实验四 密码算法

实验目的

了解现代密码学的基本原理和数论的基础知识，掌握非对称密码体制的著名代表 RSA 加密算法的工作原理和流程，并设计实现一个简单的密钥系统。

实验内容

了解加/解密的基本原理和工作过程，用公开密钥对明文进行加密，并用私人密钥对密文进行解密，构造一个简单的 RSA 公开密钥系统。

实验步骤

1、RSA 算法是由麻省理工学院的 Ron Rivest,Adi Shamir 和 Len Adleman 于 1977 年研制并于 1978 年首次发表的一种算法，是第一个能同时用于加密和数字签名的算法，且易于理解和操作，因此作为一种通用公开密钥加密方式而受到推崇。

RSA 是一种分组密码，其中明文和密文都是小于某个 n 的从 0 到 n-1 的整数，则分组的二进制值长度必须小于或等于 $\log_2 n$。若以 M 表示明文分组，而 C 表示密文分组，则加密和解密的过程如下：

$C=M^e \bmod n$

$M=C^d \bmod n=(M^e)^d \bmod n=M^{ed} \bmod n$

发送方和接受方都必须知道 n 的值。发送方知道 e 的值，而只有接受方知道 d 的值。因此这是一种**公开密钥为{e,n}**，且**私有密钥为{d,n}**的公开密钥加密算法。

此时算法要能够满足公开密钥加密的要求，则**必须满足以下条件：**

（1）有可能找到 e、d、n 的值，使得**对所有 M<n 有 M^{ed}=M mod n**。

（2）对于所有 **M<n 的值**，要计算 M^e 和 C^d 相对来说是简单的。

（3）在给定 e 和 n 时，判断出 d 是不可行的。

2、重点考虑第一个条件：

由 Euler 定理的一个推论：**给定两个素数 p 和 q 以及两个整数 n 和 m，使得 n=pq 而且 0<m<n，并且对于任意整数 k，下列关系成立：**

$\mathbf{m^{k\Phi(n)+1}=m^{k(p-1)(q-1)+1}\equiv m \bmod n}$

其中Φ(n)是欧拉函数，也就是不超过 n 且与 n 互素的整数个数。对于素数 p 和 q，有Φ(pq)=(p-1)(q-1)。

因此**得到需要的关系**：ed=kΦ(n)+1，

等价于：ed≡1 mod Φ(n)

$\mathbf{d\equiv e^{-1} \bmod \Phi(n)}$

也就是说：**d 和 e 是以Φ(n)为模的乘法逆元。**

根据模运算规则，它的**必要条件**是：d、e 和Φ(n)是互素的，即 gcd(Φ(n),d)=1。

3、因此给出 RSA 方案：

- p，q 两个素数 （私有，选择）
- n=pq （公开，计算出）
- e，其中 gcd(Φ(n),e)=1;1<e<Φ(n) （公开，选择）
- $d \equiv e^{-1} \bmod \Phi(n)$ （私有，计算出）

私有密钥由{d,n}组成，公开密钥由{e,n}组成。

假设用户 A 公布了它的公开密钥，而用户 B 希望向 A 发送一个报文 M，那么 B 计算出 $C=M^e \bmod n$ 并传输 C。在收到这个密文时，用户 A 通过计算 $M=C^d \bmod n$ 进行解密。

4、编程一个简单的 RSA 加、解密系统。

（1）假设用户 A 选择两个素数 p 和 q，计算得到 n=pq 和 Φ(n)=(p-1)(q-1)。选择一个加密密钥 e，它小于Φ(n)且与Φ(n)互素。计算解密密钥 $d \equiv e^{-1} \bmod \Phi(n)$。则用户 A 公布公开密钥{e,n}，自己拥有私有密钥{d,n}。

（2）用户 B 使用用户 A 的公开密钥 e 和 n 对报文 M 进行加密，得到 $C=M^e \bmod n$，并发送给用户 A。

（3）用户 A 收到加密的报文后，使用自己的私有密钥 d 和 n 对加密报文 C 进行解密，恢复得到明文 $M=C^d \bmod n$。

关键部分代码（由公开密钥 e 和 n，求私有密钥 d）：

```
int ext_euclid(int a,int b,int f,int e)
//因      e*0+φ*1=φ   (1)
//        e*1+φ*1=e   (2)
//则：e*(0-1*(φ/e))+.....=φ%e
//由于e和φ互质，因此一定有某一次运算后，等式右侧的φ%e==1。
//此时左侧等式中e所乘的系数就是所要求的d，即e^-1
//将(1)式e所乘的系数用a表示，(2)式中e所乘的系数用b表示，并且令：m=φ/e;   n=φ%e
{
    int m,n,t;
    if (e==1) return b;
    m=f/e;   n=f%e;
    t=a-b*m;
    ext_euclid(b,t,e,n);
 }
```

在 main()函数中实现 RSA 加/解密过程的主要流程：

```
void main()
{
    //输入质数p和q
    int p,q;
    cout<<"输入一个质数p(如101):";
    cin>>p;
    cout<<"输入一个质数q(如113):";
    cin>>q;
```

```
	//求得n=p*q的值
	int n=p*q;
	cout<<"分组加密时，每个分组的大小不能超过n=p*q=";
	cout<<n<<endl;

	//求得φ(n)=(p-1)*(q-1)的值
	int f=(p-1)*(q-1);
	cout<<"模φ(n)=(p-1)*(q-1)=";
	cout<<f<<endl<<endl;

	//选取与φ(n)互质的公钥e
	int e;
	cout<<"输入与φ(n)互质的公钥e(如3533):";
	cin>>e;

	//由e和φ(n)生成私钥d
	int d=ext_euclid(0,1,f,e);
	while (d<=0) d+=f;
	cout<<"通过调用扩展欧几里德算法，求得密钥d为："<<d<<endl;

	//利用生成的公钥{e,n}对明文M进行加密
	int M,C;
	cout<<"现在公钥{e,n}、私钥{d,n}均已生成完毕。\n\n请输入需要传输的明文内容进
行加密(如9726)：";
	cin>>M;
	C=1;
	for(int i=1;i<=e;i++)
	{
		C=C*M%n;
	}
	cout<<"明文M="<<M<<"经加密后得到密文C=M^e(mod n)："<<C<<endl;

	//利用生成的私钥私钥{e,n}对密文C进行解密
	M=1;
	for(i=1;i<=d;i++)
	{
		M=M*C%n;
	}
	cout<<"密文C="<<C<<"经解密后得到明文M=C^d(mod n)："<<M<<endl;
}
```

5、备注：进一步还可增加以下功能，有兴趣的同学可以自行完成。

在现有程序的基础上可以增加随机生成质数的功能，通常采用生成一个随机数并判断其

是否是质数的办法。判断一个数是否是质数，通常采用快速测试和费马测试结合的算法，经过这两个算法判断为素数的数，是素数的可能性非常的大(但也不能百分之百保证是素数)。

判断两个数是否互质，则可以通过求它们的最大公约数来判断。

另外，由于实际应用中 RSA 算法通常需要选择大质数进行运算，因此实际系统中还应考虑大数运算的功能，对超出整型变量表示范围的数进行运算。

思考

1、RSA 加密的安全性是如何保证的？

RSA 的安全性依赖于大整数因式分解的难度。虽然迄今为止尚无法证明，但一般“相信”RSA 系统的安全性等价于大整数因子分解的难度。

2、RSA 的速度

由于进行的都是大数计算，使得 RSA 最快的情况也比 DES 慢上 100 倍，无论是软件还是硬件实现。速度一直是 RSA 的缺陷。一般来说只用于少量数据加密。

3、RSA 的缺点主要有：

（1）产生密钥很麻烦，受到素数产生技术的限制，因而难以做到一次一密。

（2）分组长度太大，为保证安全性，n 至少也要 600 bits 以上，使运算代价很高，尤其是速度较慢，较对称密码算法慢几个数量级。且随着大数分解技术的发展，这个长度还在增加，不利于数据格式的标准化。

4、针对 RSA 算法有哪些攻击方法？

（1）强行攻击：对所有的私有密钥都进行尝试。

（2）数学攻击：等效于对两个素数乘积的因子分解，主要包括三种：

- 将 n 分解为两个素数因子 p 和 q。这样就可以计算 $\Phi(n)=(p-1)\times(q-1)$，然后可以确定 $d=e^{-1} \bmod \Phi(n)$。
- 在不先确定 p 和 q 的情况下直接确定 $\Phi(n)$。这样同样可以确定 $d=e^{-1} \bmod \Phi(n)$。
- 不先确定 $\Phi(n)$ 而直接确定 d。

（3）定时攻击：依赖于解密算法的运行时间。

5、RSA 算法如何用于数字签名？

由于 e,d 可以互换。在用于数字签名时,发送方只须用己方的解密密钥 d 先 "加密"即可,因为只有发送方自己知道自己的 d, 收方只有用对应的 e "解密" 才能知道明文, 同时也验证了发方的身份。