

BUUCTF MISC刷题笔记(四)

原创

[z.volcano](#) 于 2021-05-08 14:21:45 发布 8700 收藏 6

分类专栏: [# buuoj # 刷题](#)

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/weixin_45696568/article/details/116515458

版权



[buuoj](#) 同时被 2 个专栏收录

7 篇文章 1 订阅

订阅专栏



刷题

8 篇文章 0 订阅

订阅专栏

BUUOJ

Misc

[\[RCTF2019\]disk](#)

[\[MRCTF2020\]小O的考研复试](#)

[\[HDCTF2019\]你能发现什么蛛丝马迹吗](#)

[\[BSidesSF2019\]table-tennis](#)

[\[CFI-CTF 2018\]webLogon capture](#)

[key不在此处](#)

[很好的色彩呢?](#)

[\[INSHack2018\]Self Congratulation](#)

[\[GUET-CTF2019\]520的暗示](#)

[greatestescape](#)

[\[ACTF新生赛2020\]frequency](#)

Misc

[\[RCTF2019\]disk](#)

这个题本来是有提示的, buu这里没给

An otaku used VeraCrypt to encrypt his favorites.

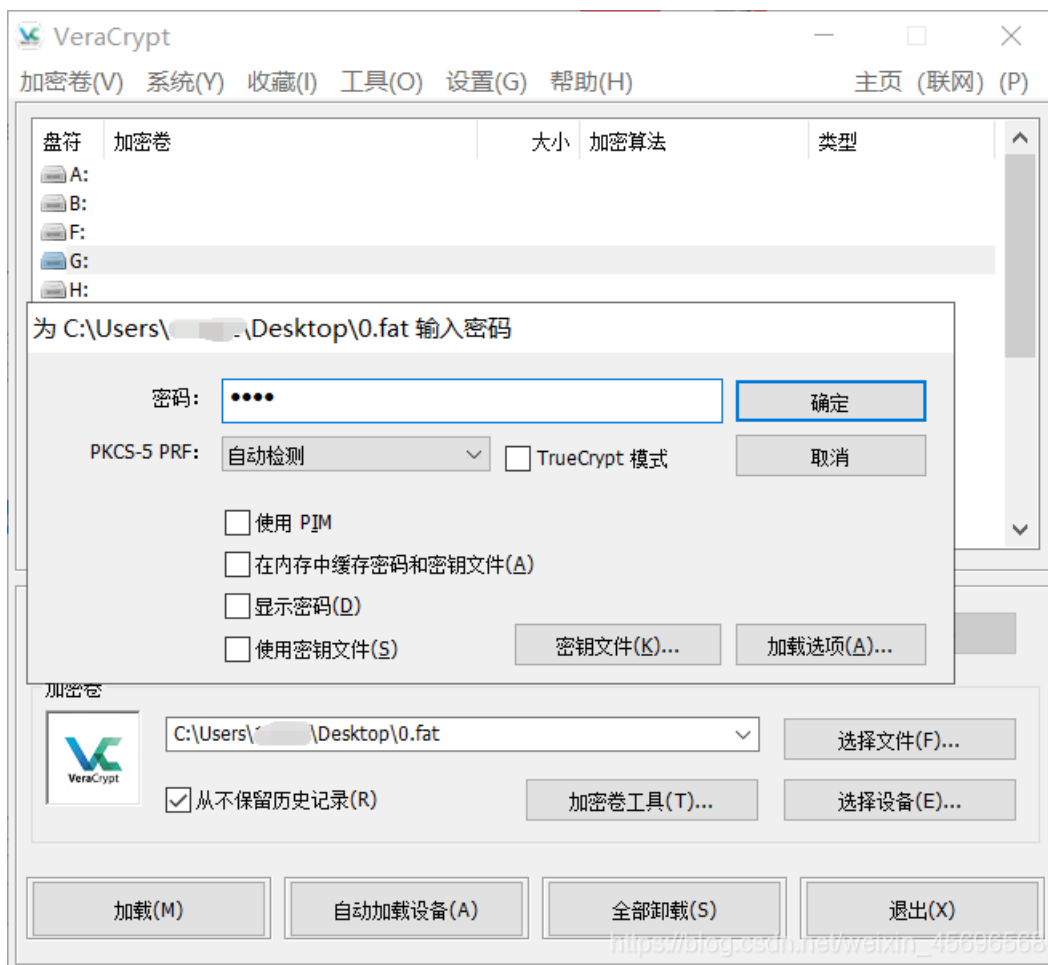
Password: rctf

Flag format: rctf{a-zA-Z0-9_}

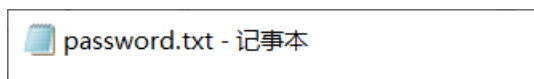
下载得到 `encrypt.vmdk`，winhex瞅一下，得到半段flag: `rctf{unseCure_quick_form4t_vo1ume}`，出题人还是很贴心的，怕我们看不到，重复了这么多次

```
70 | rctf{unseCure_qu  
6D | ick_form4t_volum  
71 | erctf{unseCure_q  
75 | uick_form4t_volu  
5F | merctf{unseCure_  
31 | quick_form4t_vo1  
65 | umerctf{unseCure  
6F | _quick_form4t_vo  
72 | lumerctf{unseCur  
76 | e_quick_form4t_v  
75 | olumerctf{unseCu  
5F | re_quick_form4t_  
43 | volumerctf{unseC
```

用7z解压这个vmdk文件，得到0.fat，然后用 VeraCrypt 挂载 0.fat



挂载完后打开对应的盘，有两个文件，txt和 `70056639_useless_file_for_ctf_just_ignore_it.jpg`



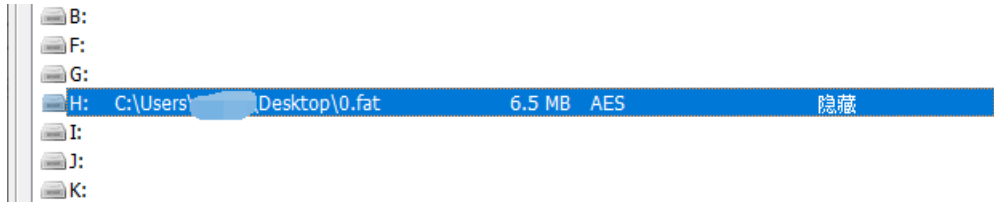
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

Password 2: RCTF2019

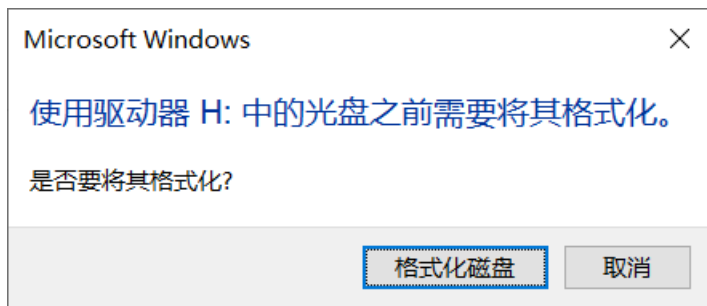
You're late... So sad

https://blog.csdn.net/weixin_45696568

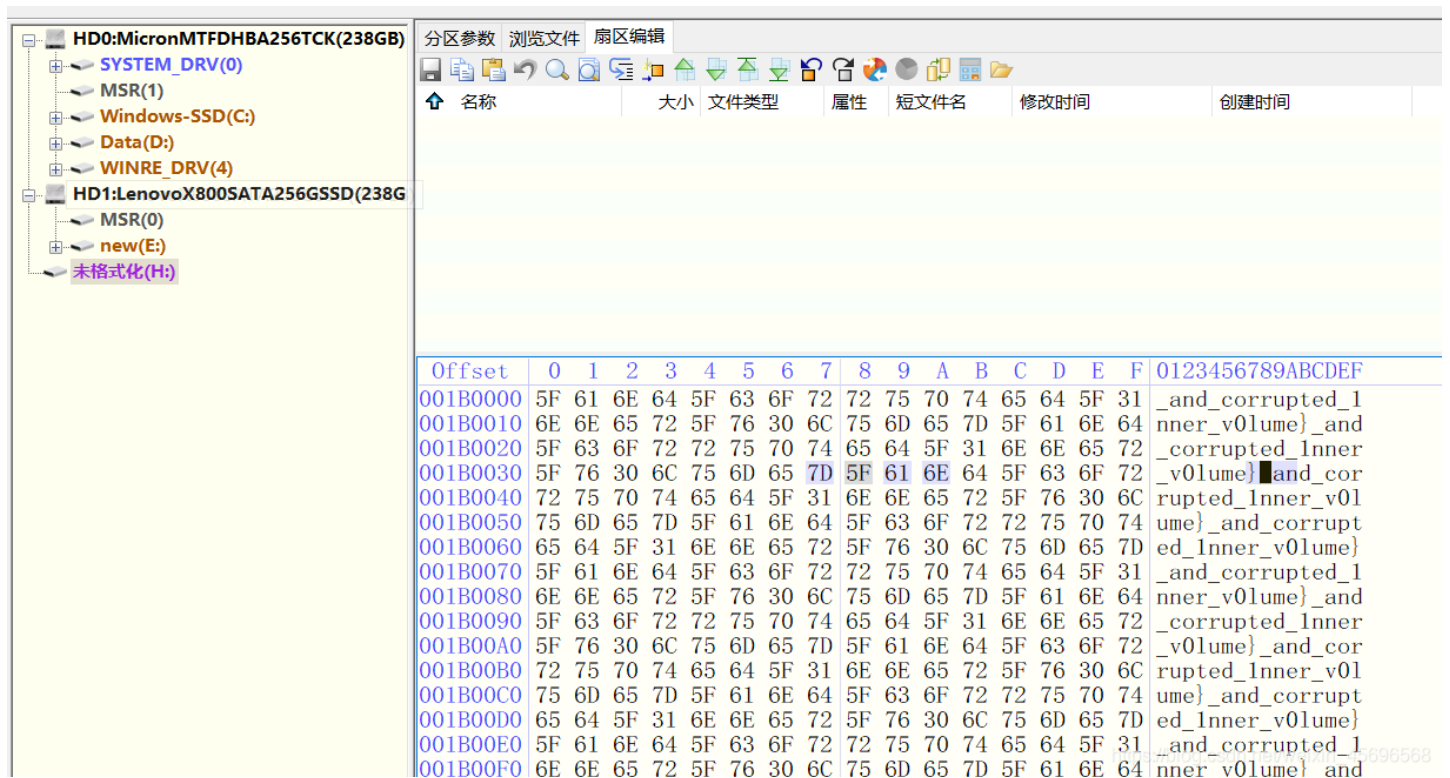
注意到jpg的图片名, 和txt中的提示, 回过头使用密码 RCTF2019 重新挂载, 注意到这是一个隐藏分区



这个隐藏分区无法直接访问, 所以使用 DiskGenius



找到另一半flag, 当然也可以使用winhex查看, 工具 → 打开磁盘 → 选择磁盘



最终flag: rctf{unseCure_quick_form4t_vo1ume_and_corrupted_inner_v0lume}

[MRCTF2020]小O的考研复试

简单的数学题

今天得题目很简单

$$\underbrace{kk \dots k}_{n \uparrow k}$$

$$\underbrace{22 \dots 2}_{19260817 \uparrow 2}$$

而上述第二个式子对 $(1e9+7)$ 取模的值是我们要上交的flag.

这题目很简单吧。大家快A掉吧。

flag直接提交数字即可。

https://blog.csdn.net/weixin_45696568

做数学题就行，跑出来结果是 `577302567`

[HDCTF2019]你能发现什么蛛丝马迹吗

拿到一个镜像文件，分析一波

```
python vol.py -f memory.img imageinfo
```

```
INFO      : volatility.debug      : Determining profile based on KDBG search...
           Suggested Profile(s) : Win2003SP0x86, Win2003SP1x86, Win2003SP2x86 (Instantiated with Win2003SP2x86)
           AS Layer1           : IA32PagedMemoryPae (Kernel AS)
           AS Layer2           : FileAddressSpace (/home/volcano/桌面/volatility/memory.img)
           PAE type            : PAE
           DTB                  : 0xe02000L
           KDBG                 : 0x8088e3e0L
           Number of Processors : 1
           Image Type (Service Pack) : 1
           KPCR for CPU 0      : 0xffdff000L
           KUSER_SHARED_DATA   : 0xffdf0000L
           Image date and time : 2019-04-25 08:43:06 UTC+0000
           Image local date and time : 2019-04-25 16:43:06 +0800
```

看一下进程：`python vol.py -f memory.img --profile=Win2003SP1x86 pslist`

Offset(V)	Name	PID	PPID	Thds	Hnds	Sess	Wow64	Start	Exit
-----------	------	-----	------	------	------	------	-------	-------	------

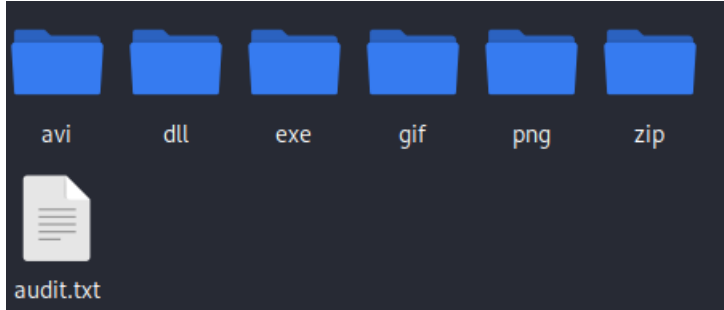
0x81f8f020	System	4	0	56	319	-----	0		
0xfe2f8448	smss.exe	380	4	3	18	-----	0	2018-12-07	16:20:54 UTC+0000
0xfe2caa60	csrss.exe	516	380	12	509	0	0	2018-12-07	16:21:00 UTC+0000
0xfe304298	winlogon.exe	580	380	25	504	0	0	2018-12-07	16:21:04 UTC+0000
0xfe2fdd88	services.exe	648	580	16	303	0	0	2018-12-07	16:21:05 UTC+0000
0xfe2e5530	lsass.exe	660	580	38	458	0	0	2018-12-07	16:21:05 UTC+0000
0xfe2f9290	vmacthlp.exe	880	648	1	26	0	0	2018-12-07	16:21:06 UTC+0000
0xfe34d658	svchost.exe	932	648	6	93	0	0	2018-12-07	16:21:07 UTC+0000
0xfde05020	svchost.exe	984	648	10	268	0	0	2018-12-07	16:21:07 UTC+0000
0xfddf4c08	svchost.exe	1040	648	10	138	0	0	2018-12-07	16:21:08 UTC+0000
0xfdde020	svchost.exe	1072	648	15	168	0	0	2018-12-07	16:21:08 UTC+0000
0xfdde9a70	svchost.exe	1096	648	79	1271	0	0	2018-12-07	16:21:08 UTC+0000
0x81e5a7d0	spoolsv.exe	1668	648	14	151	0	0	2018-12-07	16:21:26 UTC+0000
0xfe7385e8	msdtc.exe	1700	648	16	166	0	0	2018-12-07	16:21:26 UTC+0000
0xfddb7b18	svchost.exe	1800	648	2	54	0	0	2018-12-07	16:21:27 UTC+0000
0xfddb1020	svchost.exe	1848	648	2	37	0	0	2018-12-07	16:21:27 UTC+0000
0xfdda8020	VGAAuthService.e	1920	648	2	65	0	0	2018-12-07	16:21:28 UTC+0000
0xfdc6eb18	vmtoolsd.exe	300	648	8	244	0	0	2018-12-07	16:21:36 UTC+0000
0xfe3d5600	svchost.exe	484	648	16	135	0	0	2018-12-07	16:21:40 UTC+0000
0xfe3d4cb0	dllhost.exe	736	648	22	239	0	0	2018-12-07	16:21:41 UTC+0000
0xfe30ed88	dllhost.exe	1052	648	22	236	0	0	2018-12-07	16:21:42 UTC+0000
0xfdc40638	wmiprvse.exe	1368	932	9	215	0	0	2018-12-07	16:21:44 UTC+0000
0xfdc1bb18	explorer.exe	1992	1664	16	386	0	0	2018-12-07	16:21:50 UTC+0000
0xfdc1ad88	vssvc.exe	2040	648	7	112	0	0	2018-12-07	16:21:51 UTC+0000
0xfdbd3418	vmtoolsd.exe	1596	1992	6	166	0	0	2018-12-07	16:22:01 UTC+0000
0xfdbd2110	ctfmon.exe	1840	1992	1	69	0	0	2018-12-07	16:22:01 UTC+0000
0xfdbbc330	conime.exe	1792	1636	1	32	0	0	2018-12-07	16:22:16 UTC+0000
0xfdba5320	wmiprvse.exe	1128	932	8	165	0	0	2018-12-07	16:22:24 UTC+0000
0xfdb00020	vmtoolsd.exe	2224	1636	5	116	0	0	2018-12-07	16:22:44 UTC+0000

0xt0b90930	wuauclt.exe	2224	1096	5	116	0	0	2018-12-07 16:22:44	UTC+0000
0xfdb6a638	DumpIt.exe	3660	1992	1	26	0	0	2019-04-25 08:43:04	UTC+0000

发现最后这个 `DumpIt.exe` 有点可疑，把它dump下来

```
python vol.py -f memory.img --profile=Win2003SP1x86 memdump -p 1992 --dump-dir=.
```

再foremost分离一下



其中png里面有两张有用的图



```
key: Th1s_1s_K3y00000
iv: 1234567890123456
```

扫码得到: `jfXvUoypb8p3zvmPks8kJ5Kt0vmEw0xUZyRG0icraY4=`

这里给出了偏移量iv, 应该是AES加密

AES加密模式: ECB 填充: zeropadding 数据块: 128位 密码: Th1s_1s_K3y000 偏移量: 12345678901234 输出: base64

待加密、解密的文本: jfXvUoypb8p3zvmPks8kJ5Kt0vmEw0xUZyRGOicraY4=

↑ 将你电脑文件直接拖入试试^-^

AES加密 AES解密

AES加密、解密转换结果(base64了): flag{FOuNd_s0m3th1ng_1n_M3mory}

https://blog.csdn.net/weixin_45696568

[BSidesSF2019]table-tennis

给了一个流量包, wireshark打开

从这个位置开始, 打印了一串字符串

No.	Time	Source	Destination	Protocol	Length	Info
1112	172.217.5.100	192.168.10.212	ICMP	Echo (ping) reply	98	id=0x1332, seq=1/256, ttl... 98
1135	192.168.10.212	172.217.5.100	ICMP	Echo (ping) request	98	id=0x1334, seq=1/256, ttl... 98
1140	172.217.5.100	192.168.10.212	ICMP	Echo (ping) reply	98	id=0x1334, seq=1/256, ttl... 98
1180	192.168.10.212	172.217.5.100	ICMP	Echo (ping) request	98	id=0x1336, seq=1/256, ttl... 98
1185	172.217.5.100	192.168.10.212	ICMP	Echo (ping) reply	98	id=0x1336, seq=1/256, ttl... 98
1214	192.168.10.212	172.217.0.36	ICMP	Echo (ping) request	98	id=0x1338, seq=1/256, ttl... 98

> Frame 1140: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface wlp4s0, id 0
> Ethernet II, Src: Cisco_62:ee:ff (b0:fa:eb:62:ee:ff), Dst: IntelCor_4b:f0:c3 (e4:b3:18:4b:f0:c3)
> Internet Protocol Version 4, Src: 172.217.5.100, Dst: 192.168.10.212
> Internet Control Message Protocol

Offset	Hex	ASCII
0000	e4 b3 18 4b f0 c3 b0 fa eb 62 ee ff 08 00 45 00	·K· ·b· ·E·
0010	00 54 00 00 00 00 37 01 05 f0 ac d9 05 64 c0 a8	·T· ·7· · ·d·
0020	0a d4 00 00 87 b6 13 34 00 01 d0 dc 74 5c 00 00	· · · · ·4 · ·t\ · ·
0030	00 00 eb e8 0d 00 00 00 00 00 6f 62 28 22 51 31	· · · · · · · · · ·ob("Q1
0040	52 47 6f 62 28 22 51 31 52 47 6f 62 28 22 51 31	RGob("Q1 RGob("Q1
0050	52 47 6f 62 28 22 51 31 52 47 6f 62 28 22 51 31	RGob("Q1 RGob("Q1
0060	52 47	RG

到这里结束

1255	192.168.10.212	172.217.5.100	ICMP	Echo (ping) request	id=0x133a, seq=1/256, ttl...	98
1259	172.217.5.100	192.168.10.212	ICMP	Echo (ping) reply	id=0x133a, seq=1/256, ttl...	98
1277	192.168.10.212	172.217.0.36	ICMP	Echo (ping) request	id=0x133c, seq=1/256, ttl...	98
1282	172.217.0.36	192.168.10.212	ICMP	Echo (ping) reply	id=0x133c, seq=1/256, ttl...	98
1306	192.168.10.212	172.217.6.68	ICMP	Echo (ping) request	id=0x133e, seq=1/256, ttl...	98

```
> Frame 1282: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface wlp4s0, id 0
> Ethernet II, Src: Cisco_62:ee:ff (b0:fa:eb:62:ee:ff), Dst: IntelCor_4b:f0:c3 (e4:b3:18:4b:f0:c3)
> Internet Protocol Version 4, Src: 172.217.0.36, Dst: 192.168.10.212
> Internet Control Message Protocol
```

```
0000 e4 b3 18 4b f0 c3 b0 fa eb 62 ee ff 08 00 45 00  ...K...  b...E
0010 00 54 00 00 00 00 37 01 0b 30 ac d9 00 24 c0 a8  ·T...7·  0...$..
0020 0a d4 00 00 fc 41 13 3c 00 01 d1 dc 74 5c 00 00  .....A·<  ....t\..
0030 00 00 ba ba 04 00 00 00 00 00 5a 31 41 77 62 6d  .....  ..Z1Awbm
0040 64 39 5a 31 41 77 62 6d 64 39 5a 31 41 77 62 6d  d9Z1Awbm d9Z1Awbm
0050 64 39 5a 31 41 77 62 6d 64 39 5a 31 41 77 62 6d  d9Z1Awbm d9Z1Awbm
0060 64 39                                     d9
```

https://blog.csdn.net/weixin_45696568

字符串拼接起来得到 `Q1RGe0p1c3RBuZ0FiMHV0UDFuZ1Awbmd9`

base64解码得到 `CTF{JustAS0ngAb0utP1ngP0ng}`

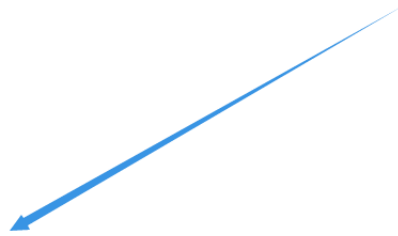
[CFI-CTF 2018]webLogon capture

简简单单

No.	Source	Destination	Protocol	Info	Length
1	10.10.40.241	10.10.20.4	HTTP	POST /auth/login/ HTTP/1.1 (application/json)	902
2	10.10.20.4	10.10.40.241	HTTP	HTTP/1.1 401 UNAUTHORIZED (application/json)	634
3	10.10.40.241	10.10.20.4	TCP	47428 → 80 [ACK] Seq=837 Ack=569 Win=246 Len=0...	66
4	10.10.40.241	10.10.20.4	HTTP	POST /auth/login/ HTTP/1.1 (application/json)	902
5	10.10.20.4	10.10.40.241	HTTP	HTTP/1.1 401 UNAUTHORIZED (application/json)	634
6	10.10.40.241	10.10.20.4	TCP	47428 → 80 [ACK] Seq=1673 Ack=1137 Win=255 Len...	66
7	10.10.40.241	10.10.20.4	HTTP	POST /auth/login/ HTTP/1.1 (application/json)	902
8	10.10.20.4	10.10.40.241	HTTP	HTTP/1.1 401 UNAUTHORIZED (application/json)	634
9	10.10.40.241	10.10.20.4	TCP	47428 → 80 [ACK] Seq=2509 Ack=1705 Win=264 Len...	66

> Frame 1: 902 bytes on wire (7216 bits), 902 bytes captured (7216 bits) on interface wlp5s0, id 0
> Ethernet II, Src: IntelCor_6f:68:73 (ac:ed:5c:6f:68:73), Dst: Routerbo_55:a1:de (6c:3b:6b:55:a1:de)
> Internet Protocol Version 4, Src: 10.10.40.241, Dst: 10.10.20.4
> Transmission Control Protocol, Src Port: 47428, Dst Port: 80, Seq: 1, Ack: 1, Len: 836
> Hypertext Transfer Protocol
▼ JavaScript Object Notation: application/json
 ▼ Object
 > Member Key: email
 ▼ Member Key: password
 String value: %20%43%46%49%7b%31%6e%73%33%63%75%72%33%5f%6c%30%67%30%6e%7d%20
 Key: password

0290	34 50 67 2d 37 4f 78 5f	7a 79 31 52 34 2d 5f 78	4Pg-70x_ zy1R4- x
02a0	45 59 49 63 4f 6f 36 74	45 6f 4b 30 7a 4b 6c 4e	EYIc0o6t EoK0zKlN
02b0	46 4c 47 76 56 63 6d 63	7a 5f 5a 45 4d 69 6e 37	FLGvVcmc z_ZEMin7
02c0	31 79 4c 30 30 69 72 53	58 4a 57 35 51 74 6f 39	1yL00irs XJW5Qto9
02d0	38 66 30 77 32 36 4b 45	32 4d 2e 44 6d 64 69 76	8f0w26KE 2M.Dmdiv
02e0	67 2e 46 75 77 43 65 42	70 57 31 74 31 48 74 5a	g.FuwCeB pWit1HTZ
02f0	6f 5f 70 57 44 4b 39 2d	4f 6f 38 54 51 0d 0a 44	o_pWdK9- Oo8TQ...D
0300	4e 54 3a 20 31 0d 0a 43	6f 6e 6e 65 63 74 69 6f	NT: 1..C onnectio
0310	6e 3a 20 6b 65 65 70 2d	61 6c 69 76 65 0d 0a 0d	n: keep- alive...
0320	0a 7b 22 65 6d 61 69 6c	22 3a 22 74 65 73 74 40	..{"email ":"test@
0330	65 78 61 6d 70 6c 65 22	2c 22 70 61 73 73 77 6f	example" ,"passwo
0340	72 64 22 3a 22 25 32 30	25 34 33 25 34 36 25 34	rd": "%20 %43%46%4
0350	39 25 37 62 25 33 31 25	36 65 25 37 33 25 33 33	9%7b%31% 6e%73%33
0360	25 36 33 25 37 35 25 37	32 25 33 33 25 35 66 25	%63%75%7 2%33%5f%
0370	36 63 25 33 30 25 36 37	25 33 30 25 36 65 25 37	6c%30%67 %30%6e%7
0380	64 25 32 30 22 7d		d%20"}]



	Input length: 6 lines:		
	<pre>%20%43%46%49%7b%31%6e%73%33%63%75%72%33%5f%6c%30%67%30%6e%7d%20</pre>		
<table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Output</td> <td style="text-align: right;">start: 22 time: end: 22 length: length: 0 lines:</td> </tr> </table>		Output	start: 22 time: end: 22 length: length: 0 lines:
Output	start: 22 time: end: 22 length: length: 0 lines:		
<pre>CFI{1ns3cur3_log0n}</pre> <p style="text-align: right;">https://blog.csdn.net/weixin_45696568</p>			

key不在此处

扫码得到网址

```
https://cn.bing.com/search?
q=key%E4%B8%8D%E5%9C%A8%E8%BF%99%E9%87%8C&m=1021089710337556653100525310297505354515550505052102525
65552549954102985610151519851503755668&qsn&form=QBRE&sp=-1&sc=0-
38&sk=&cvid=2CE15329C18147CBA4C1CA97C8E1BB8C
```



直接访问是得不到flag的，这里注意

到 `m=10210897103375566531005253102975053545155505050521025256555254995410298561015151985150375568`

这一串字符，可以分割为102、108、97... 写脚本转换

```
from urllib.parse import unquote

s="10210897103375566531005253102975053545155505050521025256555254995410298561015151985150375568"
x=len(s)//2
flag=""
for i in range(x):
    if len(s) != 0:
        if int(s[:3])< 127:
            flag += chr(int(s[:3]))
            s = s[3:]
        else:
            flag += chr(int(s[:2]))
            s = s[2:]
print(unquote(flag, 'utf-8')) #url解码一次
```

跑完出flag

很好的色彩呢？

下载得到一个gif



这六个竖杠，看起来颜色有细微的差别，用 `ps` 打开，使用取色器查看每部分的颜色

结果如下：

```
8b8b61  
8b8b61  
8b8b70  
8b8b6a  
8b8b65  
8b8b73
```

发现只有最后两位不同，把它们连起来：`6161706a6573`

直接提交是错的，需要转成字符

Recipe

From Hex

Delimiter
Auto

Input

616170616573

Output

aapjes
https://blog.csdn.net/weixin_45696568

`flag{aapjes}`

[INSHack2018]Self Congratulation

图片这个位置有点怪



放大看



把黑的换成1，白的换成0，这块数据不大，挺容易的

00110001001
10010001100
11001101000
01101010011
01100011011
10011100000

再转成字符, 在线网站

汉字二进制转换器

001100010011001000110011001101000011010100110110001101110011100000

✕ UTF-8 汉字转 2进制
✓ UTF-8 2进制 转汉字
转换 清空 复制

12345678

本页面可实现任意字符(非GB18030字符集)与2、26进制编码的相互转换。注意:

https://blog.csdn.net/waixia_45695565

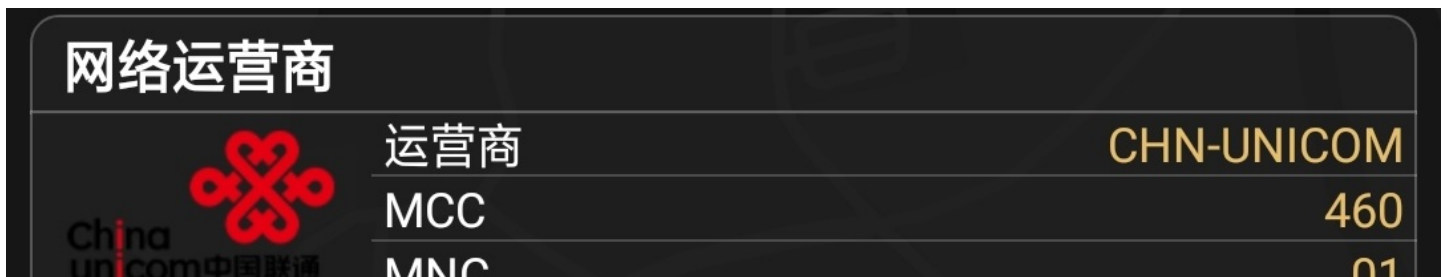
flag{12345678}

[GUET-CTF2019]520的暗示

好家伙, dat文件, 前几天刚在bugku做过, 可以参考我这篇博客的 [FileStoragedat](#)



解出来这样一张图



服务小区

数据网	LTE
小区类型	LTE
TAC	30542
PCI	150
ECI	180375296(704591-0)
EARFCN ▼ ▲	1650/19650
FREQUENCY ▼ ▲	1850/1755 MHz
BAND	3(FDD)

信号强度

RSSI
-57

RSRP
-89

RSRQ
-9

SINR
11.6

-100

-120

1分钟前

40S前

20S前

现在

邻小区

NO VALUE

数据导出

https://blog.csdn.net/weixin_45696568

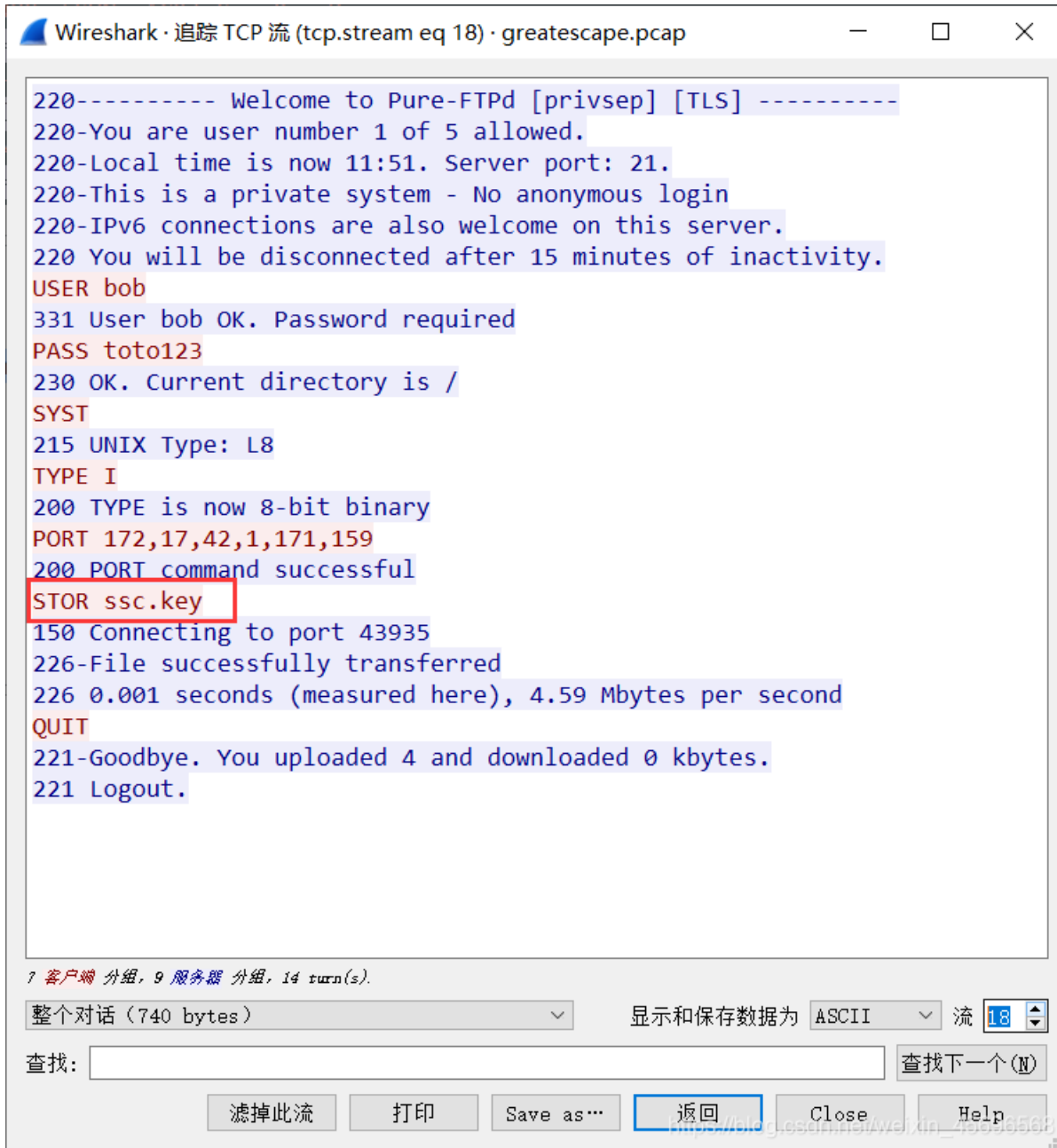
然后就是 [根据LTE定位基站地址](#)，不过没找到合适的查询网站

flag{桂林电子科技大学花江校区}

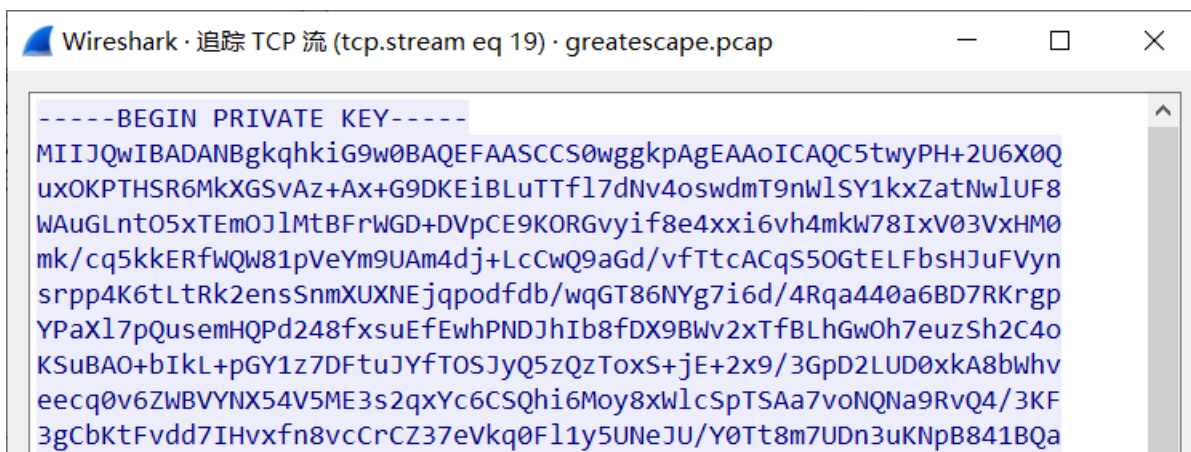
gatescape

得到一个流量包，用wireshark分析，追踪TCP流，挨个看

在流18这里看到一个 `ssc.key`，而且这里是 `TLS` 协议

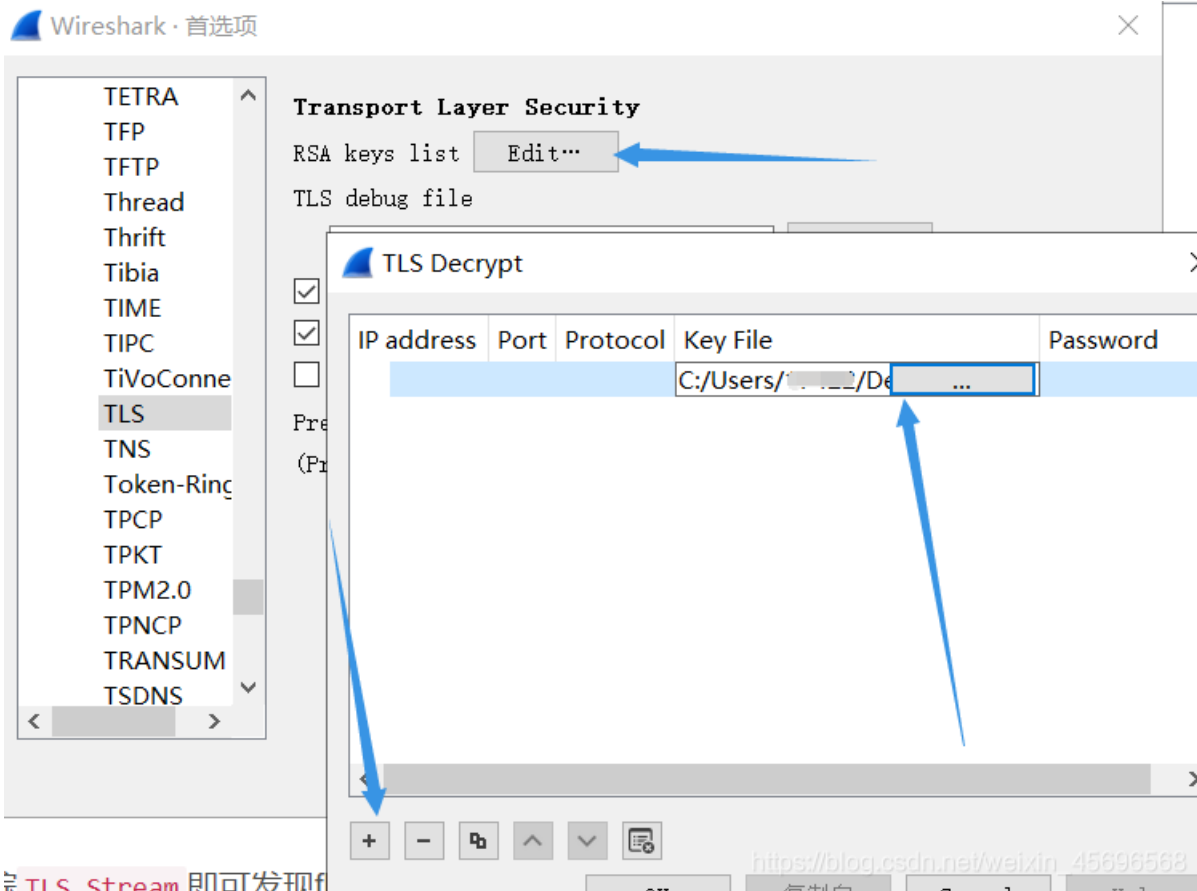


再往下看，在流19发现私钥，把它保存为 `1.key`

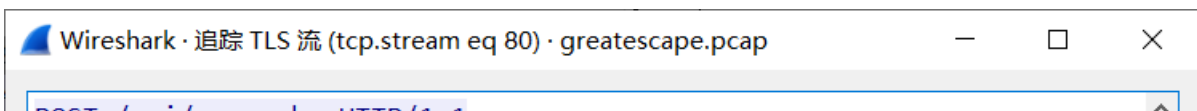




接下来 编辑 -> 首选项 -> protocols -> TLS，再把1.key导入，就可以解密出内容了



再去流80追踪TLS流



```
POST /api/user.php HTTP/1.1
Host: ssc.teaser.insomnihack.ch
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:50.0) Gecko/
20100101 Firefox/50.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
Referer: https://ssc.teaser.insomnihack.ch/login
Content-Length: 38
Cookie: PHPSESSID=3u5dqmfudc7ap1di0nmfjgtjm3
FLAG: INS{OkThatWasWay2Easy}
Connection: keep-alive
```

```
action=login&name=rogue&password=rogueHTTP/1.1 200 OK
Date: Fri, 20 Jan 2017 11:52:03 GMT
Server: Apache
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Content-Length: 36
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8
```

```
{"status": "SUCCESS", "name": "rogue"}
```

```
GET / HTTP/1.1
```

```
Host: ssc.teaser.insomnihack.ch
```

分组 2236, 6 客户端 分组, 6 服务器 分组, 11 turn(s). 点击选择。

整个对话 (5327 bytes)

显示和保存数据为 ASCII

查找:

查找下一个(N)

滤掉此流

打印

Save as...

返回

Close

Help

[ACTF新生赛2020]frequency

·你猜 **flag** 在哪？有两截哦。 ←

a2draGxmY290bnRpdWZwZ2hodGN3dWprY2ttb3ducGNrbXdseWd0bHBtZmtneW
FhaWh1Y2RsYXRveXVjb2lnZ3JwbGt2a2Ftcmt0cXp4ZW1taXdrbGh1YWVrY2Vvb
HBvY2ZtdGFobWdmbWF2YWpuYmNwbWx0anRwdWZqY2FwY3RvanBqYmZm
Ympid2h1YWxnZ3lqbmFtY2JmeWFjamJheGtpeGxtbXFpa3NtcHRxeW9qZXJ0Zm
VrdGR4ZHh4YnRyeGNhbmd5bXNpbWh2dXdrdGV4c2dscnRwZ2FrdGJtZnVjZ3Zu
bXRqdWZvZWt5bXRsaW14ZGlqanB4eWl0YWJwbWt1Y2NubGtwb2V0Z2NkY3B
vc2tpenZ5eHJ0enhyYXh0bm9paHFjeGZvYWFhbHBhanlja2VrYnljZnZqb21sbGtha
md5bWdmZGNycGVxa2xmc2NtZWppY3BqaWtjcHBhY3h5ZXZma3ljcHBia2R6Y

这串字符的结尾没有 = ，不确定是什么

同时在详细信息里发现一串字符



0ZWZtemNxaWRvZmd0ZnFnYmFkaWNubWhvdGlvbm9iZnlubGdvenRkYXZ2aW14b2JvdGlra2Z4d2lyb3JwZmNjdXpob3BoZmRjaWVrY2p5b21amtjZ2Zmam51bmhvcGFkdGZndG1sdA==

两段拼在一起，base64解码

```
import base64
f = open("1.txt", "r")
txt=f.read().replace(".", "")
f.close()
print(base64.b64decode(txt))
```

得到的结果保存在2.txt中，结合题目名，开始 **词频统计**

```
f = open("2.txt", "r")
txt=f.read().replace(" ", "")
f.close()
d={}
for i in txt:
    d[i] = d.get(i,0)+1
ls=list(d.items())
ls.sort(key=lambda x:x[1],reverse =True)
for i in ls:
    print(i[0],end="")
```

结果: actfplokmiijnuhbygvrdxeszwq{

所以: flag{plokmiijnuhbygvrdxeszwq}