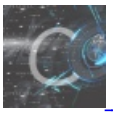
2021-07-06

原创

无名函数 于 2021-07-06 23:23:33 发布 39 收藏

分类专栏： Buu-crypto

本文链接：https://blog.csdn.net/m0_57291352/article/details/118529552



[ACTF新生赛2020]crypto-classic0

题目

hint

```
哼，压缩包的密码？这是小Z童鞋的生日吧==
```

cipher

```
Ygvd mq[lYate[elghqvakl}
```

然后还有一个要密码的压缩包

解题

已知密码是生日，那就是全是数字，并且是八位数字（年份加月日）

使用工具Ziperello

得到密码为19990306

解压得到一个c代码：

```c
#include<stdio.h>

char flag[25] = ***

int main()
{
 int i;
 for(i=0;i<25;i++)
 {
  flag[i] -= 3;
  flag[i] ^= 0x7;
  printf("%c",flag[i]);
 }
 return 0;
}
```

其中^=是按位异或且赋值运算符

可以写出解密程序

```
#include<stdio.h>
char flag[25] = "Ygvd mq[lYate[elghqvakl}";
int main()
{
 int i;
 for(i=0;i<25;i++)
 {
  flag[i] ^= 0x7;
        flag[i] += 3;
  printf("%c",flag[i]);
 }
 return 0;
}
```

运行得到

```
actf*my_naive_encrytion}
```

答案

flag{my_naive_encrytion}

[AFCTF2018]可怜的RSA

题目

flag.enc

```
GVd1d3viIXFfcHapEYuo5fAvIiUS83adrtMW/MgPwxVBSl46joFCQ1plcnlDGfL19K/3PvChV6n5QGohzfVyz2Z5GdTlaknxvHDUGf5HCukokyPw
K/1EYU7NzrhGE7J5jPdi0Aj7xi/Odxy0hGMgpaBLd/nL3N8O6i9pc4Gg3O8soOlciBG/6/xdfN3SzSStMYIN8nfZZMSq3xDDvz4YB7TcTBh4ik4w
YhuC77gmT+HWOv5gLTNQ3EkZs5N3EAopy11zHNYU80yv1jtFGcluNPyXYttU5qU33jcp0Wuznac+t+AZHeSQy5vk8DyWorSGMiS+J4KNqSVlDs12
EqXEqqJ0uA==
```

public.key

```
-----BEGIN PUBLIC KEY-----
MIIBJDANBgkqhkiG9w0BAQEFAAOCAREAMIIBDAKCAQMlsYv184kJfRcjeGa7Uc/4
3pIkU3SevEA7CZXJfA44bUbBYcrf93xphg2uR5HCFM+Eh6qqnybpIKl3g0kGA4rv
tcMIJ9/PP8npdpVE+U4Hzf4IcgOaOmJiEWZ4smH7LWudMlOekqFTs2dWKbqzlC59
NeMPfu9avxxQ15fQzIjhvcz9GhLqb373XDcn298ueA80KK6Pek+3qJ8YSjZQMrFT
+EJehFdQ6yt6vALcFc4CB1B6qVCGO7hICngCjdYpeZRNbGM/r6ED5Nsozof1oMbt
Si8mZEJ/Vlx3gathkUVtlxx/+jlScjdM7AFV5fkRidt0LkwosDoPoRz/sDFz0qTM
5q5TAgMBAAE=
-----END PUBLIC KEY-----
```

解题

公钥解析

详细信息

密钥类型	RSA
密钥强度	2070
PN(e)	65537
PN(n)	79832181757332818552764610761349592984614744432279135328398999801627880283610900361281249973175805069916210179560506497075132524902086881120372213626641879468491936860976686933630869673826972619938321951599146744807653301078602657794957961833150277630398348556604648543103954170846714140826022009859276124501067859234750189417626958051045972963367346806846714419974456373182636210260881103340088781375478028262809944349017001608783860699801749045660130158024485677724116238262817472456609542454137815197942953361975556885435379921971422580532204537576665378402764164756027593749507152838902322307415427373195698197939884314431
DER格式	30820124300d06092a864886f70d01010105000382011003082010c0282010325b18bf5f389097d17237866bb51cff8de922453749ebc403b0995c97c0e386d46c161cadff77c69860dae4791c214cf8487aaaa9f26e920a977834906038aefb5c30827dfcf3fc9e9769544f94e07cdfe0872039a3a6262116678b261fb2d6b9d32539e92a153b3675629bab3942e7d35e30f7eef5abf1c50d797d0cc88e1bdccfd1a12ea6f7ef75c3727dbdf2e780f3428ae8f7a4fb7a89f184a365032b153f8425e845750eb2b7abc02dc15ce0207507aa950863bb8480a78028dd62979944d6c633fafa103e4db28ce87f5a0c6ed4a2f2664427f565c7781ab6191456d971c7ffa395272374cec0155e5f91189db742e4c28b03a0fa11cffb03173d2a4cce6ae530203010001

得到：

```
e=65537
n=79832181757332818552764610761349592984614744432279135328398999801627880283610900361281249973175805069916210179560506497075132524902086881120372213626641879468491936860976686933630869673826972619938321951599146744807653301076026577949579618331502776303983485566046485431039541708467141408260220098592761245010678592347501894176260958051045972963367346806846714419974456373182636210260881103340088781375478028262809944349017001608783860699801749045660130158024485677724116238262817472456609542454137815197942953361975556885435379921971422580532204537576665378402764164756027593749507152838902322307415427373195698197939884314431
```

分解n得到：

```
p=3133337
q=254783260649374192922001721363994977190818429145282283164559062116931183219713999360047291348411629741442462714864396957860365881174246118819559509962196468073788222782856382615820991083394389495730341012151411561564087428438200480668308638143623798857203950823184628500029016056897618763191511473527300909575569408421442998873946787436077669378280944783364011594490358783068537162165483742734623865083073677131120730040113834189678949305540675824532489810220119228833744427368480459206763413618712317871634414675330768900817218821793691687872877247696426653999925560521448458786001262839688902730675753420617762449394
```

解密：

```
import gmpy2
from base64 import b64decode
from Crypto.Util.number import long_to_bytes
from Crypto.Cipher import PKCS1_OAEP
from Crypto.PublicKey import RSA

c='GVd1d3viIXFfcHapEYuo5fAvIiUS83adrtMW/MgPwxVBSl46joFCQ1plcnlDGfL19K/3PvChV6n5QGohzfVyz2Z5GdTlaknxvHDUGf5HCukok
yPwK/1EYU7NzrhGE7J5jPdi0Aj7xi/Odxy0hGMgpaBLd/nL3N8O6i9pc4Gg3O8soOlciBG/6/xdfN3SzSStMYIN8nfZZMSq3xDDvz4YB7TcTBh4i
k4wYhuC77gmT+HWOv5gLTNQ3EkZs5N3EAopy11zHNYU80yv1jtFGcluNPyXYttU5qU33jcp0Wuznac+t+AZHeSQy5vk8DyWorSGMiS+J4KNqSVlD
s12EqXEqqJ0uA=='
e=65537
n=798321817573328185527646107613495929846147444322791353283989998016278802836109003612812499731758050699162101795605064970751325249020868811203722136266418794684919368609766869336308696738269726199383219515991467448076533010760265779495796183315027763039834855660464854310395417084671414082602200985927612450106785923475018941762695805104597296336734680684671441997445637318263621026088110334008878137547802826280994434901700160878386069980174904566013158024485677724116238262817472456609542454137815197942953361975556885435379921971422580532204537576665378402764164756027593749507152838902322307415427373195698197939884314430
p=3133337
q=254783260649374192922001721363994977190818429145282283164559062116931183219713999360047291348411629741442462714864396957860365881174246118819559509962196468073788222782856382615820991083394389495730341012151411561564087428438200480668308638143623798857203950823184628500029016056897618763191511473527300909575569408421442998873946787436077669378280944783364011594490358783068537162165483742734623865083073677131120730040113834189678949305540675824532489810220119228833744427368480459206763413618712317871634414675330768900817218821793691687872877247696426653999925560521448458786001262839688902730675753420617762449390
```

注意，不能通过m = gmpy2.powmod(c,d,n)解，因为c不是数字
运行

```
b'afctf{R54_|5_$0_B0rin9}'
```

答案
flag{R54_|5_$0_B0rin9}

[RoarCTF2019]babyRSA

题目

```
import sympy
import random

def myGetPrime():
    A= getPrime(513)
    print(A)
    B=A-random.randint(1e3,1e5)
    print(B)
    return sympy.nextPrime((B!)%A)
p=myGetPrime()
#A1=2185696345246163043734827843419143400006607675041902749385246351346986526206434083661383106660230095977263239777348731756033905665829995446416926446723440 7
#B1=2185696345246163043734827843419143400006607675041902749385246351346986526206434083661383106660230095977263239777348731756033905665829995446416926446714059 6
q=myGetPrime()
#A2=1646611311583922811976788789930882002574926093386344688822416716985761217866413954572634086740679075456022751601379626994143807681819461703030485185841892 7
#B2=1646611311583922811976788789930882002574926093386344688822416716985761217866413954572634086740679075456022751601379626994143807681819461703030485185835102 6

r=myGetPrime()

n=p*q*r
#n=8549266378627529215983160339108387617514935430932767300871662765071816058563972310079334753464962833041663125566090130753390990043141344752426233223265915304706790869348194712106907045156282241735765643217187095118467313255421369012330804269736196998636037506095470292065636414415414581283855836533417293593144142409627020614069181466231856269692576799193736978262790840823908735803316541002069015206771571111273225203858843289675840589870901034246788226436273 3
c=pow(flag,e,n)
#e=0x1001
#c=7570088302166957773932931679545070620450263580231073147715699883471082077024521946870324530200999893206708038397756029970806047622208963020997262975596514031752603468045248336091737881224436588452718605634188861556433556076505355015575836227162233001743340302726112756122558591248477782958850121396111069045198762550270133148514163968435642731690512299575982524113387273436271604181981994864566280329241880220443087452134210841362363515047596312122009523677642 8
#so,what is the flag?
```

解题

自己写了个程序改了半天还是不对

还是膜拜大神的吧□

```
from sympy import nextprime
from Crypto.Util.number import *
from gmpy2 import invert

def get_p_q(A,B):
    tmp = 1
    # calculate remain value (mod A) of (A-1)(A-2)(A-3)...(B+1)
    for i in range(B+1,A-1):
        tmp *= i
        tmp %= A

    tmp_inv = invert(tmp,A)
    result = nextprime(tmp_inv)
    return result

A1=2185696345246163043734827843419143400006607675041902749385246351346986526206434083661383106660230095977263239777348731756033905665829995446416926446723440 7
```

```
B1=2185696345246163043734827843419143400006607675041902749385246351346986526206434083661383106660230095977263239
7773487317560339056658299954464169264467140596

A2=1646611311583922811976788789930882002574926093386344688822416716985761217866413954572634086740679075456022751
6013796269941438076818194617030304851858418927
B2=1646611311583922811976788789930882002574926093386344688822416716985761217866413954572634086740679075456022751
6013796269941438076818194617030304851858351026

n=8549266378627529215983160339108387617514935430932767300871662765071816058563972310079334753464962833041663125 5
6609013075339099004314134475242623322326591530470679086934819471210690704515628224173576564321718709511846731325
5421369012330804269736196998636037506095470292065636414415414581283855836533417293593144142409627020614069181466
2318562696925767991937369782627908408239087358033165410020690152067715711112732252038588432896758405898709010342
467882264362733
e=0x1001
c=7570088302166957773932931679545070620450263580231073147715699883471082077024521946870324530200999893206708038 3
9775602997080604762220896302099726297559651403175260346804524833609173788122443658845271860563418886155643355607
6505355015575836227162233001743340302726112756122558591248477782958850121396111069045198762550270133148514163968
4356427316905122995759825241133872734362716041819819948645662803292418802204430874521342108413623635150475963121
220095236776428

p = get_p_q(A1,B1)
q = get_p_q(A2,B2)
print(p)
print(q)
# p = 1276519424397216455160791032620569392845781005616561979809403385593761615670426423039762716291920053306063
2145483596565558091231273615394752384352856548 51
# q = 1324217549358358410841132414377378086242618338201775312963397893321367477048776538798528295657419727405616
2861584407275172775868763712231230219112670015751

r = n // p // q
print(r)
# r = 5057572094237208127867754008134739503717927865750318894982404287656747895573075881186030840558129423864679
8866460664774370204506548488398614556613852054 33

phn = (p - 1) * (q - 1) * (r - 1)
d = invert(e, phn)
print(d)
# d = 2324599156893108993557539813953317990215191132550427818689536812372468413287836259074537201698796337810205
6924287587028702166372731411906405181410326380814220943063812165970658883369631308421395770179828382024820676516
2611882764567374347764043403813748593049448849477726979154453016414490233746272145732925391613209597794180432758
8920242152106970587841482357878144116076691406837701742838077562588602338501962349978498082262941579588422850449
8888092721097658433
m = pow(c,d,n)
print(m)
# 4956218809645863041056304441735881834191326557137372526697661212652610652840494474504461412623207407381393625 9
453
print(long_to_bytes(m))
```

出自：https://www.cnblogs.com/vict0r/p/13563073.html

解析：

从题目代码中可以知道：
p=(B1!)%A1
q=(B2!)%A2

又由威尔逊定理知道，(A−1)!≡ −1 mod A

而B=A-random.randint(1e3,1e5)，所以在B的前面补上(A−1)(A−2)(A−3)…(B+1)就有(A−1)(A−2)(A−3)…(B+1)∗(B!)%A=−1%A

于是再整理一下又有

(A−2)(A−3)…(B+1)∗(B!)%A=1%A

这就意味这可由(A−2)(A−3)…(B+1)模A的逆元求得(B!)%A的值。

再取nextprime就可得到p或q的值。

答案

flag{wm-CongrAtu1ation4-1t4-ju4t-A-bAby-R4A}