

领航杯部分Writeup

原创

[LetheSec](#) 于 2019-11-25 09:08:47 发布 2284 收藏 4

分类专栏: [wp](#) 文章标签: [wp](#)

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/qq_42181428/article/details/103205987

版权



[wp 专栏收录该内容](#)

11 篇文章 0 订阅

订阅专栏

编码解码

```
V2VsY29tZXRvdGhlUG1sb3RDdXA=
```

签到题, base64解码一下得到:

```
WelcometothePilotCup
```

取证分析

给了一个pacp.pcap, CTF Wiki里面收录过这一道例题...

wireshark 打开发现全部为DNS协议, 查询名为大量字符串 `xxxxx.skullsec1abs.org`

好像是BSides San Francisco CTF 2017原题, 参考这篇文章: <https://volatilevirus.home.blog/2018/12/30/bsidessf17-ctf-dnscap-write-up/>

先用下面命令进行提取:

```
tshark -r pacp.pcap -T fields -e dns.qry.name > hex
```

然后用如下脚本得到flag.png:

```
from scapy.all import *

r = rdpcap("pacp.pcap")

a = ""

b = ""
c = ""
new = ""

f = open("flag.png", "w")

for i in range (0, len(r)):
    if r[i].haslayer(DNSQR) and not r[i].haslayer(DNSRR):
        a = r[i][DNSQR].qname
        b = a.replace(".skullseclabs.org.", "")
        b = b.replace(".", "").decode("hex")[9:]

        if b == c:
            continue

        c = b

        if 6 < i < 365:
            new = new + b

f.write(new)

f.close()
```

直接打不开，用010 Editor看一下：

▼ 编辑方式: 十六进制(16) ▼	运行脚本 ▼	运行模板 ▼	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	0123456789ABCDEF
0000h:	57	65	6C	63	6F	6D	65	20	74	6F	20	64	6E	73	63	61			Welcome to dnsca
0010h:	70	21	20	54	68	65	20	66	6C	61	67	20	69	73	20	62			p! The flag is b
0020h:	65	6C	6F	77	2C	20	68	61	76	65	20	66	75	6E	21	21			elow, have fun!!
0030h:	0A	63	6F	6D	6D	61	6E	64	20	28	73	69	72	76	69	6D			.command (sirvim
0040h:	65	73	29	00	00	00	2C	ED	80	01	00	03	89	50	4E	47			es)....,i€...%PNG
0050h:	0D	0A	1A	0A	00	00	00	0D	49	48	44	52	00	00	01	00			IHDR....
0060h:	00	00	01	00	08	04	00	00	00	F6	7B	60	ED	00	00	00			ö{'í...
0070h:	04	67	41	4D	41	00	01	86	A0	31	E8	96	5F	00	00	00			.gAMA..† 1è-...
0080h:	02	62	4B	47	44	00	FF	87	8F	CC	BF	00	00	00	09	70			.bKGD.ÿ†.İç....p
0090h:	48	59	73	00	00	0B	13	00	00	0B	13	01	00	9A	9C	18			HYs.....šœ.
00A0h:	00	00	00	07	74	49	4D	45	07	E1	02	02	05	0D	35	24			tIME.á....5\$
00B0h:	D3	81	E9	00	00	2C	08	49	44	41	54	78	DA	ED	9D	77			Ó.é...IDATxÚí.w
00C0h:	9C	1B	D5	D5	F7	BF	A3	AE	95	56	DB	AB	D7	DE	5D	7B			œ.ÖÖ÷çİ@•VÛ«×Ë]{
00D0h:	77	ED	75	B7	71	C1	D8	98	66	C0	38	26	C6	80	C1	0F			wíu`qÄØ`fÀ8&ÆÉÁ.
00E0h:	09	84	04	48	42	42	48	F2	52	9E	87	84	27	A4	40	EA			...HBBHòRž†,, 'ı@é
00F0h:	43	78	52	48	21	8D	BC	40	48	42	89	43	31	25	74	B0			C×RH!.4@HB%C1&t°
0100h:	81	60	03	2E	B8	E0	EE	B5	D7	F6	F6	5E	D4	E6	F9	63			.`...àîp×öö^Ôæùc
0110h:	65	79	46	1A	69	34	5A	49	33	32	FC	E6	83	B1	2C	8D			eyF.ı4ZI32üæf†,.
0120h:	74	E7	DE	DF	3D	E7	DC	73	CE	3D	57	E0	E4	82	05	7B			tçßß=çÛsî=Wàä,.{
0130h:	E8	72	50	40	31	E9	14	90	4E	3E	E9	14	E0	C1	8E	03			æxPß1ù O>ù ááž



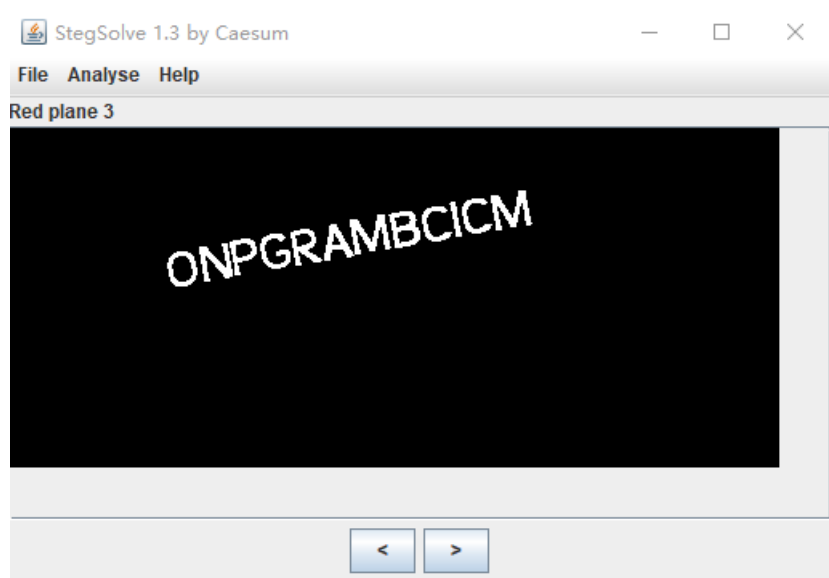
删掉上面选中的部分，再打开，得到flag: b91011fc

lsb

给了下面这张图片：



根据题目名字lsb，用Stegsolve打开，移位到下面位置得到flag：ONPGRAMBCICM



恢复与解密

题目描述：公安人员在犯罪分子的电脑中发现一些磁盘文件，但是发现关键信息已经被删除，现需要你对磁盘进行恢复，并对恢复出来的一些秘密文件里面的加密信息进行解密。注意：通过strings获取到的yc4pl0fvjs2k1t7T为假flag，请尝试使用其他正确的做题方式获取flag。

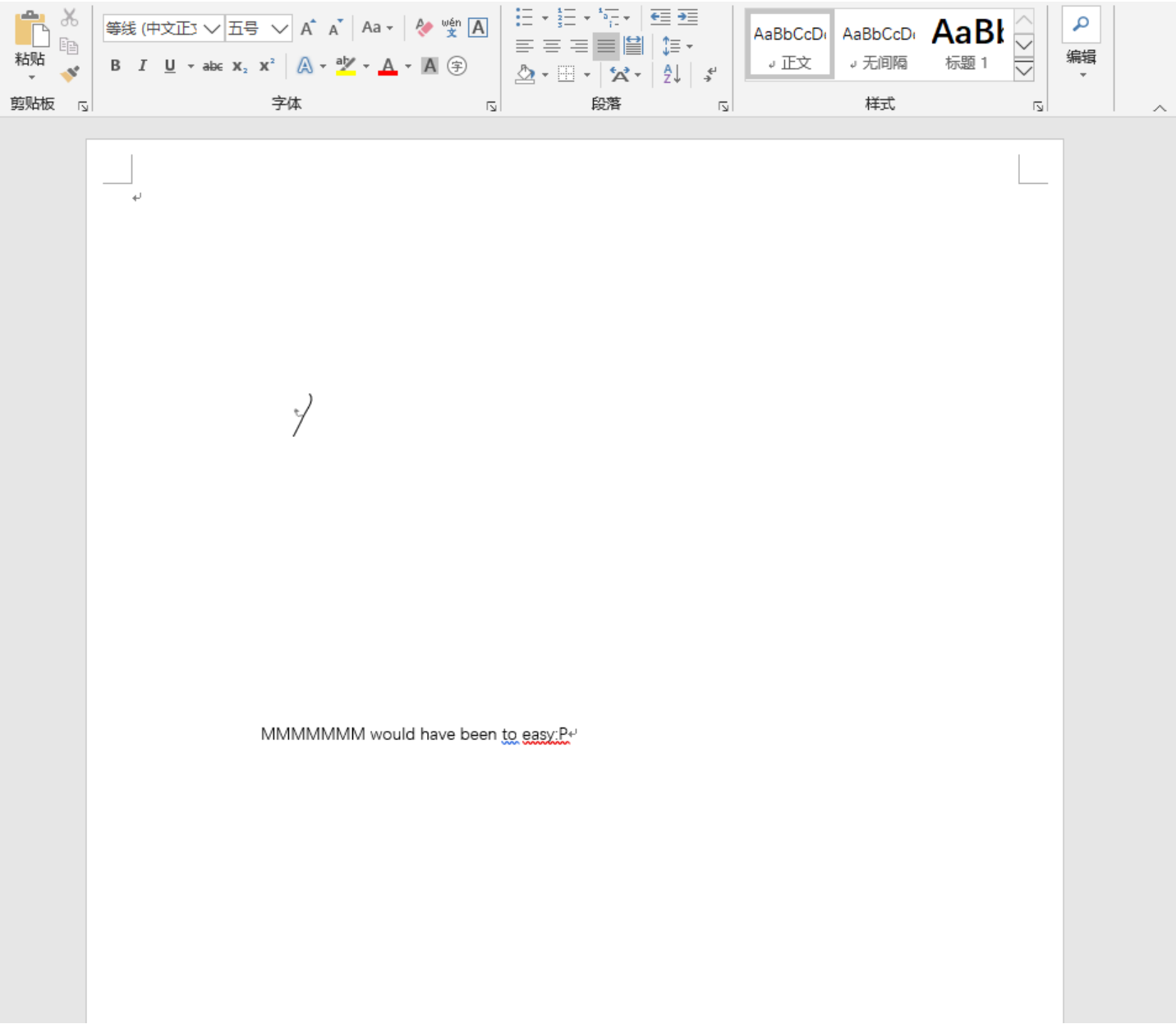
所以可以判断这题为磁盘恢复，并且有假flag。

附件给的是一个xty.img，在kali下面命令进行挂载：

```
mount -o loop xty.img /mnt
```

```
root@Kali:~/CTF/领航杯# mount -o loop xty.img /mnt
root@Kali:~/CTF/领航杯# cd /mnt/
root@Kali:/mnt# ls -a
.  ..  .hide  lost+found  .ls
```

发一下一个隐藏文件夹 `.hide`，里面有一个 `secret.odg`，打开得到如下内容：



好像没什么用，想到题目描述中的磁盘恢复，所以需要用到sstat 和 ext3grep这两个工具，先用下面命令查看相关信息：

```
fsstat xty.img
```

```
root@Kali:~/CTF/领航杯# fsstat xty.img
FILE SYSTEM INFORMATION
-----
File System Type: Ext3
Volume Name:
Volume ID: 809a1caaa174a48d5a4aeac6914601ca

Last Written at: 2019-11-22 13:05:07 (CST)
Last Checked at: 2014-02-07 09:28:31 (CST)

Last Mounted at: 2019-11-22 13:05:07 (CST)
Unmounted properly
Last mounted on: /mnt

Source OS: Linux
Dynamic Structure
Compat Features: Journal, Ext Attributes, Resize Inode, Dir Index
InCompat Features: Filetype, Needs Recovery,
Read Only Compat Features: Sparse Super,

Journal ID: 00
Journal Inode: 8

METADATA INFORMATION
-----
Inode Range: 1 - 1281
Root Directory: 2
Free Inodes: 1266

CONTENT INFORMATION
-----
Block Range: 0 - 5119
Block Size: 1024
Reserved Blocks Before Block Groups: 1
Free Blocks: 3880

BLOCK GROUP INFORMATION
-----
Number of Block Groups: 1
Inodes per group: 1280
Blocks per group: 8192
```

注意这里 **Root Directory: 2** 中的2是我们要用到的。

再用ext3grep工具，查看目录如下：

```
ext3grep --inode 2 xty.img
```



```

Writing analysis so far to 'xty.img.ext3grep.stage2'. Delete that file if you want to do this
stage again.
The first block of the directory is 184.
Inode 2 is directory "".
Directory block 184:
.-- File type in dir_entry (r=regular file, d=directory, l=symlink)
|-- D: Deleted ; R: Reallocated
Indx Next | Inode | Deletion time | Mode | File name
=====+=====+-----+-----+=====
0 1 d 2 drwxr-xr-x .
1 2 d 2 drwxr-xr-x ..
2 5 d 11 drwx----- lost+found
3 5 r 12 D 1558974332 Tue May 28 00:25:32 2019 rrw-r--r-- 0secret
4 5 r 12 D 1558974332 Tue May 28 00:25:32 2019 rrw-r--r-- secret
5 6 d 13 drwxr-xr-x .hide
6 end d 17 drwxr-xr-x .ls
7 8 r 16 D 1558974209 Tue May 28 00:23:29 2019 rrw-r--r-- secret.1
8 9 r 18 D 1558974209 Tue May 28 00:23:29 2019 rrw-r--r-- secret.2
9 10 r 19 D 1558974209 Tue May 28 00:23:29 2019 rrw-r--r-- secret.3
10 11 r 20 D 1558974209 Tue May 28 00:23:29 2019 rrw-r--r-- secret.4
11 12 r 21 D 1558974209 Tue May 28 00:23:29 2019 rrw-r--r-- secret.5
12 13 r 22 D 1558974209 Tue May 28 00:23:29 2019 rrw-r--r-- secret.6
13 14 r 23 D 1558974209 Tue May 28 00:23:29 2019 rrw-r--r-- secret.7
14 15 r 24 D 1558974209 Tue May 28 00:23:29 2019 rrw-r--r-- secret.8
15 end r 25 D 1558974209 Tue May 28 00:23:29 2019 rrw-r--r-- secret.9
16 end r 26 D 1558974209 Tue May 28 00:23:29 2019 rrw-r--r-- secret.10

```

可以看到很多secret文件，实际上内容都一样，用下面命令将其恢复出来：

```
ext3grep --restore-file secret xty.img
```

```

root@kali:~/CTF/领航杯# ext3grep --restore-file secret xty.img
Running ext3grep version 0.10.2
WARNING: I don't know what EXT3_FEATURE_COMPAT_EXT_ATTR is.
WARNING: EXT3_FEATURE_INCOMPAT_RECOVER is set. This either means that your partition is st
ill mounted, and/or the file system is in an unclean state.
Number of groups: 1
Minimum / maximum journal block: 198 / 1227
Loading journal descriptors... sorting... done
The oldest inode block that is still in the journal, appears to be from 1391737510 = Fri F
eb 7 09:45:10 2014
Number of descriptors in journal: 246; min / max sequence numbers: 23 / 121
Loading xty.img.ext3grep.stage2... done
Restoring secret

```

打开该文件发现下面一串字符：

```
aWdxNDs3NDFS0zFpa1I1MWliT08waWdx
```

打开(O)  secret
~/CTF/领航杯/RESTORED_FILES

```
aWdxNDs3NDFS0zFpa1I1MWliT08waWdx
```

base64一下得到：

```
igq4;741R;1ikR51ib000igq
```

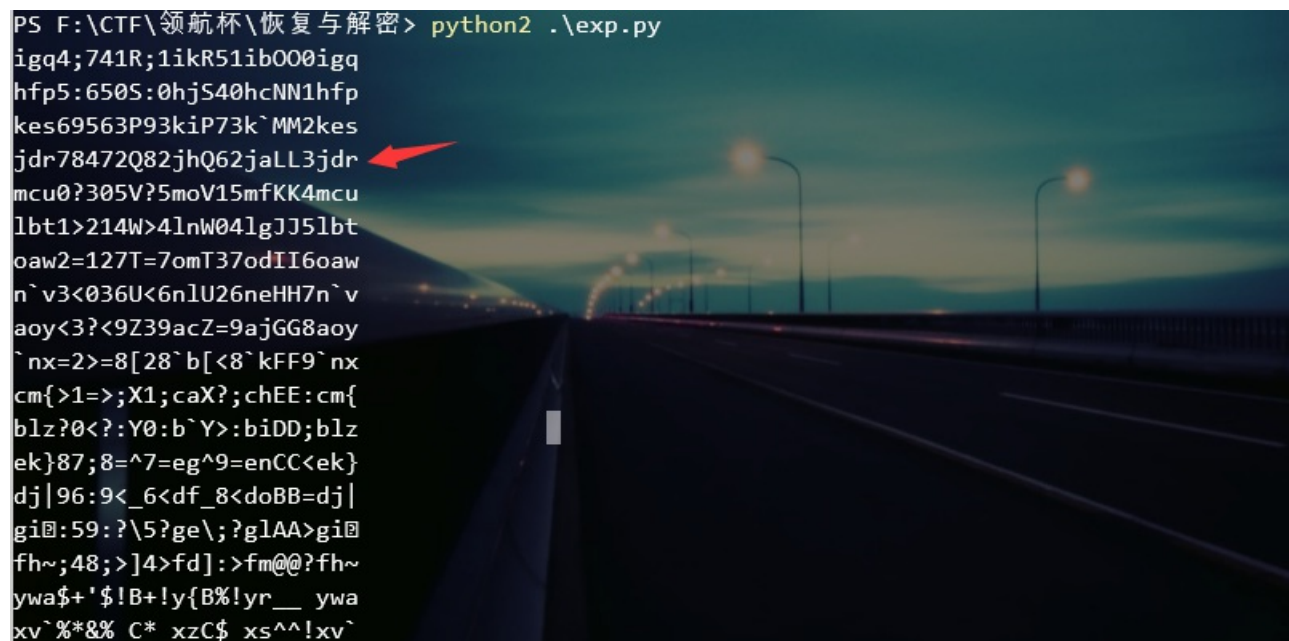
异或解密脚本如下：

```
import string

c = "igq4;741R;1ikR51ib000igq"

for i in range(0,200):
    p = ""
    for j in range(len(c)):
        p += chr(ord(c[j])^i)
    print (p)
```

得到的唯一一个无特殊字符的即为flag: jdr78472Q82jhQ62jaLL3jdr



```
PS F:\CTF\领航杯\恢复与解密> python2 .\exp.py
igq4;741R;1ikR51ib000igq
hfp5:650S:0hjS40hcNN1hfp
kes69563P93kiP73k`MM2kes
jdr78472Q82jhQ62jaLL3jdr
mcu0?305V?5moV15mfKK4mcu
lbt1>214W>4lnW04lgJJ5lbt
oaw2=127T=7omT37odII6oaw
n`v3<036U<6nlU26neHH7n`v
aoy<3?<9Z39acZ=9ajGG8aoy
`nx=2>=8[28`b[<8`kFF9`nx
cm{>1=>;X1;caX?;chEE:cm{
blz?0<?:Y0:b`Y>:biDD;blz
ek}87;8=^7=eg^9=enCC<ek}
dj|96:9<_6<df_8<doBB=dj|
gi@:59:?\5?ge\;?glAA>gi@
fh~;48;>]4>fd]:>fm@@?fh~
ywa$+'$!B+!y{B%!yr__ ywa
xv`%*8% C* xzC$ xs^^!xv`
```

文件提取


```
root@kali:~/CTF/领航杯/文件提取# strings flag.exe
data:image/jpeg;base64,/9j/4AAQSkZJRgABAQEASABIAAD/2wBDABELDA8MChEPDg8TEhEUGSobGRcXGTMkJh4
qPDU/Pjs10jlDS2BRQ0daSDk6U3FUWmNma2xrQFB2fnRofWBpa2f/2wBDARITExkWGTebGzFnRTpFZ2dnZ2dnZ2dnZ
2dnZ2dnZ2dnZ2dnZ2dnZ2dnZ2dnZ2dnZ2dnZ2dnZ2dnZ2f/wAARCAaAYYDASIAAhEBAxEB/8QAGwAAAQUBAQ
BAQAAAAAAAAAAAAAAAAAECAwAwf/xABIEAABAwIEAwQFBgwFAwUAAAABAAIDBBEFEiExBkFREYJhgTJxkaGxBxQXQ
lPRFSmZQ1JUYNKSssHwFiQ24fFzk6ImNERVY//EABUBAQEAAAAAAAAAAAAAAAAAB/8QAFREBAQAAAAAAAAAAAA
AAAAAAAAH/2gAMAwEAAhEDEQA/ANYhCFECFLKnjqkpquIwSvaxxGYdQbexc/pDoL/APtqj/x+9Bqamoipoy+aa0Jv6
T3BoSxSNmiD43Nc070acwKwHEfFFDjNA200CZsrXhZXPAs3rsdU/h/jGHDM0jpKinkcI7hrmkA6/FB6CEiyX0i0H6r
Uf+P3ob8odA7T5rP67t+9VWuTX0DGkkgADcmwTIJWzwxYm9F704eohcMY1wWt5f5eT+UqI6wVMNS0ugljla0bXBy7j
QLI/Jvc4XUk7mW3uWtZAb6I0MlVDFKy0SjXv0a1zspd6gnySsijL5HBjRu5xssVxWWjJHDQD9nmaP3ja/krrjgf+m
5SDYh7fjZBdxPZJGHxuDmnZzdbroqDgcl3DV0Sbkl3uNlfgoEzeBSgos0gVDiHFVHh2Jijjk4kAuA0bf+wgvkIQg
EIQgEIQgEIQgEIQgEIQgEIQgELmZ2hJ84ag6oXJs7U8PaeaBxAGqaHEpC0m5uEgcG7nzQ0t3lCrcQjplL5zyPTklxLE
W0NKZCRt19681xjHZK6ZxDu76I9SC5xDjV7JXCEZuR10CqpuLa2R12vA9qok3fyQavD+MKlRlSAOV1hvFLJXWeQNRb
Vedh3Ic+aVsymbG2dt9Cg9qglZPG1wN9F02K8+4Z4nkjeIqh7jsm263VL0yojDgRsgKN5pya3mnIC4br5ISFCBUIX0e
UQSS01axpcbeG6DAMc0tNVcwlS2TCydre0c3024uHw1B52utt+BcNLSPmFLb/pN+5ZWn4wwqnqXztw50Ur9H0Zlu7
nrt/ftfX6BqQIL4i1mJA0GLvNA3H8QwvC660jciwilqJTZhka3Lf0n9NvFAGLCPdG0uokZpIBI7Jtl57heMQMx44l
iTXSvJLmBqGjuR9Vvf6lomfKFQkWFtTg2+rl0vtCDR/gTDP8A6+l/7TfuWT4/o6SLpqT5vTRRPc92rGBtWbT6lK+kS
j/U5x63NXK044wmqYG1FBJIAbgPalwug1eFaYZSj/8AFvwWTxx/Er5K5rG2o70F7N9AX169VpsErosRw6KohjMUZ0D
SF0xc2wetPSB/8pQefcKy40Ipm4TG10Zcm7nNbYHz529a0VbwvX4hI2abFC2TL3mtaco0/dXH50y1mE1byQG9rq4nT
Qe4K3xPiWgwurJkmRzn0bmGRuYW239aDG4jgxwfiDDo5JzUPe5r3E9c3w0Wt44/0zPy7zeX7QWUxnGIMX4loJ6QyZY
3Mbdwy65lq+0P9MVH77P5ggzXD/DFTiGGR1LcSfBG4nutBdsbdbLU80YLLhEuRjqp1Q6qt0Yg6Wv1VFwzxThuG4LFT
```

图片转换Base64 在线把图片转换成Base64

然后strings一下这个图片得到flag: flag{068EEF6A7BAD3FDF}

```
%H\ouGj
%GNB
NMo4
8;Ki
# >0i
flag{068EEF6A7BAD3FDF}
```

给了一个pngdecode.docx:

朋友，密码是 8 位

你自己猜哦？



一开始分析了好久这个图片，但是无果...然后对整个docx文档foremost提取一下，得到flag: 360HA360

CTF / 领航杯 / output / jpg ▾



00000014.jpg



00000028.jpg

OldCypher-Easy

给了密文如下:

Vvr Ifnvaus Bdwokv Gbtrzsa Vkqgofntja rrlznxk eflvkojzcdue rs “mzg oez ff pjkhvtx ok kqzioeg vgfsf.” Zyil au vvy kokaeyrp avuwfnzv, bnl fcry eom ucdgaie mzg qhxiagl dfrguta gh huk wixdf ce oks ijggrtk-dtq uqvketsbkq sulnswvw btj. Taw fssoeimaqb sutulwu gbrvlr gp huk towwu hugk htng prke ulwf tbx teglwfvkj th wpoorv sxutsg ifmfmpwpgkihf . Dig iiyilquegghr fqknjryl wpqbsgalkgg zath fgts gnrn mzkq: vz uetdu kvzy mxujoaojml xqf rtjukapu vtkezjkh1, zv cafkehkj fhj glpnrnzapu fktrxl msly, grhlqqbrj fhj cignvnmaeogoeg nkgff, kcevltaot anuvwbty agv gzrikihfu, rvmz ttd eofn, rnw eqfr. Cztagwh nzkefhvwam ko ijqjvja a vgodykke vzcfnikekabogofn, pw ychru stq vvnz dowwtb pxppmgif nvyv bfxcybvs mzg ggauy hx oognvmtlkqnr kevzpwdavs ygt grilrbfi rvmzttd kbsuimtlkca, ypsmwog, ntu dbkvfvhltxv ec zvlttlkca yrgtapgg guvxjuoeorl tlvopqj. fesi{42s96q9hw73d79v4xf5308t18o8519is10c512u2}

最后一串明显像flag的格式，在这个网站进行维吉尼亚解密: <https://www.guballa.de/vigenere-solver>

得到flag: flag{42e96d9bf73d79c4fd5308f18b8519cb10c512b2}

Cipher Text:

```
Vvr Ifnvaus Bdwokv Gbtrza Vkqgofntja rrlznxk eflvkojcdue
rs "mzg oez ff pjkhvtx ok kqzioeg vgfsf." Zyil au
vvykokaeoyrp avuwfnzv, bnl fery eom ucdgaie mzg qhxiegl
dfrguta gh huk wixdf ce oks ijggrtk-dtq ugvketbxxq
sulnswvwtj. Taw fssoeimagb sutulwu gbrvlr gp huk towwu
hugk htng prke ulwf tbx teglwfvkj th wpoorv sxutsg
ifmfmpwpgkih. Dig iiyilquegghr fqknjryl wpqbsgalkgg zath
fgts gnrn mzkq: vz uetdu kvzy mxujoaojml xqf rtjukapu
vtkezjkh, zvcafkehkj fhj glpnrnzapu fktrxl msly,
arhlagbri fhi gignvmaeodga nkaiff. kcevltsaot anuywtj
```

Cipher Variant: Classical Vigenere ▾

Language: English ▾

Key Length: 3-30
(e.g. 8 or a range e.g. 6-10)

Break Cipher Clear Cipher Text

Result

[Clear text \[hide\]](#)

Clear text using key "congrats":

```
communication. But cryptography nowadays encompasses much more
than this: it deals with mechanisms for ensuring integrity,
techniques for exchanging secret keys, protocols for
authenticating users, electronic auctions and elections, digital
cash, and more. Without attempting to provide a complete
characterization, we would say that modern cryptography involves
the study of mathematical techniques for securing digital
information, systems, and distributed computations against
adversarial attacks.
flag{42e96d9bf73d79c4fd5308f18b8519cb10c512b2}
```

MasterofZip-Middle

level1

给了一个压缩包，和一个Readme.md内容如下：

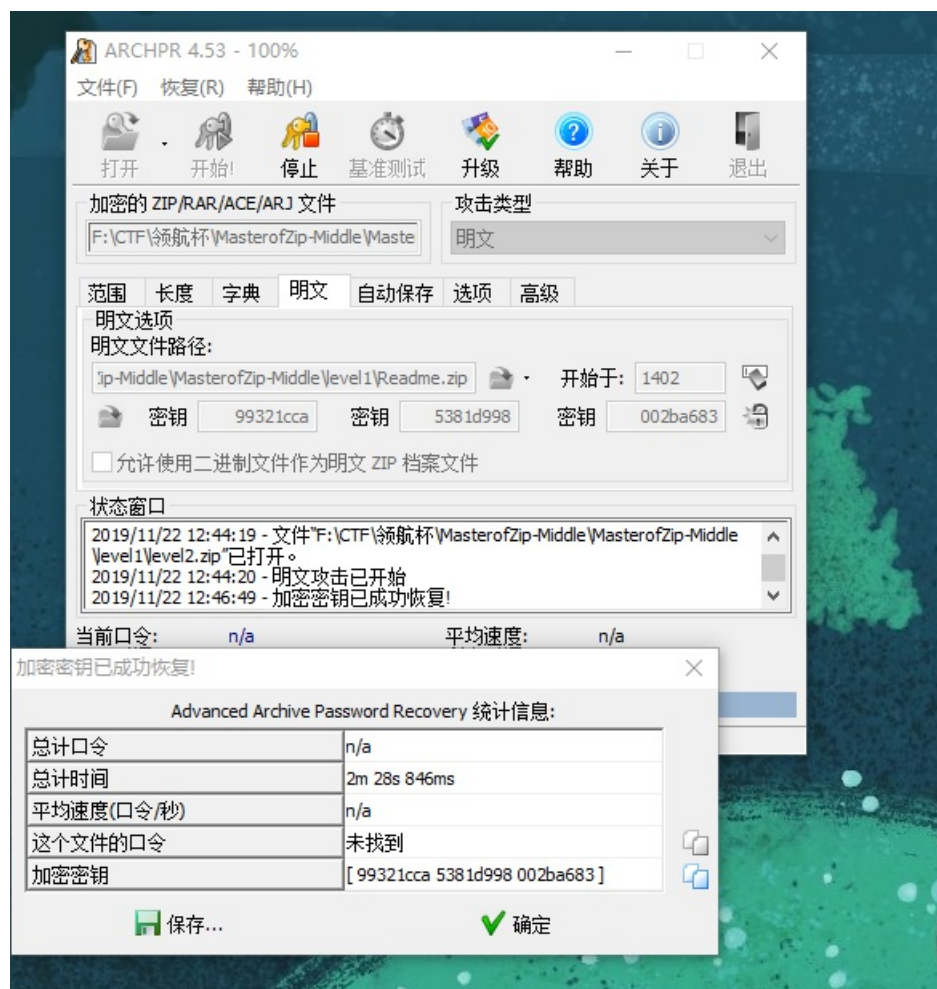
小明喜欢用自己的手机号来加密zip，我们通过社工只查到了他手机号的前三位为199，你能解开这个压缩包吗？（手机号为伪造手机号，请不要试图拨打或社工）

于是爆破199开头的手机号，得到密码：19950453796



level2

第一层解压后得到一个压缩包和一个Readme.txt，内容不重要，但是看到level2.zip里面也有一个Readme.txt，所以尝试明文攻击，成功恢复密钥并解压：



level3

第二层解压后，只给了一个压缩包，于是尝试伪加密：

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	0123456789ABCDEF
1660h:	9C	76	8E	E5	F0	7D	D2	12	F9	4D	66	C3	0F	52	C2	40	ævžăđ}ò.ùmġĂ.RĂ@
1670h:	00	4F	75	D2	20	87	81	85	AD	5D	69	50	C9	5E	F2	EE	.OuÒ ‡....-]iPÉ^òî
1680h:	23	4E	C0	B0	9F	D8	D7	11	C0	83	40	97	B8	DE	BC	A9	#NÀ°ŸØ×.Àf@- ,E¼@
1690h:	2E	18	CA	64	CD	6D	F6	07	81	29	91	DF	BB	9F	2C	B8	..Êdímö..) `ß»Ÿ,
16A0h:	69	5B	67	C0	7D	8B	25	0A	AA	10	9D	E0	AF	07	7C	C9	i[gÀ}<%.^...à~. É
16B0h:	A4	95	F1	27	13	AD	C9	F2	D7	9B	F6	01	29	5C	AF	0E	¤•ñ'-.Éð×>ö.)\`.
16C0h:	66	C1	E3	03	A7	45	F2	07	9B	F6	92	85	34	23	75	C1	fĀā.ŒÈð.>ö'...4#uĀ
16D0h:	EF	4B	23	A3	86	52	90	4D	80	B3	89	04	3D	91	8E	D9	iK#Ł†R.M€°%.=`ŽŮ
16E0h:	57	87	C2	F1	80	4C	F8	DD	45	51	C2	E0	15	09	61	09	W†Āñ€LøŸEQĀā...a.
16F0h:	E6	33	B2	30	6C	03	F9	2D	29	12	BD	02	71	33	AC	61	æ3°0l.ù-).¼.g3¬a
1700h:	97	AD	57	6F	F1	BE	B5	7D	FB	44	CA	28	BC	D5	45	0B	--Wøñ³µ}ûDÊ(ŸOE.
1710h:	1D	46	96	29	A4	A9	F7	4F	B0	77	E5	5A	15	3A	04	A7	.F-) ¯@÷O°wāZ.:.S
1720h:	A2	45	47	12	27	30	EF	54	2F	8A	AC	BC	FA	E4	9F	0C	¢EG.'0iT/Š¬4úāŸ.
1730h:	1F	FF	85	05	75	82	8D	AE	53	A2	8D	68	BF	EC	3E	11	.Ÿ...u,.@S¢.h¿ì>.
1740h:	78	A1	8C	CC	BF	22	18	5E	FC	E6	67	50	4B	01	02	3F	x;@İ¿".^üægPK..?
1750h:	03	14	03	00	00	08	00	5A	14	75	4F	CD	0F	ED	93	25	Z.uOI.í"¢
1760h:	17	00	00	FA	1E	00	00	08	00	24	00	00	00	00	00	00	...ú.....\$......
1770h:	00	20	80	A4	81	00	00	00	00	66	6C	61	67	2E	70	6E	. €¤.....flag.pn
1780h:	67	0A	00	20	00	00	00	00	00	01	00	18	00	80	5F	C9	g..€_É
1790h:	28	3E	A0	D5	01	80	5F	C9	28	3E	A0	D5	01	80	5F	C9	(> Œ.€_É(> Œ.€_É
17A0h:	28	3E	A0	D5	01	50	4B	05	06	00	00	D0	00	01	00	01	(> Œ.PK.....
17B0h:	00	5A	00	00	00	4B	17	00	00	00	00						.Z...K.....

将箭头位置修改为0，成功解压，得到下面这张图片：

???

然后修改发图片高度：

起始页	3.zip	flag2.png ×																
▼	编辑方式: 十六进制(H) ▼	运行脚本: IsASCII.lsc ▼	▶	运行模板: ZIP.bt ▼														
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	0123456789ABCDEF	
0000h:	89	50	4E	47	0D	0A	1A	0A	00	00	00	0D	49	48	44	52	%PNG.....IHDR	
0010h:	00	00	02	80	00	0A	01	FF	08	02	00	00	00	BA	B3	4B	...€...Ÿ.....°°K	
0020h:	B3	00	00	1E	C1	49	44	4A	54	78	9C	ED	DD	79	BC	54	³...ÁIDATxœíŸŸ4T	
0030h:	75	FD	F8	F1	73	E5	B2	A7	82	C8	A6	A8	2C	81	A2	86	uýøñsā²~,È!~, .¢†	
0040h:	91	62	26	09	62	9A	82	50	A2	A1	52	0F	C5	85	32	F5	`b&.bš,P¢;R.Ā...2š	
0050h:	91	B8	DB	43	13	B3	1E	45	65	2E	F9	E0	61	19	6E	59	`ŮC.°.Ee.ùāa.nŸ	
0060h:	D9	62	AE	29	B8	24	6E	09	E8	A9	25	B2	4A	22	EE	B2	Ůb@),Œn.^@%²J"î²	
0070h:	CA	66	0C	65	7E	7F	9C	87	E7	3B	BF	BB	CC	1C	B8	57	ÊfĀe~.œ†¢;¿»İ.ŸW	
0080h:	DF	D7	7C	3E	FF	1A	CE	9C	33	E7	33	E7	9C	99	D7	9D	ß× >Ÿ.îœ3¢3¢œ™×	

得到flag: flag{4537ec3bd52ba2b41c4a780db841efc3ddccc4a4}

???

flag{4537ec3bd52ba2b41c4a780db841efc3ddccc4a4}

数据包分析-Easy

题目给了一个http3.pcap，进行流量分析，根据题目描述，直接导出HTTP对象列表，发现flag.php:

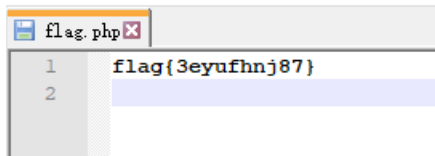
Wireshark · 导出 · HTTP 对象列表

分组	主机名	内容类型	大小	文件名
92	192.168.111.157	text/html	1334 bytes	xssbot
304	192.168.111.157	text/html	11 kB	\
309	192.168.111.157	image/png	3338 bytes	ubuntu-logo.png
312	192.168.111.157	text/html	17 bytes	flag.php

文本过滤器:

Save Save All Close Help

导出后，即可在里面看到flag: flag{3eyufhnj87}



EasyRSA

题目给了加密脚本如下：

```
from Crypto.Util.number import getPrime, inverse
flag = 'flag{a-z0-9}'
nbits = 2048
p = getPrime(nbits / 2)
q = getPrime(nbits / 2)
assert p != q
N = p * q
e = 0x10001
phiN = (p - 1) * (q - 1)
d = inverse(e, phiN)
phint = d % (p - 1)
qhint = q % (p - 1)

def str2int(s):
    return int(s.encode('hex'), 16)

with open('pubkey.txt', 'w') as f:
    f.write(str(e) + '\n')
    f.write(str(N) + '\n')
    f.write(str(phint) + '\n')
    f.write(str(qhint) + '\n')

plain = str2int(flag)

c = pow(plain, e, N)
with open('cipher.txt', 'w') as f:
    f.write(hex(c))
```

但实际上没什么用，只是告诉了我们pubkey.txt里面分别对应的内容，即：

```
e = 65537
N = 169697521655091326276302669687488543303407016921254276195598364883502982347355714803530786149755803784673559
5233375531393551651377355216339295265632149026845255660485896689995624210700841055865792434429565193929732800793
2245741660910510032969527598266270511004857674534802203387399678231880894252328431133224653544948661283777645985
0282076095266548166451555589151977450625691245874123787160498140406706650794800556448734707566029933872619395669
5880629659978294346014158204515097103121121861709128328411857371402926633122732739872426517035264679406870278964
5980810005549376399535110820052472419846801809110186557162127
phint = 17816257752910288702696852575211080903295430127287054677825469139515376426236217692464411221899486713749
9094640516445986741064682559131062261837911628429379409097029216526333474939300999933541308990379662432616803961
8287078192646490488534062803960418790874890435529393047389228718835244370645215187358081805
qhint = 10450978538445496868237560522155647855468044038637044818500580506745232482415364474390893285539835615564
3329103327081015390485282420117627363425726507632705012654406746474893754388523772804945201683491546048002186656
28586180057648386859933274414030182106920793492451577530884172876623074281199949317487086975
```

而这里的phint和qhint实际上就是常说的dp和dq。

还给了密文如下：

```
0x7b5d1ea2d92df27239817ce8d885e1f66569dd41e075efc13d09dd1df673a8fba68ec7487c1552028e9eb9ba6663983f96d01925bbdfd1
8398e44f970257fa0f96b6ec915d05d637ebb4c8f4c56c44b2bb46bd1afe5a67acd640585dccf1681155308c0663cb57fccdc10c097c454d
afdd2a96ccd08e9a2a8b0a9727bbe9945e579b0652d1c1d826305f0dd716cfb647cadb8eca1a0286dfb938b60b89981403d4faa6df54cfac
0fa4699c97aeba6e82ab575cd6aa4421018cf9b404836c02b5301dbc475a0bcc5eef86bcbeb89a73355dbeb80e7b4d23c7a39f32c6b61381
25c73892633f46b0bf1114aa67f09e1d394dfa4020e318f7d8004b84fc835b1ee9L
```

所以这题我们有c、e、n、dp、dq，足够解出明文了。

参考这篇文章：[https://skysec.top/2018/08/24/RSA%E4%B9%8B%E6%8B%92%E7%BB%9D%E5%A5%97%E8%B7%AF\(1\)/](https://skysec.top/2018/08/24/RSA%E4%B9%8B%E6%8B%92%E7%BB%9D%E5%A5%97%E8%B7%AF(1)/)

最终解密脚本如下：

```
import gmpy2
import libnum
e = 65537
n = 169697521655091326276302669687488543303407016921254276195598364883502982347355714803530786149755803784673559
5233375531393551651377355216339295265632149026845255660485896689995624210700841055865792434429565193929732800793
2245741660910510032969527598266270511004857674534802203387399678231880894252328431133224653544948661283777645985
0282076095266548166451555589151977450625691245874123787160498140406706650794800556448734707566029933872619395669
5880629659978294346014158204515097103121121861709128328411857371402926633122732739872426517035264679406870278964
5980810005549376399535110820052472419846801809110186557162127
dp = 17816257752910288702696852575211080903295430127287054677825469139515376426236217692464411221899486713749909
4640516445986741064682559131062261837911628429379409097029216526333474939300999933541308990379662432616803961828
7078192646490488534062803960418790874890435529393047389228718835244370645215187358081805
c = "0x7b5d1ea2d92df27239817ce8d885e1f66569dd41e075efc13d09dd1df673a8fba68ec7487c1552028e9eb9ba6663983f96d01925b
bdfd18398e44f970257fa0f96b6ec915d05d637ebb4c8f4c56c44b2bb46bd1afe5a67acd640585dccf1681155308c0663cb57fccdc10c097
c454dafdd2a96ccd08e9a2a8b0a9727bbe9945e579b0652d1c1d826305f0dd716cfb647cadb8eca1a0286dfb938b60b89981403d4faa6df5
4cfac0fa4699c97aeba6e82ab575cd6aa4421018cf9b404836c02b5301dbc475a0bcc5eef86bcbeb89a73355dbeb80e7b4d23c7a39f32c6b
6138125c73892633f46b0bf1114aa67f09e1d394dfa4020e318f7d8004b84fc835b1ee9L"
c = int(c[:-1], 16)
for i in range(1, 65538):
    if (dp*e-1)%i == 0:
        if n%(((dp*e-1)/i)+1)==0:
            p=((dp*e-1)/i)+1
            q=n/(((dp*e-1)/i)+1)
            phi = (p-1)*(q-1)
            d = gmpy2.invert(e, phi)%phi
            print libnum.n2s(pow(c, d, n))
```

运行得到flag: flag{6b85823e6f121a7bb3407ff2e9f5f2f27efcc5a6}

```
PS F:\CTF\领航杯\EasyRAS\EasyRSA> python2 .\exp.py  
flag{6b85823e6f121a7bb3407ff2e9f5f2f27efcc5a6}
```