




XCTF-Normal_RSA 萌新版解题WriteUp

原创

小程序员  于 2022-04-20 18:24:01 发布  1441  收藏

分类专栏: [XCTF Crypto RSA](#) 文章标签: [安全](#) [pycharm](#) [网络安全](#) [经验分享](#) [娱乐](#)

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/m0_68417968/article/details/124303014

版权



[XCTF 同时被 3 个专栏收录](#)

2 篇文章 0 订阅

订阅专栏



[Crypto](#)

1 篇文章 0 订阅

订阅专栏



[RSA](#)

1 篇文章 0 订阅

订阅专栏

对于这道RSA题目,网上的Wp有用各种方法解出的,但是对于我这种什么高科技都不会的萌新,只能想用较为直接的方式解题。

对于较为生疏的文件后缀,我们都会搜一下相关资料,譬如:

下载解题文件后,一个以.enc为后缀,一个以.pem为后缀,

ENC文件较为棘手,搜了一下,ENC格式是使用“Encore”软件制作的文件,可以使用Adobe Encore打开。对于需要下载使用新软件的,建议搁置一会儿。

PEM文件是用于对安全网站进行身份验证的Base64编码的证书文件,它可能包含私钥、证书颁发机构(CA)服务器证书或组成信任链的其他各种证书。PEM文件通常从基于Linux的Apache或Nginx Web服务器中导入,并且与OpenSSL应用程序兼容。

"SSL是利用公开密钥的加密技术(RSA)来作为用户端与服务器端在传送机密资料时的加密通讯协定。"这句话更坚定了我们用RSA解答这道题的信心。

从上面我们发现PEM文件是Base64编码的证书文件,不妨用记事本打开文件,

```
pubkey.pem - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
-----BEGIN PUBLIC KEY-----
MDwwDQYJKoZIhvcNAQEBBQADKwAwKAIhAMJjauXD2OQ/
+5erCQKPGqxsC/bNPXDryigb/+1/vjDdAgMBAAE=-----END PUBLIC
KEY-----
```

这里面是加密的pem公钥,使用SSL解密工具 复制它解密时记得全选复制,并不是只复制中间的Base64编码!只复制Base64编码无法解密!

公钥文件 *

```
MDwwDQYJKoZIhvcNAQEBBQADKwAwKAIhAMJjauXD20Q/+5erCQKPGqxsC/bNPXDr  
yigb/+1/vjDdAgMBAAE=
```

解析

❗ 公钥解析失败!

CSDN @程序员小祥

公钥文件 *

```
-----BEGIN PUBLIC KEY-----  
MDwwDQYJKoZIhvcNAQEBBQADKwAwKAIhAMJjauXD20Q/+5erCQKPGqxsC/bNPXDr  
yigb/+1/vjDdAgMBAAE=  
-----END PUBLIC KEY-----
```

解析

详细信息

密钥类型	RSA
密钥强度	256
DER格式	303c300d06092a864886f70d0101010500032b003028022100c2636ae5c3d8e43ffb97ab09028f1aac6c0bf6cd3d70ebca281bffe97f7be30dd0203010001
PN(e)	65537
PN(n)	87924348264132406875276140514499937145050893665602592992418171647042491658461

CSDN @程序员小祥

由此我们得到N.

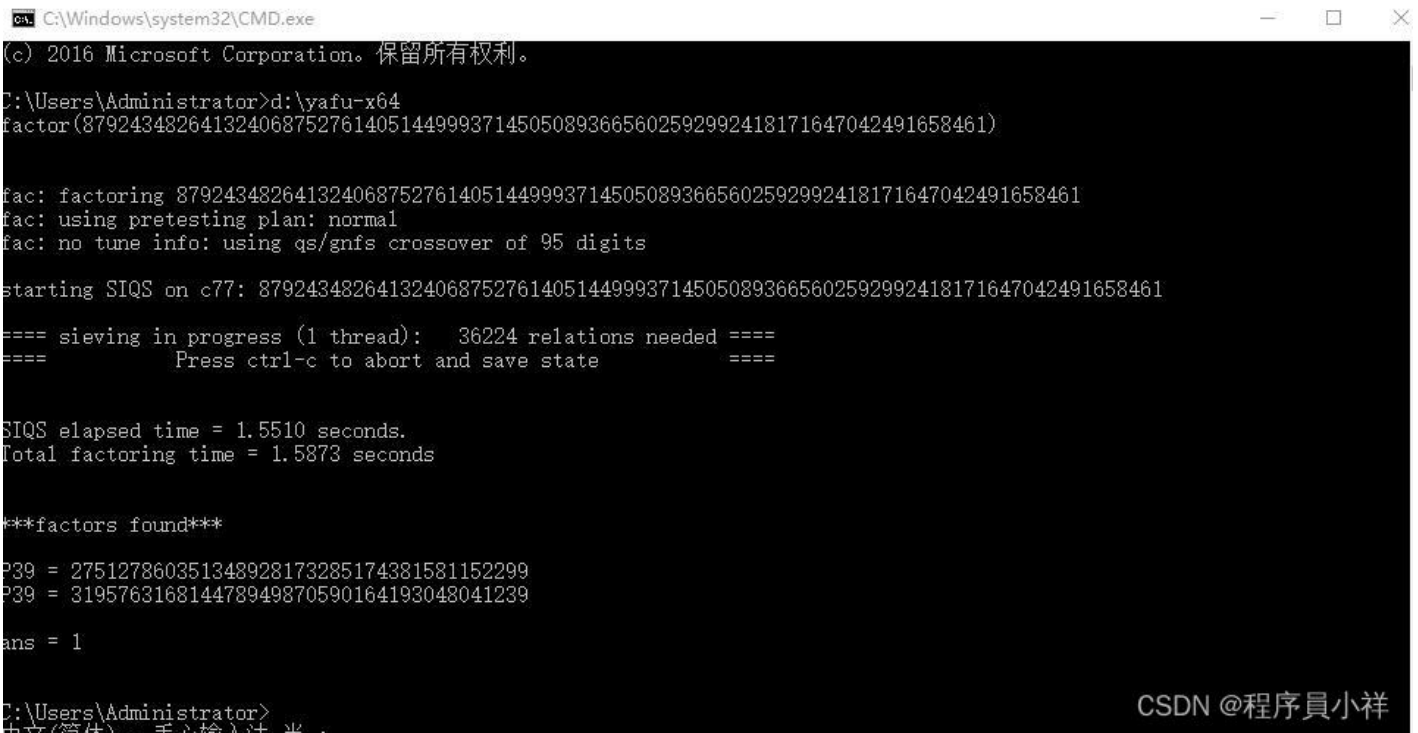
这是"一道基于N分解的RSA"

为什么叫它N呢?因为解出N后,我们先用yafu(什么是yafu可以搜一下,安装及使用也很简单,在站内有详细教程)分解一下,看看能不能分解出p和q,(这是由n找到p和q的方法,分解n是最显然的攻击方法)

这里注意一下,我们刚才用网站分解得到的N被网站割裂开了,

```
PN(n)
8792434826413240687527614051449993714505089366560259299241817164
7042491658461
```

其实应该放在一行.如果我们直接这样复制到yafu中,会出现经典的"mismatched parents",记得把她改成一行再输入yafu



```
C:\Windows\system32\CMD.exe
(c) 2016 Microsoft Corporation. 保留所有权利。
C:\Users\Administrator>d:\yafu-x64
factor(87924348264132406875276140514499937145050893665602592992418171647042491658461)

fac: factoring 87924348264132406875276140514499937145050893665602592992418171647042491658461
fac: using pretesting plan: normal
fac: no tune info: using qs/gnfs crossover of 95 digits

starting SIQS on c77: 87924348264132406875276140514499937145050893665602592992418171647042491658461

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

SIQS elapsed time = 1.5510 seconds.
Total factoring time = 1.5873 seconds

***factors found***
P39 = 275127860351348928173285174381581152299
Q39 = 319576316814478949870590164193048041239

ans = 1

C:\Users\Administrator>
```

CSDN @程序员小祥

到目前,RSA 已经有了n,e,p,q,还缺少c

还记得吗?我们还有一个文件!

遇到没把握的文件建议用Winhex打开看看16进制编码,(这算是方法吧)

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
00000000	6D	3E	B7	DF	23	EE	E1	D3	87	10	BE	BA	78	A0	87	8E	m>·B#iáÓ! %qx !!
00000010	DE	9C	65	BD	3D	08	49	6D	DA	64	92	41	99	11	0C	79	!e½= ImÚd'A! y

我们根据C语言编程基础,表示16进制,在前加0x

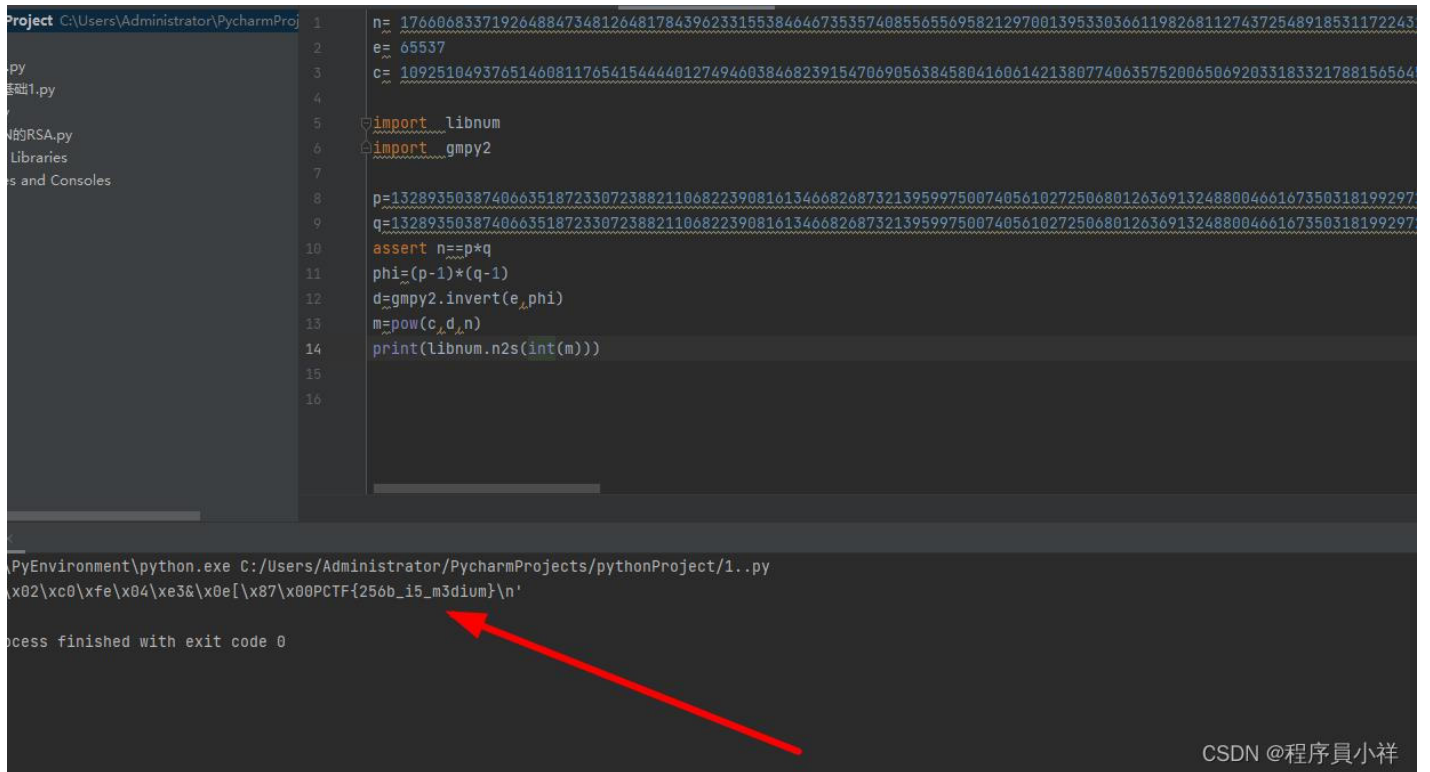
所以c=

0x6d3eb7df23eee1d38710beba78a0878e0e9c65bd3d08496dda64924199110c79

带入解题的RSA python脚本(就这么几行,多研究一下,我也是囤别人写的的),解得flag

```
Project C:\Users\Administrator\PycharmProj 1 n= 176606833719264884734812648178439623315538464673535740855655695821297001395330366119826811274372548918531172243
2 e= 65537
3 c= 109251049376514608117654154444012749460384682391547069056384580416061421380774063575200650692033183321788156564
4
5 import libnum
6 import gmpy2
7
8 p=1328935038740663518723307238821106822390816134668268732139599750074056102725068012636913248800466167350318199297
9 q=1328935038740663518723307238821106822390816134668268732139599750074056102725068012636913248800466167350318199297
10 assert n==p*q
11 phi=(p-1)*(q-1)
12 d=gmpy2.invert(e,phi)
13 m=pow(c,d,n)
14 print(libnum.n2s(int(m)))
15
16

PyEnvironment\python.exe C:/Users/Administrator/PycharmProjects/pythonProject/1..py
\x02\x00\xfe\x04\xe3&\x0e[\x87\x00PCTF{256b_i5_m3dium}\n'
Process finished with exit code 0
```



CSDN @程序员小祥

整篇文章全用截图,是因为我有时候会直接复制,懒得自己再操作一遍(狗头