defcon-ctf qualifer crypto writeup

原创

逃课的小学生 于 2020-05-29 16:17:01 发布 471 收藏
分类专栏： crypto ctf


crypto 同时被 2 个专栏收录
20 篇文章 1 订阅
订阅专栏


ctf
30 篇文章 2 订阅
订阅专栏

ooo-flag-sharing

阅读题目，我们发现这是一个密钥分享的密码系统。题目使用100*5的矩阵A对填充为5*1明文向量C(向量第一个数是明文)加密得到100*1的向量B。在系统中保存部分向量数据b1,b2……，并将剩下的向量数据b3,b4……分发。当我们和系统由于5个向量数据b1,b2,b3,b4,b5。我们就可以将A中对应向量行组合成一个新矩阵，并求得逆矩阵A^{-1}。我们将A^{-1}的第一行(s1,s2,s3,s4,s5)和密文向量(b1,b2,b3,b4,b5)相乘即为明文。即m=s1*b1+s2*b2+s3*b3+s4*b4+s5*b5。在redeem_actual_flag中系统中有两个向量B的数据bn,bm。和用户发送3个向量B的数据组合解密。我们可以将b5更改为b5+a*s5^-1。这时我们发现解密结果变为m+a。已知m的最低为是OOO{，并且在redeem_actual_flag()中进行检验flag的头部是OOO{。由于整个操作在模p之下，所以我们尝试改变a，且不改变m的低位“OOO{"。当m+a>=p的时候。我们发现解密的结果结尾不再是OOO{。当m+a<p的时候，我们发现解密的结果结尾是OOO{。这时我们找到临界值a，计算p-a获得m。

首先我们首先使用share_actual_flag()获得一组密文[(numi,bnumi),(numj,bnumj),(numk,bnumk)]

这里我们发现在上述方案中我们需要知道p。我们首先在相同的flagid下输入密文[(0,1),(1,0),(2,0),(3,0),(4,0)]。我们可以得到s1。然后我们更改密文为[(0,2),(1,0),(2,0),(3,0),(4,0)]……[(0,n),(1,0),(2,0),(3,0),(4,0)]。获得a*s1。当我们发现a*s1>a*s1mopp。我们相减获得p。

其次我们需要知道s5^-1是多少。我们可以尝试输入向量(0,0,0,0,1)从而获得s5，继而获得s5^-1。我们需要知道在redeem_actual_flag()中系统中存储的bn和bm具体是对应哪一行。所以我们需要进行爆破。不断输入密文[(i,0),(j,0),(numi,0),(numj,0),(numk,1)]解出a5，再计算a5^-1。我们将b5改为b5+(1<<32)*s5^-1。当验证成功时。说明我们爆破成功，已经获得a5^-1。我们再按照上面的方案求解处flag

```
#!/usr/bin/env python3
import ast
import copy
from pwn import *
from Crypto.Util.number import *

r = remote('ooo-flag-sharing.challenges.ooo', 5000)
#r = remote('127.0.0.1', 20000)

name = 'hahaha'
r.sendlineafter('Username: ', name)

def redeem(secret_id, shares):
```

```
    r.sendlineafter('Choice: ', '2')
    r.sendlineafter("Enter the secret's ID: ", secret_id)
    r.sendlineafter("Enter your shares of the secret: ", str(shares))
    r.recvuntil("Your secret is: ")
    secret = r.recvline().strip()
    return int.from_bytes(eval(secret), 'little')

def store_flag():
    r.sendlineafter('Choice: ', '3')
    r.recvuntil("Our secret's ID is: ")
    secret_id = r.recvline().strip().decode()
    r.recvuntil("Your shares are: ")
    shares = ast.literal_eval(r.recvline().strip().decode())
    return secret_id, shares

def redeem_flag(secret_id, shares):
    r.sendlineafter('Choice: ', '4')
    r.sendlineafter("Enter the secret's ID: ", secret_id)
    r.sendlineafter("Enter your shares of the secret: ", str(shares))
    return r.recvline().startswith(b'Congrats')

secret_id, shares = store_flag()
a = redeem(secret_id, [(0, 1)] + [(i, 0) for i in range(1, 5)])

now = 1
while True:
    now += 1
    aa = redeem(secret_id, [(0, now)] + [(i, 0) for i in range(1, 5)])
    P = a * now - aa
    if P > 0:
        print(f'P = {P}')
        break

shares = sorted(shares)

for i in range(1, 100):
    for j in range(i + 1, 100):
        if i in [shares[x][0] for x in range(3)] or j in [shares[x][0] for x in range(3)]:
            continue

        print(i, j)

        fake_shares = sorted([(s[0], 1 if s == shares[-1] else 0) for s in shares] + [(i, 0), (j, 0)])
        a = redeem(secret_id, fake_shares)
        ai = inverse(a, P)

        fail = False
        for k in range(1, 3):
            newshares = copy.deepcopy(shares)
            newshares[2] = (newshares[2][0], (newshares[2][1] + ai * (k << 32)) % P)
            valid = redeem_flag(secret_id, newshares)
            if not valid:
                fail = True
                break

        if not fail:
            L = 0
            H = P >> 32
            while L != H:
```

```
            print(((P >> 32) - L).to_bytes(32, 'little'))
            print(((P >> 32) - H).to_bytes(32, 'little'))
            M = (L + H) >> 1
            newshares = copy.deepcopy(shares)
            newshares[2] = (newshares[2][0], (newshares[2][1] + ai * (M << 32)) % P)
            valid = redeem_flag(secret_id, newshares)
            if valid:
                L = M + 1
            else:
                H = M
        exit()
```

coooppersmith

对文件逆向，我们发现这是一个RSA加密。首先需要我们输入一个120位的数字。之后系统在我们的数字后加8位并将其更改为一个素数。然后借助该素数生成RSA的私钥和公钥。当公钥中的n小于消息长度的时候将会报错。所以我们需要输入一个尽量小的数字。既保证运算的简单。有保证可以对消息进行加密。这里我们尝试输入数字
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
系统将会在[0,0x10000]中随机生成一个数s和0x10000000000000000000000000000相加为x。当x是素数的时候，就从[1,x]之中随机生成a,b。使得p=(2*x*a)+1，q=(2*x*b)+1。我们获得n=p*q。之后将n和e组合成公钥输出。并输出又该公钥加密的消息。

我们发现n=p*q=((2*x*a)+1)*((2*x*b)+1)=(4*x*x*a*b)+(2*x*a)+(2*x*b)+1。而(n-1)%x=0。我们可以借助此首先将x爆破出来。然后继续推算a，b。由于n-1=(4*x*x*a*b)+(2*x*a)+(2*x*b)，(n-1)/(2*x)=(2*x*a*b)+a+b。由于a+b<2*x。所以((n-1)/(2*x))%(2*x)=a+b.然后我们计算((n-1)/(2*x)-(a+b))=2*x*a*b。((n-1)/(2*x)-(a+b))/(2*x)=a*b。我们已知a+b和a*b。那么$a-b=\sqrt{(a+b)^2-4*a*b}$，a=((a+b)+(a-b))/2，b=a+b-a。我们已知a，b。即可轻松获得p,q。我们获取私钥后即可对消息解密。

```
import gmpy2
from Crypto.PublicKey import RSA
def shuchu(mingwenstr):
    if mingwenstr[len(mingwenstr)-1]=='L':
        mingwenstr=mingwenstr[2:len(mingwenstr)-1]
    else:
        mingwenstr=mingwenstr[2:len(mingwenstr)]
    if not len(mingwenstr)%2==0:
            mingwenstr='0'+mingwenstr
    i=len(mingwenstr)
    mingwen=""
    while i>=1:
        str1=mingwenstr[i-2:i]
        if int(str1,16)>33 and int(str1,16)<126:
            mingwen=chr(int(str1,16))+mingwen
        else :
            mingwen=" "+mingwen
        i=i-2
    print mingwen

pubkey="""-----BEGIN RSA PUBLIC KEY-----
MD0CNkrHjOABLwGYp+ILQURK13nBx4JzR7PBTERgWlTfizn2WLs8xvD0S0w8v2BR
3jkoQ3Lk2al72QIDAQAB
-----END RSA PUBLIC KEY-----"""
key=RSA.importKey(pubkey)
n=key.n
e=key.e
prime=0x100000000000000000000000000000
for i in xrange(0x1000000):
    primex=prime+i
    if gmpy2.is_prime(primex):
        if (n-1)%primex==0:
            prime=primex
            print primex
            break

aandb=((n-1)/(2*prime))%(2*prime)
c=0x216f28e567436bb97d6d5bc084be2f816eb7b3d6d2f4d7765de28f9cf5279c63ce7272dd9902fe0d0e03209189f9cf694e5c832
amulb=(((n-1)/(2*prime))-aandb)/(2*prime)
print gmpy2.iroot(pow(aandb,2)-4*amulb,2)[1]
asubb=gmpy2.iroot(pow(aandb,2)-4*amulb,2)[0]
a=(aandb+asubb)/2
b=aandb-a
print a,b
assert a+b==aandb
assert a*b==amulb
p=2*prime*a+1
q=2*prime*b+1
assert n==p*q
print p,q
phi=(p-1)*(q-1)
d=gmpy2.invert(e,phi)
m=pow(c,d,n)
shuchu(hex(m))
```