[CTF] 攻防世界MISC高手区部分题目WriteUp（2）

原创

berdb 于 2020-12-13 19:51:15 发布 2126 收藏

分类专栏： 笔记

本文链接：https://blog.csdn.net/qq_37969220/article/details/110850757

笔记 专栏收录该内容

3 篇文章 0 订阅

订阅专栏

记录一些有意思的题目

目录索引

- 7-2
- ewm
- Mysterious-GIF
- crc
- 4433
- challenge_how_many_Vigenère
- 流量分析
- 未完待续...

7-2

下载解压得到一个文件夹，里面有一堆文件，对所有文件名进行 base64 解码

```
# coding=utf-8
import os
import base64

for name in os.listdir("/root/桌面/problem"):
    print name
    # 由于文件名不是4的倍数，需要在后面补‘=’
    missing_padding = 4 - len(name) % 4
    if missing_padding:
        name += '=' * missing_padding
    print str(base64.b64decode(name.encode()))
```

看到输出的信息里只有这个文件解码后不是乱码：**YWluaWRleGluZ3podWFuZw**

```
ainidexingzhuang
YWluaWRleGluZ3podWFuZw
YWPpG3JEmRbOnXx00Xifd7
```

打开文件后看到里面存在一对花括号，那么很有可能这些数字解码后就是flag

> 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 74 94 22 42 91 23 {82 42 82 52 63 21 42 22 73 21 }00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

这次使用的是9键加密，第一个数字代表9键键盘下的数字，第二个数字代表该数字下的第几个字母。解密后再用一次凯撒加密即可得到 flag

> wdflag{ylyoselfve}

ewm

下载解压后得到一个文件夹，打开看到里面是一堆二维码的部分图块，按大小被分为了 small 和 big 两部分，总共两个二维码，可以把图片拖到 word 里拼图，拼完的二维码扫出来就是 flag，而且两组图片拼图的结果都是一样的，这里给一个拼图的脚本（并不能完美的拼出来，但已经可以扫出来了）

```
import os
from PIL import Image
# 由于二维码并没有被完整的切开，都保留了一部分边缘
# 所以可以遍历所有图片来找到两个边缘符合的图片进行拼图

# 设置已知的图片
def set_image(in_file, x, y):
    code[y][x] = in_file
    file.remove(in_file)

# 找到一张图片的右边一张
def find_right(in_file):
    result = ''
    pixes = Image.open(os.path.join(current_path, in_file)).load()
    # 计算出已知图片的右侧像素值（以二进制存储）
    pix = 0
    for row in range(width):
        r, g, b = pixes[width-1, row]
        pix <<= 1
        if r + g + b > 255:
            pix += 1
    # 用同样的方法在剩下的图片里找到与已知图片边缘相符的图片
    for fi in file:
        pixes = Image.open(os.path.join(current_path, fi)).load()
        pix2 = 0
        for row in range(width):
            r, g, b = pixes[0, row]
            pix2 <<= 1
            if r + g + b > 255:
                pix2 += 1
        if pix == pix2:
            result = fi
            file.remove(fi)
            break
    return result

# 找到一张图片的下边一张
def find_bottom(in_file):
    result = ''
    pixes = Image.open(os.path.join(current_path, in_file)).load()
    pix = 0
    for col in range(width):
```

```
        r, g, b = pixes[col, width-1]
        pix <<= 1
        if r+g+b > 255:
            pix += 1
    for fi in file:
        pixes = Image.open(os.path.join(current_path, fi)).load()
        pix2 = 0
        for col in range(width):
            r, g, b = pixes[col, 0]
            pix2 <<= 1
            if r + g + b > 255:
                pix2 += 1
        if pix == pix2:
            result = fi
            file.remove(fi)
            break
    return result


if __name__ == '__main__':
 # 读取所有拼图文件
    current_path = r"C:\Users\28919\Desktop\big"  # r"C:\Users\28919\Desktop\small"
    file = [name for name in os.listdir(current_path)]
 # 计算拼图的大小
    n = int(len(file) ** 0.5)
 # 创建储存图片的数组
    code = [['' for i in range(n)] for i in range(n)]
 # 设置单张图片的边长
    width = 51
    # width = 42

    fail = False
 # 给出3个定位点，让脚本根据已知的图片进行拼图
    set_image('ebb9e03faca4_big.jpg', 0, 0)
    set_image('be557e464b98_big.jpg', 4, 0)
    set_image('13d9bb15c1c5_big.jpg', 0, 4)
 # 由于存在多张边缘相似的图片，对small进行拼图的时候需要多给一个数据
 # 但还是不能完美拼出来，但是拼出的结果可以扫出flag
    # set_image('f56e68b804ae_small.jpg', 0, 0)
    # set_image('dee457f4f9e0_small.jpg', 5, 0)
    # set_image('8e92d38c074c_small.jpg', 0, 5)
    # set_image('bb57a891b8c1_small.jpg', 0, 1)

    for y in range(n):
        if fail:
            break
        for x in range(n):
         # 对未知的图片进行查找
            if code[y][x] == '':
                print('Find%d,%d' % (x, y))
                if x == 0:
                 # 如果是最左边的图片需要通过上方的图片来拼图
                    code[y][0] = find_bottom(code[y-1][0])
                else:
                 # 通过左边的图片进行拼图
                    code[y][x] = find_right(code[y][x-1])
                # 找不到就直接退出
                if code[y][x] == '':
                    fail = True
```

```
                    break
                else:
                    for i in code:
                        print(i)
    if fail:
        print('拼图失败!')
# 根据已找到的图片拼出最终的图片
    image = Image.new('RGB', (n*width, n*width), (255, 255, 255))
    for y in range(n):
        for x in range(n):
            if code[y][x] != '':
                p = Image.open(os.path.join(current_path, code[y][x]))
                image.paste(p, (width * x, width * y))
    image.save(r'C:\Users\28919\Desktop\flag.png')
```

脚本拼图的结果：

> flag{g00d_g00d_study_1jf8988}

Mysterious-GIF

下载得一个 gif 文件，分帧看看不到什么异常，丢到 kali 里用 binwalk 跑一遍

```
root@kali:~/桌面# binwalk out.gif

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0×0             GIF image data, version "89a", 440 x 608
2670386       0×28BF32        Zip archive data, at least v1.0 to extract, compres
2783320       0×2A7858        End of Zip archive, footer length: 22
2783420       0×2A78BC        End of Zip archive, footer length: 22
```

发现有一个 zip 文件，用 foremost 命令分离出来，得到一个 temp.zip 文件

这压缩包里就一个256字节的文件却占用112K的空间，明显不对劲，再用 binwalk 命令跑一下

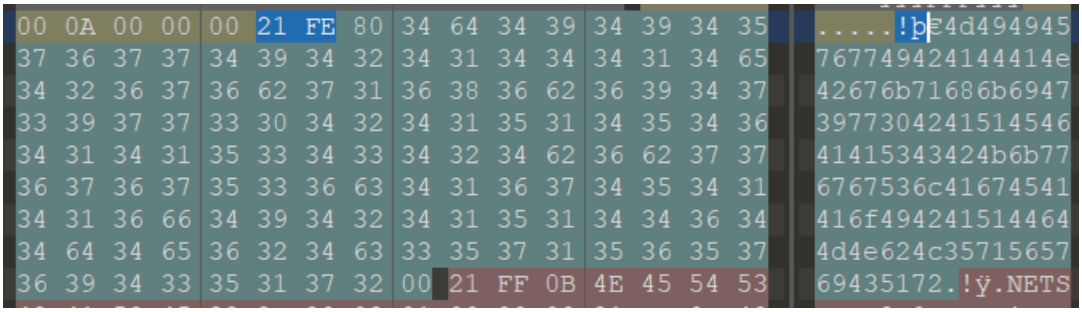
```
108182        0×1A696         End of Zip archive, footer length: 22
108204        0×1A6AC         Zip archive data, at least v1.0 to extract, comp
108608        0×1A840         End of Zip archive, footer length: 22
108630        0×1A856         Zip archive data, at least v1.0 to extract, comp
109034        0×1A9EA         End of Zip archive, footer length: 22
109056        0×1AA00         Zip archive data, at least v1.0 to extract, comp
109460        0×1AB94         End of Zip archive, footer length: 22
109482        0×1ABAA         Zip archive data, at least v1.0 to extract, comp
109886        0×1AD3E         End of Zip archive, footer length: 22
109908        0×1AD54         Zip archive data, at least v1.0 to extract, comp
110312        0×1AEE8         End of Zip archive, footer length: 22
110334        0×1AEFE         Zip archive data, at least v1.0 to extract, comp
110738        0×1B092         End of Zip archive, footer length: 22
110760        0×1B0A8         Zip archive data, at least v1.0 to extract, comp
111164        0×1B23C         End of Zip archive, footer length: 22
111186        0×1B252         Zip archive data, at least v1.0 to extract, comp
111590        0×1B3E6         End of Zip archive, footer length: 22
111612        0×1B3FC         Zip archive data, at least v1.0 to extract, comp
112016        0×1B590         End of Zip archive, footer length: 22
112038        0×1B5A6         Zip archive data, at least v1.0 to extract, comp
112442        0×1B73A         End of Zip archive, footer length: 22
112464        0×1B750         Zip archive data, at least v1.0 to extract, comp
112868        0×1B8E4         End of Zip archive, footer length: 22

root@kali:~/桌面/output/zip#
```

发现有一堆文件，再把它们分离出来（这里用binwalk分离的话可以自动解压），得到 partaa.enc 到 partke.enc 共计265个文件，enc文件是RSA加密后的文件，所以还需要找到私钥进行解密

先来了解一下 gif 的文件结构 https://blog.csdn.net/xlvector/article/details/589214

以 0x21FE 开头的注释扩展 (Comment Extension) 可以记录一些信息，我们在 010Editor 里搜索 21FE 看到有一些16进制字符



21FE后的80表示注释的长度，gif 的每一帧都有一个这样的注释，可以一个个将其复制出来，也可以在 kali 里用 strings 命令跑出来（最后一行开头的8是多余的，需要删掉）

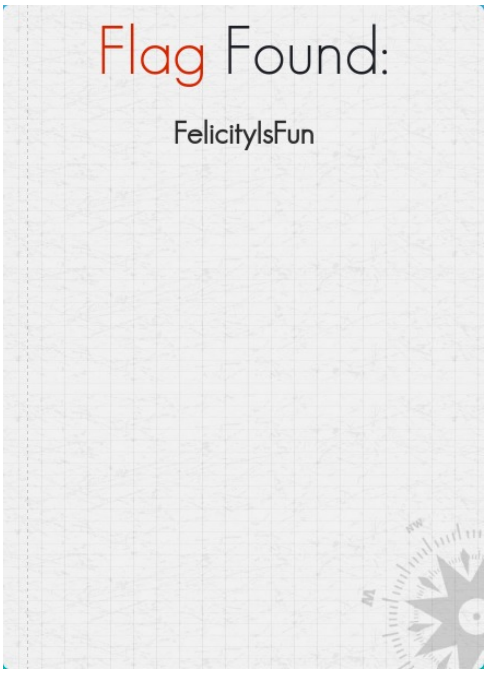
```
root@kali:~/桌面# strings -20 out.gif
   !!!""""###$$$%%%&&&'''((((*)**+++,,,---...///0001112223334445556667778889999:::;;;<<<===>>>???@@@AAABBBCCCDDDEEEFFFGGGHHHIIIJJJK
mmnnnooopppqqqrrrssstttuuuvvvwwwxxxyyyzzz{{{|||}}}~~~
4d494945767749424144414e42676b71686b6947397730424151454641415343424b6b776767536c41674541416f494241514644d4e624c3571565769435172
58327736397 12f377933536849507565707478664177525162524f72653330633655772f6f4b3877655a547834346d30414c6f75685634364b63514a6b687271
67384f79337335593546563177434b3736367532574c775672574d49554a47316a444472527659504963513555714370354544514369 6f524d4763555a456732
75766c3134324c44424161654f4c7a464d3465324a637a532b307238356d50527243537 86a4c4b4c614c774949516e5a5849705853555256 2f776a6877575231
664a474738512b756345417061587363 4e435546343462506d344850434a2f306d7244435457482f59324350564a6b4e6b2b6f305637564f74484b734d4c344e
434e414a483434572f4952774a6e744e572b4e3848726770526b467567686d4e6a63776c456b7274554b4731735243792f2f57687544756e5632706853525176
486f74425a76796441674d424141454367674542414a79614e336d6c6b756e772b617137704b5561677636437a64424779643 4a78447941706b314b374f4938
54426873464d2f33744246654131592f41762b7568434c727967726b4279652f6963372f2b30356f3853392b65674d6b52584e484b4175795233675269 6b7759
7678454b634a676a5a5a4c5246567941593 72f6c477634774e4268336249504466463144673973 7a596e6b774948396c4c454679656d4d37344169415 96c6371
6456645a49452f6271325a344a4f307439484367485a4e6651374a645266656a4e4a5156595544303151753564 4d744f523465494d6462576b68625658773254
45304837785178746a7754367a557270714576764f3765334648457342495836355652585 4c6276394f6f61794a786352715838654a6b5269344c2f597a6f34
4b5470456d6b64754a4c587346777433617151546 6a5a48584f6c5454344e4564734832703054734341454367594541 2f653162737a4f3061312b6342614451
6a514f2b50763942734a464442336f314c477555484d4e53384644706e334a3436556b59796b353276496130763454636a715353484e4976585730544445556b
624e4641314870557856786c773042696565683842673348665879514268587664544437 6b306c73446d4d456f3455504b595336443563595779727768643 56e
4344304c72786562674f373552784c35514549452b34435649516b4367594541337638522f584b52564a5045346145656737705134755476634678 6a454e4e65
787866364b34787059344c6639636c4976646563526843327431486452 2b7853756c552f52536152727a4863773378672b4c31754f3073317435533766336e75
3361696849586b636747537158435a3156487a426d6543343054567366417962733771 4a3078593654357138 4d6e78324f62355a336c49477579686a65566174
4649493253647a324a2f5543675945417465413135 7a516f6a53504e482b62676d62424e7174657632475 6a536f374932556b777243316e4559515334636266
506444436464306668d6444585534766e2b66575539686b58706d4b7043592b416363626c56554e744e4d4b6649423453484d78716a426961386f31616e5a35
726b6e6f5638576d4a4f50645a6259666478422f4b443245443444424364756494c794179756577315066574 36f64586969502f5a356a67742b6b4367594541
727378716f61306f316f3975695237752b48735835494e6f5854394f4f4759337151445 76a55526e61435779774d756a525979355a774b363930416f6f4c5253
744e555633334b3453416868384b7153714f6830652b34636b576235417066 6c38634b3561362b763838543039583841416459355042473335465622b7673357a
54704d75626c54434b4a773259617a4754385579564e386666353344e4f3951426f4533686d45796f642f45436759413 96b307879434a7a6376654b6478727936
706a51786b39465a7a3341744570316f635635566f776167562b4f4e657a2f4863634638474a4f384244716c5036586b634c5574736e7036745952654549434 9
484b7735432b667741586c4649746d30396145565458772b787a4c4a623253723667415450574d3571566175627866736 2356d58482f77443969434c684a536f
8724b3052485a6b7450624573357974447371424864355046467 73d3d
root@kali:~/桌面#
```

16进制转为字符串，给它加个文件头即可得到RSA的私钥

```
-----BEGIN PRIVATE KEY-----
MIIEvwIBADANBgkqhkiG9w0BAQEFAASCBKkwggSlAgEAAoIBAQDdMNbL5qVWiCQrX2w69q/7y3ShlPueptxfAwRQbROre30c6Uw/oK8weZ
Tx44m0ALouhV46KcQJkhrqg8Oy3s5Y5FV1wCK766u2WLwVrWMlUJG1jDDrRvYPlcQ5UqCp5EDQCioRMGcUZEg2uvl142LDBAaeOLzFM
4e2JczS+0r85mPRrCSxjLKLaLwIlQnZXlpXSURV/wjhwWR1fJGG8Q+ucEApaXscNCUF44bPm4HPCJ/0mrDCTWH/Y2CPVJkNk+o0V7VOtHK
sML4NCNAJH44W/IRwJntNW+N8HrgpRkFughmNjcwlEkrtUKG1sRCy//WhuDunV2phSRQvHotBZvydAgMBAAECggEBAJyaN3mlkunw+aq7
pKUagv6CzdBGyd9JxDyApk1K7OI8TBhsFM/3tBFeA1Y/Av+uhCLrygrkBye/ic7/+05o8S9+egMkRXNHKAuyR3gRikwYvxEKcJgjZZLRFVyAY
7/lGv4wNBh3bIPDfF1Dg9szYnkwlH9ILEFyemM74AiAYIcqdVdZlE/bq2Z4JO0t9HCgHZNfQ7JdRfejNJQVYUD01Qu5dMtOR4elMdbWkhbVXw
2TE0H7xQxtjwT6zUrpqEvvO7e3FHEsBIX65VRXRLbv9OoayJxcRqX8eJkRi4L/Yzo4KTpEmkduJLXsFwt3aqQTbjZHXOITT4NEdsH2p0TsCA
ECgYEA/e1bszO0a1+cBaDQjQO+Pv9BsJFDB3o1LGuUHMNS8FDpn3J46UkYyk52vla0v4TcjqSSHNlvXW0TDEUkbNFA1HpUxVxlw0Bieeh
8Bg3HfXyQBhXvdTD7k0IsDmMEo4UPKYS6D5cYWyrwhd5nCD0LrxebgO75RxL5QEIE+4CVIQkCgYEA3v8R/XKRVJPE4aEeg7pQ4uTvcFx
jENNexxf6K4xpY4Lf9cIlvdecRhC2t1HdR+xSuIU/RSaRrzHcw3xg+L1uO0s1t5S7f3nu3aihlXkcgGSqXCZ1VHzBmeC40TVsfAybs7qJ0xY6T5q
8Mnx2Ob5Z3IlGuyhjeVatFII2Sdz2J/UCgYEAteA15zQojSPNH+bgmbBNqtev2GUjSo7l2UkwrC1nEYQS4cbfPdDCdd0fhMdDXU4vn+fWU9hk
XpmKpCY+AccbIVUNtNMKfIB4SHMxqjBia8o1anZ5rknoV8WmJOPdZbYfdxB/KD2ED4DBCFGVILyAyuew1PfWCodXiiP/Z5jgt+kCgYEArsxqo
a0o1o9uiR7u+HsX5INoXT9OOGY3qQDWjURnaCWywMujRYy5ZwK690AooLRStNUV33K4SAhh8KqSqOh0e+4ckWb5Apfl8cK5a6+v88T0
9X8AAdY5PBG3Teb+vs5zTpMublTCKJw2YazGT8UyVN8ff53NO9QBoE3hmEyod/ECgYA9k0xyCJzcveKdxry6pjQxk9FZz3AtEp1ocV5Vowa
gV+ONez/HccF8GJO8BDqlP6XkcLUtsnp6tYReElClHKw5C+fwAXlFltm09aEVTXw+xzLJb2Sr6gATPWM5qVaubxfsb5mXH/wD9iCLhJSorK0R
HZktPbEs5ytDsqBHd5PFFw==
-----END PRIVATE KEY-----
```

然后脚本解密所有文件并输出就得到flag了

```
# coding=utf-8
import os
import Crypto.PublicKey.RSA
import Crypto.Cipher.PKCS1_v1_5
path = r"F:\ShareFile\_out.gif.extracted"
# 读取所有文件
file = [name for name in os.listdir(path)]
flag = b''
# 载入私钥
cipher = Crypto.Cipher.PKCS1_v1_5.new(Crypto.PublicKey.RSA.importKey(open(r'F:\ShareFile\private.txt','rb').read
()))
for fi in file:
    message = open(os.path.join(path,fi),'rb').read()
    flag += cipher.decrypt(message,b'rsa')
open(r'F:\ShareFile\flag.jpg','wb').write(flag)
```



> FelicityIsFun

crc

下载解压得到一个加密的zip文件，尝试暴力破解，解不出来

名称	大小	压缩后大小	类型	修改时间	CRC32
..			文件夹		
1.txt *	6	18	文本文档	2017/11/24...	CC86365B
2.txt *	6	18	文本文档	2017/11/24...	BCEE7ED5
3.txt *	6	18	文本文档	2017/11/24...	CCCA7E74
convert.txt *	15,360	1,962	文本文档	2017/11/24...	9EBFAE5D

但是里面 1.txt、2.txt、3.txt 都只有6字节，可以根据CRC的值把原文破解出来，爆破工具：https://github.com/theonlypwner/crc32

```
(python2) C:\Users\28919\Desktop>python crc32.py reverse 0xCC86365B
4 bytes: {0x65, 0xd7, 0x1e, 0xf0}
```

```
verification checksum: 0xcc86365b (OK)
alternative: 05J728 (OK)
alternative: 0EvF7h (OK)
alternative: 2ysXnu (OK)
alternative: 3y2iul (OK)
alternative: R9DrOf (OK)
alternative: WQkoQX (OK)
alternative: avuKGt (OK)
alternative: dO875V (OK)
alternative: dSwk4B (OK)
alternative: forum_ (OK)
alternative: go3DvF (OK)
alternative: ldpDP2 (OK)
alternative: r6wKtc (OK)
alternative: s66zoz (OK)
alternative: yQGfVS (OK)
```

看到得到的结果中只有 **forum_** 是一个有意义的字符，猜测原文为 **forum_**
但是文件内容太少不能进行明文攻击，继续尝试爆破出来另外两个文件

```
(python2) C:\Users\28919\Desktop>python crc32.py reverse 0xBCEE7ED5
4 bytes: {0x1c, 0xeb, 0xe5, 0x41}
verification checksum: 0xbcee7ed5 (OK)
alternative: 2VSYDo (OK)
alternative: 5OTgnD (OK)
alternative: 7sQy7Y (OK)
alternative: 91ctf_ (OK)
alternative: AVfsVk (OK)
alternative: N5K_u8 (OK)
alternative: OYyCje (OK)
alternative: PgLPQi (OK)
alternative: aYUJmn (OK)
alternative: c425Xo (OK)
alternative: cePT4s (OK)
alternative: d1zWsP (OK)
alternative: pt05kx (OK)
alternative: rTzw3q (OK)
```

```
(python2) C:\Users\28919\Desktop>python crc32.py reverse 0xCCCA7E74
4 bytes: {0x3f, 0x09, 0x32, 0xe4}
verification checksum: 0xccca7e74 (OK)
alternative: 1Atmmb (OK)
alternative: 6XsSGI (OK)
alternative: EXFyUM (OK)
alternative: KWYIiC (OK)
alternative: QmOjH5 (OK)
alternative: Spkdxd (OK)
alternative: TilZRO (OK)
alternative: Uub7HB (OK)
alternative: ZfsjnA (OK)
alternative: cN3O_z (OK)
alternative: com_66 (OK)
alternative: dW4quQ (OK)
alternative: kXjpRF (OK)
alternative: lAmNxm (OK)
alternative: nOEmLx (OK)
alternative: r24Q5p (OK)
alternative: rcVOYl (OK)
alternative: tf_ciJ (OK)
```

在结果中找到另外两个有意义的字符：**91ctf_** 、**com_66** 连起来就是 **forum_91ctf_com_66**
尝试用这串字符作为密码解压，解压成功，得到一个内容为二进制的 txt 文件，转为字符串得到一个 HTML 标签：

<img
src="
GlgHlqN2eqtoYljfQfuGiQ/Pr9+/cv4P/5609/APgOhAQBIUFASBAQEgSEBAEhQUBIEPj78G9fX19jnyOXD5onv43Dh7/7GPkLLnmvSecnyooE
ASFBQEgQEBIEhAQBIROA9mDJftq8wHf09PJux/l7hPmD8DrT5QVCQJCgoCQICAkCgJAKCgJAgICQI/MSB7ORXu3dD0qdPTv6ZrEgQEBIEhAQBIROA
+va1y/22wk7PaJTfqPv1E/bliQUJlEBASBIQEASFBQEgQEBIEhASBrye2H26wZAS5f9z5M58oKxlEhAQBIUFASBAQEgSEBIEhQUBIGPaAywH9kRYKAkCAgJAgICQJCgoCQICAkCAkCwD96oQOE6
84L1QAAAABJRU5ErkJggg==">

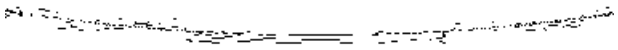
Base64转图片得到一张二维码，扫描即得到 flag

```
flag{owid0-o91hf-9iahg}
```

4433

下载解压得到一张 jpg 图片，用 StegSolve 打开发现图片太大显示不全，用 Photoshop 将图片缩小后打开，在 Red plane 1 里发现了一个疑似二维码的东西

再打开原图找到指定位置发现一个清晰的二维码

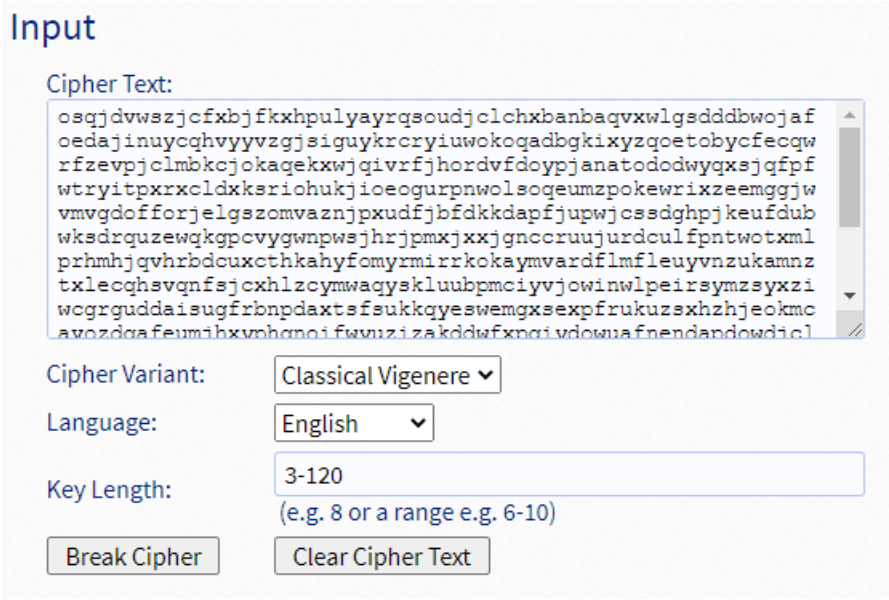


扫描得到 …–.----…–… ，应该是摩斯电码，但不知道怎么分割，根据题目名称尝试以 **4433** 进行分割解码得到**VYGUD**，然而flag并不是这个，在摩斯电码中存在一些常用的缩写，VY 代表 VERY，GUD 代表 GOOD，所以正确的flag是 **VERYGOOD**

> flag{VERYGOOD}

challenge_how_many_Vigenère

下载解压得到一段密文，题目让我们恢复明文，又根据 Vigenère 知为维吉尼亚加密，直接用爆破网站暴力破解
https://www.guballa.de/vigenere-solver（打开可能有点慢）
维吉尼亚密码的密钥越长越容易破解，调整好参数后破解：



得到密钥和明文：

> 密钥：ohihzkssefkmqxqbkihybnynvndzkdlqvhwhgywaftmeteecqprzjczvnmhnzwyasmlwbwvaqitejbfofycejjlcbpk
> 明文：
> aliceleavestheteapartyandentersthegardenwhereshecomesuponthreelivingplayingcardspaintingthewhiterosesonarosetreeredbecauseth
> equeenofheartshateswhiterosesaprocessionofmorecardskingsandqueensandeventhewhiterabbitentersthegardenalicethenmeetstheking
> andqueenthequeenafiguredifficulttopleaseintroduceshertrademarkphraseoffwithhisheadwhichsheuttersattheslightestdissatisfactionwitha
> subjectaliceisinvitedorsomemightsayorderedtoplayagameofcroquetwiththequeenandtherestofhersubjectsbutthegamequicklydescendsint
> ochaosliveflamingosareusedasmalletsandhedgehogsasballsandaliceonceagainmeetsthecheshirecatthequeenofheartsthenordersthecatt
> obebeheadedonlytohaveherexecutionercomplainthatthisisimpossiblesincetheheadisallthatcanbeseenofhimbecausethecatbelongstothed
> uchessthequeenispromptedtoreleasetheduchessfromprisontoresolvethematter

稍作处理得到：

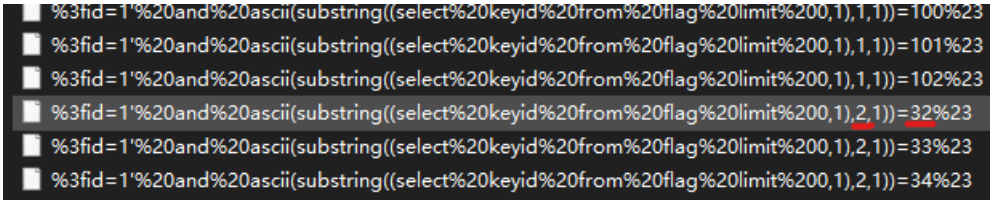
> Alice leaves the tea party and enters the garden where she comes upon three living playing cards painting the white roses on a rose tree red because the queen of hearts hates white roses. A procession of more cards, kings and queens and even the white rabbit enters the garden. Alice then meets the king and queen. The queen, a figure difficult to please, introduces her trademark phrase “off with his head”, which she utters at the slightest dissatisfaction with a subject. Alice is invited (or some might say ordered) to play a game of croquet with the queen and the rest of her subjects, but the game quickly descends into chaos. Live flamingos are used as mallets and hedgehogs as balls, and alice once again meets the cheshire cat. The queen of hearts then orders the cat to be beheaded, only to have her executioner complain that this is impossible since the head is all that can be seen of him. Because the cat belongs to the duchess. The queen is prompted to release the duchess from prison to resolve the matter.

看到这不用去 Google 也能猜到是爱丽丝梦游仙境，英文名称为 Alice’s Adventures in Wonderland，去掉空格和 ’ 再把大写换成小写，用破解得到的密钥加密即得到 flag

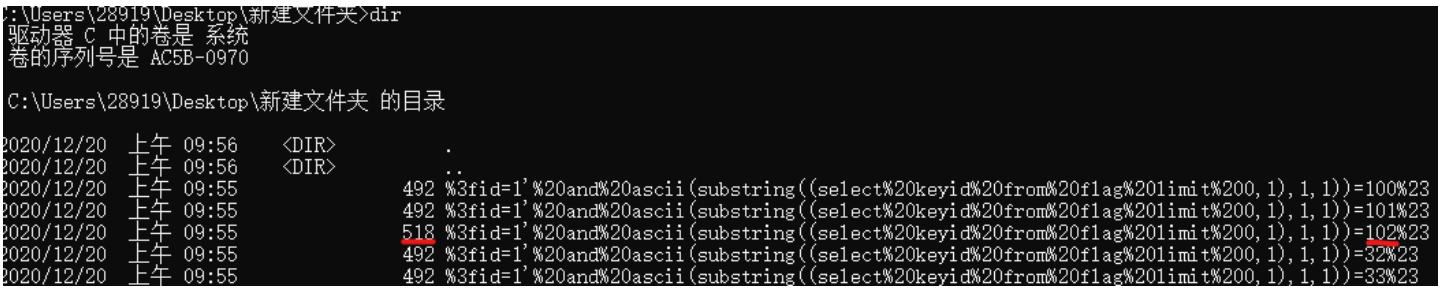
> LCTF{osqjdcsvzjxfkoutsvdmoqcegnqc}

流量分析

下载得到一个流量包，全部是SQL注入的流量，在文件菜单里选择导出对象将所有HTTP对象导出



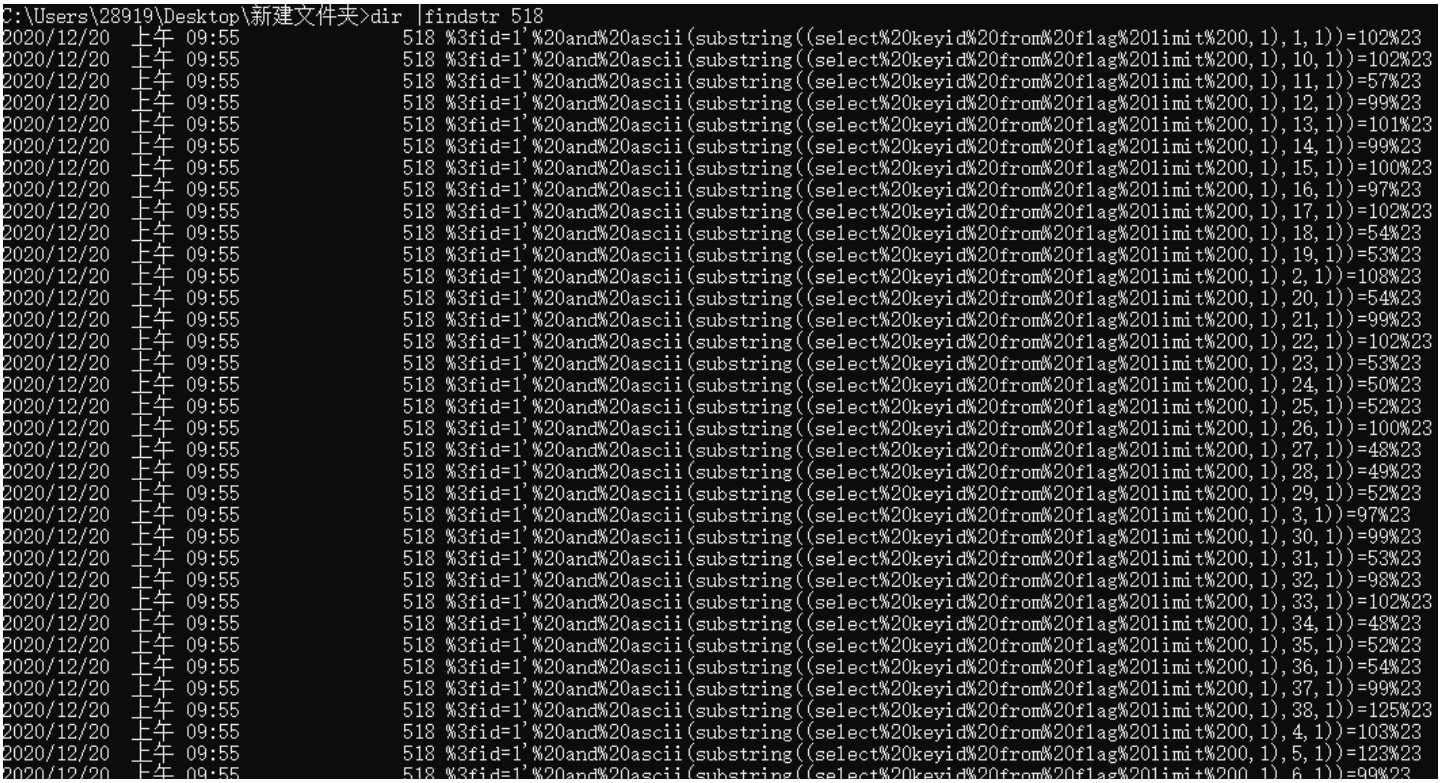
观察文件名不难发现：只有倒数第四和倒数第二个数字在变化，用 dir 命令跑一遍发现有部分文件大小和其他的不一样，而且都是倒数第二个数字最大的

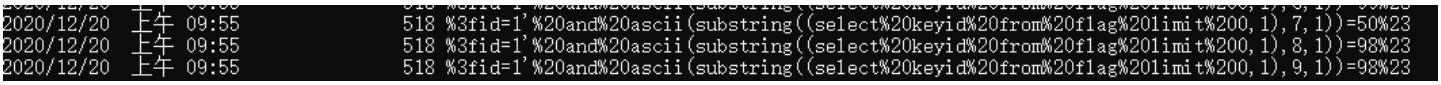


将492字节和518字节的文件后缀名改为 html 打开对比一下

感觉518字节的文件应该有我们需要的信息

用 **dir | findstr 518** 打印出所有518字节的文件（Linux下用 **ls -l | grep 518** ）





按照倒数第四个数字的大小顺序将倒数第二个数字依次排列得到

> 102 108 97 103 123 99 50 98 98 102 57 99 101 99 100 97 102 54 53 54 99 102 53 50 52 100 48 49 52 99 53 98 102 48 52 54 99 125

将以上 Ascii 码转为字符得到 flag

> flag{c2bbf9cecdaf656cf524d014c5bf046c}

未完待续...