陇原战役 Misc题目WriteUP WP CTF竞赛

原创

青少年网络安全 于 2022-03-18 13:36:33 发布 148 收藏
分类专栏： CTF-WP 文章标签： p2p 网络协议 网络 网络安全

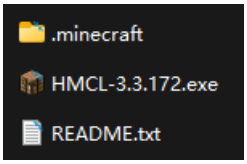
soEasyCheckin

base32，但有问题，倒数出现了0$，0–>O,$—>S，得到一串hex。 结果hex那里又有问题，具体是出在83¥6988ee这里 根据规律,每6个字节的第1个字节为e，然后把¥替换成e 得到社会主义核心价值观编码，但是还是有个地方是错误的，中间有一段为：和谐斃明平 然后把他那个斃随便改一下，我改的“富强” 解码得到的SET{Qi2Xin1Xie2Li4-Long3Yuan0Zhan4Yi4} 根据拼音，可以知道是Yuan2 所以最终flag为：

SET{Qi2Xin1Xie2Li4-Long3Yuan2Zhan4Yi4}

打败病毒

下载压缩包，解压后是HMCL和Minecraftn



查看MC核心，发现被修改过

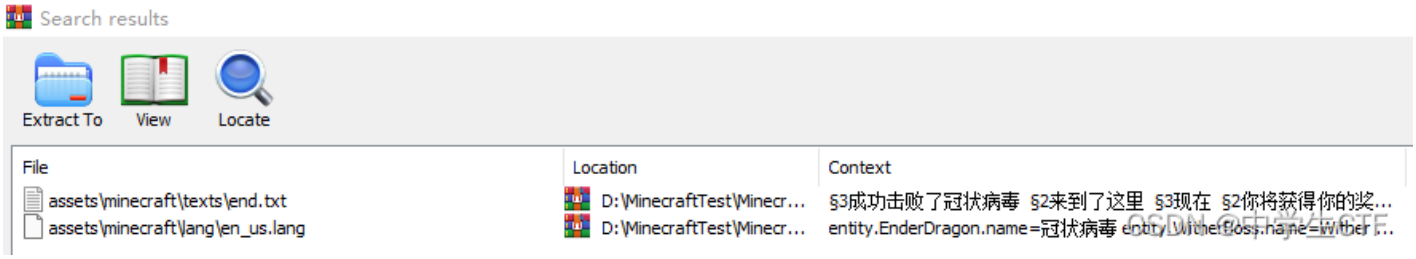
查看附带的文件，只有一个存档

猜测flag在存档或者核心里，创建帐户，启动游戏，进入存档

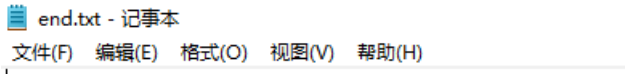

HMCL 3.3.172
账户
没有游戏账户
点击此处添加账户
游戏
陇原战"疫"
版本列表
游戏目录
当前目录
.minecraft
启动器
启动器设置
新建账户
登录方式
离线模式
用户名
f00001111
确定
取消
启动游戏
陇原战"疫"
CSDN @中学生CTF


Minecraft 1.12.2
MINECRAFT
JAVA EDITION
中国加油
Singleplayer
Multiplayer
Minecraft Realms
Options...
Quit Game
Minecraft 1.12.2/HMCL 3.3.172
Copyright Mojang AB. Do Not distribute!
CSDN @中学生CTF

可以看到要打败冠状病毒，使用作弊打败或在核心寻找冠状病毒



可以看到冠状病毒在游戏里是末影龙，flag在终末之诗



§3恭喜你

§2PLAYERNAME

§3成功击败了冠状病毒

§2来到了这里

§3现在

§2你将获得你的奖励

§3仔细看

§611F9sACbBBBWKTiClYDtNF2yIEfThXdfIGPxF

§3Hint:2

类似Base64串和提示可以猜测为Base62，放到CyberChef解码

Recipe

From Base62

Alphabet
0-9A-Za-z

Input

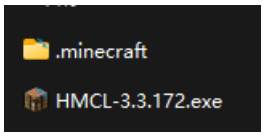
```
11F9sACbBBBWKTiClYDtNF2yIEfThXdfIGPxF
```

Output

```
SETCTF{Fi9ht1ng_3ItH_V1rUs}
```


SOS

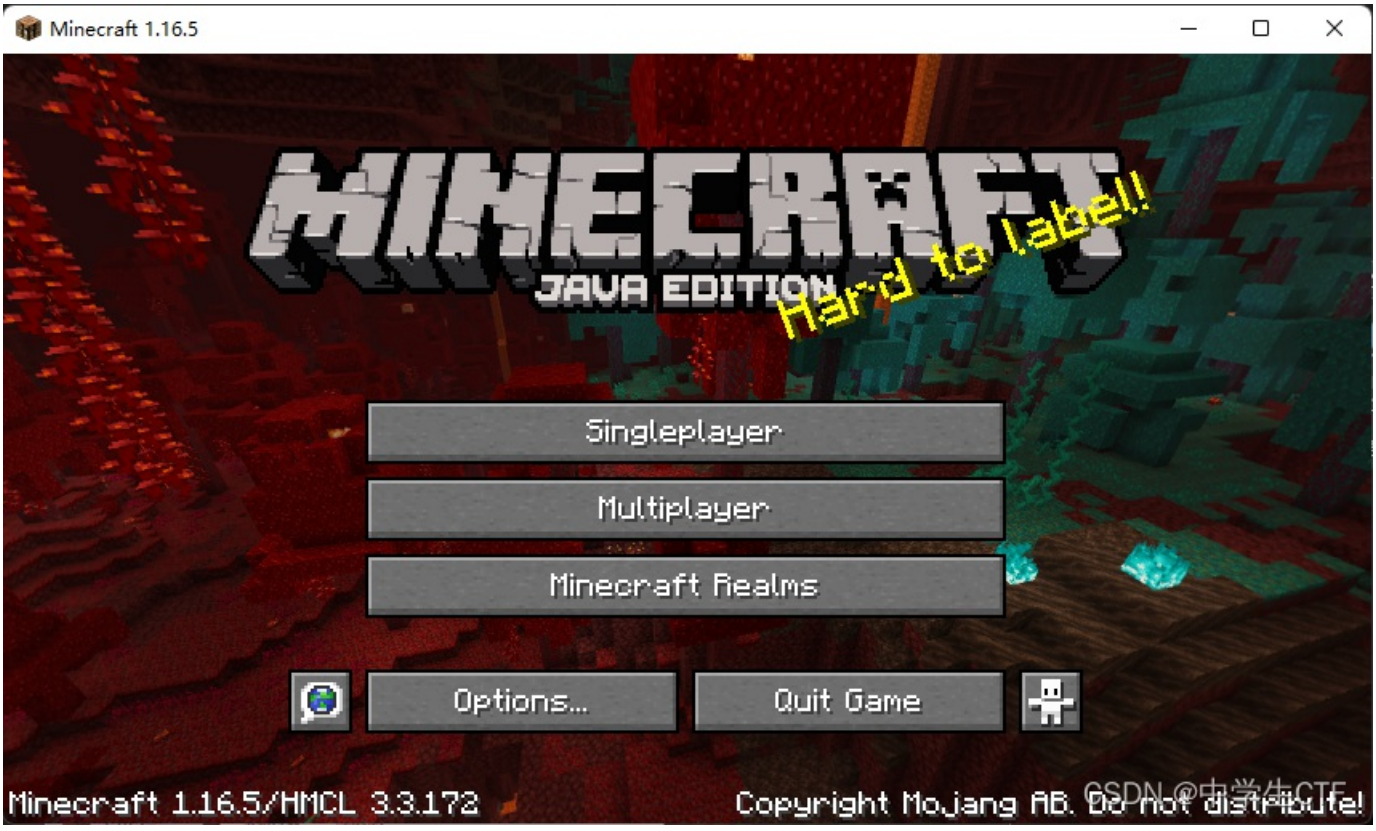
下载压缩包，解压后是HMCL和Minecraft

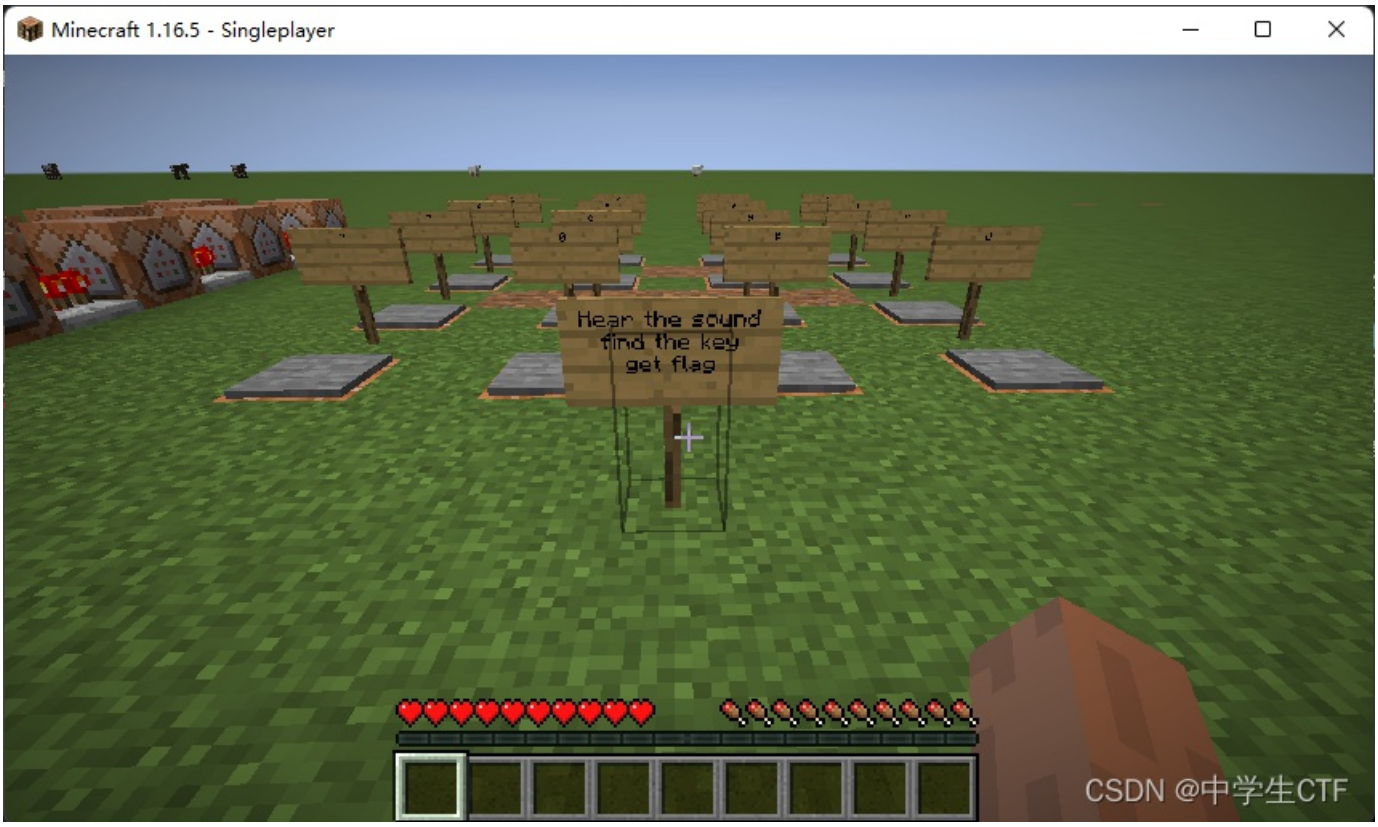


查看MC核心，没有被修改

查看附带的文件，有一个存档和一个资源包

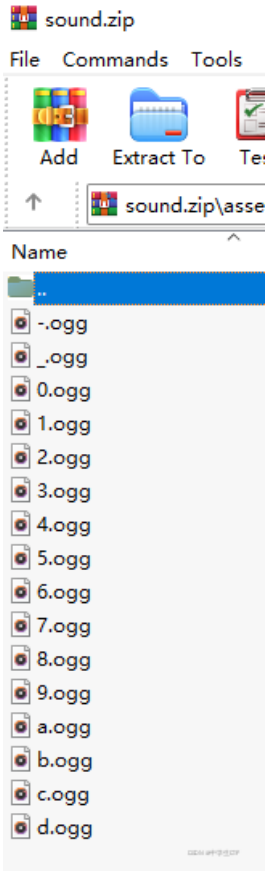
猜测flag在资源包或者存档里，创建帐户，启动游戏，进入存档



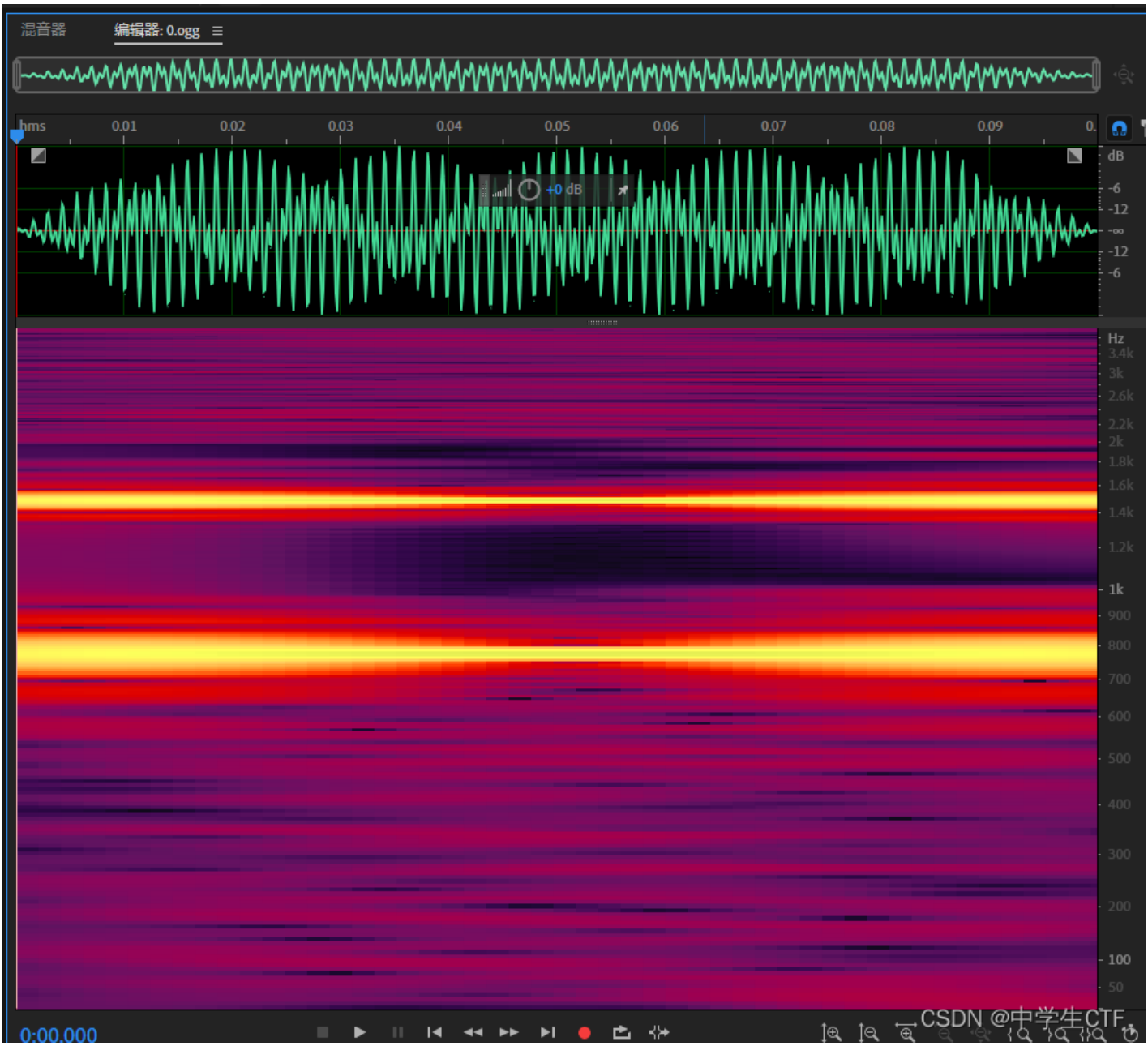


听到有拨号音，旁边红石电路在工作，调整速度

中继器调到3后可以听到清晰的拨号音，顺序查看命令方块可知为0-9，在资源包中找到0-9



使用Audition查看，得到DTMF音对应的字符

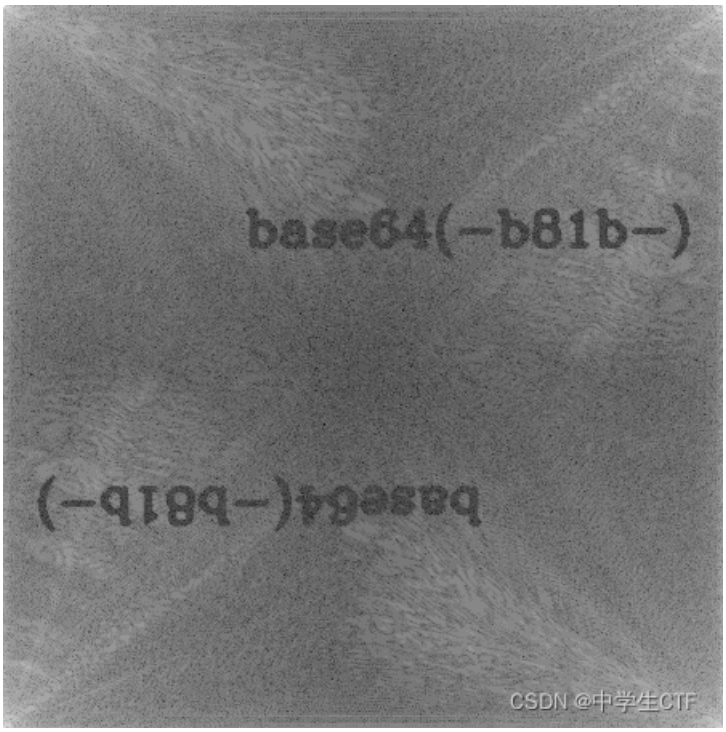


0-9解出来6830AB1C75，踩对应的字符得到flag

附赠原版DTMF资源包文件分享

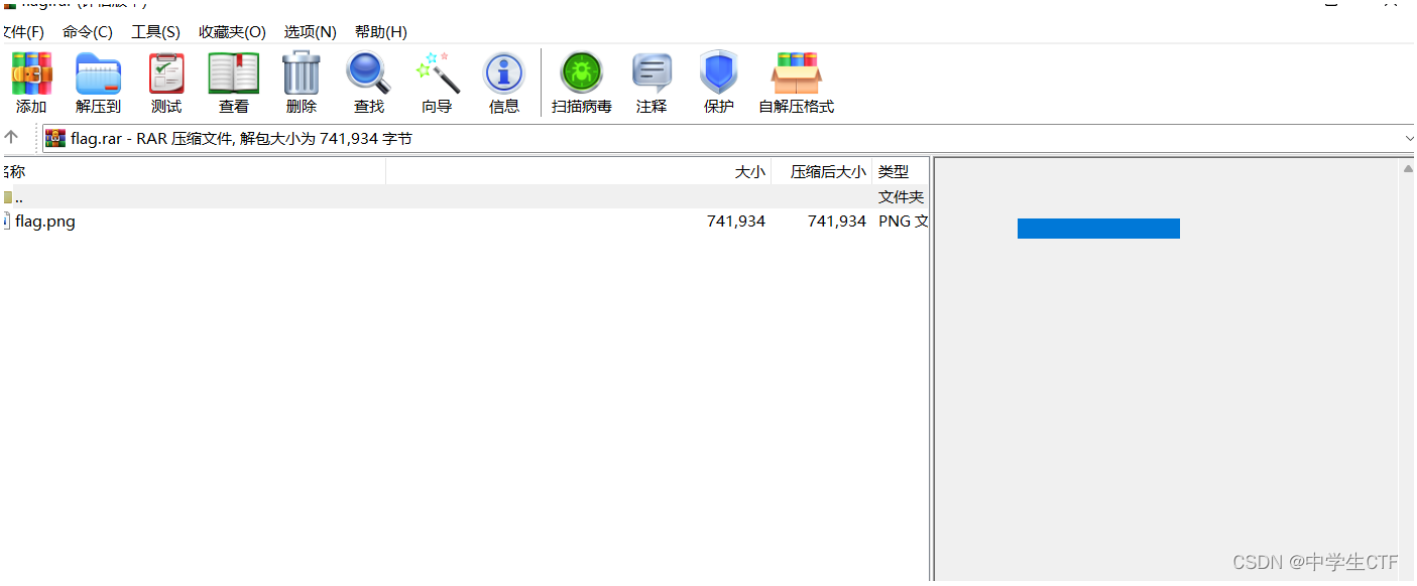
EasySteg

jk.png 盲水印提出：-b81b-



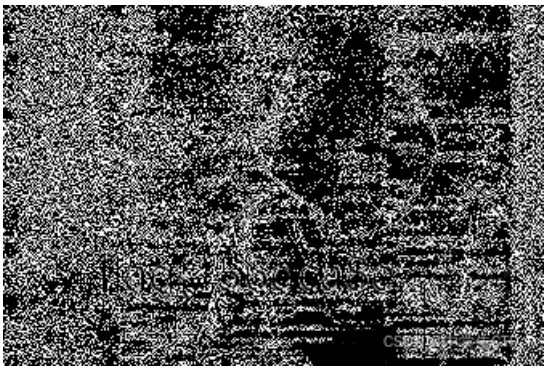
base64之后是LWI4MWIt

注释有东西



tab和空格转10

分出一张图片



oursecret解密，密码是LWI4MWIt

得到

n9S2B3I6U6L1O3R2F3Y1G2H3v9N2M1Z1D6T1h18o16u17b9s7r4g8f10a14m2i20p2e14w7q2y8P1J2E2C1V1

结合上面的01二进制，猜测是哈夫曼树

```
#!/usr/bin/env python

# -*- coding: utf-8 -*-

# 统计字符出现频率，生成映射表

def count_frequency(text):
    chars = []
    ret = []


    for char in text:
        if char in chars:
            continue
        else:
            chars.append(char)
            ret.append((char, text.count(char)))

    return ret



# 节点类

class Node:
    def __init__(self, frequency):
        self.left = None
        self.right = None
        self.father = None
        self.frequency = frequency

    def is_left(self):
        return self.father.left == self



# 创建叶子节点

def create_nodes(frequency_list):
```

```
    return [Node(frequency) for frequency in frequency_list]


# 创建Huffman树

def create_huffman_tree(nodes):
    queue = nodes[:]

    while len(queue) > 1:
        queue.sort(key=lambda item: item.frequency)
        node_left = queue.pop(0)
        node_right = queue.pop(0)
        node_father = Node(node_left.frequency + node_right.frequency)
        node_father.left = node_left
        node_father.right = node_right
        node_left.father = node_father
        node_right.father = node_father
        queue.append(node_father)

    queue[0].father = None
    return queue[0]



# Huffman编码

def huffman_encoding(nodes, root):
    huffman_code = [''] * len(nodes)


    for i in range(len(nodes)):
        node = nodes[i]
        while node != root:
            if node.is_left():
                huffman_code[i] = '0' + huffman_code[i]
            else:
                huffman_code[i] = '1' + huffman_code[i]
            node = node.father

    return huffman_code



# 编码整个字符串

def encode_str(text, char_frequency, codes):
    ret = ''
    for char in text:
        i = 0
        for item in char_frequency:
            if char == item[0]:
                ret += codes[i]
            i += 1

    return ret



# 解码整个字符串
```

```
def decode_str(huffman_str, char_frequency, codes):
    ret = ''
    while huffman_str != '':
        i = 0
        for item in codes:
            if item in huffman_str and huffman_str.index(item) == 0:
                ret += char_frequency[i][0]
                huffman_str = huffman_str[len(item):]
            i += 1

    return ret



if __name__ == '__main__':
    char_frequency=[('n', 9), ('S', 2), ('B', 3), ('I', 6), ('U', 6), ('L', 1), ('O', 3), ('R', 2), ('F', 3
    nodes = create_nodes([item[1] for item in char_frequency])
    root = create_huffman_tree(nodes)
    codes = huffman_encoding(nodes, root)
    huffman_str = '1110111111000100001111000011010001101111100110100011110111100010100111110111001101111100
    origin_str = decode_str(huffman_str, char_frequency, codes)
    print('Decode result:',origin_str)
```

得到：

The text to encode:nSBIULORFYGHvNMlOUZSDTNhnoubuosrgfbouvasmruiohauiopewgvfbwuivpqynqwUPlFJDDBUEDU

16进制转字符串：

=F0=9F=98=A9=F0=9F=92=8C=F0=9F=98=8F=F0=9F=93=93=F0=9F=90=9E=F0=9F=8E=A3=F0==9F=97=

是Quoted-printable编码

解码得到：

□□□□□□□□□□□□□□□□□□□□□□□□□□□

尝试一番之后，可用emoji-cracker解决

https://github.com/pavelvodrazka/ctf-writeups/tree/d957cab439f796304bdb005e45b333cd83203c04/hackyeaster2018/challenges/egg17/files/cracke

-> 4157-9f39-4c41f4a4facb}

原图像是分数小波变换处理后的，算法很简单，简单处理一下

```
import cv2
import numpy as np
from pywt import dwt2, idwt2

D = cv2.imread('text_d.png',0)

zeros = np.ones(shape = H.shape)
rimg = idwt2((zeros,(D,D,D)), 'haar')
cv2.imwrite("output.png",np.uint8(rimg))
```

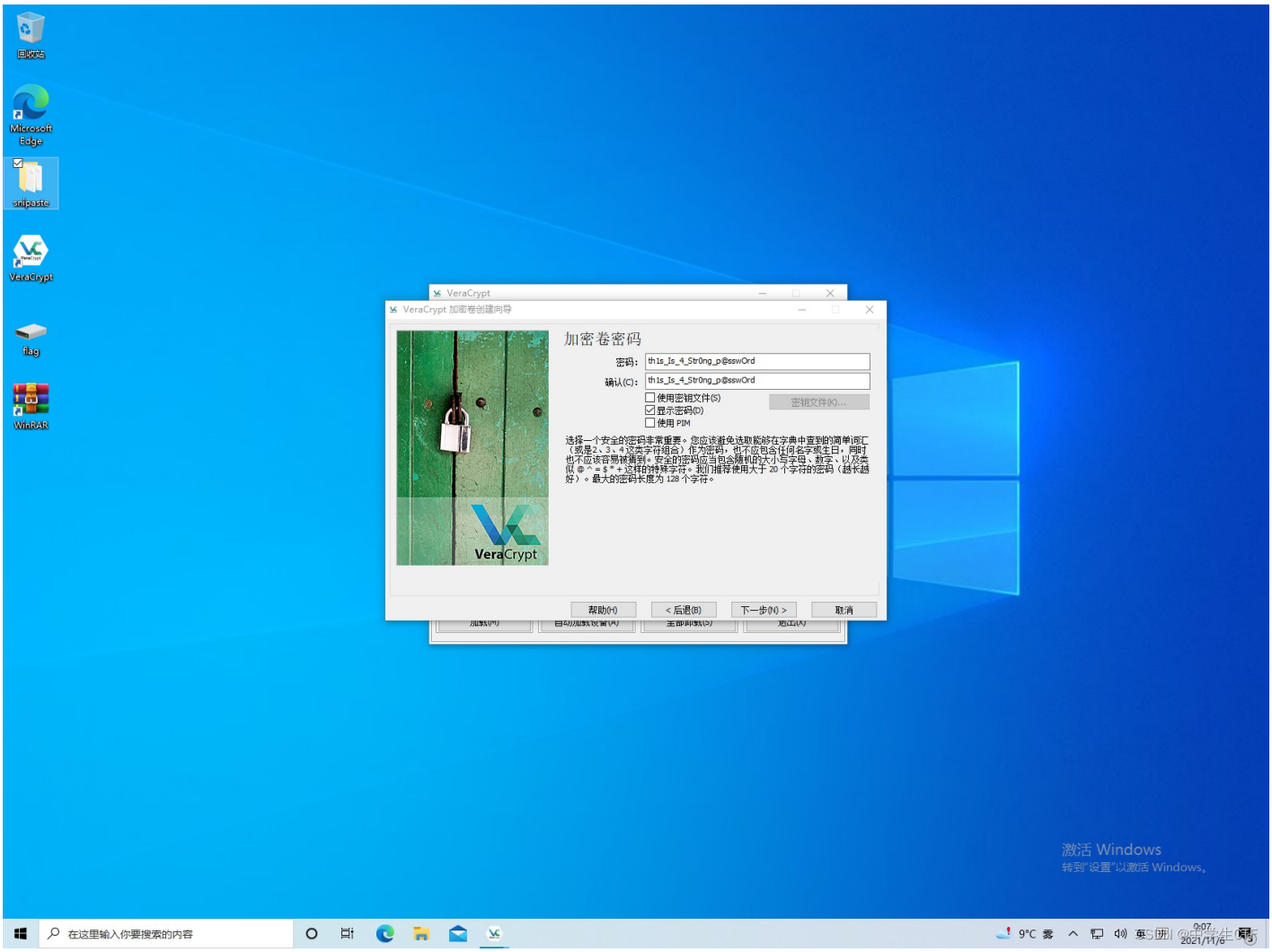
得到flag{156cca8e

flag{156cca8e-b81b-4157-9f39-4c41f4a4facb}

checkin

part 1

根据题目描述，关注到截图软件 `snipaste`，从其历史截图的缓存中发现 `veracrypt` 的密码：



part 2

将桌面上的 `flag.vhd` 挂载为本地磁盘后，解密得到 `task.py` `cip.png` 以及 `flag.zip`，查看 `flag.zip` 注释可以发现提示 `who is invited?`，那么我们解密 cip.png 即可得到邀请函，exp如下：

```
# from PIL import Image
# import sys
#
# # if len(sys.argv) != 3:
# #     print("Usage: %s [infile] [outfile]" % sys.argv[0])
# #     sys.exit(1)
#
# image = Image.open("cip.png").convert("F")
# width, height = image.size
# result = Image.new("F", (width, height))
#
# ROUNDS = 32
#
# f = open('tuples.csv', 'w')
# for i in range(width):
#     for j in range(height):
#         value = 0
#         di, dj = 1337, 42
#         for k in range(ROUNDS):
#             di, dj = (di * di + dj) % width, (dj * dj + di) % height
#             f.write('%d,%d\n' % (di,dj))
#             value += image.getpixel(((i + di) % width, (j + dj + (i + di)//width) % height))
#         result.putpixel((i, j), value / ROUNDS)
#
# f.close()
# result = result.convert("RGB")
# result.save("cip2.png")

from PIL import Image

image = Image.open("cip.png").convert("F")
width, height = image.size
result = Image.new("F", (width, height))

ROUNDS = 32

lines = iter(open('tuples.csv', 'r').readlines()[::-1])
for i in range(width):
    for j in range(height):
        value = 0
        for k in range(ROUNDS):
            line = next(lines)
            di = int(line.split(",")[0])
            dj = int(line.split(",")[1])
            value += image.getpixel(((i - di) % width, (j - dj + (i - di)//width) % height))
        result.putpixel((i, j), value / ROUNDS)

result = result.convert("RGB")
result.show()
result.save("out.png")
```

即可得到邀请函中的名字是 `every ctfer`

part 3

`flag.data` 是 /dev/input/event* 中的键盘记录，这一点已经在比赛当中的hint给出，解析即可得到flag，exp如下：

```
#!/usr/bin/python
import struct
import time
import sys

infile_path = "./flag.data"

FORMAT = 'llHHI'
EVENT_SIZE = struct.calcsize(FORMAT)

#open file in binary mode
in_file = open(infile_path, "rb")

event = in_file.read(EVENT_SIZE)

keys = {
        "0" : "[NULL]",
        "1" : "[ESC]",
        "2" : "1",
        "3" : "2",
        "4" : "3",
        "5" : "4",
        "6" : "5",
        "7" : "6",
        "8" : "7",
        "9" : "8",
        "10" : "9",
        "11" : "0",
        "12" : "-",
        "13" : "=",
        "14" : "[BACKSPACE]",
        "15" : "[TAB]",
        "16" : "q",
        "17" : "w",
        "18" : "e",
        "19" : "r",
        "20" : "t",
        "21" : "y",
        "22" : "u",
        "23" : "i",
        "24" : "o",
        "25" : "p",
        "26" : "[",
        "27" : "]",
        "28" : "[ENTER]",
        "29" : "[LEFT_CTRL]",
        "30" : "a",
        "31" : "s",
        "32" : "d",
        "33" : "f",
        "34" : "g",
        "35" : "h",
        "36" : "j",
        "37" : "k",
        "38" : "l",
        "39" : ";",
        "40" : "'",
        "41" : "`",
        "42" : "[LEFT_SHIFT]",
```

```
        "43" : "\\",
        "44" : "z",
        "45" : "x",
        "46" : "c",
        "47" : "v",
        "48" : "b",
        "49" : "n",
        "50" : "m",
        "51" : ",",
        "52" : ".",
        "53" : "/",
        "54" : "[RIGHT_SHIFT]",
        "55" : "[GRAY*]",
        "56" : "[LEFT_ALT]",
        "57" : "[SPACE]",
        "58" : "[CAPS]",
        "59" : "[F1]",
        "60" : "[F2]",
        "61" : "[F3]",
        "62" : "[F4]",
        "63" : "[F5]",
        "64" : "[F6]",
        "65" : "[F7]",
        "66" : "[F8]",
        "67" : "[F9]",
        "68" : "[F10]",
        "69" : "[NUM_LOCK]",
        "70" : "[SCROLL_LOCK]",
        "71" : "[PAD_7]",
        "72" : "[PAD_8]",
        "73" : "[PAD_9]",
        "74" : "[GRAY_MINUS]",
        "75" : "[PAD_4]",
        "76" : "[PAD_5]",
        "77" : "[PAD_6]",
        "78" : "[GRAY_PLUS]",
        "79" : "[PAD_3]",
        "80" : "[PAD_2]",
        "81" : "[PAD_1]",
        "82" : "[PAD_0/INS]",
        "83" : "[PAD_./DEL]",
        "84" : "[PRTSCR/SYSRQ]",
        "87" : "[F11]",
        "88" : "[F12]",
        "90" : "PAUSE/BREAK",
        "91" : "[INSERT]",
        "92" : "[HOME]",
        "93" : "[PG_UP]",
        "94" : "[GRAY/]",
        "95" : "[DELETE]",
        "96" : "[END]",
        "97" : "[PG_DN]",
        "98" : "[RIGHT_ALT]",
        "99" : "[RIGHT_CTRL]",
        "100" : "[UP_ARROW]",
        "101" : "[LEFT_ARROW]",
        "102" : "[DOWN_ARROW]",
        "103" : "[RIGHT_ARROW]",
        "104" : "[GRAY_ENTER]",
```

```
        "105" : "[MOUSE]",
        "183" : "[PRTSC]"
        }
count = 0
while event:
    count += 1
    (tv_sec, tv_usec, type, code, value) = struct.unpack(FORMAT, event)

    if type != 0 or code != 0 or value != 0:
        # print("Event type %u, code %u, value %u at %d.%d" % \
        #     (type, code, value, tv_sec, tv_usec))
        if count == 3:
            print(count, end="")
            print(" ", end="")
            print(keys[str(code)], end="")
    else:
        # Events with code, type and value == 0 are "separator" events
        count = 0
        print()

    event = in_file.read(EVENT_SIZE)

in_file.close()
```

SET{Wow_Y0u_c4n_r3A11y_Forensic}