

BUUCTF Reverse RSA

原创

A_dmins 于 2019-07-22 21:29:23 发布 962 收藏

分类专栏: [CTF题 一天一道CTF BUUCTF](#)

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/qq_42967398/article/details/96773826

版权



[CTF题 同时被 3 个专栏收录](#)

115 篇文章 11 订阅

订阅专栏



[一天一道CTF](#)

52 篇文章 5 订阅

订阅专栏



[BUUCTF](#)

24 篇文章 2 订阅

订阅专栏

BUUCTF Reverse RSA

一天一道CTF, 能多不能少

下载文件打开~

发现存在两个文件!!

flag.enc	2019/7/21 22:46	Wireshark captu...	1 KB
pub.key	2019/7/21 22:46	KEY 文件	1 KB

不用说, flag.enc肯定是加密后的, 就不用看了, 直接看pub.key, 内容如下:

```
-----BEGIN PUBLIC KEY-----
MDwwDQYJKoZIhvcNAQEBBQADKwAwKAIhAMAZLFxkrkcYL2wch21CM2kQVFpY9+7+
/AvKr1rzQczdAgMBAAE=
-----END PUBLIC KEY-----
```

RSA的一个公钥??? 貌似是的!

直接利用在线工具分解公钥得到n和e:

密钥类型	RSA
密钥强度	256
PN(e)	65537
PN(n)	86934482296048119190666062003494800588905656017203025617216654058378322103517
DER格式	303c300d06092a864886f70d0101010500032b003028022100c0332c5c64ae47182f6c1c876d42336910545a58f7eefefc0bcaaf5af341ccdd0203010001

ok得到了n, 那就可以分解n得到p和q了~~

利用yafu这个工具进行分解, 得到p, q:

```
选择命令提示符
fac: factoring 86934482296048119190666062003494800588905656017203025617216654058378322103517
fac: using pretesting plan: normal
fac: no tune info: using qs/gnfs crossover of 95 digits
div: primes less than 10000
fmt: 1000000 iterations
rho: x^2 + 3, starting 1000 iterations on C77
rho: x^2 + 2, starting 1000 iterations on C77
rho: x^2 + 1, starting 1000 iterations on C77
pml: starting B1 = 150K, B2 = gmp-ecm default on C77
ecm: 30/30 curves on C77, B1=2K, B2=gmp-ecm default
ecm: 74/74 curves on C77, B1=11K, B2=gmp-ecm default
ecm: 149/149 curves on C77, B1=50K, B2=gmp-ecm default, ETA: 0 sec

starting SIQS on c77: 86934482296048119190666062003494800588905656017203025617216654058378322103517

==== sieving in progress (1 thread): 36224 relations needed ====
==== Press ctrl-c to abort and save state ====
36260 rels found: 19012 full + 17248 from 185427 partial, (1841.44 rels/sec)

SIQS elapsed time = 112.6568 seconds.
Total factoring time = 132.7595 seconds

***factors found***
p39 = 285960468890451637935629440372639283459
p39 = 304008741604601924494328155975272418463

ans = 1
```

得到pq就好办了~

直接计算d, 然后利用rsa模块生成私钥~

然后就可以直接对文件进行解密了~

```
import hashlib
import gmpy2
import rsa

p = 285960468890451637935629440372639283459
q = 304008741604601924494328155975272418463
e = 65537

n = 86934482296048119190666062003494800588905656017203025617216654058378322103517

d = gmpy2.invert(e,(q-1)*(p-1))
print(d)

d = 81176168860169991027846870170527607562179635470395365333547868786951080991441

key = rsa.PrivateKey(n,e,d,p,q)
print(key)

with open("output\\flag.enc","rb") as f:
    print(rsa.decrypt(f.read(),key).decode())
```

得到:

```
81176168860169991027846870170527607562179635470395365333547868786951080991441
PrivateKey(86934482296048119190666062003494800588905656017203025617216654058378322103517, 65537, 81176168860169991027846
870170527607562179635470395365333547868786951080991441, 285960468890451637935629440372639283459, 30400874160460192449432
8155975272418463)
flag{decrypt_256}
```

flag: `flag{decrypt_256}`

总结:

更深入的了解了RSA算法的奥秘~

同时也学会了yafu工具的使用, 私钥生成函数的使用

利用rsa模块和gmpy2模块对RSA进行解密~

过程虽有坎坷, 但还是能够跨越过去的!!!