox"/>	<input type="checkbox"/>	323	
10	null	200	<input type="checkbox"/>	<input type="checkbox"/>	323	

这里需要注意勾选你要覆盖到的功能区域，否则Marco可能会发现没有生效。

Session handling rule editor

Details

Scope

Tools Scope

Select the tools that this rule will be applied to.

☒ Target
 ☒ Scanner
 ☒ Repeater

☒ Spider
 ☒ Intruder
 ☒ Sequencer

☒ Extender
 ☒ Proxy (use with caution)

URL Scope

Use the configuration below to control which URLs this rule applies to.

☐ Include all URLs
 ☐ Use suite scope [defined in Target tab]
 ☒ Use custom scope

☒ Use advanced scope control

Include in scope

Add

Edit

Remove

Enabled	Protocol	Host / IP range	Port	File
<input checked="" type="checkbox"/>	HTTPS	^wx\.\heabo\com\$	^443\$	^/mina/saveCustomerInfo.*

0x07 参考

- <https://github.com/NetSPI/BurpExtractor>
- <http://www.freebuf.com/articles/web/156735.html>
- 给自己的博客埋个广告 <http://am4zing.com.cn>

点击收藏 | 4 关注 | 2

上一篇 : Pwn2Own 2018 Safa... 下一篇 : Linux反弹shell (一) 文件...

1. 1 条回复



[Hulk](#) 2019-02-14 09:47:19

好文，精读了

0 回复Ta

[登录](#) 后跟帖

先知社区

[现在登录](#)

热门节点

[技术文章](#)

[社区小黑板](#)

目录

[RSS](#) [关于社区](#) [友情链接](#) [社区小黑板](#)