实战绕过双重waf(玄武盾+程序自身过滤)结合编写sqlmap的tamper获取数据

> 先知社区，先知安全技术社区

前言

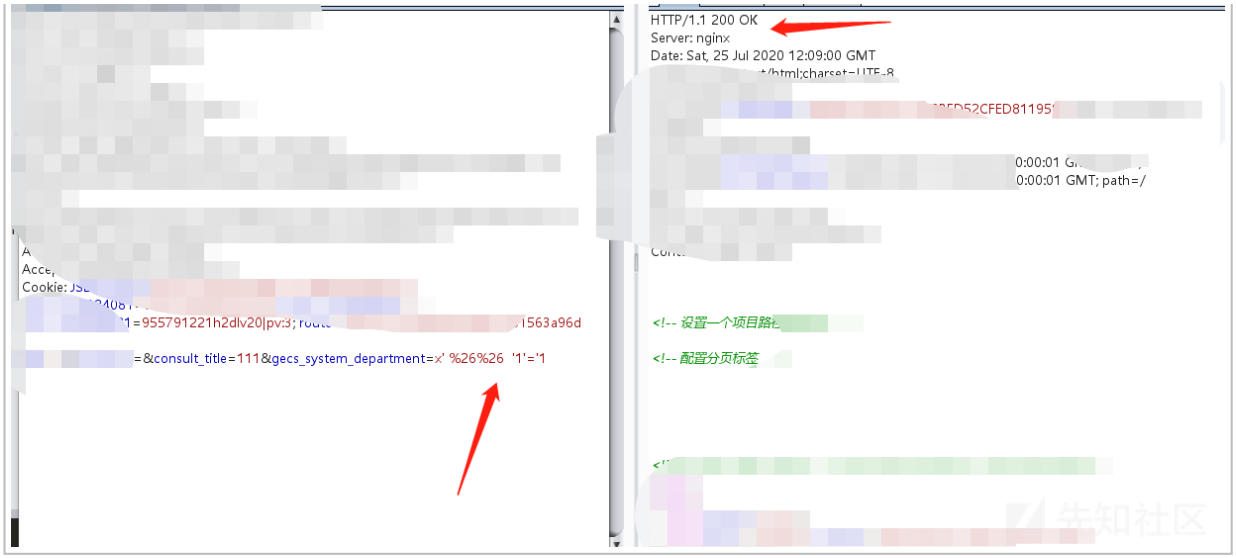
某次渗透中遇到的注入存在双层 waf 的情况, 第一层是云 waf 第二层似乎是本地的安全狗或者是代码层面的防护, 后端数据库是 mysql,jsp 环境，最终通过多次 fuzz 的方式成功找到突破口, 并且利用 sqlmap 的 tamper 脚本来获取数据.

正文

首先来看一下这个双层 waf 的大概情况, 首先是云 waf，利用一个简单的'and'1'='1 触发 waf

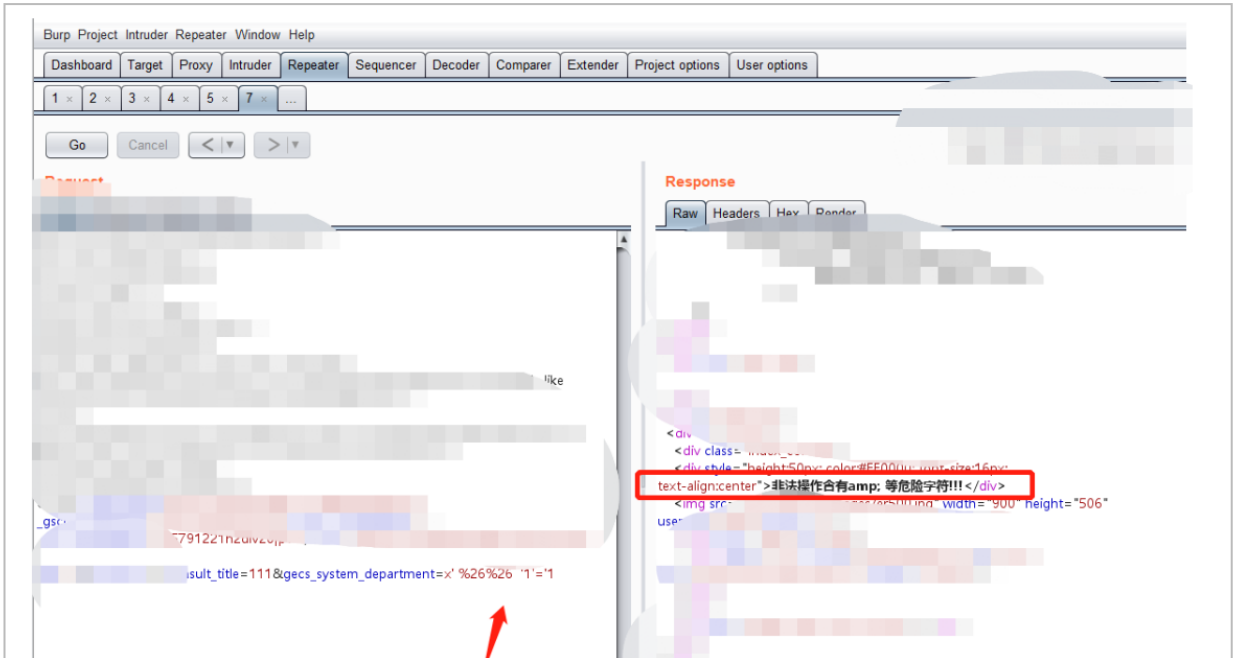
(https://xzfile.aliyuncs.com/media/upload/picture/20200726101123-4d95a996-cee5-1.png)
渲染后的样子

(https://xzfile.aliyuncs.com/media/upload/picture/20200726101138-56b4c8cc-cee5-1.png)
有兄弟看出这个是啥牌子的 waf 嘛 有的话希望告知一下 小弟没看出来
因为后端是 mysql 所以直接用 && 替换了 and 并且 url 编码一下 因为是特殊字符 直接就过了 我是没想到的 0_0



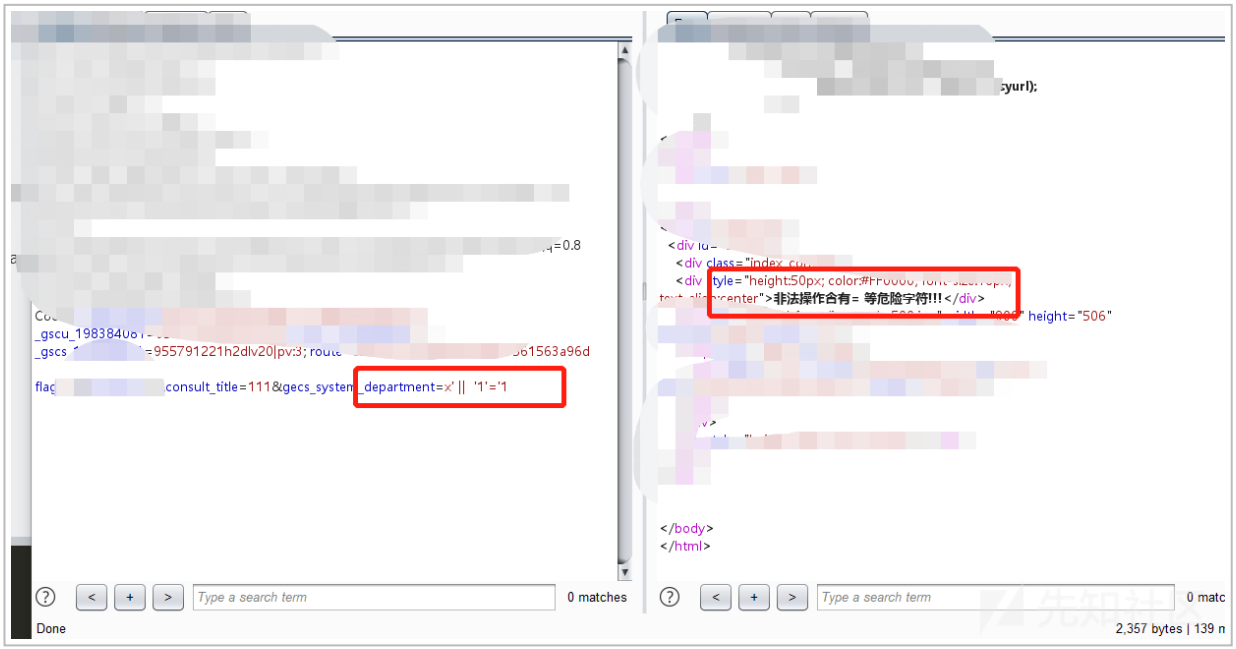
(https://xzfile.aliyuncs.com/media/upload/picture/20200726101153-5f6081a0-cee5-1.png)

其他更多的 payload 我还没测试我在回包里发现还有检测, 应该是本地的安全狗或者是代码层面的检测



(https://xzfile.aliyuncs.com/media/upload/picture/20200804170200-286a5f9c-d631-1.png)

换一个 or 对应的 || 试试



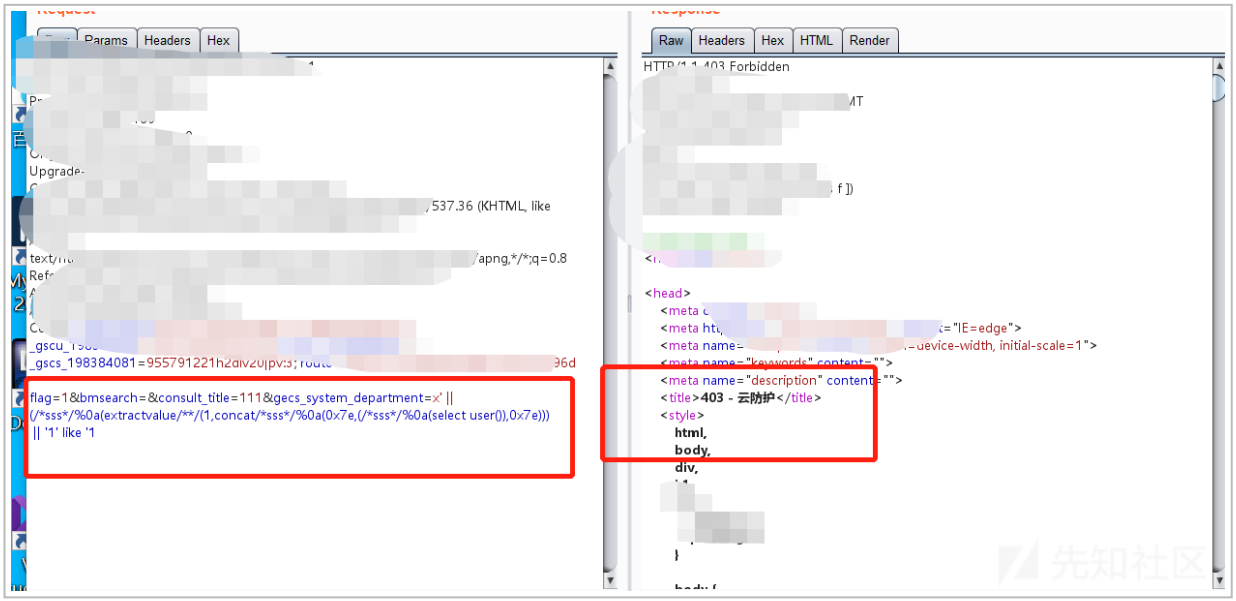
(https://xzfile.aliyuncs.com/media/upload/picture/20200726101218-6e7c1988-cee5-1.png)

偶哟 还不错哦 检测了 = 号 我们用 like 来替换一下

(https://xzfile.aliyuncs.com/media/upload/picture/20200726101229-74f86d02-cee5-1.png)

成功过去了 我低头沉思 虽然过去了 但是好像没有什么卵用 单纯 or 用 || 替换 = 用 like 替换 那么其他的 payload 会不会被检测到了 我觉得多半是会被检测到的 我们来试一试 因为这个返回是把错误打印出来的 多半是支持报错注入的

找个报错注入的 payload 试一试 稍微加点注释和换行



(https://xzfile.aliyuncs.com/media/upload/picture/20200726101316-91172ed8-cee5-1.png)

理所当然的被拦截了 这都拦截不鸟的话 这云 waf 也太垃圾了 0_0

如果想要真正把这个注入利用起来, 也就是最终想要获取到数据库数据, 那么多半要找一个一劳永

逸，比较通用的绕过方法, 类似于 假定可以双写绕过 那么利用 sqlmap 的 tamper 把所有关键字双写 假如只是不允许有空格 那么用其他能绕过检测的东西批量替换空格 获取数据 所以我想寻找到类似上述两种情景的这么一个突破口

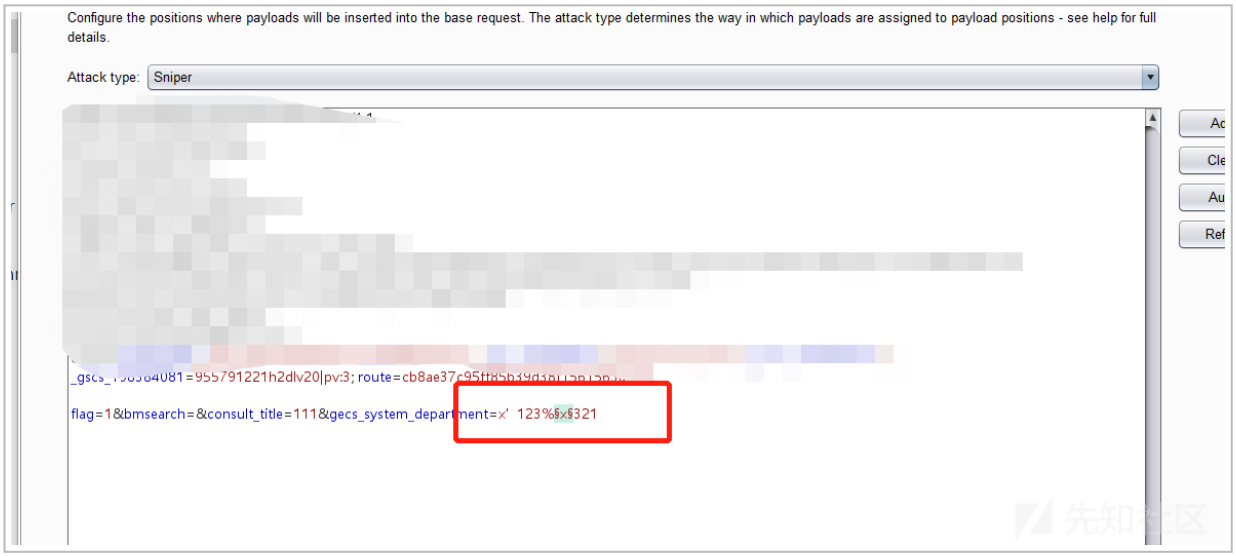
于是我开始 fuzz 还有一个比较好的地方是 因为如果语法出错 返回包会把具体位置 语句 给你爆出来 所以我们可以对着 返回包来 fuzz

首先 如果语法有问题那么 返回包会显示具体情况

(https://xzfile.aliyuncs.com/media/upload/picture/20200726101339-9ea85c5c-cee5-1.png)

我特地构造了一个 payload 来 fuzz

fuzz payload 如下

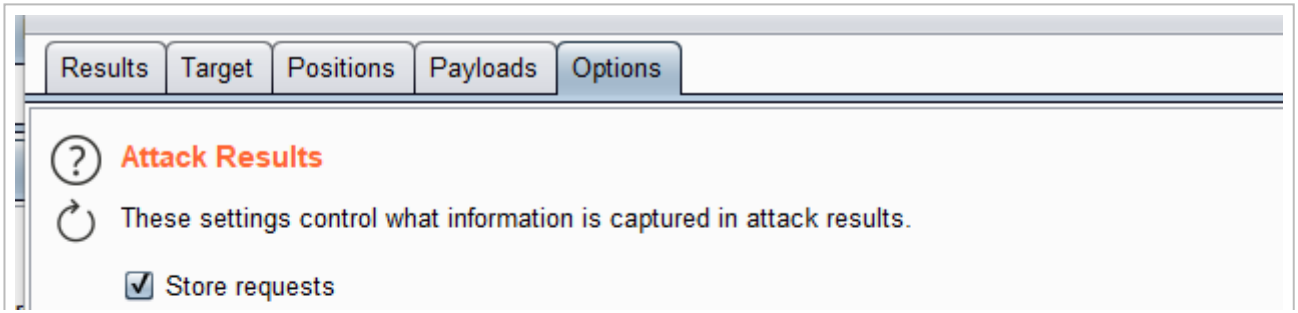


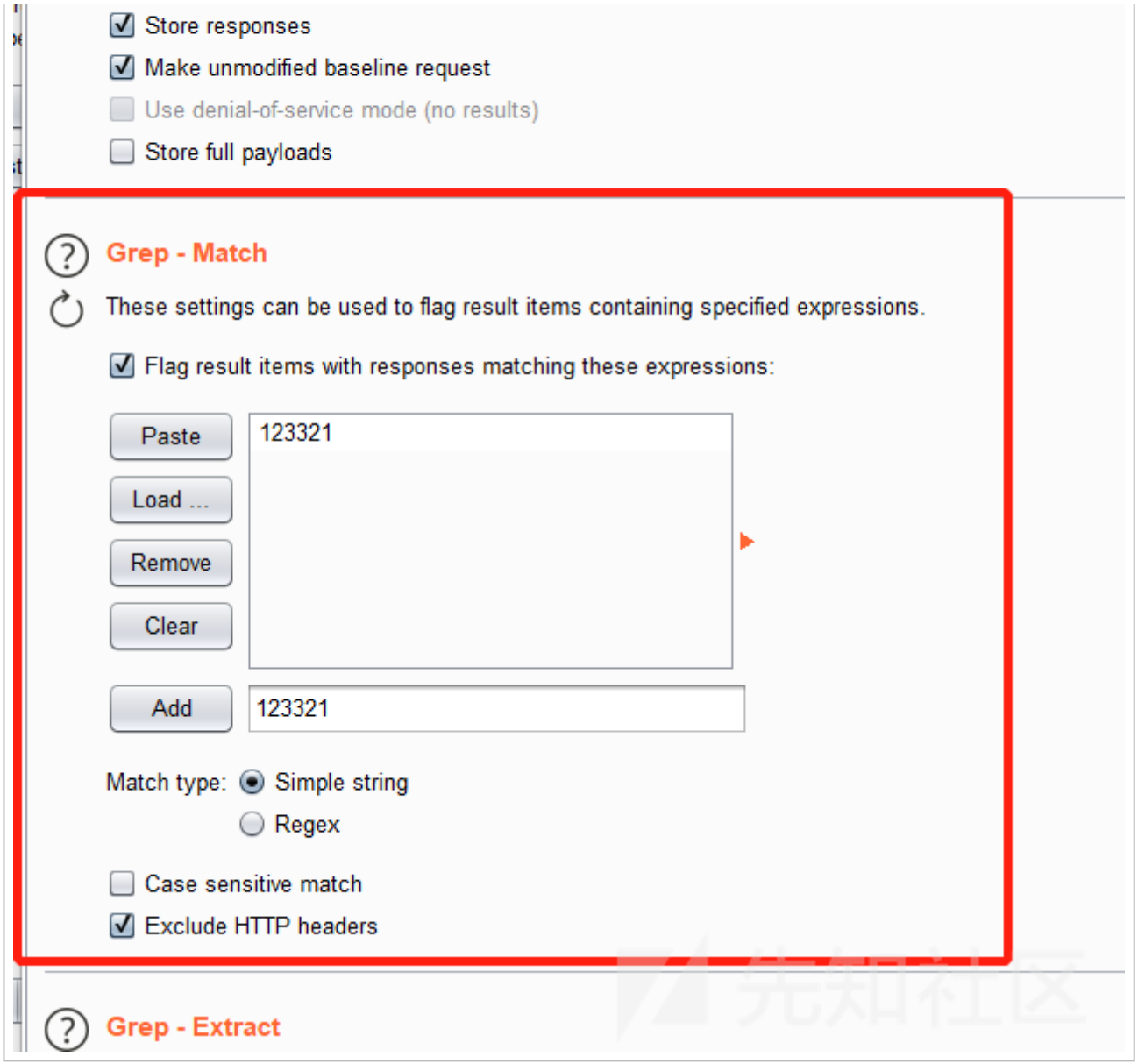
(https://xzfile.aliyuncs.com/media/upload/picture/20200726101352-a69803f4-cee5-1.png)

我想找到一个会被替换为空的字符 如果存在这样一个字符那么肯定可以绕过 waf 利用 sqlmap 获取数据 如果存在这么一个字符 返回结果里肯定有 123321 那么我只需要在结果里寻找这个 123321 就好

加载 16 进制数据 00 01 一直到 100(256 的 16 进制) 测试所有字符 看有没有机会

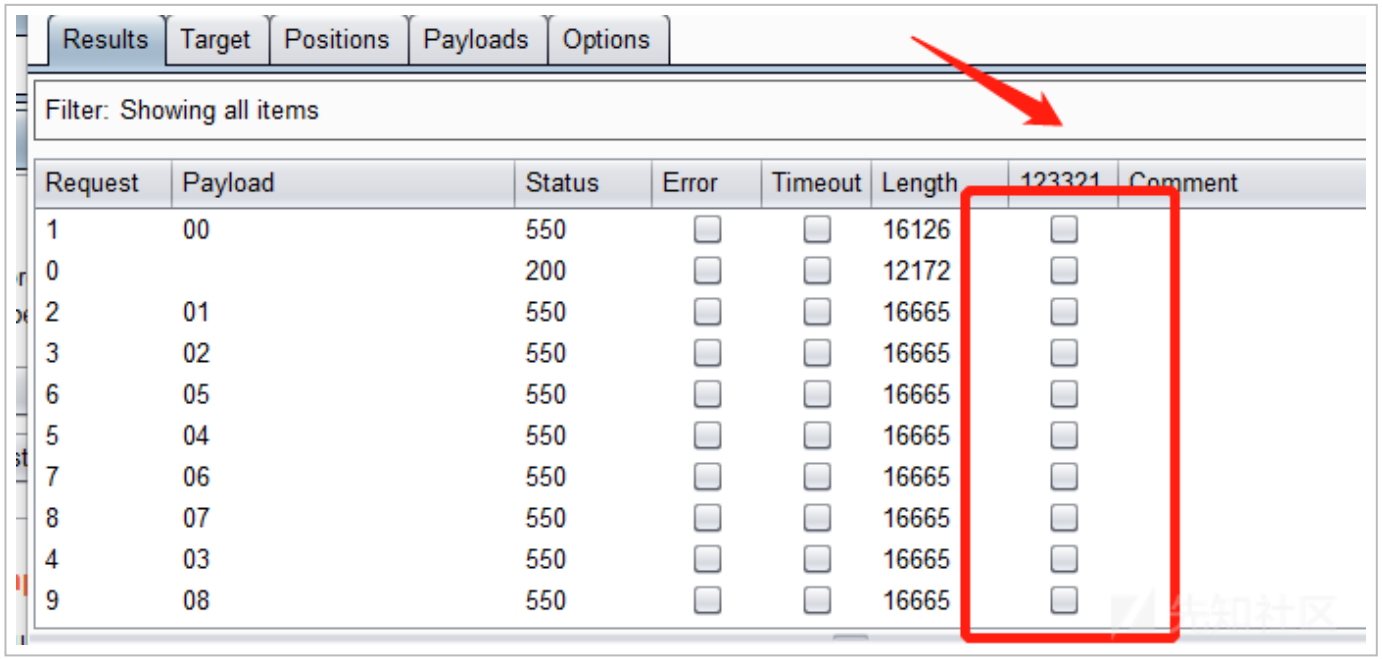
在 intruder 爆破结束后的 option 选项里 可以设置要寻找的数据 下图就是设置要寻找的 123321





(https://xzfile.aliyuncs.com/media/upload/picture/20200726101426-baae384a-cee5-1.png)

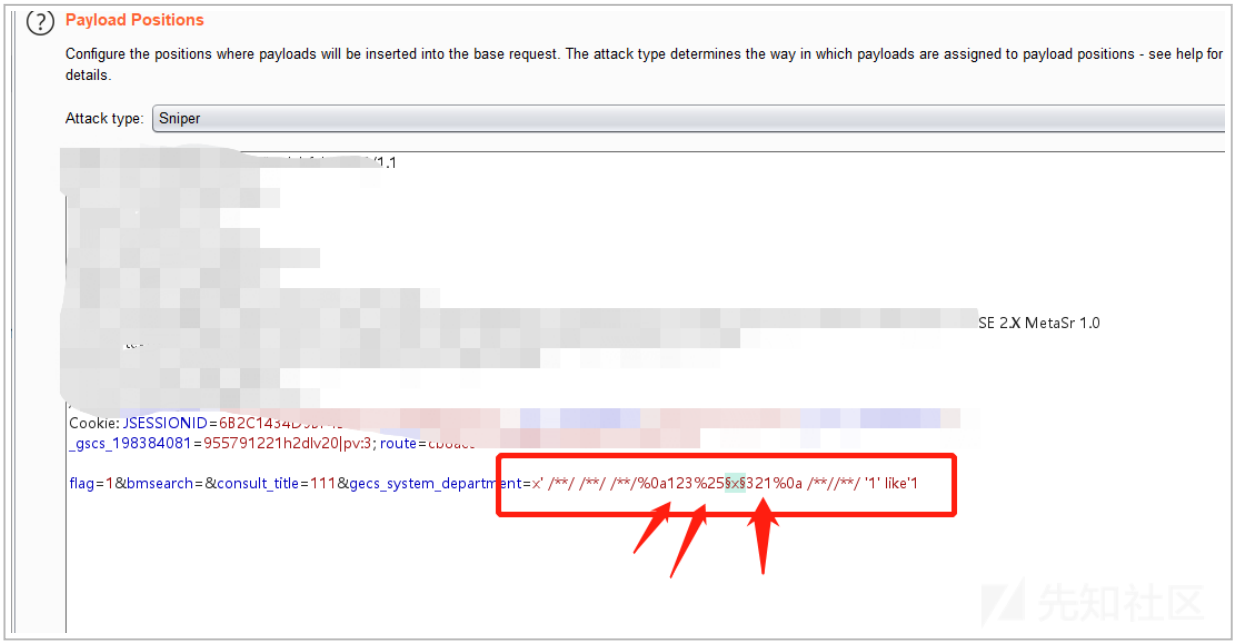
结果没有发现这样一个字符



(https://xzfile.aliyuncs.com/media/upload/picture/20200726101438-c1aae5f8-cee5-1.png)

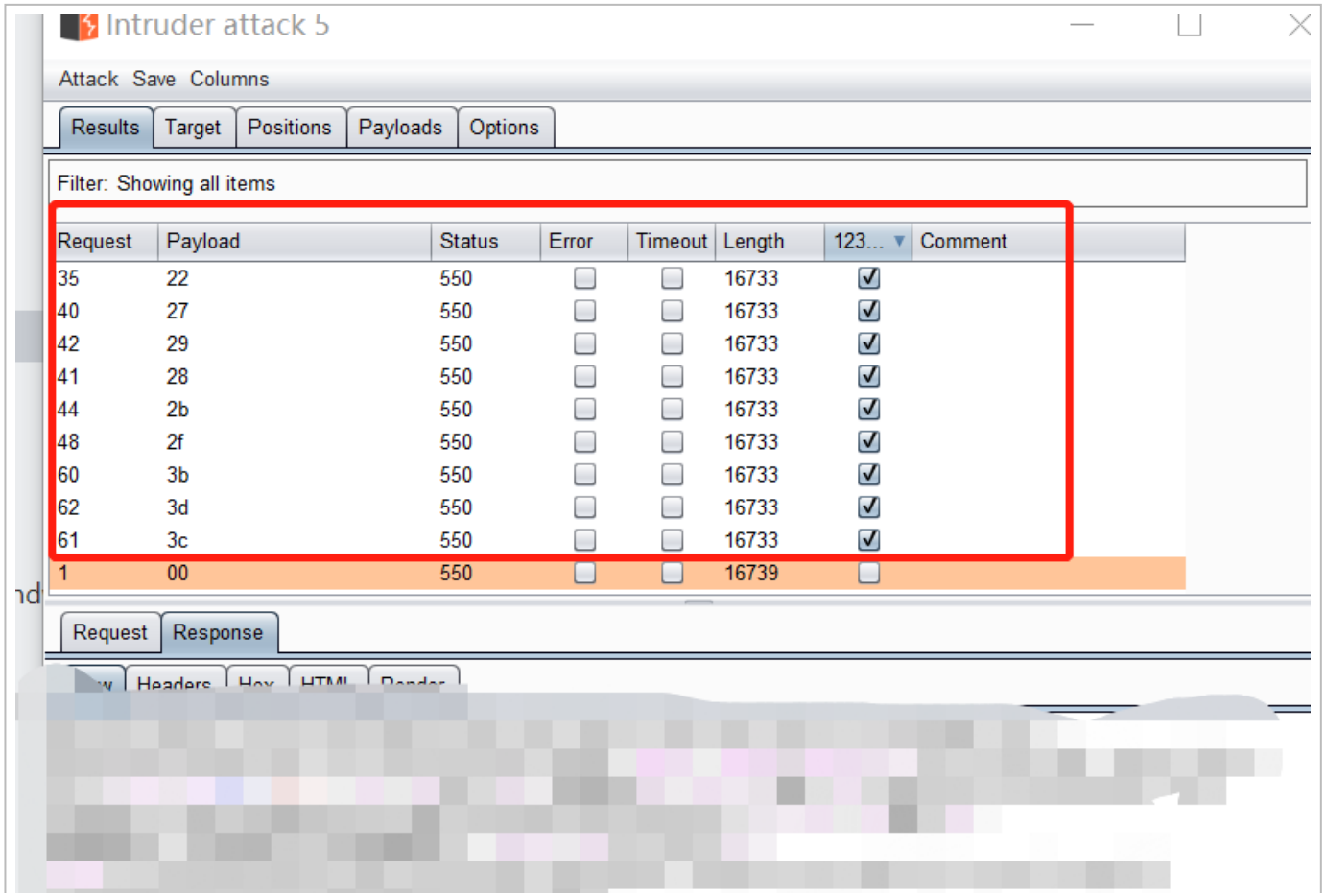
不慌 我们继续来 fuzz 我们来用两次 urlencode 试试 看后端怎么处理

这次构造的 payload 如下 就是 urlencode 了两次 对要测试的字符 我只要在结果里寻找 123321 就好



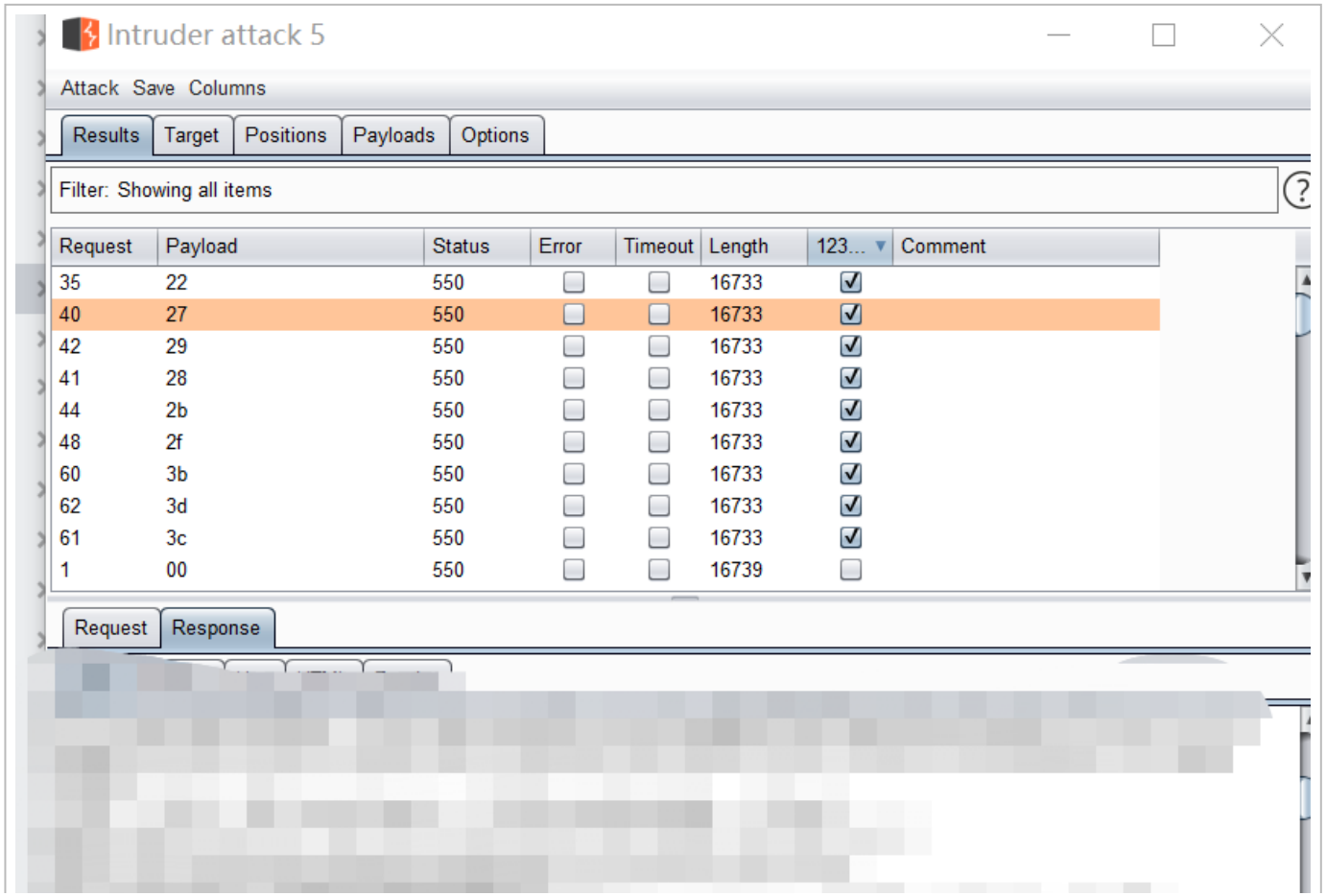
(https://xzfile.aliyuncs.com/media/upload/picture/20200726101452-ca69f06c-cee5-1.png)

let's go


Intruder attack 5
Attack Save Columns
Results Target Positions Payloads Options
Filter: Showing all items
Request Payload Status Error Timeout Length 123... Comment
35 22 550 16733
40 27 550 16733
42 29 550 16733
41 28 550 16733
44 2b 550 16733
48 2f 550 16733
60 3b 550 16733
62 3d 550 16733
61 3c 550 16733
1 00 550 16739
Request Response

(https://xzfile.aliyuncs.com/media/upload/picture/20200726101503-d0d66296-cee5-1.png)

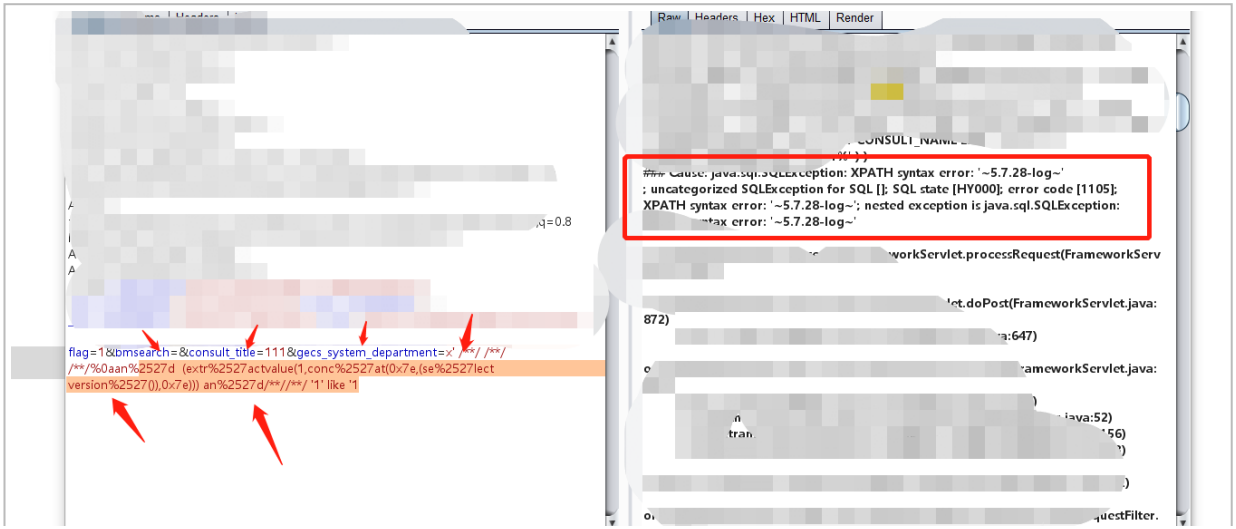
bingo 有结果了 0_0 运气不错 第二次 fuzz 就中奖了 奥里给

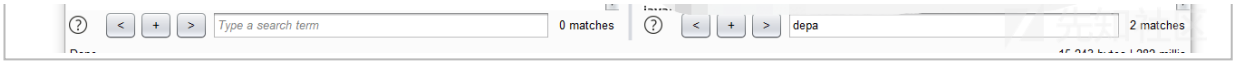

Intruder attack 5
Attack Save Columns
Results Target Positions Payloads Options
Filter: Showing all items
Request Payload Status Error Timeout Length 123... Comment
35 22 550 16733
40 27 550 16733
42 29 550 16733
41 28 550 16733
44 2b 550 16733
48 2f 550 16733
60 3b 550 16733
62 3d 550 16733
61 3c 550 16733
1 00 550 16739
Request Response

(https://xzfile.aliyuncs.com/media/upload/picture/20200726101517-d9164cd2-cee5-1.png)

可以看到 %2527 被替换为空了只剩下 123321 因为 post 的时候 会自动 urldecode 一次 所以后端程序其实是把 %27 替换为空了

这么多组合都可以用来绕 waf 嘿嘿 简单测试一下 报错注入





(https://xzfile.aliyuncs.com/media/upload/picture/20200726101534-e369a774-cee5-1.png)
哦哟 舒服了 接下来就是修改 sqlmap tamper 来获取数据

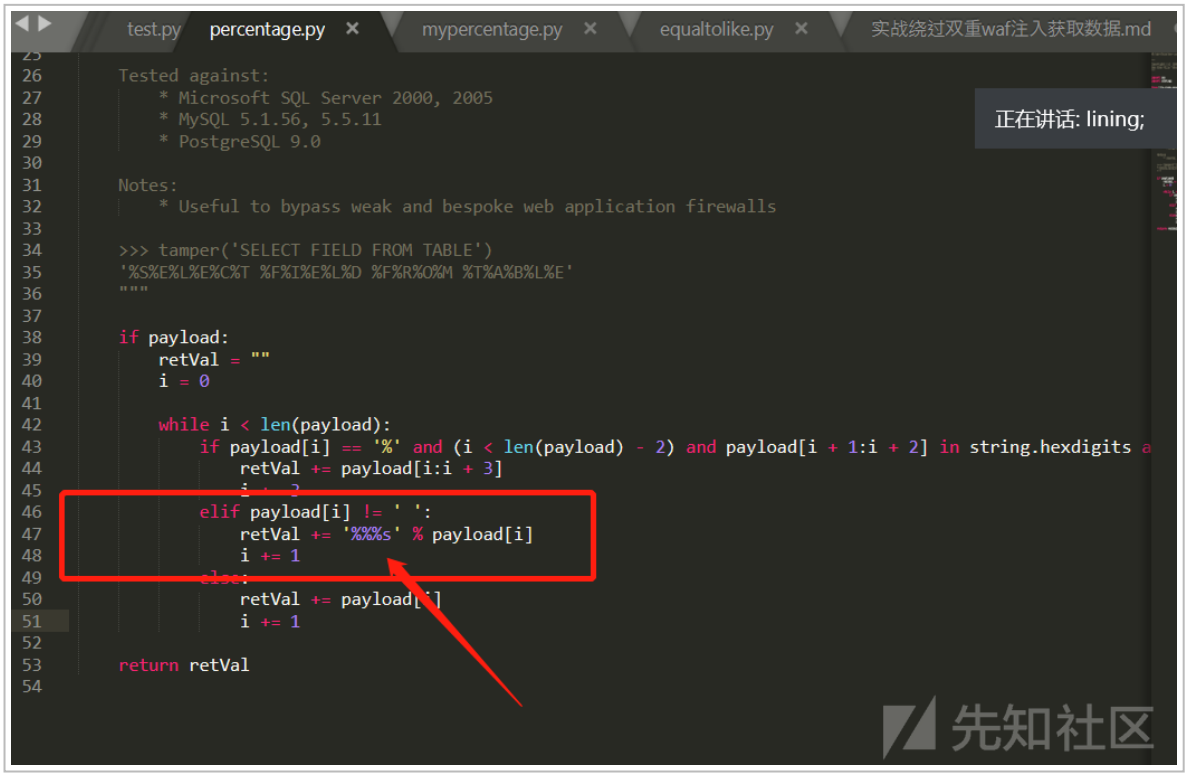
什么是 sqlmap 的 tamper
sqlmap 是一款人见人爱的自动化 SQL 渗透工具，能够以良好的引擎发现给定 URL 中的可注入处，并自动化的完成注入。但是由于 SQL 注入的影响过于广泛，致使现代程序的防护越来越严

密。sqlmap/tamper 是官方给出的一些绕过脚本，可以配合渗透测试人员完成更高效更高质量的测试

至于 sqlmap 有哪些 tamper 作用是啥 可以参考
https://www.freebuf.com/sectool/179035.html
(https://www.freebuf.com/sectool/179035.html)

修改 sqlmap-tamper - 获取数据

不得不感叹一句 sqlmap 真乃神器 已经写好的 tamper 功能基本就满足大多数场景了 在这里我的目标是在每一个字符前加上 %2527 在 sqlmap 自带的 tamper 中已经有一个叫 percentage.py 的，这个的功能是在每个字符前加 % 我只需要改成加 %2527 就行 非常简单 我们先看一下这个 tamper 大概长啥样



(https://xzfile.aliyuncs.com/media/upload/picture/20200726101551-ed351ca2-cee5-1.png)

框起来的部分是关键 sqlmap 把原来的 payload 每一个字符前加 % 后给 retVal 并且重新拼接起来 那么我只需要改一句

```
retVal += '%%2527%s' % payload[i]
```

就好

还有 = 号之前说过是会被检测的 只需要 把 = 号 换成 like 在最后加上

```
retVal = re.sub(r"\s*=\s*", " LIKE ", retVal)
```

最终的 tamper 如下

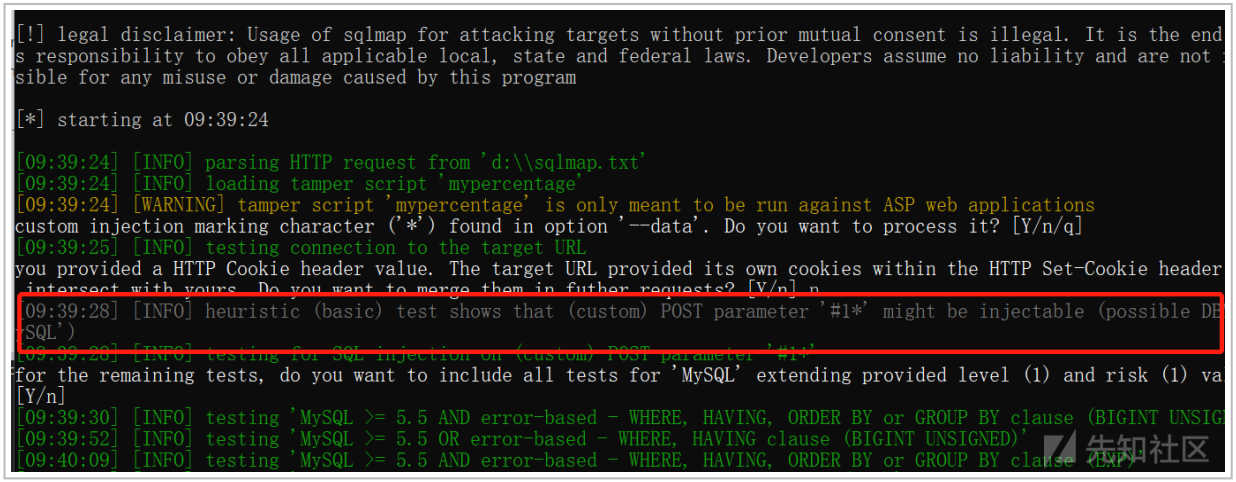
(https://xzfile.aliyuncs.com/media/upload/picture/20200726101602-f42ff496-cee5-1.png)

sqlmap 真乃神奇也 简直奥里给

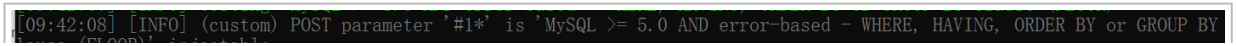
好了 现在就来一把梭

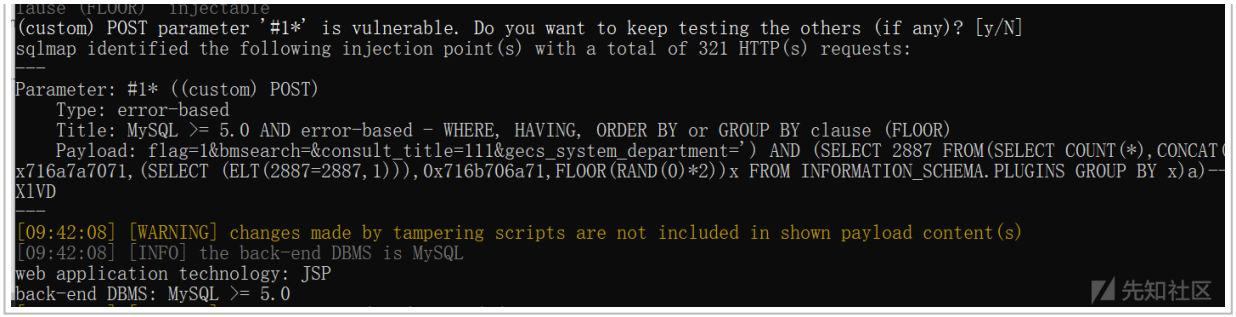
sqlmap 命令如下； python2 sqlmap.py -r "d:\sqlmap.txt" --dbms=mysql --technique=E --tamper mypercentage

我开始了 嘿嘿



(https://xzfile.aliyuncs.com/media/upload/picture/20200726101623-00480f0c-cee6-1.png)





(https://xzfile.aliyuncs.com/media/upload/picture/20200726101640-0a64ec44-cee6-1.png)