BUUCTF 每日打卡 2021-4-28

原创

Σ2333! 于 2021-04-28 22:53:09 发布 189 收藏

分类专栏： crypto 文章标签： 密码学 加密解密

本文链接：https://blog.csdn.net/weixin_52446095/article/details/116244826

引言

数分考完了

明天蓝帽杯

可怜的RSA

附件给了公钥：

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

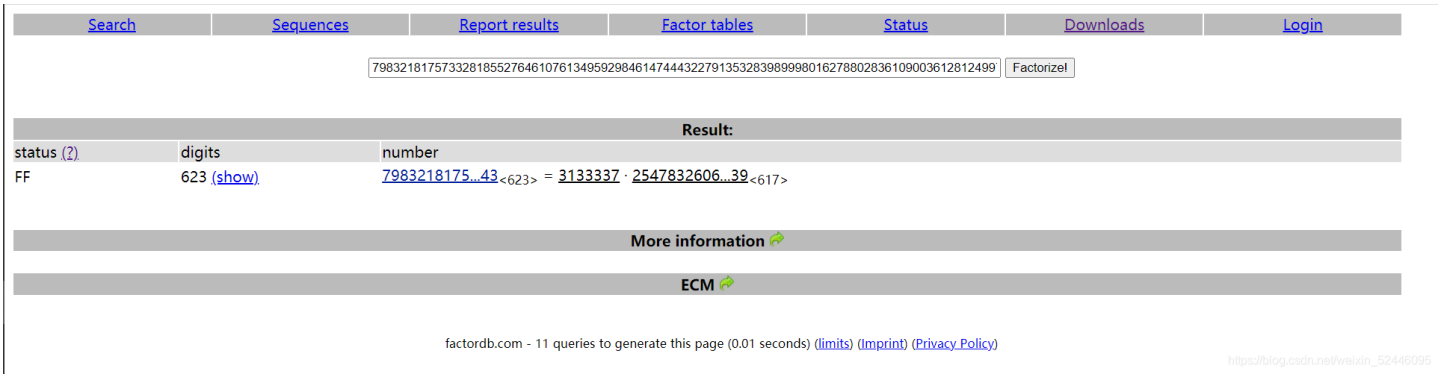
提取公钥代码如下：

```
from Crypto.PublicKey import RSA

with open("public.key","r") as f:
    key = RSA.import_key(f.read())
    print(key.n)
    print(key.e)
```

得到

```
e = 65537
n = 79832181757332818552764610761349592984614744432279135328398999801627880283610900361281249973175805069916210179560506497075132524902086881120372213626641879468491936860976686933630869673826972619938321951599146744807653301076026577949579618331502776303983485566046485431039541708467141408260220098592761245010678592347501894176269580510459729633673468068467144199744563731826362102608811033400887813754780282628099443490170016087838606998017490456601315802448567772411623826281747245660954245413781519794295336197555688543537992197142258053220453757666537840276416475602759374950715283890232230741542737319569819793988431443
```

然后爆破就可以求出 p,q




```
p = 3133337
q = 254783260649374192922001721363994977190818429145282283164559062116931183219713999360047291348411629741442462
714864396957860365881174246118819559509962196468073788222782856382615820991083394389495730341012151411561564087 4
284382004806683086381436237988572039508231846285002901605689761876319151147352730090957556940842144299887394678
743607766937828094478336401159449035878306853716216548374273462386508307367713112073004011383418967894930554067 5
824532489810220119228833744427368480459206763413618712317871634414675330768900817218821793691687872877247696426 6
539999255605214484587860012628396889027306757534206177624493 9
```

附件密文如下：

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

容易发现经过了 base64 加密
尝试解密：

```
import base64
from Crypto.Util.number import *
# p,q,n,e
c = bytes_to_long(base64.b64decode(b'GVd1d3viIXFfcHapEYuo5fAvIiUS83adrtMW/MgPwxVBSl46joFCQ1plcnlDGfL19K/3PvChV6n
5QGohzfVyz2Z5GdTlaknxvHDUGf5HCukokyPwK/1EYU7NzrhGE7J5jPdi0Aj7xi/Odxy0hGMgpaBLd/nL3N8O6i9pc4Gg3O8soOlciBG/6/xdfN3
SzSStMYIN8nfZZMSq3xDDvz4YB7TcTBh4ik4wYhuC77gmT+HWOv5gLTNQ3EkZs5N3EAopy11zHNYU80yv1jtFGcluNPyXYttU5qU33jcp0Wuznac
+t+AZHeSQy5vk8DyWorSGMiS+J4KNqSVlDs12EqXEqqJ0uA=='))

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

print(long_to_bytes(m))
```

发现是一堆乱码

```
79832181757332818552764610761349592984614744432279135328398999801627880283610900361281249973175805069916210179560506497075132524902086881120372213626665537
b'\xb2u~\xc4\xd0\xbeV\x1c\x80\x0b\x16F\xee[\xe9\xf4\xbfJ\xd2\x17p\xc2c\xfcvb\x14\xe4\x8c`A\xd4\'Iy=\x11\xeah:\xb9n\x0b\x01\xd2\x03V\xaa\xb2C{\xc3yP\x8
```

嗯？
最后求助 wp
代码如下：

```
from Crypto.Cipher import PKCS1_OAEP
import base64
from Crypto.Util.number import *
# p,q,n,e
c = base64.b64decode(b'GVd1d3viIXFfcHapEYuo5fAvIiUS83adrtMW/MgPwxVBSl46joFCQ1plcnlDGfL19K/3PvChV6n5QGohzfVyz2Z5G
dTlaknxvHDUGf5HCukokyPwK/1EYU7NzrhGE7J5jPdi0Aj7xi/Odxy0hGMgpaBLd/nL3N8O6i9pc4Gg3O8soOlciBG/6/xdfN3SzSStMYIN8nfZZ
MSq3xDDvz4YB7TcTBh4ik4wYhuC77gmT+HWOv5gLTNQ3EkZs5N3EAopy11zHNYU80yv1jtFGcluNPyXYttU5qU33jcp0Wuznac+t+AZHeSQy5vk8
DyWorSGMiS+J4KNqSVlDs12EqXEqqJ0uA==')

phi = (p-1)*(q-1)
d = inverse(e, phi)
key_info = RSA.construct((n, e, d, p, q))
key = RSA.importKey(key_info.exportKey())
key = PKCS1_OAEP.new(key)
print(key.decrypt(c))
```

结果为：afctf{R54_|5_$0_B0rin9}
啊这
多西得？
我们知道，
Crypto.Util 包中 long_to_bytes 方法是将字符串转化成二进制然后转化成十进制
示例如下：

```
>>> from Crypto.Util.number import *
>>> print(bytes_to_long(b'AB'))
16706
>>> print(bin(ord('A')))
0b1000001
>>> print(bin(ord('B')))
0b1000010
>>> print(hex(ord('A')))
0x41
>>> print(hex(ord('B')))
0x42
>>> print(int('4142',16))
16706
>>> print(int('100000101000010',2))
16706
```

然后我又找了 Crypto.Cipher 包中的 decrypt 方法源码

```
    def decrypt(self, ciphertext):
        """Decrypt a message with PKCS#1 OAEP.
        :param ciphertext: The encrypted message.
        :type ciphertext: bytes/bytearray/memoryview
        :returns: The original message (plaintext).
        :rtype: bytes
        :raises ValueError:
            if the ciphertext has the wrong length, or if decryption
            fails the integrity check (in which case, the decryption
            key is probably wrong).
        :raises TypeError:
            if the RSA key has no private half (i.e. you are trying
            to decrypt using a public key).
        """

        # See 7.1.2 in RFC3447
        modBits = Crypto.Util.number.size(self._key.n)
        k = ceil_div(modBits,8) # Convert from bits to bytes
        hLen = self._hashObj.digest_size

        # Step 1b and 1c
        if len(ciphertext) != k or k<hLen+2:
            raise ValueError("Ciphertext with incorrect length.")
        # Step 2a (O2SIP)
        ct_int = bytes_to_long(ciphertext)
        # Step 2b (RSADP)
        m_int = self._key._decrypt(ct_int)
        # Complete step 2c (I2OSP)
        em = long_to_bytes(m_int, k)
        # Step 3a
        lHash = self._hashObj.new(self._label).digest()
        # Step 3b
        y = em[0]
        # y must be 0, but we MUST NOT check it here in order not to
        # allow attacks like Manger's (http://dl.acm.org/citation.cfm?id=704143)
        maskedSeed = em[1:hLen+1]
        maskedDB = em[hLen+1:]
        # Step 3c
        seedMask = self._mgf(maskedDB, hLen)
        # Step 3d
        seed = strxor(maskedSeed, seedMask)
        # Step 3e
        dbMask = self._mgf(seed, k-hLen-1)
        # Step 3f
        db = strxor(maskedDB, dbMask)
        # Step 3g
        one_pos = hLen + db[hLen:].find(b'\x01')
        lHash1 = db[:hLen]
        invalid = bord(y) | int(one_pos < hLen)
        hash_compare = strxor(lHash1, lHash)
        for x in hash_compare:
            invalid |= bord(x)
        for x in db[hLen:one_pos]:
            invalid |= bord(x)
        if invalid != 0:
            raise ValueError("Incorrect decryption.")
        # Step 4
        return db[one_pos + 1:]
```

其中也有 long_to_bytes 和 bytes_to_long 方法，原理类似（大概）
因为举例比较麻烦，所以就没有尝试
浏览了一下源码（因为不怎么看得懂），发现里面有 hash, xor 之类的字眼
而且看注释内容（也可见于官方文档）
发现它对密文长度也做了要求，如果密文长度错误会报错 ValueError
就很迷。。。

那么如果碰到了该怎么办？
那就先尝试第一种代码，再尝试第二种代码咯[扶额]

Single

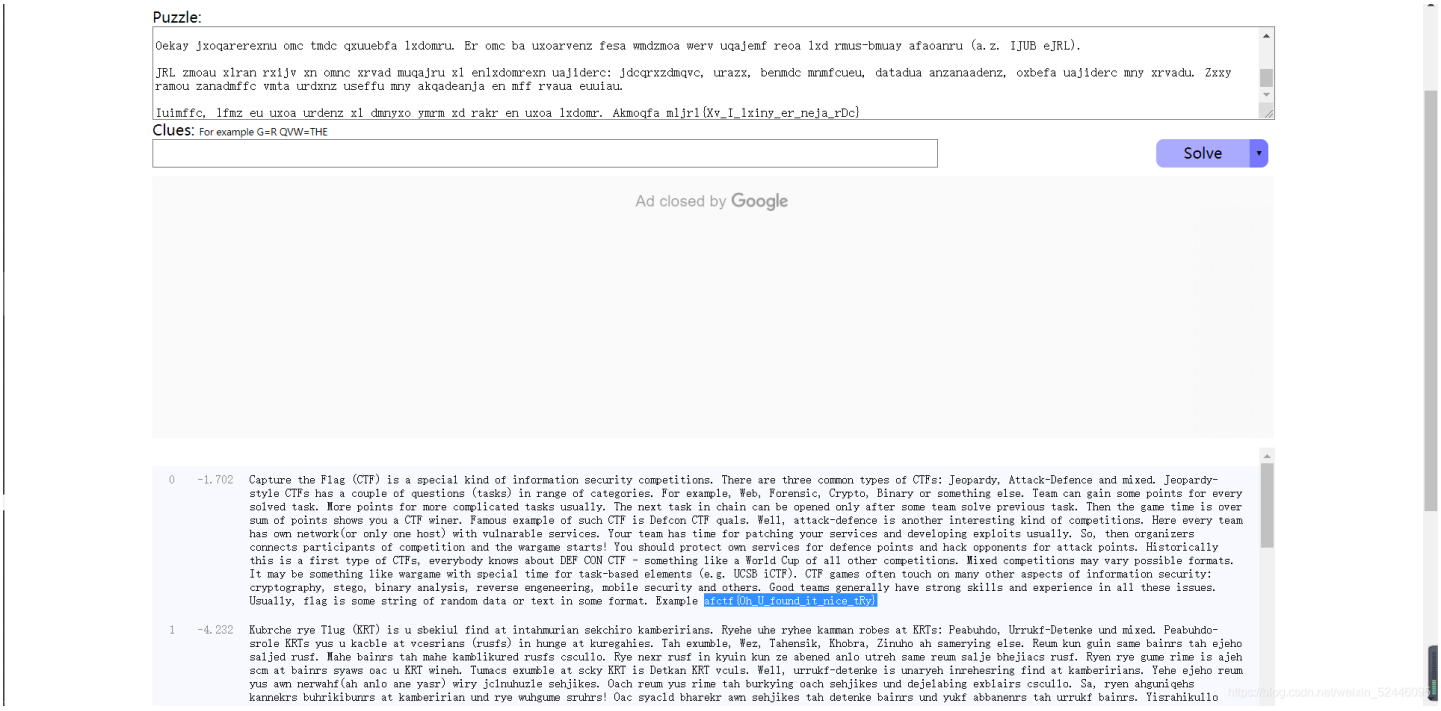
加密代码如下：

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
 freopen("Plain.txt","r",stdin);
 freopen("Cipher.txt","w",stdout);
 map<char, char> f;
 int arr[26];
 for(int i=0;i<26;++i){
  arr[i]=i;
 }
 random_shuffle(arr,arr+26);
 for(int i=0;i<26;++i){
  f['a'+i]='a'+arr[i];
  f['A'+i]='A'+arr[i];
 }
 char ch;
 while((ch=getchar())!=EOF){
  if(f.count(ch)){
   putchar(f[ch]);
  }else{
   putchar(ch);
  }
 }
 return 0;
}
```

这学期刚学 C 语言
大致看了一下
大概就是给明文随机移位了一下
密文如下：

```
Jmqrida rva Lfmz (JRL) eu m uqajemf seny xl enlxdomrexn uajiderc jxoqarerexnu. Rvada mda rvdaa jxooxn rcqau xl J
RLu: Paxqmdyc, Mrrmjs-Yalanja mny oekay.

Paxqmdyc-urcfa JRLu vmu m jxiqfa xl giaurexnu (rmusu) en dmnza xl jmrazxdeau. Lxd akmoqfa, Wab, Lxdanuej, Jdcqrx
, Benmdc xd uxoarvenz afua. Ramo jmn zmen uxoa qxenru lxd atadc uxftay rmus. Oxda qxenru lxd oxda jxoqfejmray rm
usu iuimffc. Rva nakr rmus en jvmen jmn ba xqanay xnfc mlrad uxoa ramo uxfta qdatexiu rmus. Rvan rva zmoa reoa e
u xtad uio xl qxenru uvxwu cxi m JRL wenad. Lmoxiu akmoqfa xl uijv JRL eu Yaljxn JRL gimfu.

Waff, mrrmjs-yalanja eu mnxrvad enradaurenz seny xl jxoqarerexnu. Vada atadc ramo vmu xwn narwxds(xd xnfc xna vx
ur) werv tifnmdmbfa uadtejau. Cxid ramo vmu reoa lxd qmrjvenz cxid uadtejau mny yatafxqenz akqfxeru iuimffc. Ux,
 rvan xdzmnehadu jxnnajru qmdrejeqmnru xl jxoqarerexn mny rva wmdzmoa urmdru! Cxi uvxify qdxrajr xwn uadtejau lx
d yalanja qxenru mny vmjs xqqxnanru lxd mrrmjs qxenru. Veurxdejmffc rveu eu m ledur rcqa xl JRLu, atadcbxyc snxw
u mbxir YAL JXN JRL - uxoarvenz fesa m Wxdfy Jiq xl mff xrvad jxoqarerexnu.

Oekay jxoqarerexnu omc tmdc qxuuebfa lxdomru. Er omc ba uxoarvenz fesa wmdzmoa werv uqajemf reoa lxd rmus-bmuay
afaoanru (a.z. IJUB eJRL).

JRL zmoau xlran rxijv xn omnc xrvad muqajru xl enlxdomrexn uajiderc: jdcqrxzdmqvc, urazx, benmdc mnmfcueu, datad
ua anzanaadenz, oxbefa uajiderc mny xrvadu. Zxxy ramou zanadmffc vmta urdxnz useffu mny akqadeanja en mff rvaua
euuiau.

Iuimffc, lfmz eu uxoa urdenz xl dmnyxo ymrm xd rakr en uxoa lxdomr. Akmoqfa mljrl{Xv_I_lxiny_er_neja_rDc}
```

尝试爆破



啊这
完事了

结语

这几天的博客属实水
主要是数分考试的原因
（但是差不多凉了，求求给我及格吧）
希望继续坚持