Bugku-加密-rsa(WriteUp)

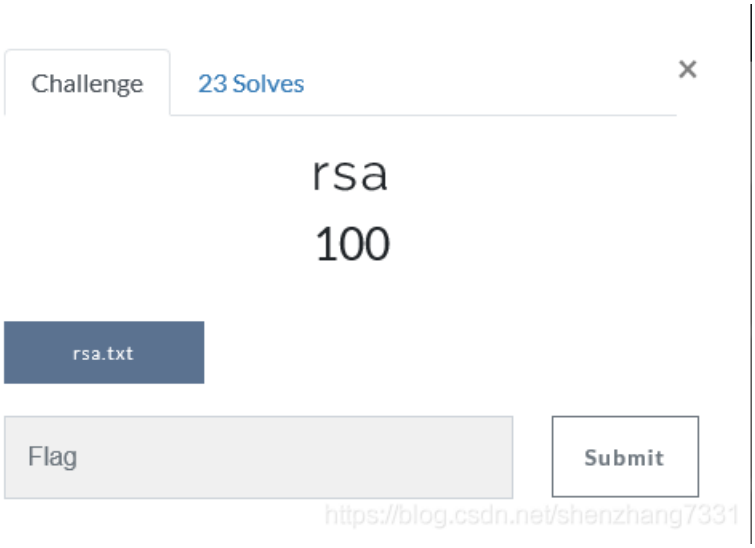
原创

段_Ross 于 2018-11-20 22:43:06 发布 6943 收藏 8

文章标签： CTF

本文链接：https://blog.csdn.net/shenzhang7331/article/details/84311280

该题目是一道rsa解密的题目




```
N :
460657813884289609896372056585544172485318117026246263899744329237492701820627
892363143604314518686388786002989248800814861248595075326277099645338694977097
061418202849911479124793990722597
e :
354611102441307572056572181827925899198345350228753730931089393275463916544456
791528916212839368121066035541008808261534500586023652767712271625785204280964
391826210972320377614967547827619

enc :
382309913162293996518235675906923010600446204121917377646323846805462562284515
407367446578858589438101776758719911114666531582571911396056999163473082949956
759190269335633260694494243911 92
```


ne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， 如下所示：

```
root@kali:~/RsaCtfTool# python RsaCtfTool.py --createpub -n 460657813884289609896372056585544172485318117026246263899744329237492701820627215956007788200590119136173
638997443292374927018206272195560077882005901191361738959890013821515360068538233263828923631436043145186863887860029892488008148612485950753262770996453386949770974
895989001382151536006853823326382892363143604314518686388786002989248800814861248595075326277099645338694977097459168530898776007293695728101976069423971696524237755227187
600298924880081486124859507532627709964533869497709745916853089877600729369572810197606942397169652423775522718706141820284991147912479399072259706141820284991147912479399072259
061418202849911479124793990722597 -e 354611102441307572056572181827925899198345350228753730931089393275463916544
456626894245415096107834465778409532373187125318554614722599301791528916212839368121066035541008808261534500586023652767712271625785204280964
236527677122716257852042809646880046803283001248496804771053025193773700925781078271168213918262109723203776149675478276197 > test.pem
root@kali:~/RsaCtfTool# python RsaCtfTool.py --publickey test.pem --private > test.key
root@kali:~/RsaCtfTool# python RsaCtfTool.py --key test.key --dumpkey
[*] n: 460657813884289609896372056585544172485318117026246263899744329237492701820627219556007788200590119136173
895989001382151536006853823326382892363143604314518686388786002989248800814861248595075326277099645338694977097459168530898776007293695728101976069423971696524237755227187
061418202849911479124793990722597
[*] e: 354611102441307572056572181827925899198345350228753730931089393275463916544566268942454150961078344657784095323731871253185546147225993017915289162128393681210660355410088082615345005860236527677122716257852042809646880046803283001248496804771053025193773700925781078271168213918262109723203776149675478276197
[*] d: 826466797229427501729333977237178332216882214947197683422108239340936369189 5
[*] p: 159918469709932133220726269015607499326863257664034048640233418107353192490663709160906409262190793688455104440314003222291477716829611324204818973628431 99
[*] q: 288057917712602594868569027290204386866703544412962471482078628360646578497353436182070981639017872873685697684725213446355673342993567600805074546402070 03
```


使用pqe直接解密密文，得到flag,代码如下所示:

```
#coding:utf-8
from libnum import n2s,s2n
import base64
def gcd(a, b):   #求最大公约数
if a < b:
    a, b = b, a
while b != 0:
    temp = a % b
    a = b
    b = temp
return a

def egcd(a, b):
if a == 0:
    return (b, 0, 1)
else:
    g, y, x = egcd(b % a, a)
    return (g, x - (b // a) * y, y)

def modinv(a, m):
g, x, y = egcd(a, m)
if g != 1:
    raise Exception('modular inverse does not exist')
else:
    return x % m

if __name__ == "__main__":
p=159918469709932133220726269015607499326863257664034048640233418107353192490663709160906409262190793688455104440314003222291477716829611324204818973628431990
q=288057917712602594868569027290204386866703544412962471482078628360646578497353436182070981639017872873685697684725213446355673342993567600805074546402070030
e = 354611102441307572056572181827925899198345350228753730931089393275463916544456626894245415096107834465778409532373187125318554614722599301791528916212839368121066035541008808261534500586023652767712271625785204280964688004680328300124849680477105302519377370092578107827116821391826210972320377614967547827619
# tmp = base64.b64decode("qzogS7X8M3ZOpkUhJJcbukaRduLyqHAPblmabaYSm9iatuulrHcEpBmil7V40N7gbsQXwYx5EBH5r5V2HRcEIOXjgfk5vpGLjPVxBLyXh2DajHPX6KvbFpQ8jNpCQbUNq8Hst00yDSO/6ri9dk6bk7+uyuN0b2K1bNG5St6sCQ4qYEA3xJbsHFvMqtvUdhMiqO7tNCUVTKZdN7iFvSJqK2IHosIf7FqO24zkHZpHi31sYU7pcgYEaGkVaKs8pjq6nbnffr4URfoexZHeQtq5UAkr95zD6WgvGcxaTDKafFntboX9GR9VUZnHePiio7nJ3msfue5rkIbISjmGCAlj+w==")
#  =
d = modinv(e, (p - 1) * (q - 1))
# c=s2n(tmp)
c = 382309913162293996518235675906923010600446204121917377646323846805462562284515182388429652213947118483378324594438444468894683621541882148407367446578858589438101776758719911114666531582571911396056999163473082949956645302808168504827405306022545591237591211063383592202426377759190269335633260694494243911924
#c = 225031483444634056931067907865853799650197225351377050632290334721073031287701730297815850654473721939907812470206115171738967740183098960272963323728747481560137205796840356532306950935686580268408289864109695494835661414073083573249882362332920722000099781994315336570711188934565379141406727420346806389405536474102730682155998263607095718543239272202402139286809779368710600842078606046563228470023546348908618719147790859257980882643030144242048154566691808688844513142261099020381730517293884263384819159874220288293023868919557980548807831273449743064237407705987056818011286315950476959812697067649075359373253
n = p*q
m=pow(c,d,n)
print n2s(m)
```

运行Rsa.py解得flag

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

安装RsaCtfTool

https://www.freebuf.com/sectool/185468.html