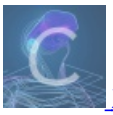
XCTF_Normal_RSA

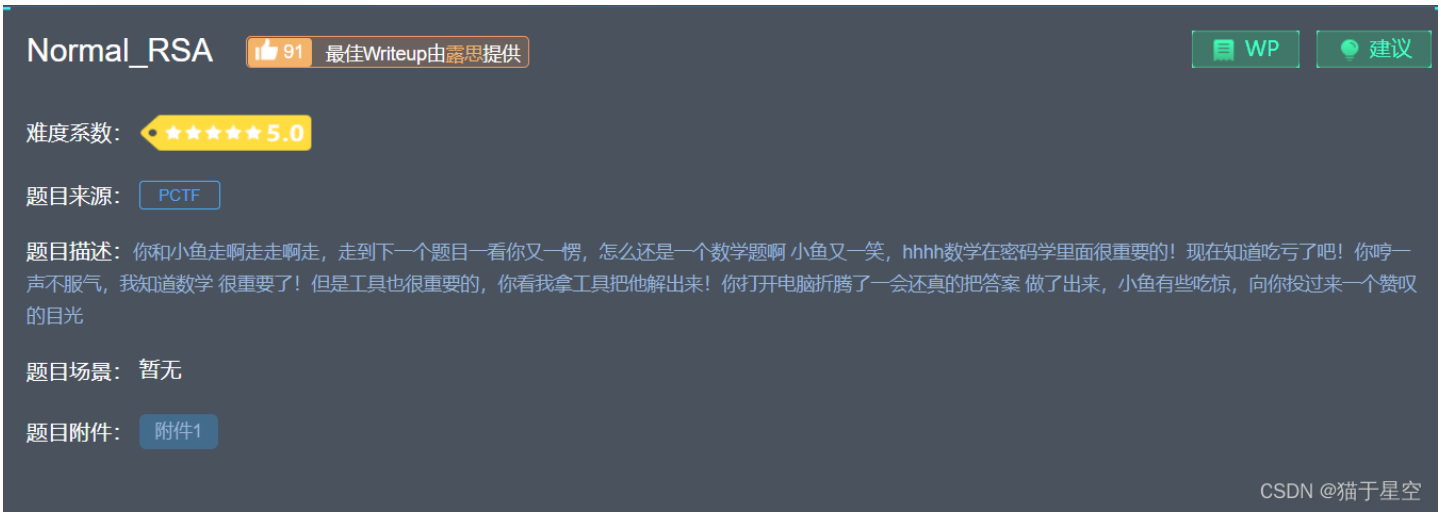
原创

永远是深夜有多好。 于 2022-01-06 16:55:40 发布 937 收藏

分类专栏： XCTF 文章标签： 安全

本文链接：https://blog.csdn.net/qq_37370714/article/details/122346286





打开发现两个文件

根据文件名字知道RSA加密后的文件和公钥

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
-----BEGIN PUBLIC KEY-----
MDwwDQYJKoZIhvcNAQEBBQADKwAwKAIhAMJjauXD2OQ/+5erCQKPGqxsC/bNPXDr
yigb/+l/vjDdAgMBAAE=
-----END PUBLIC KEY-----
```


把公钥放在这里分析

详细信息

密钥类型	RSA
密钥强度	256
DER格式	303c300d06092a864886f70d010101050003 2b003028022100c2636ae5c3d8e43ffb97ab09028f1aac6c0bf6cd3d70eb ca281bffe97fbe30dd0203010001
PN(e)	65537
PN(n)	879243482641324068752761405144999371450508936656025929924181716470424916584617042491658461


```
n=879243482641324068752761405144999371450508936656025929924181716470424916584617042491658461
e=65537
```

根据j解密公式m等于明文 c等于密文 d等于私钥 n等于大素数p和q的乘积

$$m = D(c) = c^d mod n$$

$$e * d \equiv 1 mod \varphi(n)$$

φ(n)是n的欧拉函数 φ(n)=(p-1)*(q-1)

通过yafu找到p和q，就可以使用因式分解求出私钥d

```
>C:\CTF\Crypto\RSA_yafu\yafu-x64.exe factor(879243482641324068752761405144999371450508936656025929924181716470424916584
61)


fac: factoring 879243482641324068752761405144999371450508936656025929924181716470424916584 61
fac: using pretesting plan: normal
fac: no tune info: using qs/gnfs crossover of 95 digits

starting SIQS on c77: 879243482641324068752761405144999371450508936656025929924181716470424916584 61

==== sieving in progress (1 thread):    36224 relations needed ====
====           Press ctrl-c to abort and save state           ====


SIQS elapsed time = 1.3310 seconds.
Total factoring time = 1.3550 seconds


***factors found***

P39 = 275127860351348928173285174381581152299
P39 = 319576316814478949870590164193048041239

ans = 1
```

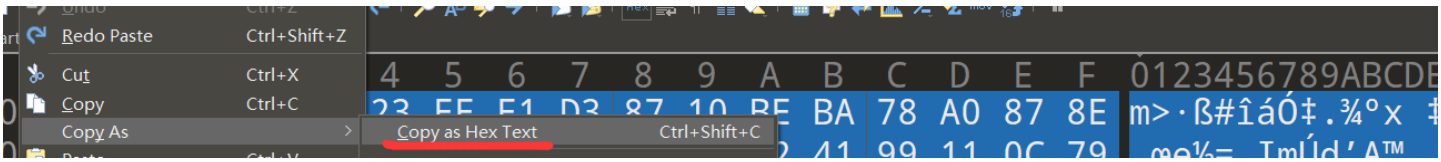


```
p = 275127860351348928173285174381581152299
q = 319576316814478949870590164193048041239
```

于是可以通过gmpy2 python库来计算私钥d mpz后面就是私钥d

```
>>> p=275127860351348928173285174381581152299
>>> q=319576316814478949870590164193048041239
>>> n=p*q
>>> e=65537
>>> phi = (q-1)*(p-1)
>>> d = gmpy2.invert(e,phi)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'gmpy2' is not defined
>>> import gmpy2
>>> d = gmpy2.invert(e,phi)
>>> d
mpz(108669487608445991682520826123784959773882712796792315398390496986219949946 73)
>>>
```

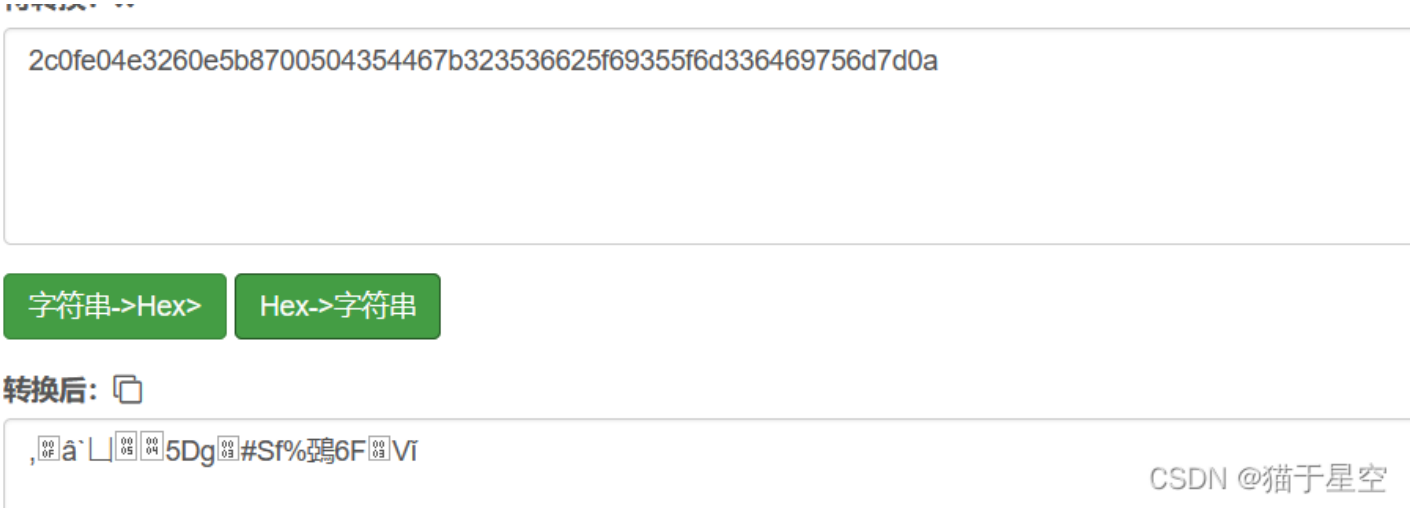

然后用winhex或者010editor等找到密文的十六进制




```
6d3eb7df23eee1d38710beba78a0878e0e9c65bd3d08496dda64924199110c79
```

至此 已经找到了私钥和密文的十六进制数据
最后根据公式得到明文 将其转化为十六进制

```
>>> c=0x6d3eb7df23eee1d38710beba78a0878e0e9c65bd3d08496dda64924199110c79
>>> n=879243482641324068752761405144999371450508936656025929924181716470424916584 61
>>> d=10866948760844599168252082612378495977388271279679231539839049698621994994673
>>> m=pow(c,d,n)
>>> m
486567776928671724041929620814591451783209446484594905503537098752560 2570
>>> hex(m)
'0x2c0fe04e3260e5b8700504354467b323536625f69355f6d336469756d7d0a'
>>>
```



发现不对，原因是这串十六进制数据只有奇数个要不就是多了一个要不就是少了一个
去掉最前面的2以后