CCTF 2016 WriteUp（部分）

原创

Bendawang 于 2016-04-26 14:55:29 发布 5464 收藏

分类专栏： WriteUp 文章标签： CCTF CTF Writeup

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_19876131/article/details/51250495

版权

WriteUp 专栏收录该内容

24 篇文章 0 订阅

订阅专栏

上周参加了CCTF，也算是练了一段时间以来第一次正式打比赛并打完了全程的，最后结果拿了个11名，还算满意，毕竟是第一次。但这次比赛强队确实不多，而且我们队和别人的差距也是很大了，毕竟也还不到半年时间，还有太多东西要学，慢慢来吧，正确明年之前能够凭实力打进一次线下赛。

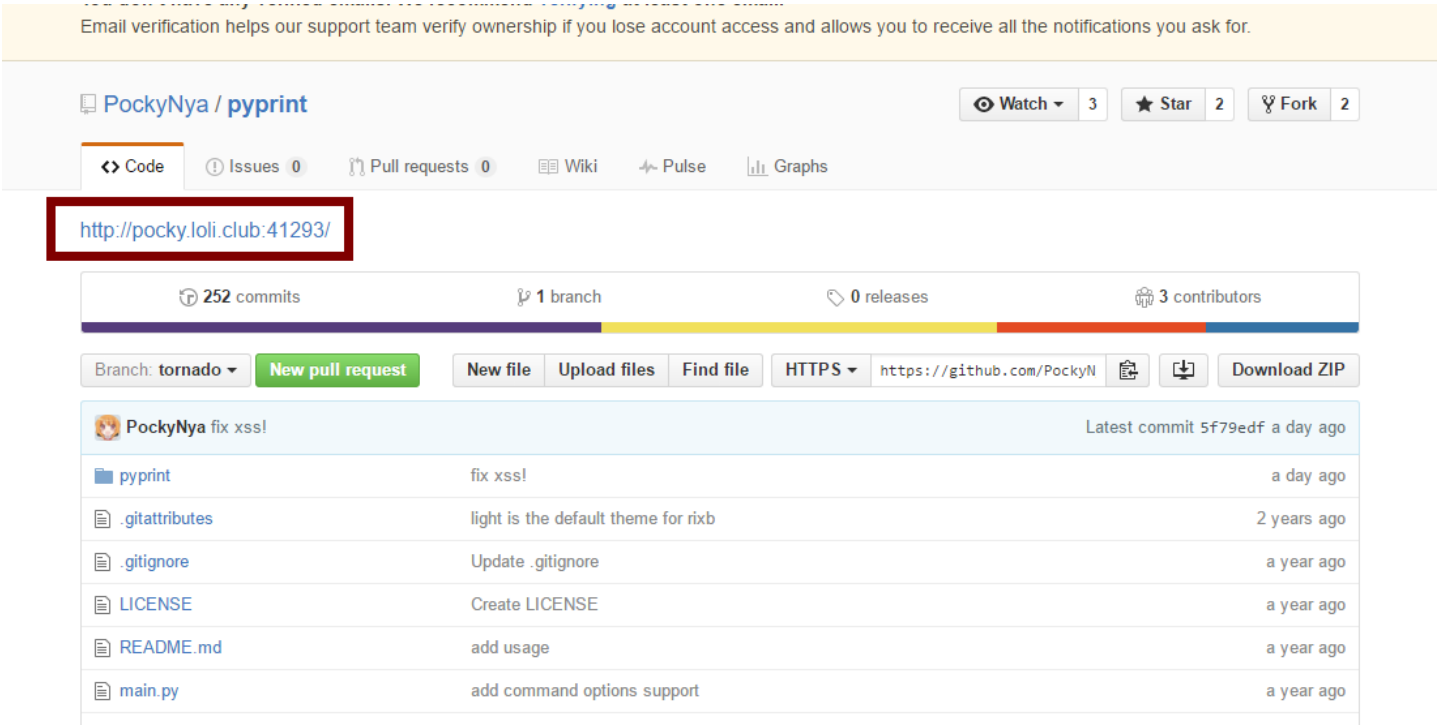
先贴一下大神的wp：http://bobao.360.cn/ctf/detail/159.html

在贴贴我们自己的wp把。

WEB 350

给了一个静态的页面，扫了扫也没发现什么东西，

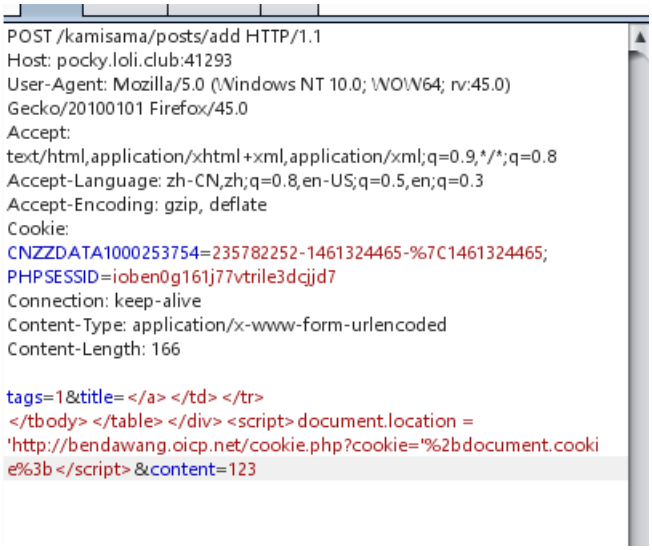
然后后来给了一个hint，说是找博客，好吧，疯狂寻找群里管理员们的博客，最后在`github`上找到了`Pocky Nya`，然后在`repositories`找到了目标网址



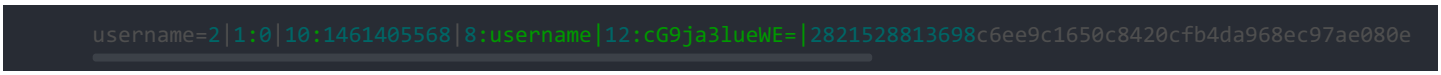
然后先把源码也下载下来，简单读了之后，大部分地方都有身份验证，而且获取参数地方也就两个，都没法XSS，后来意外中发现了源代码的`background.py`的'/kamisama/posts/add'这里是没有身份验证的，如下：

```
class AddPostHandler(BaseHandler):
    @tornado.web.authenticated
    def get(self):
        self.background_render('add_post.html', post=None)

    def post(self):
        title = self.get_argument('title', None)
        content = self.get_argument('content', None)
        tags = self.get_argument('tags', '').strip().split(',')
        if not title or not content:
            return self.redirect('/kamisama/posts/add')

        post = self.orm.query(Post.title).filter(Post.title == title).all()
        if post:
            return self.write('<script>alert("Title has already existed");window.history.go(-1);</scrip
        self.orm.add(Post(title=title, content=content, created_time=date.today()))
        self.orm.commit()
        return self.redirect('/kamisama/posts')
```

所以就可以伪造添加文章的数据包，然后设置存储型XSS获取管理员的cookie。如下所示：



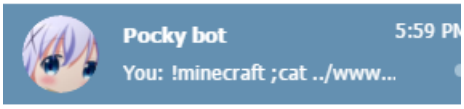
因为考虑到万一设在content里面管理员看不到就不好了，所以我把XSS设置在了title上面，虽然有点毒瘤，不过这样子就能百分百的获取到管理员的Cookie了，很快就有人登录了，结果第一次获取到的不是管理员的，是其他人获取之后登录的，虽然确实拿到cookie了，但是里面并没有flag，结果花费了N久又把各种代码重读了一遍，发现在`background.py`里面的`SignInHandler`中确实是把flag写进了cookie的，所以我又提交了一次上面的payload，这一次成功获取到了正确的cookie，如下：



然后按照ascii码转一下就出`flag`了`CCTF{CODE_AUDIT_BUSTERS}`

WEB 300

用之前获取到的cookie登录之后，访问`pocky.loli.club:41293/diaries`目录，能够看到新的提示说是telegram上布置了一个机器人，同时给出了机器人的lua代码，然后我们就去`telegram`社工一下pockynya就搜到了这个账户



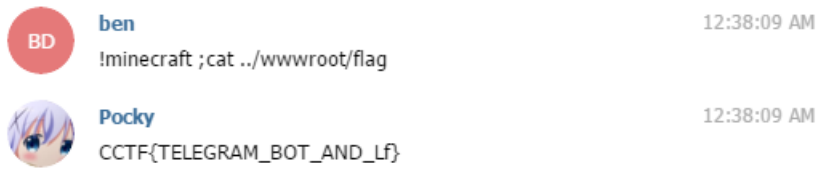
然后根据它博客里面给出的给的lua代码如下：

```
do

local function run(msg, matches)
  if matches[1] ~= '!minecraft' then
    operation = matches[1]
  else
    return "!minecraft start|stop|restart"
  end
  if string.find(operation, '&') or string.find(operation, '|') or string.find(operation, '`') then
    return "Invalid operation " .. operation
  end
  local t = io.popen('cd /home/telegram && ./mc ' .. operation)
  local a = t:read("*all")
  return a
end

return {
  description = "loli.club minecraft bot!",
  usage = "!minecraft start|stop|restart",
  patterns = {
    "^!minecraft$",
    "^!minecraft (.*)$"
  },
  run = run
}

end
```

所以我们可以用`;`断开前面的命令，然后就能执行我们的命令了，如下：

```
!minecraft ;xxx
```

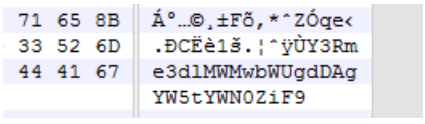
后面的`xxx`处就可以插入我们的命令行代码，最后的flag就在`../wwwroot/flag`里面
如图：



所以flag就是`CCTF{TELEGRAM_BOT_AND_Lf}`。

MISC 1

给了一个图片，在图片最后的32位有很连续的一堆字母，如下



base64解码一下得到flag，`ctf{we1c0me t0 anmactf!}`

MISC 2

这里，在第5560的内容里面发现这样的东西

`type s4cr4t.txt`

所以接下来这就是我们需要的文件内容，是个Base64编码的东西，解码之后是这个，`CCTF{do_you_like_sniffer}`，根据它的格式，他还需要的是MS打头的漏洞编码，根据随便谷歌`MS SMB 漏洞 溢出`了一下，试了几个之后就试出来了最后确切的漏洞编号就是MS08067，所以最后答案就是`MS08067CCTF{do_you_like_sniffer}`

re1:

IDA反编译后查看,发现运行要求包含三个参数,且在其中随机选一个进行测试,这里注意md5_custom函数没有用处....

```
srand(v3);
v19 = 8 * argc;
v20 = rand() % (argc - 1) + 1;
v21 = v19 - 1;
v16 = 16;
v4 = alloca(16 * ((v19 + 30) / 0x10u));
dest = (char *)(16 * ((unsigned int)&v15 >> 4));
v23 = 16 * ((unsigned int)&v15 >> 4);
v24 = v23 + v19;
*(_BYTE *)(v23 + v19) = 48;
v10 = argv[v20];
strcpy(dest, v10);
v5 = md5_custom(dest);
```

经过check函数测试

```
signed int __cdecl check(char *a1)
{
  signed int i; // [sp+Ch] [bp-4h]@1

  for ( i = 0; i <= 31; ++i )
  {
    if ( a1[i] != *(_BYTE *)(i + 134515840) )
      return 0;
  }
  return 1;
}
```

取内存中找下:发现字串f2332291a6e1e6154f3cf4ad8b7504d8

```
0 57  nth one ! . . I W
F 72  ant More Passwor
9 31  ds !!!..f2332291
1 64  a6e1e6154f3cf4ad
3 3B  8b7504d8. ...;
0 00  X...........t...
```

尝试提交,成功

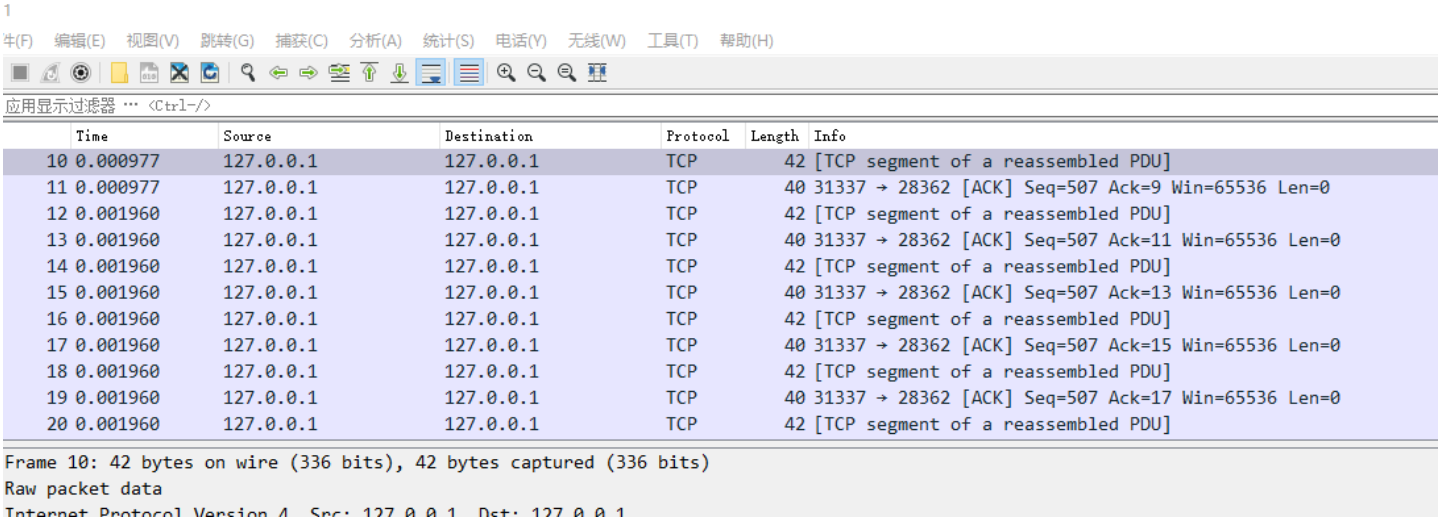
flag:CCTF{f2332291a6e1e6154f3cf4ad8b7504d8}

re2:

一个.net程序,使用Reflector反编译,得到代码

```
private static void Main(string[] args)
{
    string hostname = "127.0.0.1";
    int port = 0x7a69;
    TcpClient client = new TcpClient();
    try
    {
        Console.WriteLine("Connecting...");
        client.Connect(hostname, port);
    }
    catch (Exception)
    {
        Console.WriteLine("Cannot connect!\nFail!");
        return;
    }
    Socket socket = client.Client;
    string str2 = "Super Secret Key";
    string text = read();
    socket.Send(Encoding.ASCII.GetBytes("CTF{"));
    foreach (char ch in str2)
    {
        socket.Send(Encoding.ASCII.GetBytes(search(ch, text)));
    }
    socket.Send(Encoding.ASCII.GetBytes("}"));
    socket.Close();
    client.Close();
    Console.WriteLine("Success!");
}
```

发现这就是个和本地端口通信的程序,于是先去关闭了防火墙,在通过RAWPCAP本地回环抓包即可



得到flag::CCTF{7eb67b0bb4427e0b43b40b6042670b55}

re3:

一个简单的反汇编
发现就是两个字符串的比较

```
while ( v20 )
{
  strcpy(v19, "123456");
  strcpy(v18, "7891011");
  strcpy(v17, "12131415");
  strcpy(Str2, "numbers:");
  v15 = 7365741;
  v14 = 7631717;
  v13 = 7237475;
  v12 = 7627107;
  v11 = 6647397;
  v10 = 6582895;
  v8 = 25970;
  v9 = 0;
  v7 = 3219507;
  v6 = 3285044;
  strcat(Str2, v18);
  strcat(Str2, v17);
  strcat(Str2, v19);
  printf("Trouvez moi si vous pouvez\n");
  scanf("%s", &Str1);
```

```
00007F8 main:35
```

尝试将上面那个字符串提交,发现正确.
flag:CCTF{7891011112131415123456}

true-or-false

两个linux程序,通过IDA反汇编,发现两个程序开始时都会调用system两次,通过ascll码知道了false会将自己覆盖到true上,true会将自己删除.
然后在false里发现了print_f函数,反汇编+凯撒加密就得到了结果的flag
PPGS{yvahk-enva-ova}
\\\\\\\\\
CCTF{linux-rain-bin}

diffffffffffuse

通过IDA观察反编译出的C语言.
总共有3000个函数,3个函数为一周期。但是其中有一些周期中的,第二个函数是直接提取数据,第三个函数移位存在微小差异。通过把汇编代码提取出来生成txt文件,然后用python读取文件模拟生成c程序,即把这3*1000个函数中的第二个函数都扒取出来,生成second.c文件

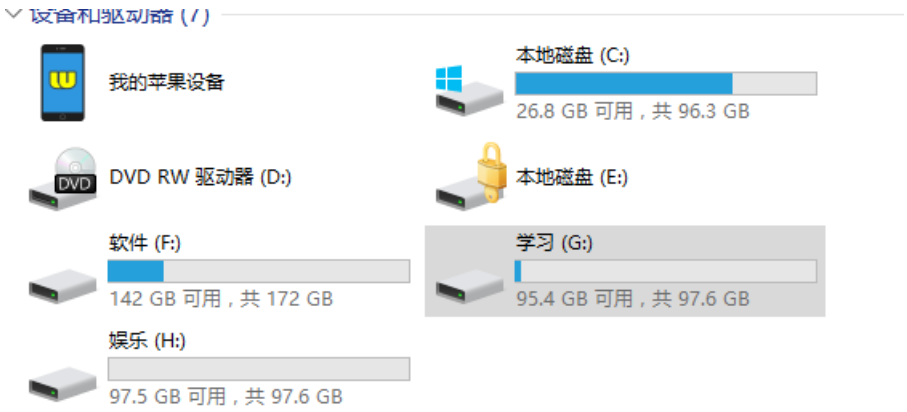
程序最后会将这3000个函数加密后的40字节与现有的40字节相比较,于是我们在IDA中把这40个字节抓取出来

```
0x83    0xec    0x5f    0xa2    0x93    0xce    0xa3    0xfb
0x5a    0x17    0x06    0xff    0x13    0x2d    0xd7    0xc4
0xbe    0xce    0x8d    0x6a    0xb8    0x15    0x26    0xfc
0x84    0x01    0x94    0x44    0xf8    0xd7    0x23    0x1c
0x4b    0xc2    0x31    0x04    0xa6    0x33    0x08    0x57
```

每一个字符的加密是独立运行的,也就意味着我们可以针对每一个字符进行单独的爆破,看看加密后的数据是否相同,通过简单爆破,最终得到flag:
CCTF{1f_Y0u_W4nNa_R3vEn93_____purpleroc}

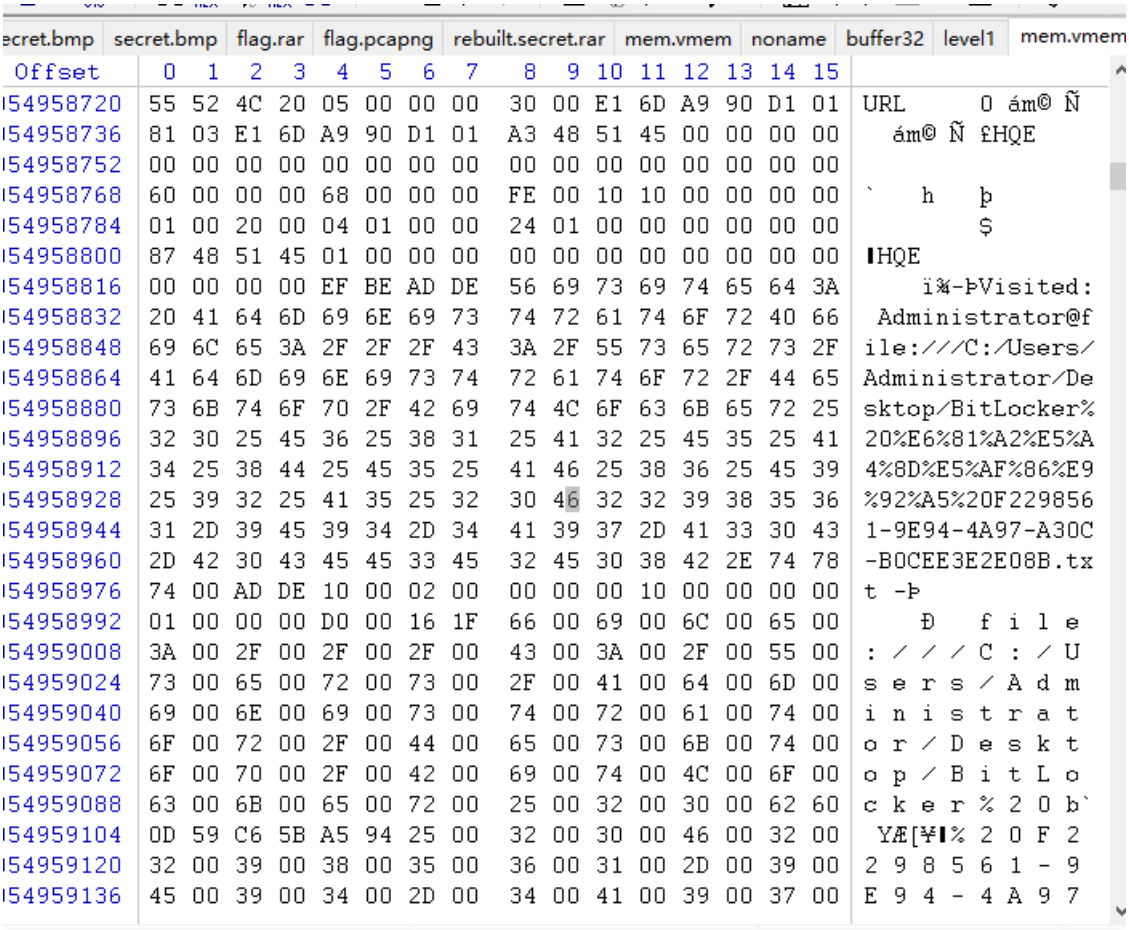
神秘文件1

拿到forensic.7z后解压得到level1与mem.vmem两个文件，观察文件开头，经过百度后得出level1为硬盘文件，mem为内存文件
恰巧本人有一个空硬盘，于是将硬盘格式化，用bootice将level1写入了硬盘，如图所示



发现硬盘被Bitlocker加密，由于存在忘记密码的可能，Bitlocker提供了文件恢复密码机制，密码为48位纯数字

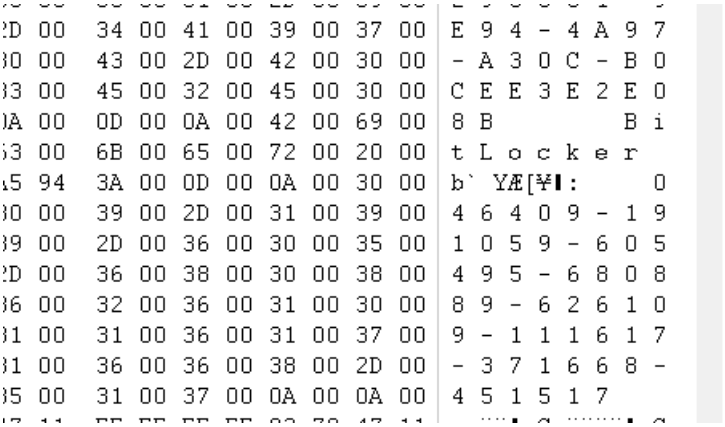
考虑了恢复密码在内存中的可能性后，用winhex打开mem，在其中搜索Bitlocker没有找到，想起了恢复密码id提示为F2298561，搜索后找到有关内容



ecret.bmp | secret.bmp | flag.rar | flag.pcapng | rebuilt.secret.rar | mem.vmem | noname | buffer32 | level1 | mem.vmem

Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
154958720	55	52	4C	20	05	00	00	00	30	00	E1	6D	A9	90	D1	01	URL 0 ám© Ñ
154958736	81	03	E1	6D	A9	90	D1	01	A3	48	51	45	00	00	00	00	ám© Ñ £HQE
154958752	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
154958768	60	00	00	00	68	00	00	00	FE	00	10	10	00	00	00	00	` h þ
154958784	01	00	20	00	04	01	00	00	24	01	00	00	00	00	00	00	$
154958800	87	48	51	45	01	00	00	00	00	00	00	00	00	00	00	00	‡HQE
154958816	00	00	00	00	EF	BE	AD	DE	56	69	73	69	74	65	64	3A	ï¾-ÞVisited:
154958832	20	41	64	6D	69	6E	69	73	74	72	61	74	6F	72	40	66	Administrator@f
154958848	69	6C	65	3A	2F	2F	2F	43	3A	2F	55	73	65	72	73	2F	ile:///C:/Users/
154958864	41	64	6D	69	6E	69	73	74	72	61	74	6F	72	2F	44	65	Administrator/De
154958880	73	6B	74	6F	70	2F	42	69	74	4C	6F	63	6B	65	72	25	sktop/BitLocker%
154958896	32	30	25	45	36	25	38	31	25	41	32	25	45	35	25	41	20%E6%81%A2%E5%A
154958912	34	25	38	44	25	45	35	25	41	46	25	38	36	25	45	39	4%8D%E5%AF%86%E9
154958928	25	39	32	25	41	35	25	32	30	46	32	32	39	38	35	36	%92%A5%20F229856
154958944	31	2D	39	45	39	34	2D	34	41	39	37	2D	41	33	30	43	1-9E94-4A97-A30C
154958960	2D	42	30	43	45	45	33	45	32	45	30	38	42	2E	74	78	-B0CEE3E2E08B.tx
154958976	74	00	AD	DE	10	00	02	00	00	00	00	10	00	00	00	00	t -Þ
154958992	01	00	00	00	D0	00	16	1F	66	00	69	00	6C	00	65	00	Ð f i l e
154959008	3A	00	2F	00	2F	00	2F	00	43	00	3A	00	2F	00	55	00	: / / / C : / U
154959024	73	00	65	00	72	00	73	00	2F	00	41	00	64	00	6D	00	s e r s / A d m
154959040	69	00	6E	00	69	00	73	00	74	00	72	00	61	00	74	00	i n i s t r a t
154959056	6F	00	72	00	2F	00	44	00	65	00	73	00	6B	00	74	00	o r / D e s k t
154959072	6F	00	70	00	2F	00	42	00	69	00	74	00	4C	00	6F	00	o p / B i t L o
154959088	63	00	6B	00	65	00	72	00	25	00	32	00	30	00	62	60	c k e r % 2 0 b`
154959104	0D	59	C6	5B	A5	94	25	00	32	00	30	00	46	00	32	00	YÆ[¥‡% 2 0 F 2
154959120	32	00	39	00	38	00	35	00	36	00	31	00	2D	00	39	00	2 9 8 5 6 1 - 9
154959136	45	00	39	00	34	00	2D	00	34	00	41	00	39	00	37	00	E 9 4 - 4 A 9 7

发现内存中的数据很多情况用00隔开，于是搜索F2298561的16进制数，每两个数用00隔开，最后找到了48位数字密码



2D	00	34	00	41	00	39	00	37	00	E	9	4	-	4	A	9	7
30	00	43	00	2D	00	42	00	30	00	-	A	3	0	C	-	B	0
33	00	45	00	32	00	45	00	30	00	C	E	E	3	E	2	E	0
0A	00	0D	00	0A	00	42	00	69	00	8	B					B	i
63	00	6B	00	65	00	72	00	20	00	t	L	o	c	k	e	r	
a5	94	3A	00	0D	00	0A	00	30	00	b`		YÆ[¥‡:					0
30	00	39	00	2D	00	31	00	39	00	4	6	4	0	9	-	1	9
39	00	2D	00	36	00	30	00	35	00	1	0	5	9	-	6	0	5
2D	00	36	00	38	00	30	00	38	00	4	9	5	-	6	8	0	8
36	00	32	00	36	00	31	00	30	00	8	9	-	6	2	6	1	0
31	00	31	00	36	00	31	00	37	00	9	-	1	1	1	6	1	7
31	00	36	00	36	00	38	00	2D	00	-	3	7	1	6	6	8	-
35	00	31	00	37	00	0A	00	0A	00	4	5	1	5	1	7		

为046409-191059-605495-680889-626109-111617-371668-451517

解锁成功！拿到flag CCTF{U_m4st_G00d_4t_F0nr4n51c}

BEST_EZ_MISC

拿到名为reverze.zip的压缩包，用winhex看了一下，很容易想到是zip伪加密

Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
00000000	50	4B	03	04	14	00	00	00	08	00	7D	BE	90	48	B3	E5
00000016	E6	15	AB	00	00	00	FF	1B	00	00	07	00	00	00	72	65
00000032	76	65	72	73	65	ED	98	41	0E	80	30	08	04	EF	BE	A2
00000048	1F	70	FF	FF	3C	E3	A1	17	A2	09	87	05	B6	04	2F	13
00000064	B5	DA	5D	68	0A	E9	FD	5E	6B	0D	06	83	C1	69	C0	3F
00000080	EC	48	07	28	92	38	00	65	48	ED	0C	12	30	B9	95	4D
00000096	38	07	69	8E	40	99	5D	37	0F	70	DC	39	1E	D6	24	47
00000112	D7	5F	5A	40	C6	5F	0B	7F	9E	02	4D	29	EC	F2	0D	41
00000128	77	1C	D6	8E	80	F9	0E	01	22	8A	FC	51	0B	7B	23	7F
00000144	45	FB	8B	99	2F	AB	AF	8B	CF	5F	84	16	04	FC	73	E0
00000160	5A	98	C9	DD	85	2E	4C	40	98	ED	DD	D1	28	F2	C7	39
00000176	2E	DA	1F	E8	D5	87	88	0D	3E	4D	35	47	6E	23	7F	5B
00000192	60	80	BF	A2	48	20	EB	98	B7	C8	5F	1A	F0	89	EB	01
00000208	50	4B	01	02	3F	03	14	03	09	00	08	00	7D	BE	90	48
00000224	B3	E5	E6	15	AB	00	00	00	FF	1B	00	00	07	00	00	00
00000240	00	00	00	00	00	00	20	80	B4	81	00	00	00	00	72	65
00000256	76	65	72	73	65	50	4B	05	06	00	00	00	00	01	00	01
00000272	00	35	00	00	00	D0	00	00	00	00	00					

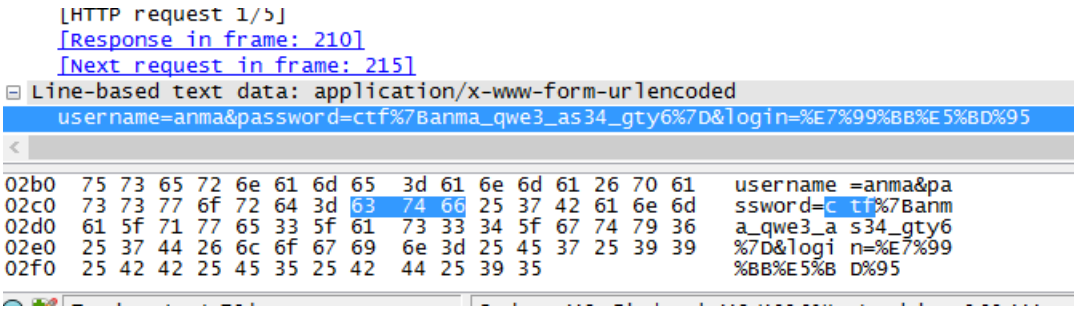
将pk 01 02 后的第5个字节改为00，发现可以解压，得到reverse

打开看了一下，发现是摩丝电码，解密后得到0,9字符串，程序跑了一下算上空格一共2048个字符，由于题目提示结果是个图片，于是尝试不同的像素来观察，最终得到如图所示结果

虽然不是很清楚，但是颠倒一下很容易看出flag： ctf{pixelnice}

misc100T2

很简单的流量分析，打开以后尝试搜索ctf直接得到结果



flag为ctf{anma_qwe3_as34_gty6}

创作打卡挑战赛

赢取流量/现金/CSDN周边激励大奖