

NewStar CTF Week5 WriteUp

Date: 10/28/2024 Author: 夏槿 中山大学网络空间安全学院 23366091 郑梓炫

这周太多ddl了，做不完根本做不完。

Crypto

easy_ecc

推导一下式子就解出来了

```
from Crypto.Util.number import * # type: ignore
# from secret import flag

p = 64408890408990977312449920805352688472706861581336743385477748208693864804529
a =
111430905433526442875199303277188510507615671079377406541731212384727808735043
b = 89198454229925288228295769729512965517404638795380570071386449796440992672131
E = EllipticCurve(GF(p), [a, b])
k = 86388708736702446338970388622357740462258632504448854088010402300997950626097
# K = k * G
# r = getPrime(256)
# c1 = m + r * K
# c2 = r * G

c1 =
E(10968743933204598092696133780775439201414778610710138014434989682840359444219 ,
50103014985350991132553587845849427708725164924911977563743169106436852927878 ,
1)
c2 =
E(16867464324078683910705186791465451317548022113044260821414766837123655851895 ,
35017929439600128416871870160299373917483006878637442291141472473285240957511 ,
1)

m = c1-k*c2

c_left =
15994601655318787407246474983001154806876869424718464381078733967623659362582
c_right =
3289163848384516328785319206783144958342012136997423465408554351179699716569

c_left = c_left//m[0]

c_right = c_right //m[1]

print(c_left)
print(c_right)

# print(long_to_bytes(c_left))

# c_left =bytes_to_long(flag[:len(flag)//2]) * m[0]
# c_right = bytes_to_long(flag[len(flag)//2:]) * m[1]
```

```

# print(f"c1 = {c1}")
# print(f"c2 = {c2}")
# print(f"cipher_left = {c_left}")
# print(f"cipher_right = {c_right}")

# '''
# c1 =
(10968743933204598092696133780775439201414778610710138014434989682840359444219 :
50103014985350991132553587845849427708725164924911977563743169106436852927878 :
1)
# c2 =
(16867464324078683910705186791465451317548022113044260821414766837123655851895 :
35017929439600128416871870160299373917483006878637442291141472473285240957511 :
1)
# c_left =
15994601655318787407246474983001154806876869424718464381078733967623659362582
# c_right =
3289163848384516328785319206783144958342012136997423465408554351179699716569
# '''

# ---

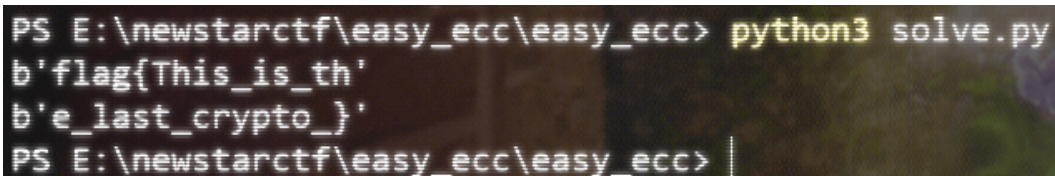
from Crypto.Util.number import * # type: ignore

num1 = 531812496965563174412251588431148136
num2 = 526357398425538015765092604513836925

print(long_to_bytes(num1))
print(long_to_bytes(num2))

# flag{This_is_the_last_crypto_}

```



```

PS E:\newstarctf\easy_ecc\easy_ecc> python3 solve.py
b'flag{This_is_th'
b'e_last_crypto_}'
PS E:\newstarctf\easy_ecc\easy_ecc> |

```

RSA?

正难则反，你不是知道e了嘛，用d加密，那么就用e解密。

然后md5爆破。

```

from Crypto.Util.number import *
from gmpy2 import *
import hashlib

```

```

s =
546151489312666996023365846820368281346591180533427446213489227026035503719116735
709840539297266889014671686337422915211678421892127557118522913540969672001876593
091930988720578649228471690606067064904045966272321573712482949765872211392905482
7469554157634284671989682162929417551313954916635460603628116503

e = 65537

n =
139458221347981983099030378716991183653410063401398496859351212711302933950230621
243347114295539950275542983665063430931475751013491128583801570410029527087462464
558398730501041018349125941967135719526654701663270142483830687281477000567117071
676521061576952568958398421029292366101543468414270793284704549051

m = pow(s,e,n)

print(long_to_bytes(m))

# 86133884de98baada58a8c4de66e15b8

# https://www.cmd5.com/

# adm0n12

mm = 'adm0n12'

def get_flag(m0): # 请用这个函数来转m得到flag
    import hashlib
    flag = 'flag{th1s_1s_my_k3y:' + m0 + '0x' +
hashlib.sha256(m0.encode()).hexdigest() + '}'
    print(flag)

get_flag(mm)

```

```

PS E:\newstarctf\RSA? > python3 solve.py
b'86133884de98baada58a8c4de66e15b8'
PS E:\newstarctf\RSA? > python3 solve.py
b'86133884de98baada58a8c4de66e15b8'
flag{th1s_1s_my_k3y:adm0n120xbfab06114aa460b85135659e359fe443f9d91950ca95cbb2cbd6f88453e2b08b}
PS E:\newstarctf\RSA? > |

```

没e也能玩

直接copy脚本来用

```

c=3120269202161957720142559841744630854438665925759426334495818041711080458520805
178405784084768856736001236734475924778755431065598226532804585398899751250693645
841409810699133417057386334269788864913590362851449743117514907927577517560444096
644216639807215788705825483950968878406889286841490148165572767657471355677142571
844750272701118221597125323385904576933334032009715562246620943818916484679590541
157237449634146738619647445670568239256307233430023256051546619598638497383330743
267698798612808953884231624447467265688928778028243538588459448568818767422119569
86853244518521508714633279380808950337611574412909

```

```

p=1080437256091867817917050904633999888378481283845071366975468851822576134931457
588482157143229991964823039581826393881800632067085751752645020300109719717998508
891239155805186135543827220698742950168415960990304964860691570612110917612735686
31799006187376088457421848367280401857536410610375012371577177832001
q=1215905511215402471148175099661351207519360845282110932753866286666412984570701
262348360533376813259520686733627534080929905533648188514391578686861314163912015
197942446591554112289078970259484360219905208534984626777973928553353640069241066
15008646396883330251028071418465977013680888333091554558623089051503
dp=112829586045939596652643489804463055008046232000788385729894697985469445770647
050300927468273892076342354439446722305370150081131801653952767428078046321161813
858608736779692294607045691723182274912685030395313291415636558116320355221349207
8850164637298628178590101973275656606694831838769040155501078857473
dq=465753573608060540392507861237141778133970652607872085323604364869823634964415
284343092342186726888124377370965799709594036170662436859564615276179355642932194
478373242278932121319331651882052815645520856234833057214005180316514179475688965
38797580895484369480168587284879837144688420597737619751280559493857

from Crypto.Util.number import *
import gmpy2

I = gmpy2.invert(q,p)

mp = pow(c,dp,p)

mq = pow(c,dq,q)

m = (((mp-mq)*I)%p)*q+mq;

print(long_to_bytes(m))

```

```

PS E:\newstarctf\没e也能玩\task1> python3 solve.py
b'flag{No_course_e_can_play}'
PS E:\newstarctf\没e也能玩\task1> |

```

End

完结撒花~