实验三公钥密码算法 RSA

1611531-信息安全-刘新慧

摘要

通过实际编程了解公钥密码算法 RSA 的加密和解密过程，加深对公钥密码算法的了解和使用。

目录

1 实验原理

序列密码和分组密码算法都要求通信双方通过交换密钥实现使用同一个密钥，这在密钥的管理、发布和安全性方面存在很多问题，而公钥密码算法解决了这个问题。

公钥密码算法是指一个加密系统的加密密钥和解密密钥是不同的，或者说不能用其中一个推导出另一个。在公钥密码算法的两个密钥中，一个是用于加密的密钥，它是可以公开的，称为公钥；另一个是用于解密的密钥，是保密的，称为私钥。公钥密码算法解决了对称密码体制中密钥管理的难题，并提供了对信息发送人的身份进行验证的手段，是现代密码学最重要的发明。

RSA 密码体制是目前为止最成功的公钥密码算法，它是在 1977 年由 Rivest、Shamir 和 Adleman 提出的第一个比较完善的公钥密码算法。它的安全性是建立在“大数分解和素性检测”这个数论难题的基础上，即将两个大素数相乘在计算上容易实现，而将该乘积分解为两个大素数因子的计算量相当大。虽然它的安全性还未能得到理论证明，但经过 20 多年的密码分析和攻击，迄今仍然被实践证明是安全的。

RSA 算法描述如下：

1、公钥

选择两个不同的大素数 p 和 q，n 是二者的乘积，即 n = pq，使

$$\varphi(n) = (p-1)(q-1)$$

$\varphi(n)$ 为欧拉函数。随机选取正整数 e，使其满足 gcd(e, $\varphi(n)$)=1 ，即 e 和 $\varphi(n)$ 互素，则将（n，e）作为公钥。

2、私钥

求出正数 d，使其满足 $e \times d = 1 mod \varphi(n)$，则将（n，d）作为私钥。

3、加密算法

对于明文 m，由 $c \equiv m^e modn$，得到密文 c。

4、解密算法

对于密文 c，由 $m \equiv c^d modn$，得到明文 m。

如果攻击者获得了 n、e 和密文 c，为了破解密文必须计算出私钥 d，为此需要先分解 n。当 n 的长度为 512 比特时，在目前还是安全的，但从因式分解技术的发展来看，512 比特并不能保证长期的安全性。为了达到更高的安全性，要求在一般的商业应用中使用 1024 比特的长度，在更高级别的使用场合，要求使用 2048 比特长度。

2 实验内容和步骤

1、为了加深对 RSA 算法的了解，根据已知参数：p = 3，q = 11，m = 2，手工计算公钥和私钥，并对明文 m 进行加密，然后对密文进行解密。

2、编写一个程序，用于生成 512 比特的素数。

3．利用 2 中程序生成的素数，构建一个 n 的长度为 1024 比特的 RSA 算法，利用该算法实现对明文的加密和解密。

4、在附件中还给出了一个可以进行 RSA 加密和解密的对话框程序 RSATool，运行这个程序加密一段文字，了解 RSA 算法原理。

3 实验结果

3.1 生成大素数

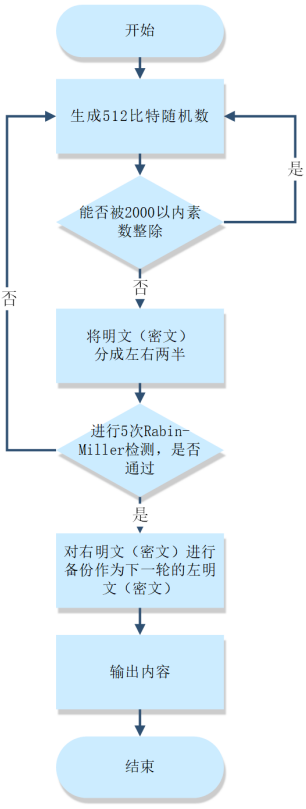
文件夹中有两个工程，一个是用来生成大素数的，一个是进行 RSA 加密解密的。

生成素数的程序运行后结果如下图，素数会存在 result.txt 中：

生成素数的原理如下：

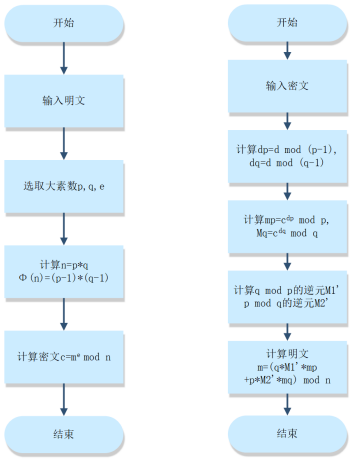
首先生成 512 比特的随机数，然后对小于 2000 的素数取模，若能整除，则重新生成随机数。如果不能整除，则继续对其进行 5 次 Rabin-Miller 检测，只要有一次不符合，就可以断定其不是素数，需重新生成素数并重新素性检测，若能够符合，则其有很大可能为素数 (概论为 96.8%)，将其输出.

程序框图如下：



3.2 RSA 加密解密

程序框图如下：

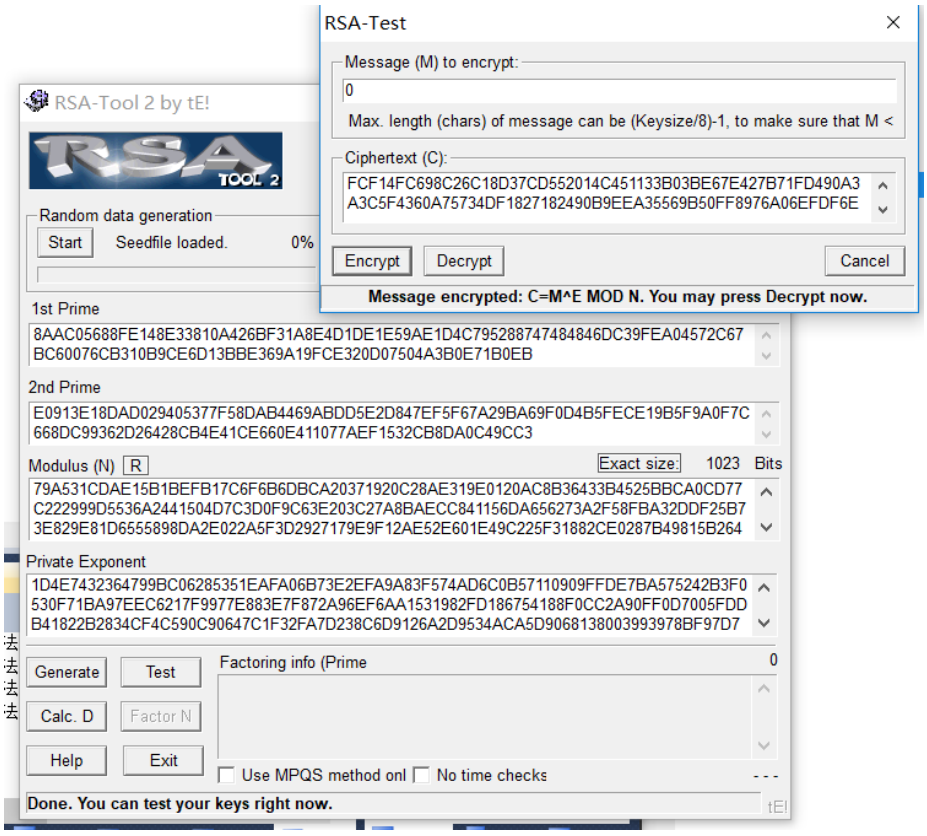


加密　　解密

代码运行后，进行加密解密结果如下：

```
n=  79A531CDAE15B1BEFB17C6F6B6DBCA20371920C28AE319E0120AC8B36433B4525BBCA0CD77C222999D5536A2441504D7C3D0F9C63E203C27A8BA
ECC841156DA656273A2F58FBA32DDF25B73E829E81D6555898DA2E022A5F3D2927179E9F12AE52E601E49C225F31882CE0287B49815B26422B8FFCCA
DE572E716FD69C54F701
运行中!

e=  10001
运行中!

d=  1D4E7432364799BC06285351EAFA06B73E2EFA9A83F574AD6C0B57110909FFDE7BA575242B3F0530F71BA97EEC6217F9977E883E7F872A96EF6A
A1531982FD186754188F0CC2A90FF0D7005FDDB41822B2834CF4C590C90647C1F32FA7D238C6D9126A2D9534ACA5D9068138003993978BF97D7F8A8F
36E5D6EB129D1A4C71C5
运行中!

m=  30
运行中!
密文  FCF14FC698C26C18D37CD552014C451133B03BE67E427B71FD490A3A3C5F4360A75734DF1827182490B9EEA35569B50FF8976A06EFDF6EA338
55104461396897B4941F5D338461BD3CD6FC9800668EBBBC16A081287FDD3ADC6463480E503F64D4F7950757267A931AFAE0FE8EF4FD7F8B8F00D184
B7AF41B08CDDD17CA10E9

运行中!

解密结果为:30
请按任意键继续. . .
```

使用 RSA-tool 进行检验，验证结果是正确的：


RSA-Test
Message (M) to encrypt:
0
Max. length (chars) of message can be (Keysize/8)-1, to make sure that M <
Ciphertext (C):
FCF14FC698C26C18D37CD552014C451133B03BE67E427B71FD490A3
A3C5F4360A75734DF1827182490B9EEA35569B50FF8976A06EFDF6E
Encrypt
Decrypt
Cancel
Message encrypted: C=M^E MOD N. You may press Decrypt now.
RSA-Tool 2 by tE!
Random data generation
Start
Seedfile loaded.
0%
1st Prime
8AAC05688FE148E33810A426BF31A8E4D1DE1E59AE1D4C795288747484846DC39FEA04572C67
BC60076CB310B9CE6D13BBE369A19FCE320D07504A3B0E71B0EB
2nd Prime
E0913E18DAD029405377F58DAB4469ABDD5E2D847EF5F67A29BA69F0D4B5FECE19B5F9A0F7C
668DC99362D26428CB4E41CE660E411077AEF1532CB8DA0C49CC3
Modulus (N)
R
Exact size:
1023
Bits
79A531CDAE15B1BEFB17C6F6B6DBCA20371920C28AE319E0120AC8B36433B4525BBCA0CD77
C222999D5536A2441504D7C3D0F9C63E203C27A8BAECC841156DA656273A2F58FBA32DDF25B7
3E829E81D6555898DA2E022A5F3D2927179E9F12AE52E601E49C225F31882CE0287B49815B264
Private Exponent
1D4E7432364799BC06285351EAFA06B73E2EFA9A83F574AD6C0B57110909FFDE7BA575242B3F0
530F71BA97EEC6217F9977E883E7F872A96EF6AA1531982FD186754188F0CC2A90FF0D7005FDD
B41822B2834CF4C590C90647C1F32FA7D238C6D9126A2D9534ACA5D9068138003993978BF97D7
Generate
Test
Factoring info (Prime
0
Calc. D
Factor N
Help
Exit
Use MPQS method onl
No time checks
Done. You can test your keys right now.