Crypto之RSA(二）

原创

Future prospects 于 2021-11-18 14:56:05 发布 39 收藏 1

分类专栏： CTF # Crypto 文章标签： python

本文链接：https://blog.csdn.net/XXJya/article/details/121400932

CTF 同时被 2 个专栏收录

3 篇文章 0 订阅

订阅专栏

Crypto

3 篇文章 0 订阅

订阅专栏

接上次的RSA，我们来接着学习RSA在CTF的常见类型题目。新涉及了python中的`sympy`库，利用里面的中国剩余定理函数。

低加密指数广播攻击

在RSA中e也称为加密指数。由于e是可以随意选取的，选取小一点的e可以缩短加密时间（比如3），但是选取不当的话，就会造成安全问题

如果选取的加密指数较低，并且使用了相同的加密指数将一个信息多次加密（广播），那么可以进行广播攻击得到明文。选取了相同的加密指数 e（这里取 e=3），对相同的明文 m 进行了加密并进行了消息的传递。

简单来说就是：加密指数e非常小 同一份明文使用不同的模数n，相同的加密指数e进行多次加密 ，拥有多份密文及其模数n，那我们就尝试使用低加密指数广播攻击。

简述原理如下：

c1 = m^e mod n1
c2 = m^e mod n2
c3 = m^e mod n3

运用中国剩余定理，在 e=3 时，可以得到 cx = m^e mod n1n2n3，再通过通过对cx进行e次开方就可以求得明文

例题

2020-CSICTF-Crypto-Quick Math

根据题目描述，我们能提取三组n，c e取3（其实这里我们是猜测e=3）如果不是3，我们也可以利用循环利用不同的e，来求解flag，找到最正确的即可

```
n1 = 86812553978993
n2 = 81744303091421
n3 = 83695120256591
c1 = 8875674977048
c2 = 70744354709710
c3 = 29146719498409
```

然后就是利用中国剩余得到结果再3次开方即可

```
import gmpy2
from sympy.ntheory.modular import *

n = [86812553978993, 81744303091421, 83695120256591]
c = [8875674977048, 70744354709710, 29146719498409]

e = 3

result, N = crt(n, c)  # 中国剩余定理,返回第一个数为结果

flag = gmpy2.iroot(gmpy2.mpz(result), e)[0].digits()  # result开n次方根 gmpy2.mpz(x) 可以为变量a赋予一个高精度的大整数（长度可达50位）

print(bytes.fromhex(str(flag)))
```

得到`flag h45t4d`

低解密指数攻击还没有遇到合适的例题，我们下次在分享受，下面我们来了解一下曾经在一些CTF比赛中出现过的关于RSA的例题

2020-BJDCTF-Crypto-EasyRSA

题目描述 难点在于z的分析

Fraction(a,b) = a/b

Derivative() 求导函数

arctan(),arth() 分别 反正切函数，反双曲正切函数

(artan x) ’ = 1/(1+x^2)

(arch x) ’ = 1/(1-x^2)

所以z = p^2 + q^2

```
from Crypto.Util.number import getPrime,bytes_to_long
from sympy import Derivative
from fractions import Fraction
from secret import flag

p=getPrime(1024)

q=getPrime(1024)

e=65537

n=p*q

z=Fraction(1,Derivative(arctan(p),p)) - Fraction(1,Derivative(arth(q),q))

m=bytes_to_long(flag)

c=pow(m,e,n)

print(c,z,n)
```

那么了解原理之后，我们就可以编写解题脚本了

```
import gmpy2

c = 79225478668577614598074915026542162830127761777895115493506729581018102813484022840983101477965494306892538035109948774201355372685494106526544796208586913241103671820256487884070415999430913862275431821577462029470995723896760843927064060843076570001046656966544091550063132039572928857437917151987819742055786547921231915849576652932083904537483691823331528098823124533597061478081989229167627737217266815889771038774541190437448891645293831880774991949329096439186966468769073273647513809531825178831345918108008489717191848087136943429854581030066760134519122210802527359489936926748993998260848486221458154610351
z = 32115748677623209667471622872185275070257924766015020072805267359839059393284316595882933372289732127274076434587519333300142473010344694803885168557548801202495933226215437763329280242113556524498457559562872900811602056944423967403777623306961880757613246328729616643032628964072931272085866928045973799374711846825157781056965164178505232524245809179235607571567174228822561697888645968559343608375331988097157145264357626738141646556353500994924115875748198318036296898604097000938272195903056733565880150540275369239637793975923329598716003350308259321436752579291000355560431542229699759955141152914708362494482
n = 15310745161336895413406690009324766200789179248896951942047235448901612351128459309145825547569298479821101249094161867207686537607047447968708758990950136380924747359052570549594098569970632854351825950729752563502284849263730127586382522703959893392329333760927637353052250274195821469023401443841395096410231843592101426591882573405934188675124326997277775238287928403743324297705151732524641213516306585297722190780088180705070359469719869343939106529204798285957516860774384001892777525916167743272419958572055332232056095979448155082465977781482598371994798871917514767508394730447974770329967681767625495394441
e = 65537

p1 = gmpy2.iroot(z + 2 * n, 2)[0] # 算出p+q 利用 (p+q)^2 = p^2 + q^2 + 2pq  pq=n 在开根号

p2 = gmpy2.iroot(z - 2 * n, 2)[0] # 同上 算出p-q

p = (p1 + p2) // 2
q = (p1 - p2) // 2

phi = (p - 1) * (q - 1)
d = gmpy2.invert(e, phi)
m = pow(c, d, n)

flag = m


print(bytes.fromhex(hex(flag)[2:]))
```

`flag BJD{Advanced_mathematics_is_too_hard!!!}`

强网杯-2019-Crypto-强网先锋-辅助

题目描述如下,其实很简单无难点，两次加密模数不同，p不同，q相同，那么我们可以回忆上一篇文章的利用公约数求解额攻击方法来求出q

```
flag=open("flag","rb").read()

from Crypto.Util.number import getPrime,bytes_to_long
p=getPrime(1024)
q=getPrime(1024)
e=65537
n=p*q
m=bytes_to_long(flag)
c=pow(m,e,n)
print c,e,n
p=getPrime(1024)
e=65537
n=p*q
m=bytes_to_long("1"*32)
c=pow(m,e,n)
print c,e,n
```

脚本如下

```
import gmpy2

c1 = 24820838937466182485444267370237504001245434520824363343985049860235017106394020609491066932794628969688390
29712099336235976221571564642900240827774719199533124053953157919850838214021934907480633441577316263853011232513
83929049830280521558621542644011081249684040988239466918117989527471942372905813238686666373576046930150790075555
59497424555955551881914084402049848743268494692274123205324989457541779606709065512270230613484822025794329764544
61477488086804856018323986796999103385565540496534422406390355987976815450744535949785073009043007159496929187184
43385928590409175461223439815205082203327858625466088411275 97
n1 = 14967030059975114950295399874185047053736587880127990542035765201425779342430662517765063258784685868107066
78947574718024471135264646977673293854464158384231379187298635750446218492407522743349863142328918798835147566678
85190854210389587594975456064984611990461126684301086241532915267311675164190213474245311019623654865937851653532
87096542347455534823985802155158965016960243942384116069879333811520423814008573868088331343357406024360002850060
08246243584734030595975938914121793991658136225129012633802995610196247414887793670193897757865472920653528850072
24239581776975892385364446446185642939137287519945974807727
e = 65537
n2 = 14624662628725820618622370803948630854094687814338334827462870357582795291844925274690253604919535785934208081
82542554153605755022704839983724339249076216773308303036822124076469369432115010430604412593420169943014697046665
74109992616308259311787318572675997503249186107900989525201135931302450105309613505927352394543376319276695420269
35873535964487595433984902529960726655481696404006628917922241666148082741874033756970724357470539589848548704
57309163391786938723932444773058754547256456149672488279949518676885832449083816912307705189033231367122038583044
43315786743380140809596532018024765162374646518092556799 79
q = gmpy2.gcd(n1, n2) # 利用公约数法

p1 = n1 // q

phi = (p1-1) * (q-1)

d = gmpy2.invert(e,phi)

flag = pow(c1,d,n1)

print(bytes.fromhex(hex(flag)[2:]))
```

`flag flag{i_am_very_sad_233333333333}`