

实验报告

57119101 王晨阳

目录

实验原理
编程实现
 扩展欧几里得
 快速乘
 快速幂
 验证互质
 RSA加密
 RSA解密
 代码
实验结果

实验原理

- 公钥(加密): (e, n)
- 私钥(解密): (d, n)

$n = p \cdot q$, p 和 q 为两个大素数

$(e, \varphi(n)) = 1$

$e \cdot d \equiv 1 \pmod{\varphi(n)}$

- 加密
 $E(P) = C \equiv P^e \pmod{n}$
- 准备好 p, q , 计算 $n = p \cdot q, \varphi(n)$
 - 假设一个与 $\varphi(n)$ 互质的 e , 求出 d
 - 使用公钥加密信息 m : $m^e \equiv c \pmod{n}$
- 解密
 $D(C) = C^d \equiv (P^e)^d \equiv P^{e \cdot d}$
 $\equiv P^{k \cdot \varphi(n) + 1} \equiv (P^{\varphi(n)})^k P \equiv P \pmod{n}$
 - 求解 $c^d \pmod{n}$

编程实现

扩展欧几里得

求解贝祖等式

```
1 void ex_gcd(long long a, long long b, long long &x, long long &y)
2 {
3     if (b==0)
4     {
5         x=1;
6         y=0;
7         return;
8     }
9     ex_gcd(b, a%b, x, y);
```

```

10     long long t=x;
11     x=y;
12     y=t-a/b*y;
13     return;
14 }

```

快速乘

防止乘法溢出long long

```

1 long long qMultiply(long long x,long long y,long long c)
2 {
3     return (x*y-(long long)((long double)x/c*y)*c+c)%c;
4 }

```

快速幂

加快幂运算

为什么不用中国剩余定理：因为我能想到的算法复杂度更高

```

1 long long qPower(long long base,long long power,long long c)
2 {
3     long long ans=1,res=base;
4     while(power)
5     {
6         if(power&1)//power为奇数
7             ans=qMultiply(ans,res,c)%c;
8         res=qMultiply(res,res,c)%c;
9         power>>=1;//power/2
10    }
11    return ans;
12 }

```

验证互质

```

1 bool is_coprime(long long a,long long b)
2 {
3     if (a==1 || b==1)    // 两个正整数中，只有其中一个数值为1，两个正整数为互
    质数
4         return true;
5     while (true)
6     {    // 求出两个正整数的最大公约数
7         long long t=a%b;
8         if (t==0) break;
9         else
10        {
11            a=b;
12            b=t;
13        }
14    }
15    return (b==1);
16 }

```

RSA加密

```
1 long long RSA_encode(long long m,long long &n,long long &d,long long
  &e,long long &phi_n)
2 {
3     long long temp;
4     n=p*q;
5     phi_n=n-p-q+1;
6     srand(time(0));
7     e=rand()%phi_n+1;
8     while (!is_coprime(e,phi_n)) e=rand()%phi_n+1;
9     ex_gcd(e,phi_n,d,temp);
10    while (d<=0)
11        d+=phi_n;
12    return (qPower(m,e,n));
13 }
```

RSA解密

```
1 long long RSA_decode(long long n,long long d,long long c)
2 {
3     return (qPower(c,d,n));
4 }
```

代码

```
1  /*****
2  *****/
3  author:王晨阳
4  description:RSA加密解密
5  *****/
6
7  #include<bits/stdc++.h>
8  using namespace std;
9
10 const long long p=10191161;
11 const long long q=10191133;
12
13 //扩展欧几里得
14 void ex_gcd(long long a,long long b,long long &x,long long &y)
15 {
16     if (b==0)
17     {
18         x=1;
19         y=0;
20         return;
21     }
22     ex_gcd(b,a%b,x,y);
23     long long t=x;
24     x=y;
25     y=t-a/b*y;
26     return;
27 }
```

```

28 //快速乘
29 long long qMultiply(long long x,long long y,long long c)
30 {
31     return (x*y-(long long)((long double)x/c*y)*c+c)%c;
32 }
33
34 //快速幂
35 long long qPower(long long base,long long power,long long c)
36 {
37     long long ans=1,res=base;
38     while(power)
39     {
40         if(power&1)//power为奇数
41             ans=qMultiply(ans,res,c)%c;
42         res=qMultiply(res,res,c)%c;
43         power>>=1;//power/2
44     }
45     return ans;
46 }
47
48 //验证互质
49 bool is_coprime(long long a,long long b)
50 {
51     if (a==1 || b==1) // 两个正整数中，只有其中一个数值为1，两个正整数为互
    质数
52         return true;
53     while (true)
54     { // 求出两个正整数的最大公约数
55         long long t=a%b;
56         if (t==0) break;
57         else
58         {
59             a=b;
60             b=t;
61         }
62     }
63     return (b==1);
64 }
65
66 //RSA加密
67 long long RSA_encode(long long m,long long &n,long long &d,long long
    &e,long long &phi_n)
68 {
69     long long temp;
70     n=p*q;
71     phi_n=n-p-q+1;
72     srand(time(0));
73     e=rand()%phi_n+1;
74     while (!is_coprime(e,phi_n)) e=rand()%phi_n+1;
75     ex_gcd(e,phi_n,d,temp);
76     while (d<=0)
77         d+=phi_n;
78     return (qPower(m,e,n));
79 }
80

```

```

81 //RSA解密
82 long long RSA_decode(long long n,long long d,long long c)
83 {
84     return (qPower(c,d,n));
85 }
86
87 //主程序
88 int main()
89 {
90     long long m,n,d,e,phi_n,c,Message;
91     printf("明文为 ");
92     scanf("%lld",&m);
93     c=RSA_encode(m,n,d,e,phi_n);
94     printf("公钥为 (%lld,%lld)\n私钥为 (%lld,%lld)\n",e,n,d,n);
95     printf("密文为 %lld\n",c);
96     Message=RSA_decode(n,d,c);
97     printf("明文为 %lld\n",Message);
98 }

```

实验结果

由上次作业得到两个素数分别为

```

1  p=10191161
2  q=10191133

```

程序运行结果为

```

1  明文为 57119101
2  公钥为 (6619,103859477175413)
3  私钥为 (96296341794739,103859477175413)
4  密文为 84116618419953
5  明文为 57119101

```

注：由于e为随机产生，每次运行的结果不完全相同！