BugkuCTF-Crypto题rsa

原创

彬彬有礼am_03 于 2021-08-31 15:02:50 发布 129 收藏

分类专栏： # BugkuCTF-Crypto 文章标签： python

本文链接：https://blog.csdn.net/am_03/article/details/120017116

BugkuCTF-Crypto 专栏收录该内容

20 篇文章 1 订阅

订阅专栏

解题流程

n，e已经给出，可以看出e特别大，在e特别大的情况下，可以使用wiener attack的方法进行破解，正好工具RsaCtfTool集成了wiener attack的方法，所以可以直接使用RsaCtfTool计算私钥。

典型的rsa…
密钥的产生：
选两个满足需要的大素数p和q，计算n= p x q, φ(n) = (p - 1) x (q - 1)，其里φ(n)是n的欧拉函数值。
选一个整数e，满足1< e < φ \varphi φ，且gcd(φ(n), e) = 1。通过d x e ≡ 1 mod φ(n)，计算得出d。
以{ e,n }为公开密钥，{d, n }为私钥。

加解密都为模幂运算
加密：c = m^e mod n，即得到密文 c
解密：m = c^d mod n，即得到明文m

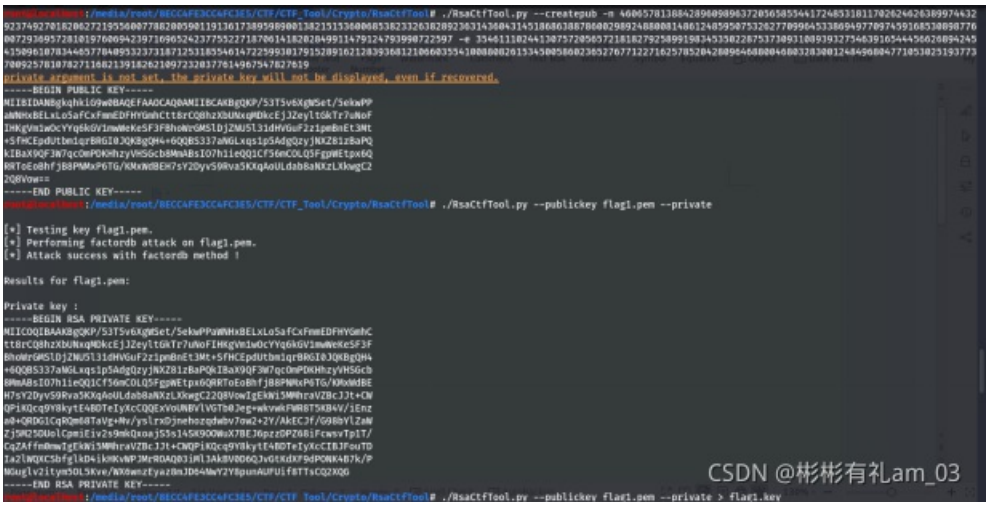
分析

题里给了 n即由两个大素数p x q得到的, e, 以及密文enc，也就是给了{e,n }公开密钥以及密文，我们需要的目标是要获得明文。
通过解密算法我们知道，对于m = c^d mod n，m是我们想要获取的，c为密文，n也是已知的，然而d是不知道的，目标就是要获取d，该题因为n和e是同数量级的，证明d会很小。

解决方案

这里推荐一个工具RsaCtfTool
下面可直接利用这个工具获得 p, q和d。
python RsaCtfTool.py --createpub -n
python RsaCtfTool.py --publickey test.pem --private > test.key
python RsaCtfTool.py --key test.key --dumpkey

注意：
生成公钥或者私钥文件可能有乱码，运行出现格式错误，删除即可



```
#!/usr/bin/python
# -*- coding=utf8 -*-
import gmpy2
from libnum import n2s

n = '460657813884289609896372056585544172485318117026246263899744329237492701820627219556007788200590119136173895989001382151536006853823326382892363143604314518686388786002989248800814861248595075326277099645338694977097459168530898776007293695728101976069423971696524237755227187061418202849911479124793990722597'

e = '354611102441307572056572181827925899198345350228753730931089393275463916544456626894245415096107834465778409532373187125318554614722599301791528916212839368121066035541008808261534500586023652767712271625785204280964688004680328300124849680477105302519377370092578107827116821391826210972320377614967547827619'

enc = '382309913162293996518235675906923010600446204121917377646323846805462562284515182388429652213947118483378324594438444468894683621541882148407367446578858589438101776758719911114666531582571911396056999163473082949956645302808168504827405306022545591237591211063383592202426377759190269335633260694494243911 92'

p = '159918469709932133220726269015607499326863257664034048640233418107353192490663709160906409262190793688455104440314003222291477716829611324204818973628431 99'

q = '288057917712602594868569027290204386866703544412962471482078628360646578497353436182070981639017872873685697684725213446355673342993567600805074546402070 03'

d = '826466797229427501729333977237178332216882214947197683422108239340936369189 5'

n1 = gmpy2.mpz(n)
enc1 = gmpy2.mpz(enc)
d1 = gmpy2.mpz(d)

r = gmpy2.powmod(enc1, d1, n1)
print (r)
s = n2s(r)
print (s)
```

结果为：

```
root@kali:~# python Rsa.py
flag{Wien3r_4tt@ck_1s_3AsY}
```

flag{Wien3r_4tt@ck_1s_3AsY}