2021振兴杯参赛后感（部分writeup）

原创

Jelly-fish 于 2021-10-20 14:54:11 发布 66 收藏

分类专栏： 赛题 文章标签： 算法 unctf wp

目录

写在前面

距离参加振兴杯已经将近一周了，各种奖项也都该出来了，最近刚好有时间回忆一下振兴杯，顺便把WP写一下加深一下经验，若有什么不对的地方还请各位大佬指正。

题目

振兴杯B模块的题目都是有关于CTF的题目，主要是包括杂项和密码，可能是因为比赛时长的限制，主办方也没有出一些逆向和PWN之类的CTF题目。
此次CTF一共包括十几道CTF题目，包括中国文化、核心价值观、快乐的回收站、美丽的风景、怪异的信息、easy-rsa、二维码等题目。

中国文化

打开题目附件后发现里面就一张ASCII码表和一份TXT文本，我们打开文本发现是一些数字（-6、-10、38、42、42、40、37、-3、38、41、-4、39、41、-10、-5、-12），仔细寻找可能的提示，发现txt文件命名中包含着题目提示（hack很喜欢中国文化并请了一个大师帮他算了一下年龄，大师给了他封信信封表面写着**甲子**二字，信的内容却是一串阿拉伯数字）
众所周知，一甲子代表**60年**因此我们将文本中每个数字统一相加60，之后与ACCII表对照得出flag

核心价值观

打开附件，发现里面拥有一份未知格式的文档和一份文本，我们先打开文本发现里面明显是经过核心价值观密码加密的密文

（富强富强富强富强富强公正富强公正富强富强富强富强富强公正友善民主富强富强富强富强富强公正富强民主富强富强富强富强富强公正富强法治富强富强富强富强富强法治友善富强富强富强富强富强富强公正富强文明富强富强富强富强富强公正富强民主富强富强富强富强富强公正富强文明富强富强富强富强富强和谐富强富强富强富强富强富强和谐富强和谐富强富强富强富强和谐富强法治富强富强富强富强富强和谐富强文明富强富强富强富强富强和谐富强爱国富强富强富强富强富强公正富强平等富强富强富强富强富强公正富强和谐富强富强富强富强富强和谐富强文明富强富强富强富强富强和谐富强民主富强富强富强富强富强和谐富强文明富强富强富强富强富强公正富强民主富强富强富强富强富强和谐富强法治富强富强富强富强富强和谐富强文明富强富强富强富强富强法治友善文明）

根据附件内容提示，另一份就是加密脚本，我们将他打开尝试进行解密：

```
def encdoe(string):
    len_str = len(string)
    if len_str % 16 != 0:
        return 0
    result = ''
    for x in range(0, len_str, 16):
        encode_char = string[x:x+16]
        temp_int = [ENSTRS.index(encode_char[y:y+2]) for y in range(0, 16, 2)]
        int_list = [temp_int[x]+temp_int[x+1] for x in range(0, 8, 2)]
        bin_temp = [bin(i).replace('0b', '') for i in int_list]
        binstr_list = []
        for b in bin_temp:
            if len(b) < 4:
                binstr_list.append(b.zfill(4))
            else:
                binstr_list.append(b)
        binstr = ''.join(binstr_list)
        result = result + chr(int(binstr, 2))
    return result
```

经过分析后我们发现这是一份残缺的解密脚本，主办方将其删除并修改了一部分，仔细分析脚本的话很快就能看出来的。
我们开始进行脚本的复原和补充得到新的解密脚本：

```
def decoder(string):
    len_str = len(string)
    if len_str % 16 != 0:
        return 0
    result = ''
    for x in range(0, len_str, 16):
        decode_char = string[x:x + 16]
        temp_int = [ENSTRS.index(decode_char[y:y + 2]) for y in range(0, 16, 2)]
        int_list = [temp_int[x] + temp_int[x + 1] for x in range(0, 8, 2)]
        bin_temp = [bin(i).replace('0b', '') for i in int_list]
        binstr_list = []
        for b in bin_temp:
            if len(b) < 4:
                binstr_list.append(b.zfill(4))
            else:
                binstr_list.append(b)
        binstr = ''.join(binstr_list)
        result = result + chr(int(binstr, 2))
    return result


ENSTRS = ("富强", "民主", "文明", "和谐", "自由", "平等",
          "公正", "法治", "爱国", "敬业", "诚信", "友善")

while True:
    decode_str = input("请输入解密字符串：\n")
    result = decoder(decode_str)
    print("解密结果:\n{}".format(result))
```

运行解密脚本之后得到flag。

怪异的信息

打开一条怪异的信息文本里面存放着未知的数字和字符串，共有两排，怀疑是密码题，仔细分析我们怀疑是伪栅栏加密：

```
342516
agl{fbdc4c645ed20bc}7@3@
```

我们将其按照每六个一排进行分组

```
342516
agl{fb
dc4c64
5ed20b
c}7@3@
```

然后将每一列按照第一排的数字进行顺序排放。

```
123456
flag{b
64dcc4
0d5e2b
37c}@@
```

最后将字符合为一排即得到flag

Easy-RSA

打开文档，查看发现RSA加密：

```
c = 32777590618821256240188457883196017403261423525673816299491555872691 9
n = 54418730685090279762910735361926742769483716360085398324278753236512 3
e = 65537
m = ???
```

题目中已经给出C、N、E因此我们需要根据这些有限的条件进行解密明文。
我们首先进行分离大质数N

```
I:\振兴杯\B模块工具\必备\yafu-1.34
λ .\yafu-x64.exe "factor(54418730685090279762910735361926742769483716360085398324278753236512 3)"
fac: factoring 54418730685090279762910735361926742769483716360085398324278753236512 3
fac: using pretesting plan: normal
fac: no tune info: using qs/gnfs crossover of 95 digits
div: primes less than 10000
fmt: 1000000 iterations
rho: x^2 + 3, starting 1000 iterations on C65
rho: x^2 + 2, starting 1000 iterations on C65
rho: x^2 + 2, starting 1000 iterations on C59
rho: x^2 + 1, starting 1000 iterations on C59
pm1: starting B1 = 150K, B2 = gmp-ecm default on C59
Total factoring time = 0.0983 seconds


***factors found***

P4 = 6343
P7 = 3125921
P11 = 18787605191
P49 = 14608454525177809798886956058979666100474450760 51

ans = 1
```


对n进行质因数分解，得到了4个质因数，根据欧拉公式构建Python代码
φ(x * y * zc) = φ(x) * φ(y) * φ(z) φ©=(x-1)(y-1)(z-1)(c-1)

```
import gmpy2
from Crypto.Util.number import long_to_bytes

c = 32777590618821256240188457883196017403261423525673816299491555872691 9
n = 54418730685090279762910735361926742769483716360085398324278753236512 3
e = 65537
p1 = 6343
p2 = 3125921
p3 = 18787605191
p4 = 14608454525177809798886956058979666100474450760 51

phi = (p1 - 1) * (p2 - 1) * (p3 - 1) * (p4 - 1)
d = gmpy2.invert(e, phi)
m = pow(c, d, n)
print(long_to_bytes(m))
```

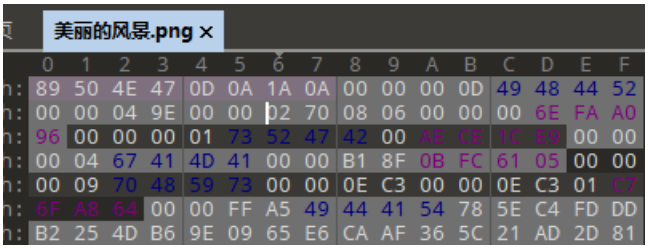
运行脚本解出flag。

二维码

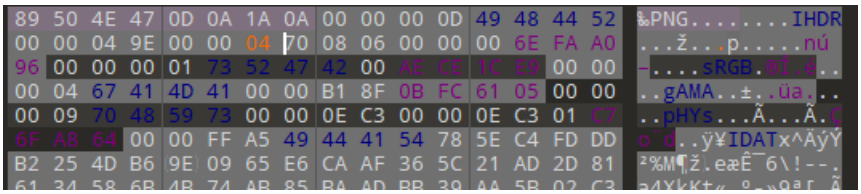
二维码这种类型的题目大概率是主办方用来给选手保分的，多数为CTF第一道题目，作为签到题，因此我们只需要利用主办方所给的QR-Search进行扫描即可得到flag

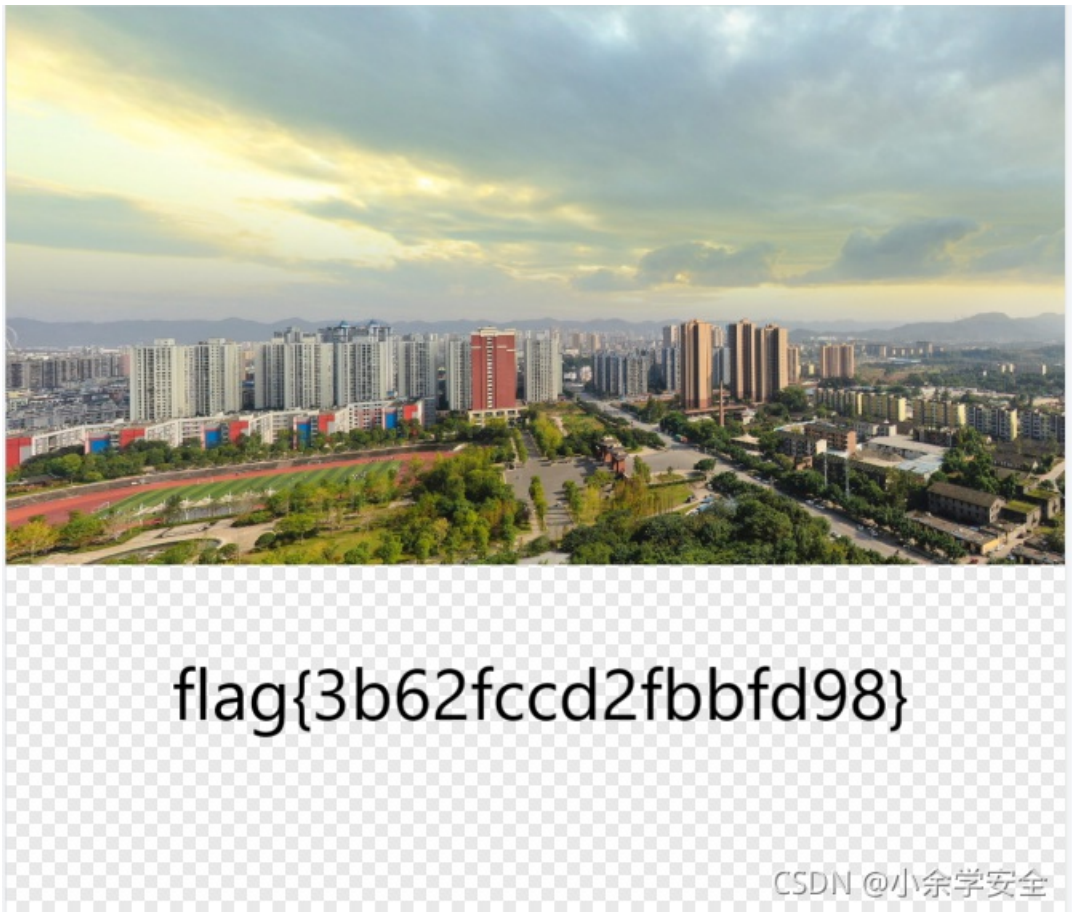
美丽的风景

打开附件是一张风景图片，根据我们的做题经验，看到图片的第一时间想到图片隐写的几种套路，无非就是修改图片的行高，宽高，伪加密，伪造文件头，lsb加密之类的图片隐写术。我们利用主办方所给的010打开图片进行分析，发现是正常的png图片头



此时我们尝试修改行高。

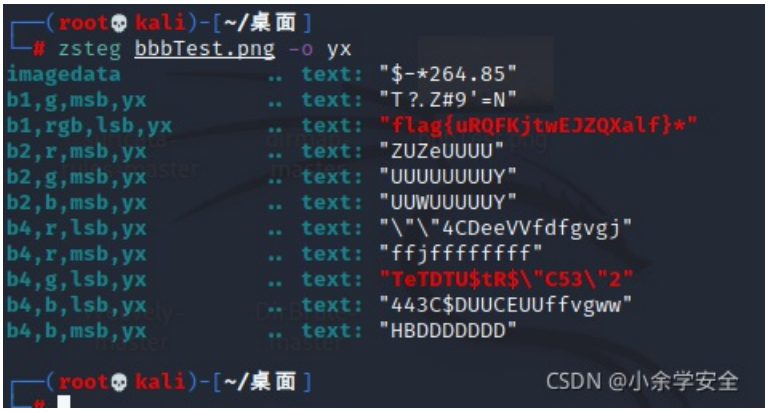




成功得到flag

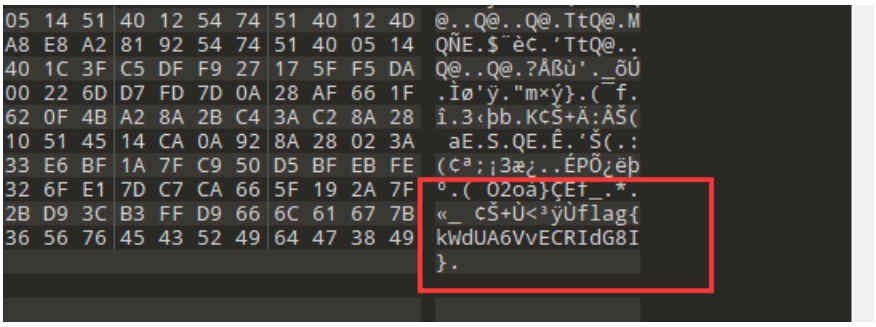
狐狸牧羊

再次看到图片类型CTF题目，我们联想到常规图片隐写术，将行高，文件头等类型分析过后发现不是这些类型的图片隐写。我们分析是否存在lsb图片隐写，查看主办方所给工具，进入kali中寻找工具，发现一个zsteg脚本工具，我们尝试利用它进行解密，得到flag



皮卡丘

同样的图片隐写，图样的010神器，我们利用010打开图片直接搜索flag发现在图片尾部存在flag。



从前有个鹅

打开附件，发现有个压缩包

从前有个鹅.zip

对压缩包进行解压

bTU.zip
嘘.txt

发现有两个文件，txt文件里面没有什么重要的重要的信息，继续解压压缩包

zV3h.zip


很多弯弯绕绕要细心.jpg

PMjV.zip


遇到困难不要怕.jpg

vRVQ.zip


要学会坚持到最后.jpg

1V0R.zip


我会嘎嘎嘎.jpg

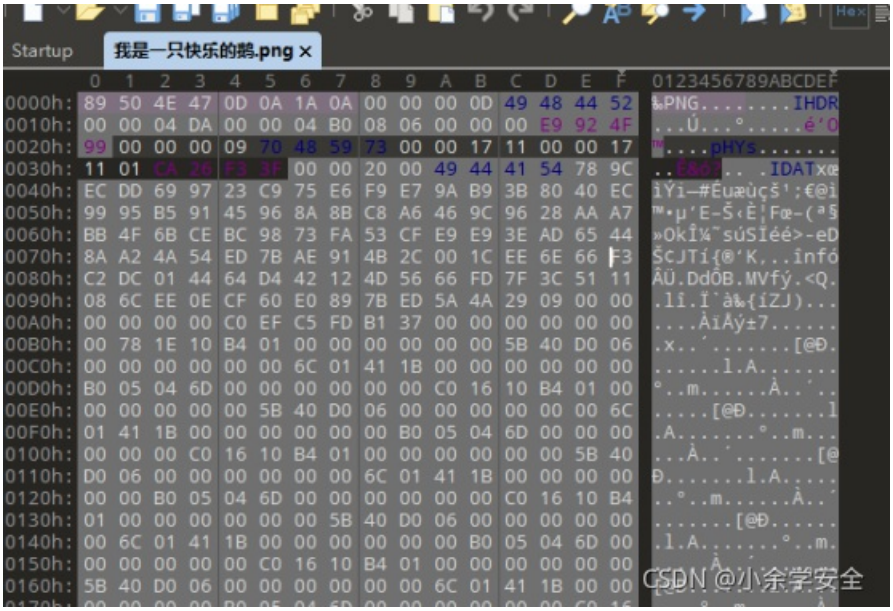
解压到最后发现一个需要解压密码的压缩文件，以及可以看到之前的压缩包与这个压缩包命名方式的不一样，对此我们将前面压缩包的名字拼接在一起可以得到一个base64值

bTUzV3hPMjVvRVQ1V0RyZzl=

对base64值解密可得到一串字符

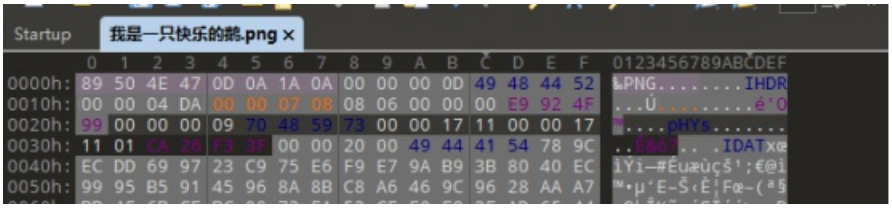
字符就是解开“发现了个大宝贝.7z”压缩包的密码，解开压缩包后可以获得一个被分割的二维码，由此得知我们还需要找到其它的二维码进行拼接

我们对当前文件进行分析，发现最后一张图片“我是一只快乐的鹅.png”与其它图片的二进制文件有差异，并且能够看到有其它文件的数据头



修改后缀后可以打开发现里面还有两个压缩文件

尝试解压发现文件需要密码，解压密码就在文件附加信息之中，解压后得到两个压缩包，第一个压缩包解开后，发现里面有一个flag.txt可惜文本里面没有关键信息，第二个压缩包解压需要密码。重新整理思路，我们发现之前每一个压缩包解开就有一张图片，我们把所有图片放在一起进行对比，而对比之前的图片，我们发现这个图片还与其它图片高度不同，由此怀疑图片被修改了高度以此来隐藏信息，修改高度后可以发现一串数字，由之前压缩包的线索可知，这是压缩包的解压密码。



解开压缩包后，我们可以得到新的二维码切割图片

奇怪这是什么.7z

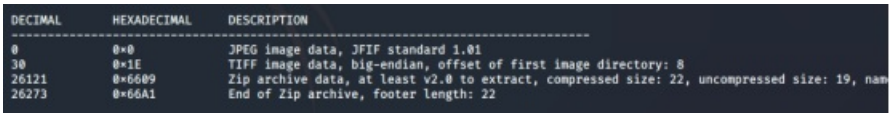

奇怪这是什么.jpg

目前已经得到了，两块二维码的切割图片，还差一块二维码切割图片，我们就能构造二维码图片了


发现了个大宝贝.jpg


奇怪这是什么.jpg

目前唯一没有处理的就是“发现了压缩包诶.zip”以及里面的“flag.txt”文件，我们重新将两个文件进行分析，当我们对“发现了压缩包诶.zip”进行隐写分析后，可以发现里面有隐藏文件

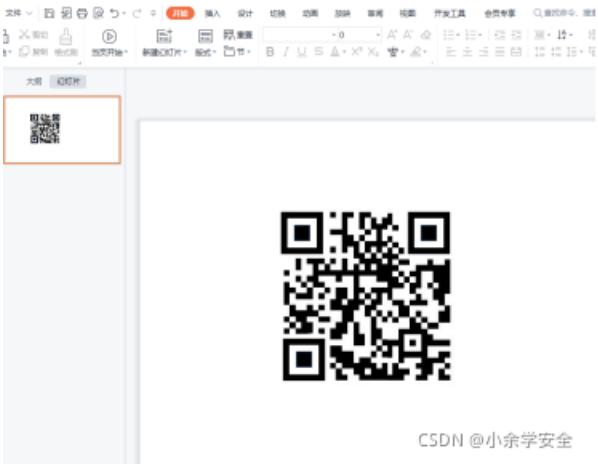


```
DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             JPEG image data, JFIF standard 1.01
30            0x1E            TIFF image data, big-endian, offset of first image directory: 8
26121         0x6609          Zip archive data, at least v2.0 to extract, compressed size: 22, uncompressed size: 19, nam
26273         0x66A1          End of Zip archive, footer length: 22
```

我们将文件进行分离出来，分离后就可以得到最后一块被切割的二维码

1.jpeg

将二维码拼接好，拼接好后，发现缺少定位点，我们手动制作定位点。



拼接好二维码的定位点后我们用软件扫描二维码，就得到了flag。

写在最后

关于振兴杯B模块的CTF题目的wp暂时就写道这里，下次有机会在做补充，还请各位大佬指正。