Cryptohack刷题记录(一) General部分 WP

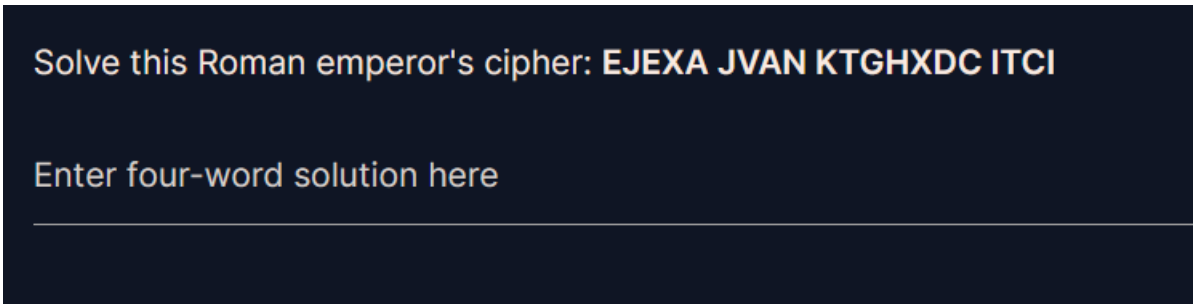
原创

Mr_AgNO3 已于 2022-01-24 00:46:00 修改 284 收藏 1
文章标签： python 密码学
于 2022-01-24 00:43:44 首次发布
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/u010883831/article/details/122659878
版权

cryptohack，很不错的一个密码学习平台。
很适合没有基础的beginner系统进行学习

文章目录

注册平台账号

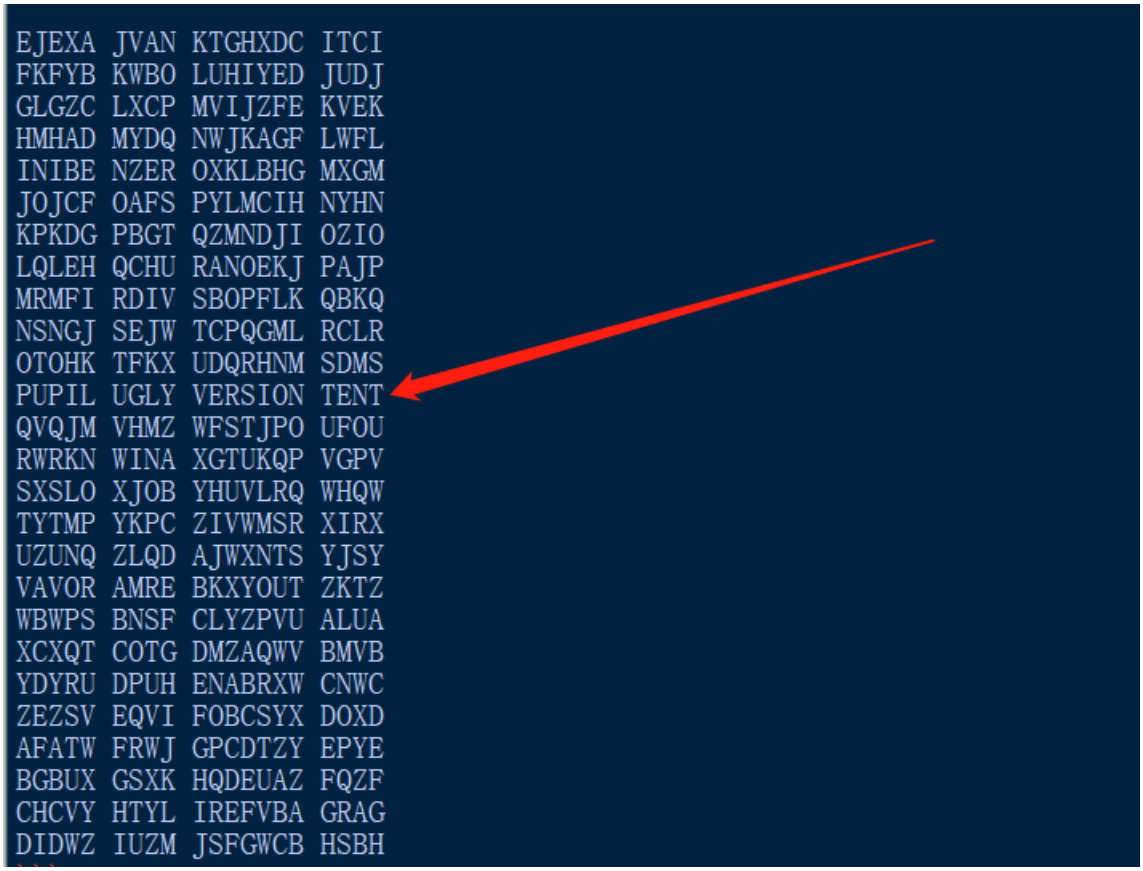


EJEXA JVAN

KTGHXDC ITCI

罗马皇帝密码，那就是凯撒了

```
for i in range(26):
for j in c:
 if j == ' ':
  print(' ',end='')
  continue
 t = ss.index(j)
 print(ss[(t+i)%26],end='')
print()
```



General

Encoding

第一題

ASCII 5 pts · 16765 Solves

ASCII is a 7-bit encoding standard which allows the representation of text using the integers 0-127.

Using the below integer array, convert the numbers to their corresponding ASCII characters to obtain a flag.

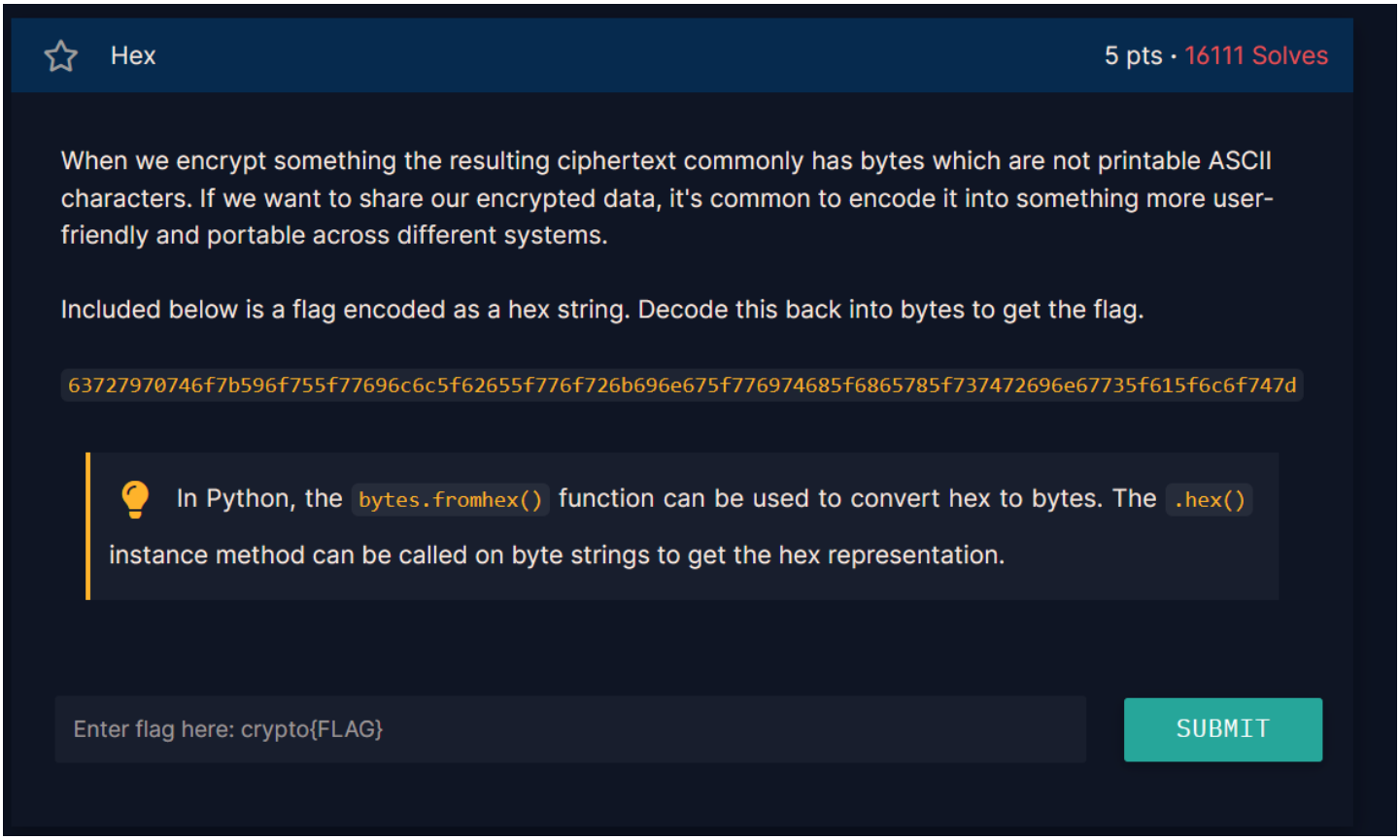
```
[99, 114, 121, 112, 116, 111, 123, 65, 83, 67, 73, 73, 95, 112, 114, 49, 110, 116, 52, 98, 108, 51, 125]
```

> In Python, the `chr()` function can be used to convert an ASCII ordinal number to a character (the `ord()` function does the opposite).

Enter flag here: crypto{FLAG} SUBMIT

```
>>> a  =[99, 114, 121, 112, 116, 111, 123, 65, 83, 67, 73, 73, 95, 112, 114, 49, 110, 116, 52, 98, 108, 51, 125]
>>> ''.join(chr(i) for i in a)
'crypto{}'
```

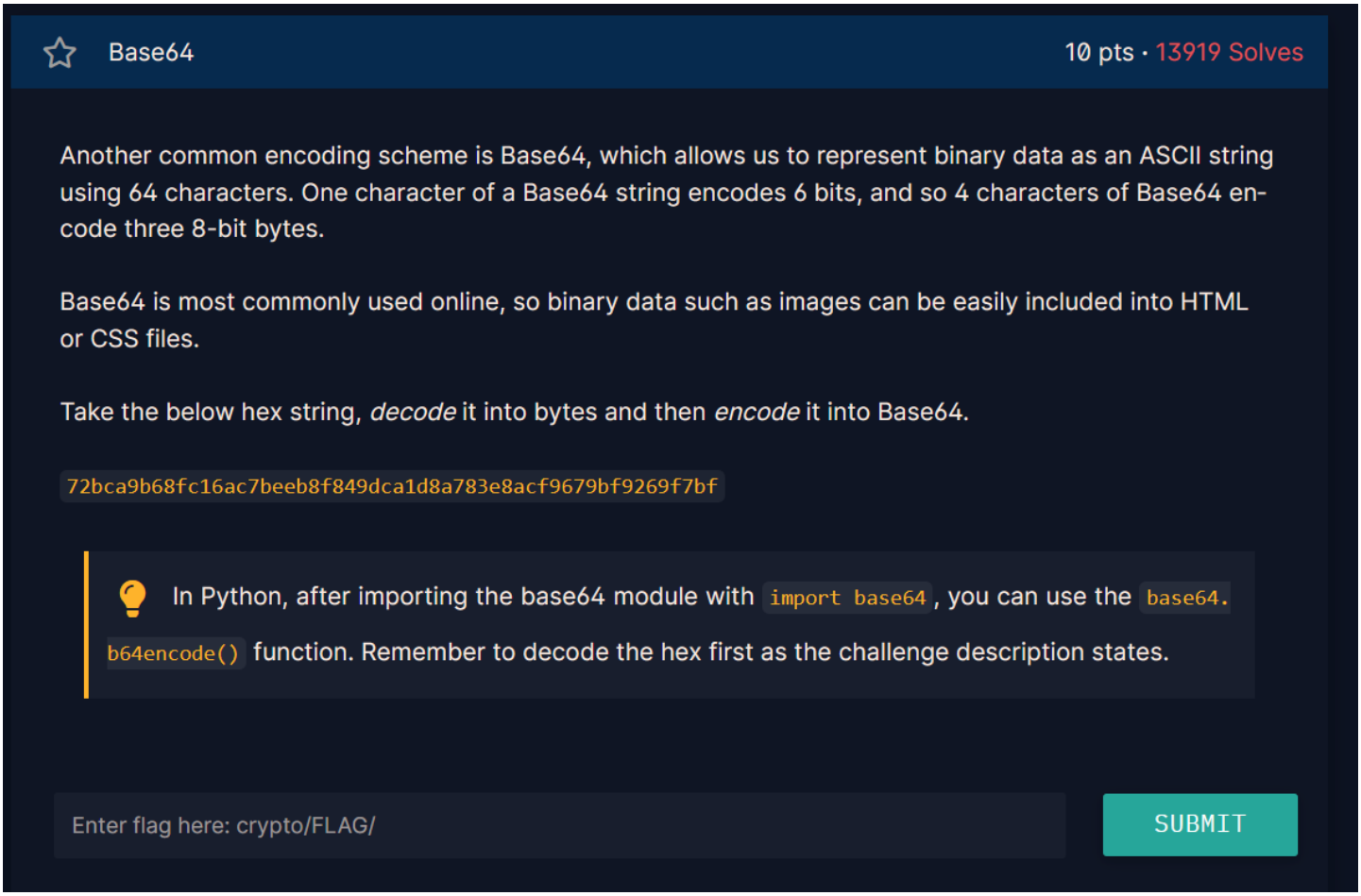
第二題



63727970746f7b596f755f77696c6c5f62655f776f726b696e675f776974685f6865785f737472696e67735f615f6c6f747d

```
>>> from binascii import *
>>> unhexlify('63727970746f7b596f755f77696c6c5f62655f776f726b696e675f776974685f6865785f737472696e67735f615f6c6f747d')
b'crypto{}'
```

第三題



72bca9b68fc16ac7beeb8f849dca1d8a783e8acf9679bf9269f7bf

```
>>> c = '72bca9b68fc16ac7beeb8f849dca1d8a783e8acf9679bf9269f7bf'
>>> from base64 import b64encode as e
>>> unhexlify(c)
b'r\xbc\xa9\xb6\x8f\xc1j\xc7\xbe\xeb\x8f\x84\x9d\xca\x1d\x8ax>\x8a\xcf\x96y\xbf\x92i\xf7\xbf'
>>> e(unhexlify(c))
b'crypto//'
```

有点意思

第四题

Bytes and Big Integers 10 pts · 11240 Solves

Cryptosystems like RSA works on numbers, but messages are made up of characters. How should we convert our messages into numbers so that mathematical operations can be applied?

The most common way is to take the ordinal bytes of the message, convert them into hexadecimal, and concatenate. This can be interpreted as a base-16 number, and also represented in base-10.

To illustrate:

```
message: HELLO
ascii bytes: [72, 69, 76, 76, 79]
hex bytes: [0x48, 0x45, 0x4c, 0x4c, 0x4f]
base-16: 0x48454c4c4f
base-10: 310400273487
```

Python's PyCryptodome library implements this with the methods `Crypto.Util.number.bytes_to_long()` and `Crypto.Util.number.long_to_bytes()`. You may first have to install PyCryptodome and import it with `from Crypto.Util.number import *`. For more details check the FAQ.

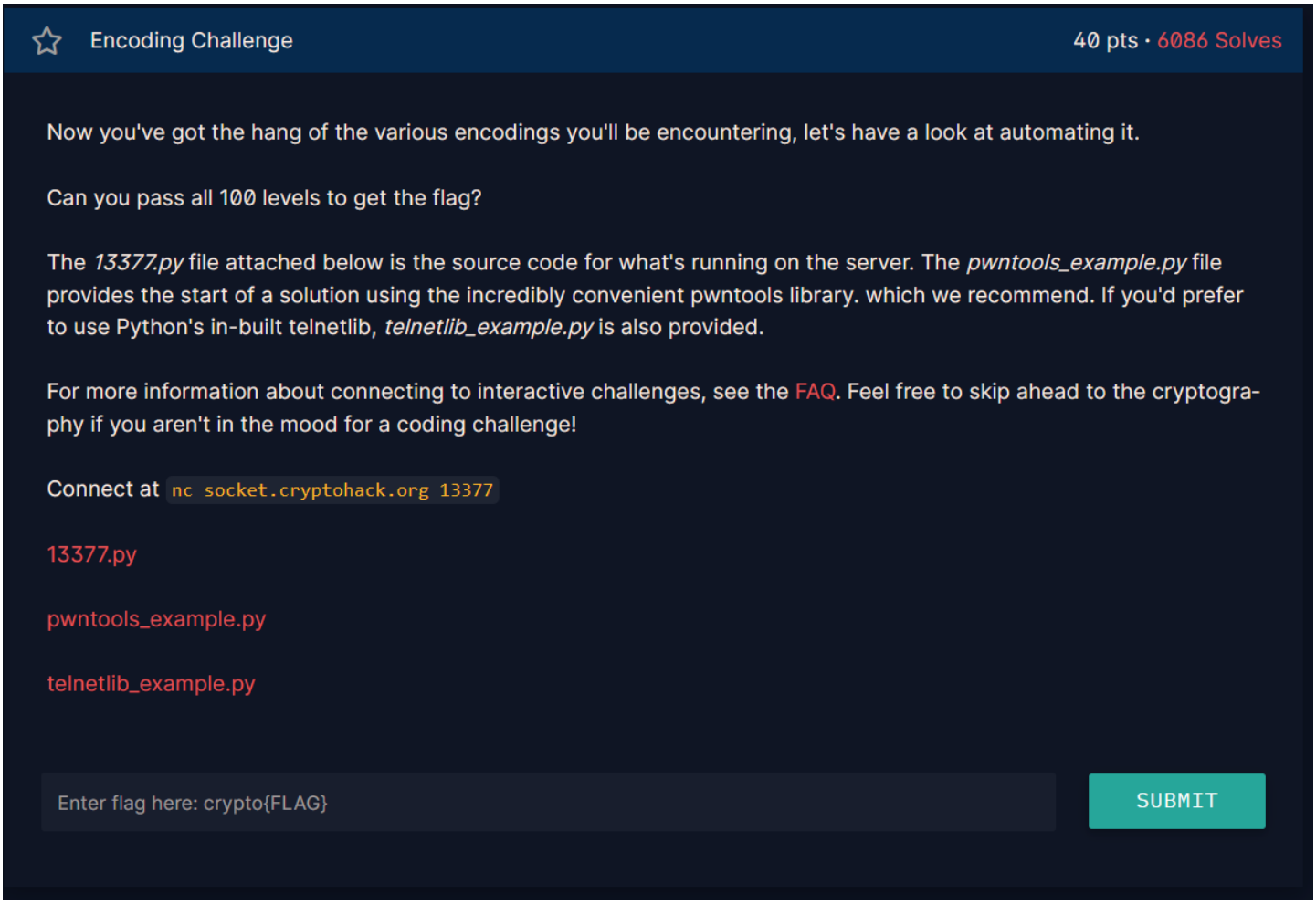
Convert the following integer back into a message:

`11515195063862318899931685488813747395775516287289682636499965282714637259206269`

Enter flag here: crypto{FLAG} SUBMIT

```
>>> from number import *
>>> l2b(11515195063862318899931685488813747395775516287289682636499965282714637259206269)
b'crypto{}'
```

第五題



nc到服务器后，会给你一个加密方式`type`和一个加密后的字符串`c`

你需要解密c之后给服务器发送过去，重复100次

服务端代码：

```
if encoding == "base64":
    encoded = base64.b64encode(self.challenge_words.encode()).decode()
elif encoding == "hex":
    encoded = self.challenge_words.encode().hex()
elif encoding == "rot13":
    encoded = codecs.encode(self.challenge_words, 'rot_13')
elif encoding == "bigint":
    encoded = hex(bytes_to_long(self.challenge_words.encode()))
elif encoding == "utf-8":
    encoded = [ord(b) for b in self.challenge_words]
```

要重复100次，所以肯定不能手动解，所以用`pwntools`的`remote`进行连续破解

```
from pwn import *
import json

r = remote('socket.cryptohack.org', 13377, level = 'debug')

def json_recv():
    line = r.recvline()
    return json.loads(line.decode())

def json_send(hsh):
    request = json.dumps(hsh).encode()
    r.sendline(request)

from binascii import *
from number import *
from base64 import b64decode as dd

from string import *
def rot13(x):
    ss = ascii_lowercase
    res = ''
    for i in x:
        if i in ss:
            res += ss[(13+ss.index(i))%26]
        else:res += i
    return res

def dec(tp,c):
    if tp == 'bigint':
        m = unhexlify(c[2:]).decode()
    elif tp == 'base64':
        m = dd(c.encode()).decode()
    elif tp == 'rot13':
        m = rot13(c)
    elif tp == 'hex':
        m = l2b(int(c,16)).decode()
    elif tp == 'utf-8':
        m = ''.join(chr(i) for i in c)
    return m

def func(received):
    print("Received type: ",end='')
    print(received["type"])
    print("Received encoded value: ",end='')
    print(received["encoded"])
    c = received["encoded"]
    tp = received['type']
    to_send = {"decoded": dec(tp,c)}
    print('to_sent:',to_send)
    json_send(to_send)
    rrr = json_recv()
    print(rrr,'================',sep = '\n')
    return rrr

rcvd = json_recv()
for i in range(100):
    rcvd = func(rcvd)
```

得到flag

```
    b'{"decoded": "giving_journals_running"}\n'
[DEBUG] Received 0x4f bytes:
    b'{"type": "hex", "encoded": "72657475726e65645f70656e64616e745f67726561746c
79"}\n'
{'type': 'hex', 'encoded': '72657475726e65645f70656e64616e745f67726561746c79'}
================
Received type: hex
Received encoded value: 72657475726e65645f70656e64616e745f67726561746c79
to_sent: {'decoded': 'returned_pendant_greatly'}
[DEBUG] Sent 0x28 bytes:
    b'{"decoded": "returned_pendant_greatly"}\n'
[DEBUG] Received 0x29 bytes:
    b'{"flag": "crypto{3nc0d3_d3c0d3_3nc0d3}"}\n'
{'flag': 'crypto{3nc0d3_d3c0d3_3nc0d3}'}
================
>>>
```

XOR

1. XOR Starter

将 label 的每一位与 13 进行异或

题目说了可以用 pwntools 里面的 xor 函数

```
>>> from pwn import *
>>> xor('label',13)
b'aloha'
```

crypto{aloha}

---

2. XOR Properties

题目给了如下信息

> k1
> k2 ^ k1
> k2 ^ k3
> flag ^ k1 ^ k2 ^ k3

分别记为 A、B、C、D，则 flag = D ^ A ^ C

```
>>> a = 0xa6c8b6733c9b22de7bc0253266a3867df55acde8635e19c73313
>>> b = 0x37dcb292030faa90d07eec17e3b1c6d8daf94c35d4c9191a5e1e
>>> c = 0xc1545756687e7573db23aa1c3452a098b71a7fbf0fddddde5fc1
>>> d = 0x04ee9855208a2cd59091d04767ae47963170d1660df7f56f5faf
>>> f = d^a^c
>>> from number import *
>>> l2b(f)
b'crypto{}'
```

3.Favourite byte

`73626960647f6b206821204f21254f7d694f7624662065622127234f726927756d`

将数字转为byte型后为

```
>>> l2b(int('73626960647f6b206821204f21254f7d694f7624662065622127234f726927756d',16))
b"sbi`d\x7fk h! O!%O}iOv$f eb!'#Ori'um"
```

已知flag前几位，为`crypto{`，与解出的字符串对应前几位进行异或便可以得到secret

```
>>> xor(b"sbi`d\x7fk",b"crypto{")
b'\x10\x10\x10\x10\x10\x10\x10'
```

得到flag

```
>>> xor(b"sbi`d\x7fk h! O!%O}iOv$f eb!'#Ori'um",'\x10')
b'crypto{}'
```

4.You either know, XOR you don’t

`0e0b213f26041e480b26217f27342e175d0e070a3c5b103e2526217f27342e175d0e077e263451150104`

尝试同同样的步骤，找出异或的字符串

```
>>> c = l2b(int('0e0b213f26041e480b26217f27342e175d0e070a3c5b103e2526217f27342e175d0e077e263451150104',16))
>>> c
b"\x0e\x0b!?&\x04\x1eH\x0b&!\x7f'4.\x17]\x0e\x07\n<[\x10>%&!\x7f'4.\x17]\x0e\x07~&4Q\x15\x01\x04"
>>> xor(c[:7],'crypto{')
b'myXORke'
>>> xor(c[-1],'}')
b'y'
```

拼接起来可以得到一段有意义的语句`myXORkey`
猜测这就是加密的字符串

```
>>> xor(c,'myXORkey')
b'crypto{}'
```

---

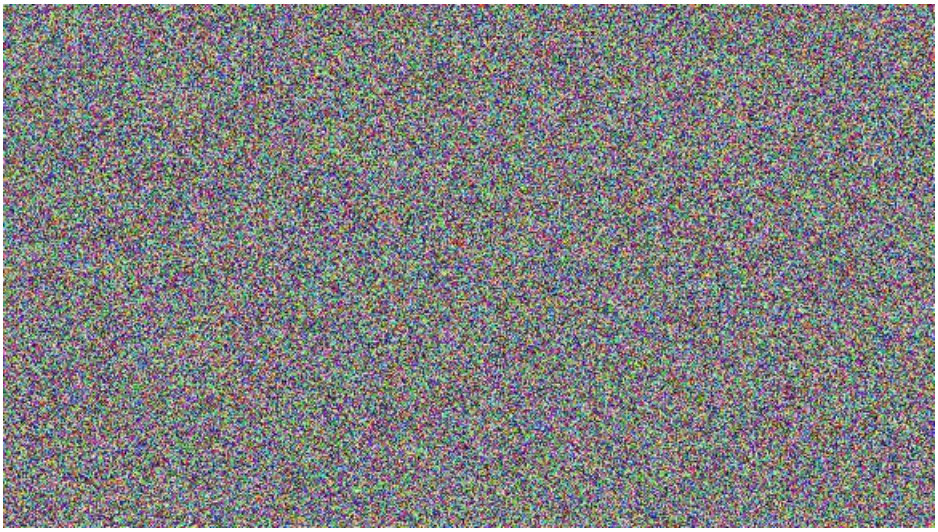
5. Lemur XOR

这是个图片题

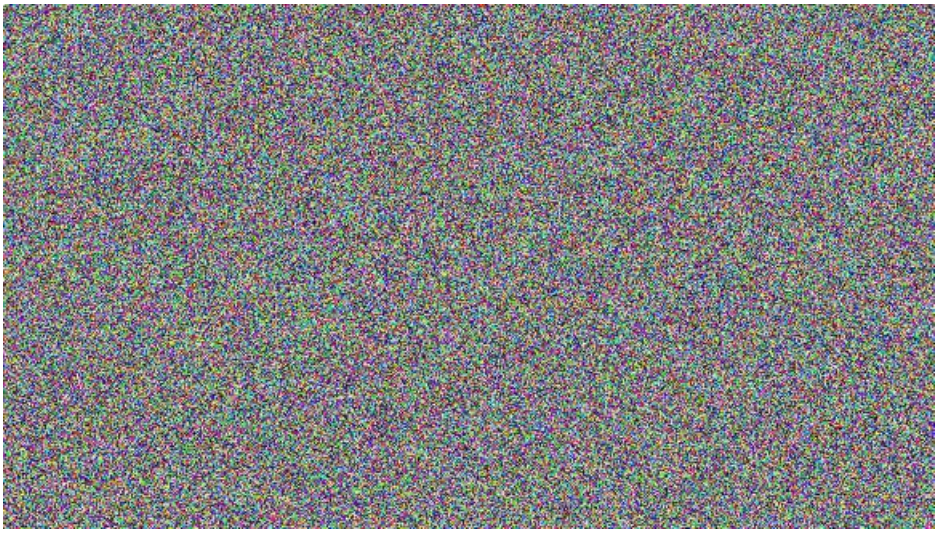
> ps.
> lemur n. 狐猴

两个图片

lemur.png:



flag.png:



异或试试，使用`PIL.Image`模块和`numpy`模块

```
from PIL import Image as img
import numpy as np

ll = img.open(r".\lemur.png")
ff = img.open(r".\flag.png")

nl = np.array(ll)
nf = np.array(ff)
img.fromarray(nl^nf).show()
```

得到flag，在图片中

MATHEMATICS

1. Greatest Common Divisor

计算GCD

属实是零基础入门题了

```
>>> a = 66528; b = 52920
>>> from gmpy2 import *
>>> gcd
<built-in function gcd>
>>> gcd(a,b)
mpz(1512)
```

---

2. Extended GCD

要求扩展欧几里得算法

使用gmpy2库自带的gcdext()函数

```
>>> gcdext
<built-in function gcdext>
>>> p = 26513;q = 32321
>>> gcdext(p,q)
(mpz(1), mpz(10245), mpz(-8404))
```

按照要求，flag是`-8404`

---

3. Modular Arithmetic 1

解方程

$$11 \equiv x \mod 6$$

---

4. Modular Arithmetic 2

费马小定理，当p为素数时，有

$$a^{} \equiv 1 \mod p$$

---

5. Modular Inverting

求逆元

b是a对m的逆元，则在模m的情况下有 $ab \equiv 1 \mod m$

使用gmpy2中的 `invert()` 函数

```
>>> invert(3,13)
mpz(9)
```

DATA FORMATS

1. Privaty-Enhanced Mail?

有点懵，啥意思

啊

给了一个pem文件，把里面的私钥信息解密出来即可

```
>>> c = open(r"./ppp").read()
>>> p = RSA.import_key(c)
>>> p.d
15682700288056331364787171045819973654991149949197959929860861228180021707316851924456205543665565810892674190059831330231436970914474774562714945620519144389785158908994181951348846017432506464163564960993784254153954067991013147600334450651934295925123499520209829322185244623410021020634354893188133164645116217369439384407104706949123362376802197462045951289591618005952163662375382964473353758188719525200269931021483288970835471842864932411915059536016688589411297909669092369411278513702024211358970910867635698847600991122910720569706363804173490195797687480547601048387904247089882604439269066737959751046 89
```

私钥d即flag

---

2. CERTainly not

> PEM和DER是证书编码的两个方式，可以互相转化
> PEM是可读的文本格式的，以"-----BEGIN XXX-----"开头，同样格式END结尾的文件，内容经过base64加密
> DER是二进制文件，不能直接看懂，直接二进制方式读取

同样使用Crypto.RSA.import_key函数

```
>>> ddd = open(r".\der.der",'rb').read()
>>> p = RSA.import_key(ddd)
>>> p
RsaKey(n=228253736920195308043062128646095127753741718239937085165098976315475136346358563756240037370680345490476779993109418374543788293513983023826296582640787754568386262075077254940306005168728523061912554929264959655363792718753104573191079360207300504762352786715282658175714339195611756650961711897584061364539879662552369637826606696265467846495007592306032735869135663290860649823175596356738233901098522262320558692346640580921742667033410014429905146941652293366212903733630083016398810887356019977409467374742266276267137547021576874204809506045914964491063393800499167416471949021995447722415959979785959569497, e=65537)
```

flag是整数内容的`模`，也就是`n`的值

---

3. SSH KEY

泪目了 他讲的好详细

主要是ssh,Secure Shell Protocol这个东西

> 明明简写是SSP为什么要叫SSH呢？？？？？？

为什么要有ssh？为了安全。

> As the internet became increasingly hostile, people realised the need for both authentication and encryption for administrative network traffic

以及SSH为什么安全

依旧使用同样方法即可

```
>>> c = open(r"./bruce_rsa_6e7ecd53b443a97013397b1a1ea30e14.pub").read()
>>> p = RSA.import_key(c)
>>> p
RsaKey(n=393140627292252344843619459982009301624147265815180155284509451857950781599060045966925960364526153292761115298494284088989875653206089485704517530014576580063349900545173887208138126700406986555739563855004111420614308540360723410929328633639355275689398460521435298870525863897945473651499731422366907590078380671539888031069594594514775513291903797388907519178597779786155722867815953888215354471779710040109643506235947412975562545383188249060356013447704323543320270894861523453698471587211334381276010281232318039154449603016365304693141472385137455487303658228238990483859766828654333742658168081779603871122840144324465516219930235201796499786667731716101408311673053587552128663185810276896109885120940097389939396493160506785600541099863184267303090107800840864961353814379995980368504156696451448980921196298453432234839442801090898431894041169896115073120431667064667697636195882852822983761079584314504824349290 9, e=65537)
```

flag还是`n`

4. Transparency

有点懵

给了一个PEM格式的公钥文件，打开文件可以得到以下信息

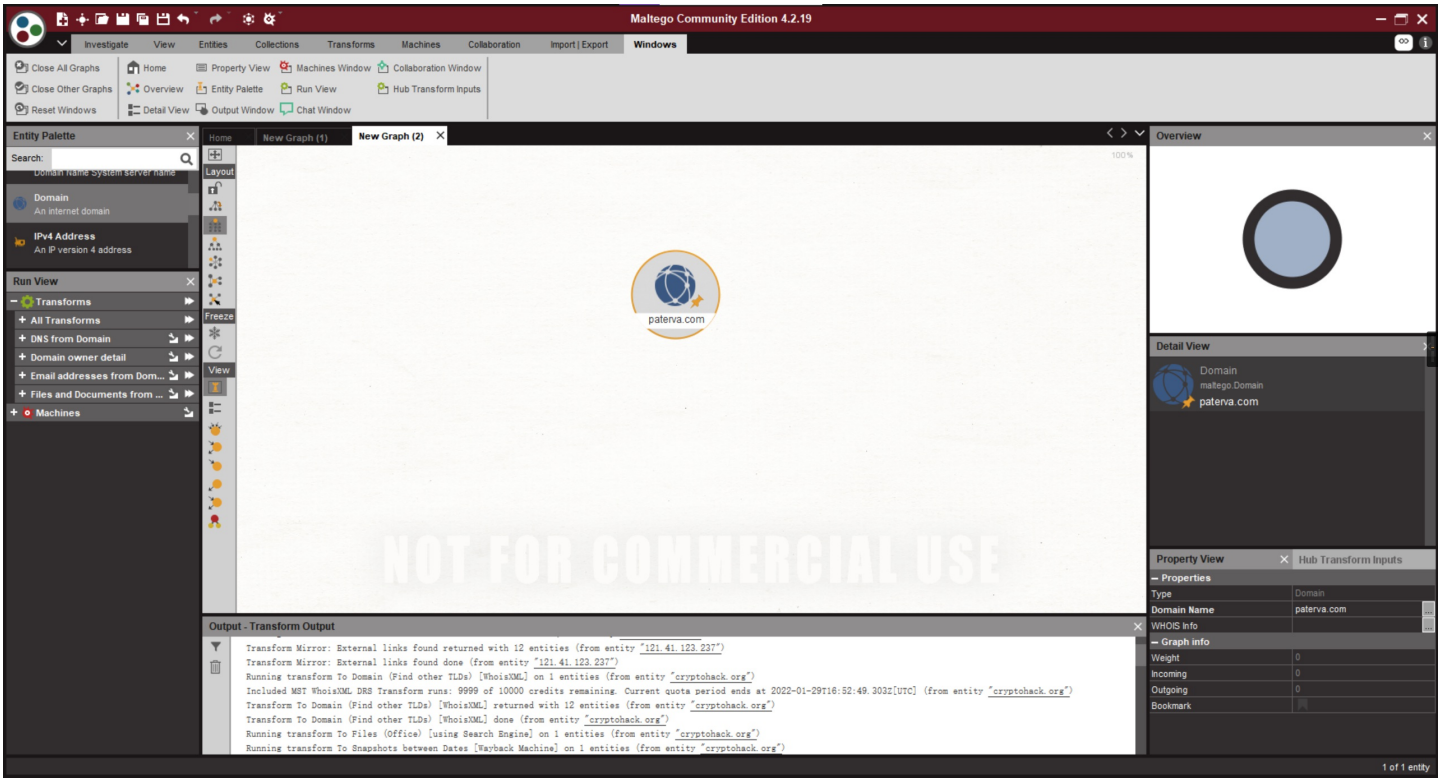
```
>>> from Crypto.PublicKey.RSA import *
>>> c = open(r".\transparency.pem").read()
>>> p = import_key(c)
>>> p
RsaKey(n=23421622285641341405633616890150413771071492791662619237015532689271209254675255214187772835143801809039951016782376679973376782695533167272817148034946155291022588458116896449130547957859630601417029406537713697722164841265084046694925746517387007853236278038029670978141921557139882067656772559964537465702212036054646836981397590682017458056432266023096481777208423697374253076626745245307575706269702325375498240059983936090218617731342155424505568392508047990989034831520127135201674146131415263027275123889726231738091952255921099644166823482030587841034849620518448903987665100805624202958323295532375280413 93, e=65537)
```

要求是 找到 cryptohack.org 下的一个，用这些参数进行证书认证的，某个子域名，flag就在这个子域名中

没有什么思路，搜了下wp，是爆破的，但也只有一张图

使用 Maltego 分析网站域名

1. 新建一个graph，从左侧Entity栏中拖入一个 Domain 实体，并修改为 cryptohack.org



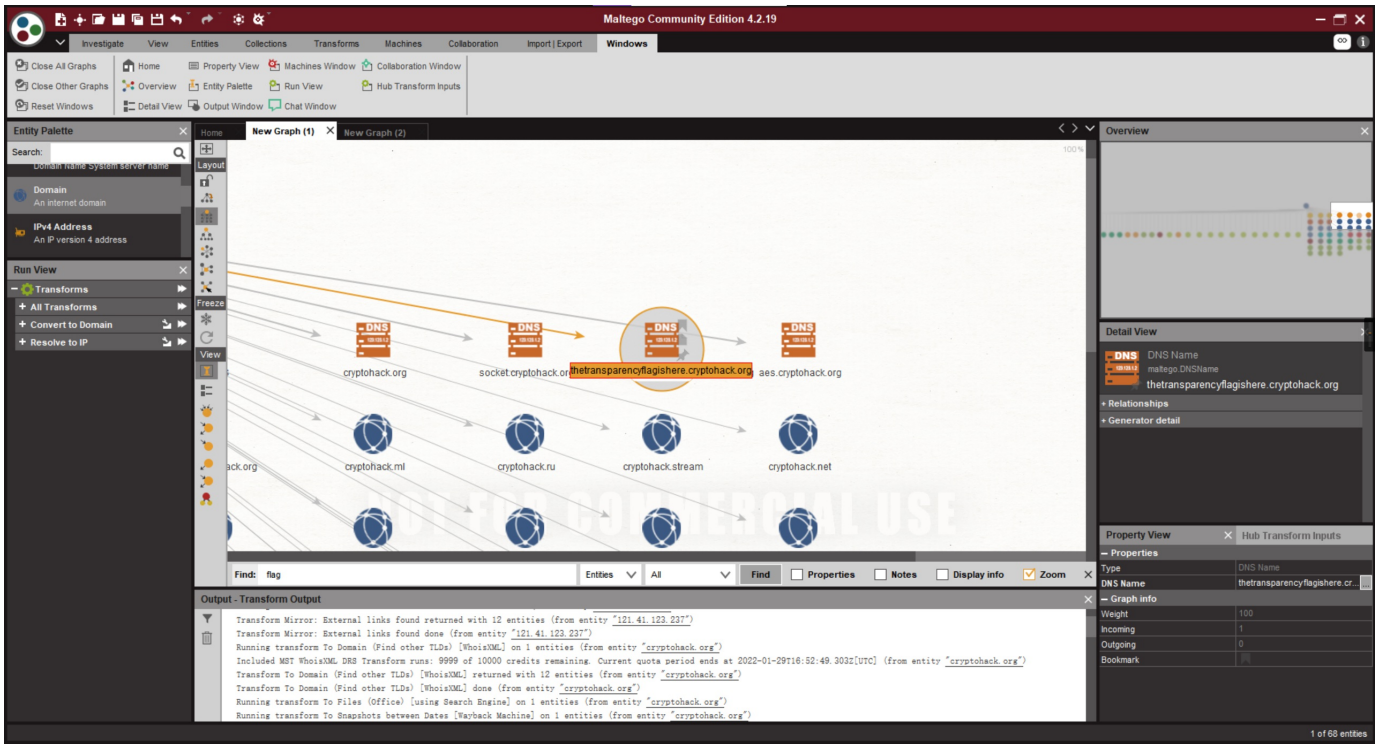
2. 右键实体，点击第一行的All Transforms右侧的双箭头

开始了自动查询域名下的所有信息

包括该域名的子域名，IP地址，DNS服务器，联系人，地址，电话号等等

等待结果

使用 Ctrl+F 搜索功能，查询题目的 Transparentcy 关键字或者直接查询 flag 字样



访问，得到flag

创作打卡挑战赛
赢取流量/现金/CSDN周边激励大奖