

攻防世界——Crypto新手练习区11题（Normal RSA）题解

原创

筠yun 于 2020-07-02 16:08:22 发布 1382 收藏 4

分类专栏: [CTF](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: https://blog.csdn.net/Catherine_qingzhu/article/details/107037306

版权



[CTF 专栏收录该内容](#)

11 篇文章 1 订阅

订阅专栏

题目分析

从题目就可以看出来这道题在考标准的RSA算法, 而且会用到工具。下载附件解压后, 发现有两个文件flag.enc和pubkey.pem。从文件的命名可知, 前者就是加密后的flag, 而后者是公钥。因为RSA是非对称加密算法, 公钥加密, 私钥解密, 可以保证信息的机密性, 而私钥加密, 公钥解密, 可以对信息进行签名, 保证信息不可以被抵赖。这里用到的工具就是openssl, 其中包含了许多常见的加密算法, 可以使用它来查看公钥私钥, 并加密解密。为了避免安装openssl的麻烦, 建议直接装一个kali虚拟机, 其中自带了openssl工具, 而且对于信息安全爱好者来说, kali真的是必不可少的学习工具。

```
kali@kali: ~/Downloads/crypto
File Actions Edit View Help
kali@kali:~/Downloads/crypto$ openssl
OpenSSL> help
Standard commands
asn1parse          ca                ciphers           cms
crl                crl2pkcs7        dgst              dhparam
dsa               dsaparam         ec                ecparam
enc              engine          errstr           gendsa
genpkey          genrsa          help             list
nseq            ocpkcs8         passwd          pkcs12
pkcs7           pkcs8           pkey            pkeyparam
pkeyutl        prime          rand            rehash
req            rsa            rsautl         s_client
s_server       s_time        sess_id        smime
speed          spkac          srp            storeutl
ts            verify        version        x509

Message Digest commands (see the `dgst' command for more details)
blake2b512      blake2s256      gost             md4
md5             rmd160          sha1             sha224
sha256          sha3-224        sha3-256        sha3-384
sha3-512        sha384          sha512          sha512-224
sha512-256     shake128         shake256         sm3

Cipher commands (see the `enc' command for more details)
aes-128-cbc     aes-128-ecb     aes-192-cbc     aes-192-ecb
aes-256-cbc     aes-256-ecb     aria-128-cbc     aria-128-cfb
aria-128-cfb1   aria-128-cfb8   aria-128-ctr     aria-128-ecb
```

kali中使用openssl

解题过程

开始我以为是用公钥直接去解密flag.enc, 后来发现这样行不通。直到我看了一篇[博客](#), 才意识到应该是利用公钥求解出私钥, 然后用私钥解密flag。接下来就按照博客中的步骤解出来了这道题。

1. 查看公钥

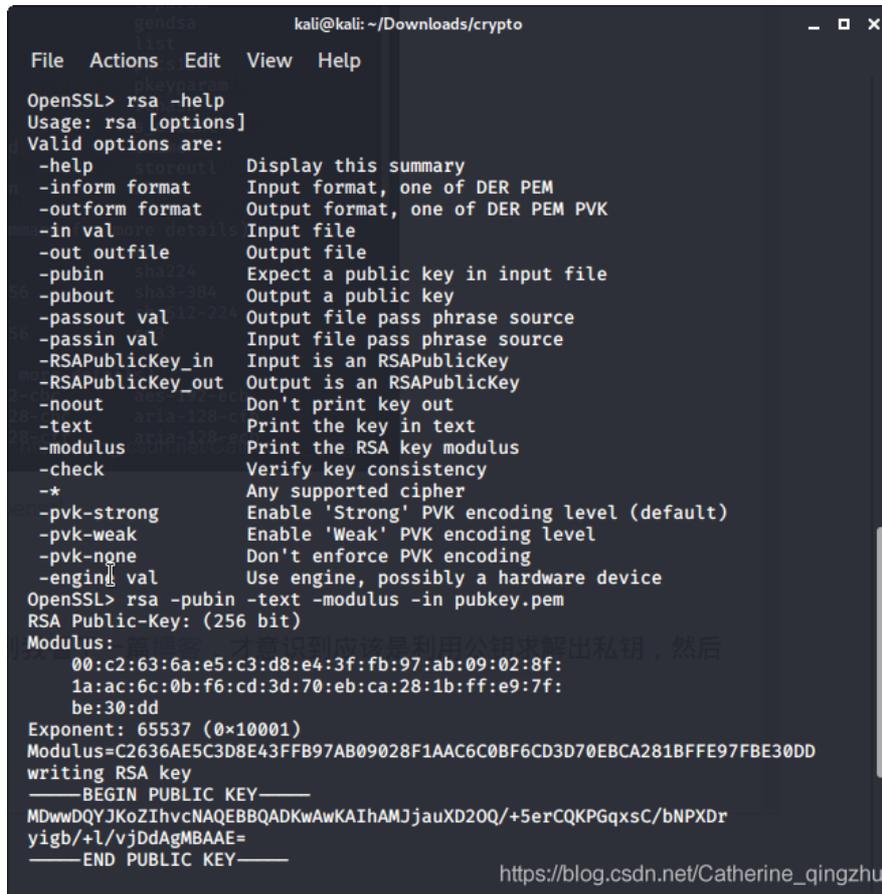
使用下面的命令, 可以查看调用rsa算法的相应参数

```
OpenSSL> rsa -help
```

查看公钥pubkey.pem的命令如下

```
OpenSSL> rsa -pubin -text -modulus -in pubkey.pem
```

输出的公钥文件如下图，



```
kali@kali: ~/Downloads/crypto
File Actions Edit View Help
OpenSSL> rsa -help
Usage: rsa [options]
Valid options are:
  -help           Display this summary
  -inform format  Input format, one of DER PEM
  -outform format Output format, one of DER PEM PVK
  -in val         Input file
  -out outfile    Output file
  -pubin          Expect a public key in input file
  -pubout         Output a public key
  -passout val    Output file pass phrase source
  -passin val     Input file pass phrase source
  -RSAPublicKey_in Input is an RSAPublicKey
  -RSAPublicKey_out Output is an RSAPublicKey
  -noout         Don't print key out
  -text          Print the key in text
  -modulus       Print the RSA key modulus
  -check        Verify key consistency
  -*            Any supported cipher
  -pvk-strong    Enable 'Strong' PVK encoding level (default)
  -pvk-weak     Enable 'Weak' PVK encoding level
  -pvk-none     Don't enforce PVK encoding
  -engine val    Use engine, possibly a hardware device
OpenSSL> rsa -pubin -text -modulus -in pubkey.pem
RSA Public-Key: (256 bit)
Modulus:
  00:c2:63:6a:e5:c3:d8:e4:3f:fb:97:ab:09:02:8f:
  1a:ac:6c:0b:f6:cd:3d:70:eb:ca:28:1b:ff:e9:7f:
  be:30:dd
Exponent: 65537 (0x10001)
Modulus=C2636AE5C3D8E43FFB97AB09028F1AAC6C0BF6CD3D70EBCA281BFFE97FBE30DD
writing RSA key
-----BEGIN PUBLIC KEY-----
MDwwDQYJKoZIhvcNAQEBBQADKwAwKAIhAMJjauXD20Q/+5erCQKPGqxsC/bNPXD
yigb/+l/vjDdAgMBAAE=
-----END PUBLIC KEY-----
https://blog.csdn.net/Catherine_qingzhu
```

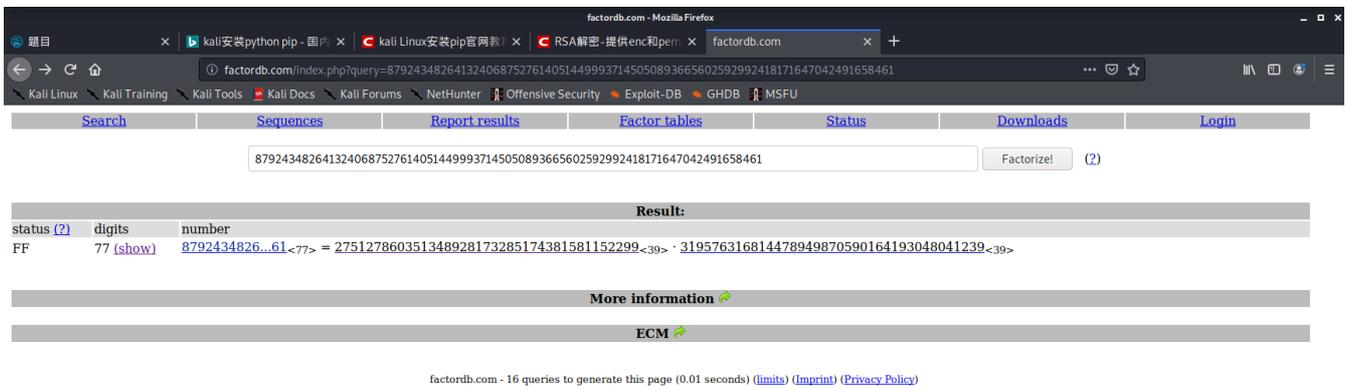
使用openssl查看公钥文件pubkey.pem

我们便得到了公钥对 (n, e)，现在n还是十六进制，还需要转化成十进制再分解。

```
n = C2636AE5C3D8E43FFB97AB09028F1AAC6C0BF6CD3D70EBCA281BFFE97FBE30DD
e = 65537
```

2. 分解整数n

分解n的工具很多，之前推荐的RSA介绍里提供了三种，我是使用factordb网站进行的分解，分解得到的结果如下，



https://blog.csdn.net/Catherine_qingzhu

分解整数n

所以我们得知计算私钥时的两个大素数p, q为,

```
p=275127860351348928173285174381581152299
q=319576316814478949870590164193048041239
```

3. 使用脚本计算私钥

这里我偷懒没有自己写,就使用了开头介绍的那位博客的博主提供的一个python脚本,生成了一个私钥文件private.pem。

```
#coding=utf-8
import math
import sys
from Crypto.PublicKey import RSA
arsa=RSA.generate(1024)
arsa.p=275127860351348928173285174381581152299
arsa.q=319576316814478949870590164193048041239
arsa.e=65537
arsa.n=arsa.p*arsa.q
Fn=long((arsa.p-1)*(arsa.q-1))
i=1
while(True):
    x=(Fn*i)+1
    if(x%arsa.e==0):
        arsa.d=x/arsa.e
        break
    i=i+1
private=open('private.pem','w')
private.write(arsa.exportKey())
private.close()
```

4. 使用私钥解密flag

使用下面的命令就可以用刚生成的私钥文件解密flag.enc了,然后就可以看到flag。

```
OpenSSL> rsautl -decrypt -in flag.enc -inkey private.pem
```