****

FUZHOU TECHNOLOGY AND BUSINESS UNIVERSITY



**实验报告**

**课程名称： 网络信息安全技术**

**系 别： 工学院 年 级： 2018级**

**专 业： 软件工程**

**班 级： 1班 学 号： 1850304048**

**姓　　名： 汤郭**

**成 绩：**

**任课教师： 林晖**

**2021年 5 月 31 日**

目录

[一、Pailiar密码算法 1](#_Toc73366233)

[**1.实验目的** 1](#_Toc73366234)

[**2.实验原理** 1](#_Toc73366235)

[**3.实验环境** 3](#_Toc73366236)

[**4.实验步骤** 4](#_Toc73366237)

[㈠查看Pai11ier算法运行结果并学习核心算法 4](#_Toc73366238)

[二、SM2密码算法 6](#_Toc73366239)

[**1.实验目的** 6](#_Toc73366240)

[**2.实验原理** 6](#_Toc73366241)

[**3.实验环境** 9](#_Toc73366242)

[**4.实验步骤** 9](#_Toc73366243)

[㈠查看SM2算法运行结果并学习核心算法 9](#_Toc73366244)

[三、MD5算法 12](#_Toc73366245)

[**1.实验目的** 12](#_Toc73366246)

[**2.实验原理** 12](#_Toc73366247)

[**3.实验环境** 15](#_Toc73366248)

[**4.实验步骤** 15](#_Toc73366249)

[㈠查看MD5算法运行结果并学习核心算法 16](#_Toc73366250)

**实验三 密码学算法**

# 一、Pailiar密码算法

**1.实验目的**

1） 学习Pailiar密码算法原理

2） 学习Pailiar密码算法编程实现

**2.实验原理**

**1) 算法原理**

a) Pascal Paillier在1999年欧密会上提出了概率加密体制——Paillier，Paillier加密方案与RSA、Goldwasser-Micali、Rabin加密方案一样，是基于合数分解的困难性，Paillier加密方案比Goldwasser-Micali密码系统更加高效，拥有很好的同态特性，而且其依赖的是一个更新的且更少研究的困难性问题—*n*次复合剩余问题。

b) 设说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152336_15090.gif，如果*g*的阶是*n*的倍数，则函数

说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152336_11589.gif

说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152336_72922.gif

  是一一映射。

c) *n*次复合剩余问题：对于上式定义的函数说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152336_55877.gif和某个说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152336_23535.gif，满足说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152337_56063.gif的唯一的*x*，称为说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152337_68595.gif中*w*的*n*次复合剩余，由*w*求*x*的问题，称为*n*次复合剩余问题。

d) 给定说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152337_54924.gif，知道*n*的分解，可以有效地求得*w*的*n*次剩余。事实上，令：

说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152337_39104.gif

  以及

说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152337_10545.gif

  不难求得：

说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152337_33345.gif

e) 密钥产生

  随机的选取两个素数*p*和*q*，且满足说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152337_67908.gif。

  计算说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152337_47374.gif和说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152337_80421.gif。

  选取随机数*g* 说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152337_95745.gif，且能保证说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152337_48958.gif存在，其中函数*L*定义为说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152357_16279.gif。此时，公钥为说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152357_44507.gif，私钥为说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152357_44234.gif。

f) 加密解密过程

  Paillier加解密: 对于明文说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152357_90366.gif，并选择随机数说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152357_47015.gif，则

  加密过程为：说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152357_31729.gif。

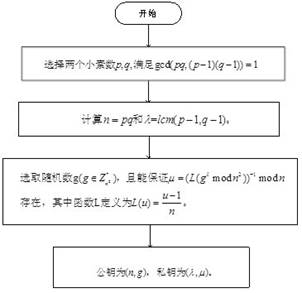
  解密过程为：说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152357_95011.gif。

**2) 算法参数**

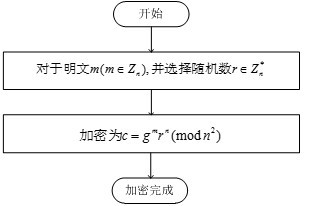
说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152336_13169.gif是Paillier的模数说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152336_82772.gif，说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152336_67990.gif分别为*n*的Euler函数和Carmichael函数，不难证明以下结果：说明: http://192.168.1.205/Public/editor/attached/image/20130724/20130724152336_61344.gif

**3) 算法流程**

a) 密钥生成流程。如图一所示

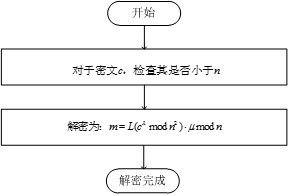
  
图一  Paillier密钥生成流程图

b) 加密流程。如图二所示



图二  Paillier加密部分流程图

c) 解密流程。如图三所示



图三  Paillier解密部分流程图

**3.实验环境**

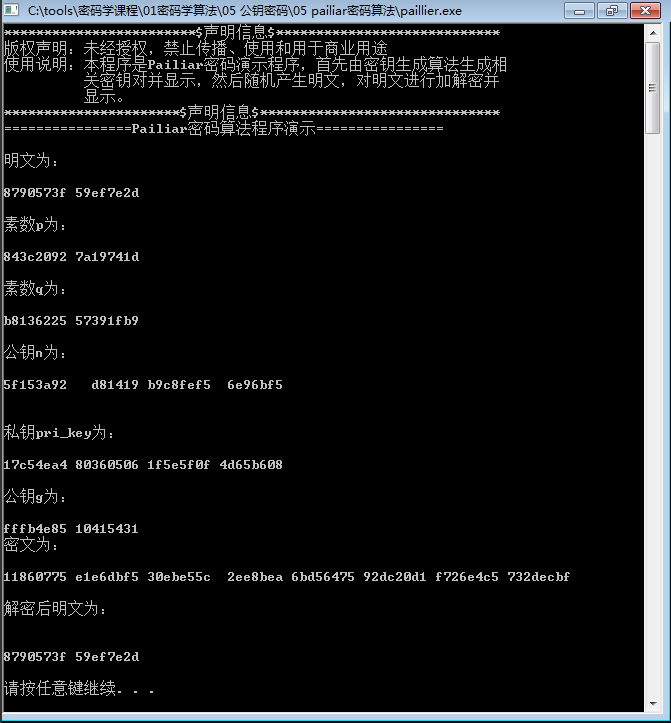
Windows 7

工具：C:\tools\密码学课程\01密码学算法\05 公钥密码\05 pailiar密码算法。

**4.实验步骤**

㈠查看Pai11ier算法运行结果并学习核心算法

1.1运行【Pailiar.exe】，程序运行结果。



1.2代码如下。

1) Pai11ier系统初始化部分代码

Paillier::Paillier\_Generated\_Parameter() {

p.GeneratePrime(); //产生素数

q.GeneratePrime();

while(p==q) //判断两个素数不等

q.GeneratePrime();

temp=p-I;

t=q-I;

t=t \*temp;

cout<<"素数p为："<<endl; //输出素数

p.print();

cout<<endl;

cout<<"素数q为："<<endl;

q.print();

cout<<endl;

cout<<"公钥n为"<<endl;

n=p \*q;

n.print();

cout<<endl;

cout<<endl;

pri \_key=ISD \_Lcm(p-I,q-I);

cout<<"私钥pri \_key为"<<endl;

pri \_key.print();

cout<<endl;

g.GeneratePrime();

cout<<"公钥g为"<<endl;

g.print();

}

//(2) paillier加解密部分代码：

Paillier::Paillier\_Decrypt\_Crypt (char s) {

r.GeneratePrime();

middle1=g.power \_mod(middle,n \*n) \*r.power \_mod(n,n \*n);

b=middle1.power \_mod(I,n \*n);

//b=middle.power \_mod(e,n); //产生密文b

cout<<"生成密文如下"<<endl;

b.print();

cout<<endl;

u.ex \_euclid(ISD\_Lfunction(g.power \_mod(pri \_key,n \*n)),n);

c=ISD \_Lfunction(b.power \_mod(pri \_key,n \*n)) \*u;

c=c.power \_mod(I,n);

//c=b.power\_mod(d,n); //解密

cout<<"解密后的结果为"<<endl<<endl;

c.print();

}

# 二、SM2密码算法

**1.实验目的**

1） 学习SM2密码算法的原理

2） 学习SM2密码算法的编程实现

**2.实验原理**

**1) 算法原理**

a) 1985年，N.Koblitz和V.Miller各自独立地提出了将椭圆曲线应用于公钥密码系统。2012年12月，国家密码管理局公布了SM2椭圆曲线公钥密码算法。SM2是我国自主知识产权的商用密码算法，是ECC（Elliptic Curve Cryptosystem）算法的一种，其加密强度为256位。SM2椭圆曲线公钥密码算法基于椭圆曲线离散对数问题，计算复杂度是指数级的，求解难度较大。在相同安全程度要求下，椭圆曲线密码较其他公钥密码算法所需密钥长度要小得多。

b) 选取一个大素数p和参数a，b共同构成一条素域上的椭圆曲线，并满足该椭圆曲线的阶的余因子较小。选择椭圆曲线上一个点作为基点G，并保证该点的阶不是余因子。计算PA=dAG作为公钥，dA则为私钥。

c) 加密过程

  假设要发送的消息为比特串M，len为M的比特长，加密者进行如下运算：

  第一步：用随机数发生器产生随机数http://192.168.1.205/Public/editor/attached/image/20130724/20130724145302_47699.gif，其中n是椭圆曲线基点G的阶次。

  第二步：计算椭圆曲线点http://192.168.1.205/Public/editor/attached/image/20130724/20130724145302_11493.gif，其中G和http://192.168.1.205/Public/editor/attached/image/20130724/20130724145303_59917.gif都在椭圆曲线上。并将http://192.168.1.205/Public/editor/attached/image/20130724/20130724145303_97188.gif的数据类型转换成比特串。

  第三步：计算椭圆曲线上的点http://192.168.1.205/Public/editor/attached/image/20130724/20130724145303_60242.gif，若S是无穷远点，则报错并退出。

  第四步：曲线点http://192.168.1.205/Public/editor/attached/image/20130724/20130724145303_79968.gif，将坐标http://192.168.1.205/Public/editor/attached/image/20130724/20130724145303_26025.gif、http://192.168.1.205/Public/editor/attached/image/20130724/20130724145303_99480.gif的数据类型转换成比特串。

  第五步：计算http://192.168.1.205/Public/editor/attached/image/20130724/20130724145303_49171.gif，http://192.168.1.205/Public/editor/attached/image/20130724/20130724145303_64916.gif是密钥派生函数，输出长度是len的比特串，若t为全零比特串，则返回步骤1。

  第六步：计算http://192.168.1.205/Public/editor/attached/image/20130724/20130724145303_69667.gif。

  第七步：计算http://192.168.1.205/Public/editor/attached/image/20130724/20130724145303_43324.gif, http://192.168.1.205/Public/editor/attached/image/20130724/20130724145303_2chapter     = 5110.gif是密码杂凑函数。

  第八步：输出密文http://192.168.1.205/Public/editor/attached/image/20130724/20130724145303_92739.gif。

d) 解密过程

  假设klen为密文中http://192.168.1.205/Public/editor/attached/image/20130724/20130724145606_29757.gif的比特长度，解密者要对密文http://192.168.1.205/Public/editor/attached/image/20130724/20130724145606_40587.gif进行解密，要完成以下运算：

  第一步：从http://192.168.1.205/Public/editor/attached/image/20130724/20130724145606_73037.gif中取出比特串http://192.168.1.205/Public/editor/attached/image/20130724/20130724145606_29631.gif，将http://192.168.1.205/Public/editor/attached/image/20130724/20130724145606_33469.gif的数据类型转换为椭圆曲线上的点，验证http://192.168.1.205/Public/editor/attached/image/20130724/20130724145606_58429.gif是否满足椭圆曲线方程，若不满足则报错并退出。

  第二步：计算http://192.168.1.205/Public/editor/attached/image/20130724/20130724145606_94724.gif，将坐标http://192.168.1.205/Public/editor/attached/image/20130724/20130724145606_32645.gif、http://192.168.1.205/Public/editor/attached/image/20130724/20130724145606_57054.gif的数据类型转换为比特串。

  第三步：计算http://192.168.1.205/Public/editor/attached/image/20130724/20130724145606_34076.gif，若http://192.168.1.205/Public/editor/attached/image/20130724/20130724145607_81362.gif为全零比特串，则报错并退出。

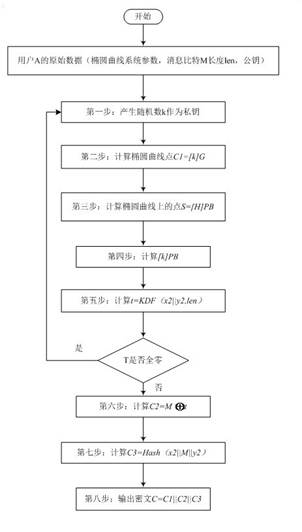
  第四步：从http://192.168.1.205/Public/editor/attached/image/20130724/20130724145607_66661.gif中取出比特串http://192.168.1.205/Public/editor/attached/image/20130724/20130724145607_28430.gif，计算http://192.168.1.205/Public/editor/attached/image/20130724/20130724145607_19167.gif。

  第五步：计算http://192.168.1.205/Public/editor/attached/image/20130724/20130724145607_87288.gif，从http://192.168.1.205/Public/editor/attached/image/20130724/20130724145607_15231.gif中取出比特串http://192.168.1.205/Public/editor/attached/image/20130724/20130724145607_54859.gif，若http://192.168.1.205/Public/editor/attached/image/20130724/20130724145607_39818.gif，则报错并退出。

  第六步：输出明文http://192.168.1.205/Public/editor/attached/image/20130724/20130724145607_61541.gif。

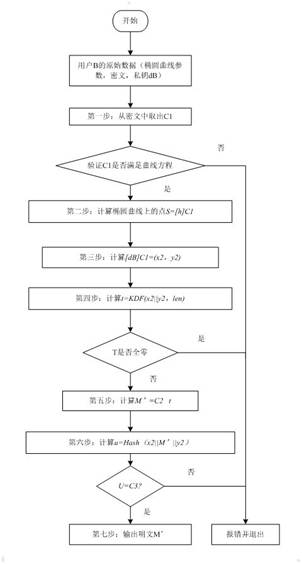
**2) 算法流程**

a) SM2算法加密过程程序实现流程图。如图一所示



图一  SM2加密算法流程

b) SM2算法解密过程程序实现流程图如下。如图二所示



图二  SM2解密算法流程

**3.实验环境**

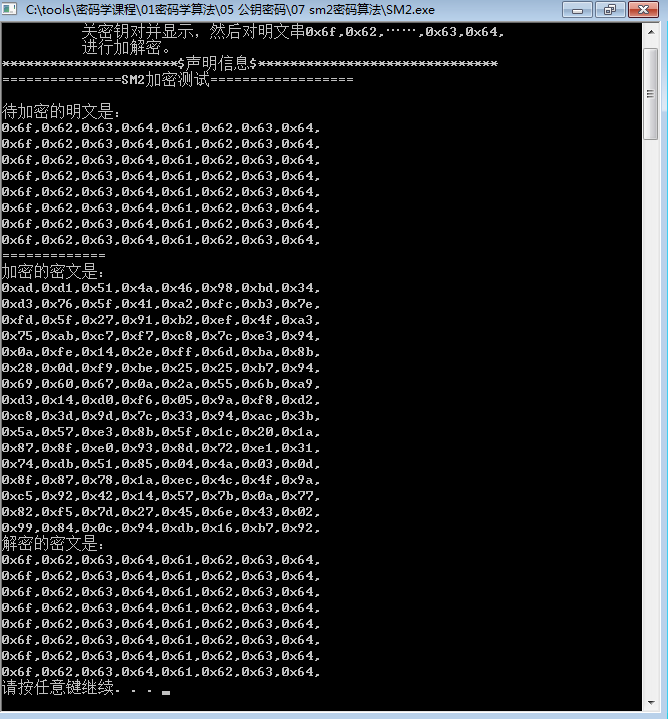
Windows 7

工具：C:\tools\密码学课程\01密码学算法\05 公钥密码\07 sm2密码算法

**4.实验步骤**

㈠查看SM2算法运行结果并学习核心算法

1.1运行文件【SM2.exe】。



1.2源码如下

SM2CurveParam para;

SM2Key key

char \*pp = PP;

char \*pa = PA;

char \*pb = PB;

char \*pGx =PGX;

char \*pGy =PGY;

char \*pn = PN;

unsignedchar pRandom [32 ];

inti;

unsignedchar plant [32 ];

unsignedint plen [2 ];

unsignedchar bHashData [64 ] = {0x6f,0x62,0x63,0x64,0x61,0x62,0x63,0x64,

0x6f,0x62,0x63,0x64,0x61,0x62,0x63,0x64,

0x6f,0x62,0x63,0x64,0x61,0x62,0x63,0x64,

0x6f,0x62,0x63,0x64,0x61,0x62,0x63,0x64,

0x6f,0x62,0x63,0x64,0x61,0x62,0x63,0x64,

0x6f,0x62,0x63,0x64,0x61,0x62,0x63,0x64,

0x6f,0x62,0x63,0x64,0x61,0x62,0x63,0x64,

0x6f,0x62,0x63,0x64,0x61,0x62,0x63,0x64

};

SM2Cipher cipher;

SM2Signature sign;

char \*pUserId ="1111111111111111";

intidLen = strlen(pUserId);

unsignedchar Hashdata [128 ] ="0x02";

intret;

unsignedchar Za [SM3 \_HASH \_LEN ];

unsignedchar Mdiesgt [SM3 \_HASH \_LEN ];

SM3 \_CTX sm3, \*psm3;

para = SM2 \_SM2CurveParam \_new(256);

Set \_SM2 \_Curve \_Param(&para, (const char \*)pp,(const char \*) pa,(const char \*) pb,(const char \*) pGx, (const char \*)pGy,(const char \*) pn, 256);

// printf("begin test n");

printf("===============SM2加密测试================== n");

printf("待加密的明文是: n");

print\_hex(bHashData,64);

psm3=&sm3;

SM3Init(psm3);

//SM3Update( psm3, Za, SM3 \_HASH \_LEN );

SM3Update( psm3,bHashData, 3);

SM3Final(Mdiesgt, psm3);

// print \_hex(Mdiesgt, 32);

for(i=0;i<TMAX;i++){

key =SM2 \_SM2Key \_new(256);

cipher = SM2 \_SM2Cipher \_new(256);

SM2 \_GenerateKeyPair(&para, &key);

printf("============= n");

ret=SM2 \_Encrypt(bHashData, 32, &key.pkey, &cipher);

ret=SM2 \_Decrypt(&cipher, &key, plant, plen);

printf("加密的密文是: n");

print \_hex(cipher.x,32);

print \_hex(cipher.y,32);

print \_hex(cipher.C,32);

print \_hex(cipher.M,32);

// print \_hex(plant, 32);

if(ret != 0 )

printf("Decrypt error n");

#if 1

SM2 \_SM2Cipher \_free(cipher);

SM2 \_SM2Key \_free(key);

#endif

}

printf("end test n");

return 0;

# 三、MD5算法

**1.实验目的**

1) 学习MD5算法的原理

2) 学习MD5算法的编程实现

**2.实验原理**

**1) 算法原理**

a) MD5 的全称是 Message-Digest Algorithm 5，在 90 年代初由 MIT 的计算机科学实验室和 RSA Data Security Inc发明，经 MD2、MD3 和 MD4 发展而来。此算法将对输入的任意有限长度的信息进行计算，产生一个128位长度的“指纹”或“报文摘要”。

b) MD5的处理过程如下：

Ø 填充消息使其成为512-bit的倍数少64-bit的比特块，填充时用一个“1”，后跟适当多个“0”，最后的64-bit用于存放消息的长度。

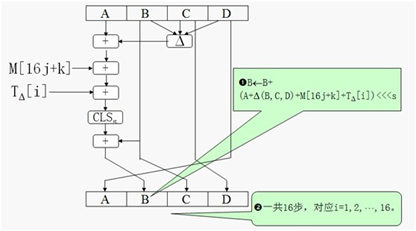
Ø 将消息分割成512-bit的消息块，每块再划分成16个32-bit的子块M1，M2，…，M16。

Ø 取4个32-bit的初始量：A0=01234567，B0=89ABCDEF，C0=FEDCBA98，D0=76543210，令A=A0，B=B0，C=C0，D=D0。

Ø 对A，B，C，D作4轮指定的计算，依次分别为：

    fF(ABCD,Yj,TF[1L16])，fG(ABCD,Yj,  
         TG[1L16])，fH(ABCD,Yj,  
         TH[1L16])，fI(ABCD,Yj,  
         TI[1L16])。

    fD(ABCD,Yj,TD[1L16])（D=F,G,H,I）的计算过程如下。如图一所示

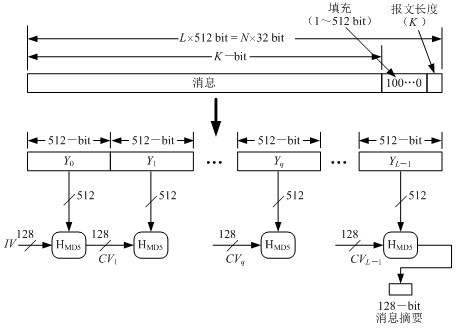


图一  压缩函数运算

Ø 再将A0，B0，C0，D0分别加到A，B，C，D上，即：A=A0+A，B=B0+B，C=C0+C，D=D0+D。

Ø 对剩余的消息块继续（4）和（5）的操作。

Ø 最后输出消息摘要值：A，B，C，D，共128-bit。如图二所示



图二 MD5生成消息摘要的过程

**2) 算法参数**

下表为F，G，H，I函数所用到的参数表：

表1 F-轮参数表

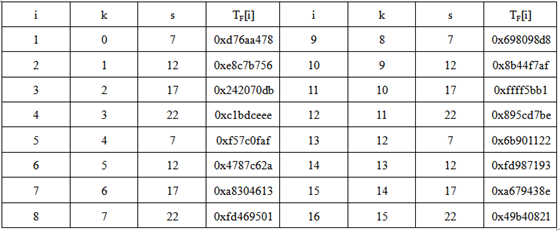


表2 G-轮参数表

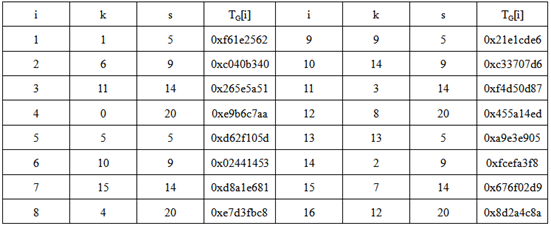


表3 H-轮参数表

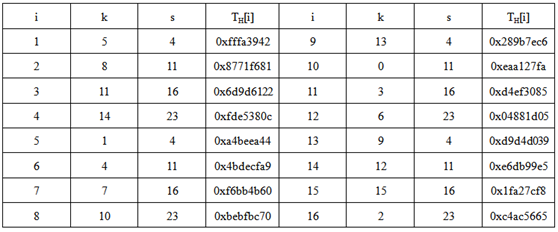
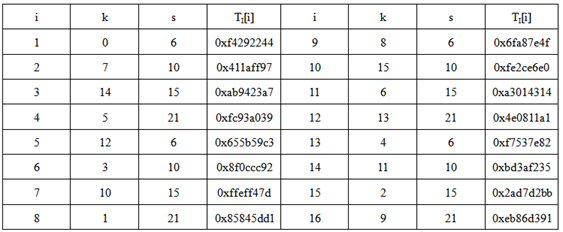
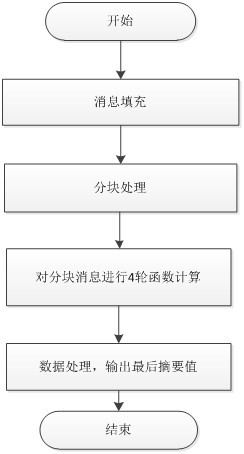


表4 I-轮参数表



**3) 算法流程**

MD5算法的程序实现流程图如下。如图三所示



图三 MD5算法流程图

**3.实验环境**

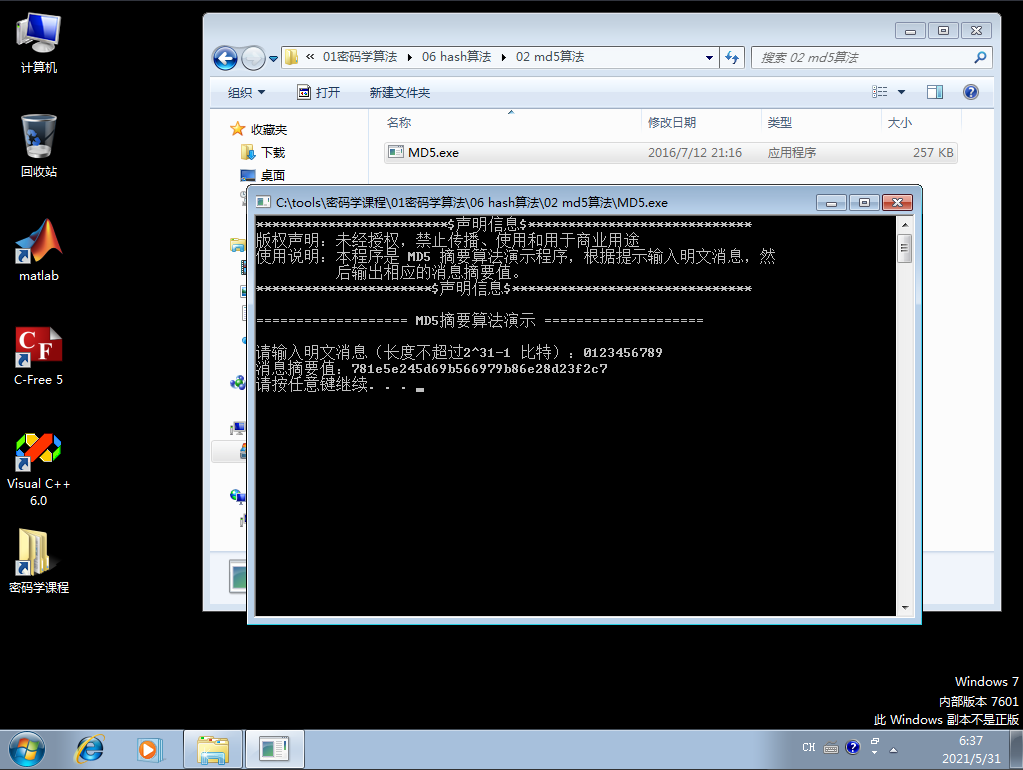
Windows 7

工具：C:\tools\密码学课程\01密码学算法\06 hash算法\02 md5算法

**4.实验步骤**

㈠查看MD5算法运行结果并学习核心算法

1.1运行程序，输入明文，即可得到摘要值。



1.2源代码如下所示。

int main()

{

int i;

unsigned long a;

unsigned long b;

unsigned long c;

unsigned long d;

unsigned long M[16];

ISD\_input();

ISD\_append();

for(i=0;i<num;i=i+64)

{

a=A;b=B;c=C;d=D;

memmove((char\\*)M,info+i,64);

//round 1

ISD\\_FF(a, b, c, d, M\[0\], 7, 0xd76aa478);

ISD\\_FF(d, a, b, c, M\[1\], 12, 0xe8c7b756);

ISD\\_FF(c, d, a, b, M\[2\], 17, 0x242070db);

ISD\\_FF(b, c, d, a, M\[3\], 22, 0xc1bdceee);

ISD\\_FF(a, b, c, d, M\[4\], 7, 0xf57c0faf);

ISD\\_FF(d, a, b, c, M\[5\], 12, 0x4787c62a);

ISD\\_FF(c, d, a, b, M\[6\], 17, 0xa8304613);

ISD\\_FF(b, c, d, a, M\[7\], 22, 0xfd469501);

ISD\\_FF(a, b, c, d, M\[8\], 7, 0x698098d8);

ISD\\_FF(d, a, b, c, M\[9\], 12, 0x8b44f7af);

ISD\\_FF(c, d, a, b, M\[10\], 17, 0xffff5bb1);

ISD\\_FF(b, c, d, a, M\[11\], 22, 0x895cd7be);

ISD\\_FF(a, b, c, d, M\[12\], 7, 0x6b901122);

ISD\\_FF(d, a, b, c, M\[13\], 12, 0xfd987193);

ISD\\_FF(c, d, a, b, M\[14\], 17, 0xa679438e);

ISD\\_FF(b, c, d, a, M\[15\], 22, 0x49b40821);

//round 2

ISD\\_GG(a, b, c, d, M\[1\], 5, 0xf61e2562);

ISD\\_GG(d, a, b, c, M\[6\], 9, 0xc040b340);

ISD\\_GG(c, d, a, b, M\[11\], 14, 0x265e5a chapter = 51);

ISD\\_GG(b, c, d, a, M\[0\], 20, 0xe9b6c7aa);

ISD\\_GG(a, b, c, d, M\[5\], 5, 0xd62f105d);

ISD\\_GG(d, a, b, c, M\[10\], 9, 0x02441453);

ISD\\_GG(c, d, a, b, M\[15\], 14, 0xd8a1e681);

ISD\\_GG(b, c, d, a, M\[4\], 20, 0xe7d3fbc8);

ISD\\_GG(a, b, c, d, M\[9\], 5, 0x21e1cde6);

ISD\\_GG(d, a, b, c, M\[14\], 9, 0xc33707d6);

ISD\\_GG(c, d, a, b, M\[3\], 14, 0xf4d50d87);

ISD\\_GG(b, c, d, a, M\[8\], 20, 0x455a14ed);

ISD\\_GG(a, b, c, d, M\[13\], 5, 0xa9e3e905);

ISD\\_GG(d, a, b, c, M\[2\], 9, 0xfcefa3f8);

ISD\\_GG(c, d, a, b, M\[7\], 14, 0x676f02d9);

ISD\\_GG(b, c, d, a, M\[12\], 20, 0x8d2a4c8a);

//round 3

ISD\\_HH(a, b, c, d, M\[5\], 4, 0xfffa3942);

ISD\\_HH(d, a, b, c, M\[8\], 11, 0x8771f681);

ISD\\_HH(c, d, a, b, M\[11\], 16, 0x6d9d6122);

ISD\\_HH(b, c, d, a, M\[14\], 23, 0xfde5380c);

ISD\\_HH(a, b, c, d, M\[1\], 4, 0xa4beea44);

ISD\\_HH(d, a, b, c, M\[4\], 11, 0x4bdecfa9);

ISD\\_HH(c, d, a, b, M\[7\], 16, 0xf6bb4b60);

ISD\\_HH(b, c, d, a, M\[10\], 23, 0xbebfbc70);

ISD\\_HH(a, b, c, d, M\[13\], 4, 0x289b7ec6);

ISD\\_HH(d, a, b, c, M\[0\], 11, 0xeaa127fa);

ISD\\_HH(c, d, a, b, M\[3\], 16, 0xd4ef3085);

ISD\\_HH(b, c, d, a, M\[6\], 23, 0x04881d05);

ISD\\_HH(a, b, c, d, M\[9\], 4, 0xd9d4d039);

ISD\\_HH(d, a, b, c, M\[12\], 11, 0xe6db99e5);

ISD\\_HH(c, d, a, b, M\[15\], 16, 0x1fa27cf8);

ISD\\_HH(b, c, d, a, M\[2\], 23, 0xc4ac5665);

//round 4

ISD\\_II(a, b, c, d, M\[0\], 6, 0xf4292244);

ISD\\_II(d, a, b, c, M\[7\], 10, 0x432aff97);

ISD\\_II(c, d, a, b, M\[14\], 15, 0xab9423a7);

ISD\\_II(b, c, d, a, M\[5\], 21, 0xfc93a039);

ISD\\_II(a, b, c, d, M\[12\], 6, 0x655b59c3);

ISD\\_II(d, a, b, c, M\[3\], 10, 0x8f0ccc92);

ISD\\_II(c, d, a, b, M\[10\], 15, 0xffeff47d);

ISD\\_II(b, c, d, a, M\[1\], 21, 0x85845dd1);

ISD\\_II(a, b, c, d, M\[8\], 6, 0x6fa87e4f);

ISD\\_II(d, a, b, c, M\[15\], 10, 0xfe2ce6e0);

ISD\\_II(c, d, a, b, M\[6\], 15, 0xa3014314);

ISD\\_II(b, c, d, a, M\[13\], 21, 0x4e0811a1);

ISD\\_II(a, b, c, d, M\[4\], 6, 0xf7537e82);

ISD\\_II(d, a, b, c, M\[11\], 10, 0xbd3af235);

ISD\\_II(c, d, a, b, M\[2\], 15, 0x2ad7d2bb);

ISD\\_II(b, c, d, a, M\[9\], 21, 0xeb86d391);

A+=a;B+=b;C+=c;D+=d;

}

unsigned char r[16];

memmove(r+0,(char \\*)&A,4);

memmove(r+4,(char \\*)&B,4);

memmove(r+8,(char \\*)&C,4);

memmove(r+12,(char \\*)&D,4);

printf("摘要值:");

for(i=0;i<16;i++) printf("%02x",r\[i\]);

printf("\\n");

return 0;

}