RSA 仿真实现

16337341 朱志儒

实验要求

实现 RSA 算法，将学号作为私钥，并生成对应的公钥，n 为任意选取的两个 512bit 的素数相乘获得，加密：SUN YAT-SEN UNIVERSITY（包含空格），给出密文，并给出解密函数，进行验证。

实验原理

RSA 密钥体制：

1) 设$n = pq$，其中 p 和 q 为素数，且$p \neq q$；
2) 根据欧拉函数，求得$r = \varphi(n) = \varphi(p)\varphi(q) = (p-1)(q-1)$；
3) 选择一个小于 r 的整数 e, 使得 e 与 r 互素, 并求得 e 的模 r 逆元 d, 即$ed \equiv 1(\mathrm{mod}\ \ r)$；

（n，e）组成公钥，（n，d）组成私钥。

RSA 参数生成算法：

1) 生成两个大素数 p 和 q；
2) $n \leftarrow pq$，且$\varphi(n) \leftarrow (p-1)(q-1)$；
3) 选择一个随机数$b\big(1 < b < \varphi(n)\big)$，使得$\gcd\big(b, \varphi(n)\big) = 1$；
4) $a \leftarrow b^{-1} mod\ \ \varphi(n)$；
5) 公钥为（n，b），私钥为（n，a）。

平方-乘算法:

```
Function exp_by_squaring(x, n)
  if n < 0  then return exp_by_squaring(1 / x, -n);
  else if n = 0  then return  1;
  else if n = 1  then return  x ;
  else if n is even  then return exp_by_squaring(x * x,  n / 2);
  else if n is odd  then return x * exp_by_squaring(x * x, (n - 1) / 2);
```

首先生成两个 512bit 的数，并使用素性检验来测试其是否为素数，如果不是则重新生成。这样就得到两个素数 p 和 q，然后计算$n = pq$，$\varphi(n) = (p-1)(q-1)$。以学号为私钥，即 d = 16337341，计算 d 模$\varphi(n)$的逆，即$e = d^{-1} mod\ \varphi(n)$。得到这些参数后，可使用 e 和 n 加密信息，使用 d 和 n 解密信息，加密和解密都使用平方-乘算法实现。

实验内容

超大整数超大次幂然后对超大的整数取模，即(base ^ exponent) mod m：

```python
def exp_mode(base, exponent, m):
    result = 1
    while exponent != 0:
        if (exponent & 1) == 1:
            # ei = 1, then mul
            result = (result * base) % m
        exponent >>= 1
        base = (base * base) % m
    return result
```

素性检验：

```python
def primeTest(n):
    q = n - 1
    k = 0
    # Find k, q, satisfied 2^k * q = n - 1
    while q % 2 == 0:
```

```
        k += 1
        q //= 2
    a = random.randint(2, n - 2)
    # If a^q mod n= 1, n maybe is a prime number
    if exp_mode(a, q, n) == 1:
        return "inconclusive"
    # If there exists j satisfy a ^ ((2 ^ j) * q) mod n == n-1, n maybe is a prime number
    for j in range(0, k):
        if exp_mode(a, (2 ** j) * q, n) == n - 1:
            return "inconclusive"
    # a is not a prime number
    return "composite"
```

寻找素数:

```
def findPrime(halfkeyLength):
    while True:
        # Select a random number n
        n = random.randint(0, 1 << halfkeyLength)
        if n % 2 != 0:
            found = True
         # If n satisfy primeTest 10 times, then n should be a prime number
            for i in range(0, 10):
                if primeTest(n) == "composite":
                    found = False
                    break
            if found:
                return n
```

求两个数字的最大公约数:

```
def gcd(a, b):
    while a != 0:
        a, b = b % a, a
    return b
```

使用扩展欧几里得算法求模逆:

```
def findModReverse(a, m):
    if gcd(a, m) != 1:
        return None
```

```
    u1, u2, u3 = 1, 0, a
    v1, v2, v3 = 0, 1, m
    while v3 != 0:
        q = u3 // v3
        v1, v2, v3, u1, u2, u3 = (u1 - q * v1), (u2 - q * v2), (u3 - q * v3), v1, v2, v3
    return u1 % m
```

生成公钥私钥:

```
# 生成公钥私钥，p、q 为两个超大质数
def gen_key(p, q):
    n = p * q
    fy = (p - 1) * (q - 1)  # 计算与 n 互质的整数个数 欧拉函数
    d = 16337341  # 选取私钥 d 注意选取与 fy 互质的数
    while gcd(d, fy) != 1:
        d += 1
    # generate d
    b = fy
    e = findModReverse(d, fy)
    # 返回：  公钥     私钥
    return (n, e), (n, d)
```

加密:

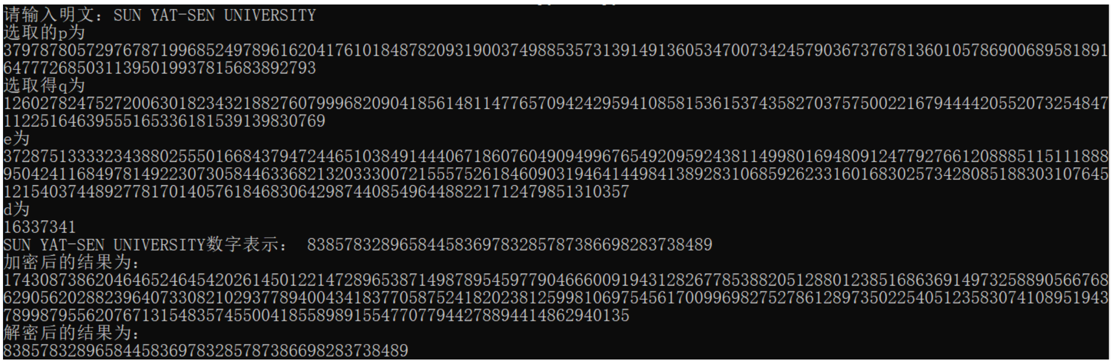
```
# 加密 m 是待加密的信息 加密成为 c
def encrypt(m, pubkey):
    n = pubkey[0]
    e = pubkey[1]
    c = exp_mode(m, e, n)
    return c
```

解密:

```
# 解密 c 是密文，解密为明文 m
def decrypt(c, selfkey):
    n = selfkey[0]
    d = selfkey[1]
    m = exp_mode(c, d, n)
    return m
```

实验结果

程序运行结果：



由图可知，SUN YAT-SEN UNIVERSITY 数字表示为

83857832896584458369783285787386698283738489

素数 p 为

3797878057297678719968524978961620417610184878209319003749885357313914913605347007342457903673767813601057869006895818916477726850311395019937815683892793

素数 q 为

12602782475272006301823432188276079996820904185614811477657094242959410858153615374358270375750022167944442055207325484711225164639555165336181539139830769

e 为

3728751333323438802555016684379472446510384914440671860760490949967654920959243811499801694809124779276612088851151111888950424116849781492230730584463368213203330072155575261846090319464144984138928310685926233160168302573428085188303107645121540374489277817014057618468306429874408549644882217124798513 10357

d 为

16337341

密文：

1743087386204646524645420261450122147289653871498789545977904666009194312826778538820512880123851686369149732588905667686290562028823964073308210293778940043418377058752418202381259981069754561700996982752786128973502254051235830741089519437899879556207671315483574550041855898915547707794427889441486294 0135

密文解密后:

838578328965844583697832857873866982837384889

由此可以看出，密文解密后与明文相同，说明 RSA 仿真实验成功。