ISSN 1009-3044
Computer Knowledge and Technology 电脑知识与技术
Vol.5,No.19, July 2009, pp.5155-5156,5178

E-mail: xsjl@cccc.net.cn
http://www.dnzs.net.cn
Tel:+86-551-5690963 5690964

IDEA 加密解密算法的设计与实现

杨建武

(华南师范大学 增城学院,广东 广州 511363)

摘要:IDEA 算法是在 DES 算法的基础上发展出来的,是一种使用 128 位密钥以 64 位分组为单位加密数据的分组密码算法。该文主要对 IDEA 算法思想进行深入的分析和研究,并在此基础上阐述了改算法的实现原理和过程,尤其对该算法的密钥扩展和加密过程进行了详细的描述,并在 C# 环境下设计、实现 IDEA 对称加解密算法。

关键词:分组密码结构;加密;解密;迭代算法

中图分类号:TP312 **文献标识码**:A **文章编号**:1009-3044(2009)19-5155-02

Design and Implementation of IDEA Encryption Algorithm

YANG Jian-wu

(Zengcheng college, South China Normal University, Guangzhou 511363, China)

Abstract: IDEA algorithm is based on the DES algorithm,it is a block cipher algorithm using 128-bit key to encrypt data with a 64-bit grouping as the unit.On the base of thorough analysis and research on the main idea of IDEA algorithm,the implementation principle and process of the IDEA algorithm are discussed,particularly,the key expansion and encryption process are discribed in details,and the IDEA symmetric encryption algorithm is designed and implemented in the C# environment.

Key words: structure of block ciphers; encryption; decryption; iterative algorithm

信息加密技术作为信息安全最有效的方法之一,可以通过不同的加密算法来实现,而加密算法的选取直接影响着信息的安全程度。IDEA 加密算法是已公开的可用算法中速度快且安全性强的分组密码算法,具有良好的应用前景。具有极强的抗攻击能力。本文主要对 IDEA 算法思想进行深入的分析和研究,并在此基础上阐述了改算法的实现原理和过程,尤其对该算法的密钥扩展和加密过程进行了详细的描述,并在 C# 环境下设计、实现 IDEA 对称加解密算法。

1 IDEA 算法简介

1.1 IDEA 算法概述

IDEA 算法是一个分组长度为 64 比特的分组密码算法,密钥长度为 128 比特,由 8 轮迭代操作实现。每个迭代都由三种函数:mod(216)加法、mod(216+1)乘法和逐位异或算法组成。整个算法包括子密钥产生、数据加密过程、数据解密过程三部分。该算法规定明文和密文块均为 64 比特,密钥长度为 128 比特,加解密相同,只是密钥各异。

1.1.1 加密过程

IDEA 总共进行 8 轮迭代操作,每轮需要 6 个子密钥,另外还需要 4 个额外子密钥输出变换,所以总共需要 52 个子密钥,这 52 个子密钥都是从 128 比特密钥中扩展出来的。

输入的明文为 8 个字符(即 64 比特),将 64 比特数据块分成 X1,,X2,,X,3,X,4 四个子块,每一子块 16 比特。这 4 个子块将作为第一轮迭代的输入,全部共 8 轮迭代。在每一轮中,这 4 个子块相互异或、相加和相乘,且与 6 个 16 比特子密钥相异或、相加和相乘。最后在输出变换中 4 个子块与 4 个子密钥进行运算。加密过程如图 1 所示。

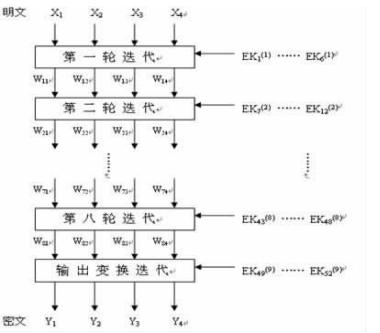

图 1 加密过程图

1.1.2 解密过程

和加密密钥一样,解密密钥也需要扩展,不同的是解密密钥是由加密密钥加法逆或乘法逆构成的,且两者一一对应。

IDEA 算法的解密过程和加密过程一样,只是加密用加密密钥 EK,解密用的是解密密钥 DK。输入的密文同样为 8 个字符(即 64 比特),将 64 比特数据块分成 Y1,,Y2,,Y,3,Y,4 四个子块,每一子块 16 比特。这 4 个子块将作为第一轮迭代的输入,全部共 8 轮迭代。解密过程如图 2。

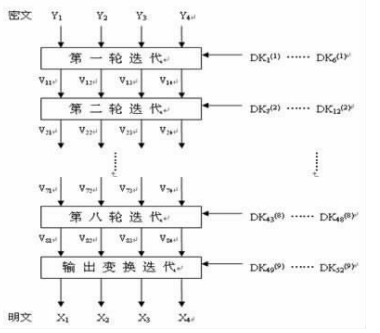

图 2 解密过程图

IDEA 算法结构图如图 3。

2 IDEA 算法的编程实现

2.1 加密密钥扩展算法的实现

在算法中,实际上是将 128 比特密钥扩展为 832 比特。每一个字节用 0 补足 16 位。具体是如何扩展的呢?在前面的扩展思想中已经知道密钥串分 8 个子块,循环左移 7 次,其中第 7 次循环是取前 4 个,所以最终形成 832 比特(16*8*6+16*4)。扩展密钥数组为 m_nKeyEncryptBox[52],部分代码如下。

```
byte[]asciiBytes=Encoding.Convert(unicode,ascii,unicode.GetBytes
(this.m_sEncryptionKey));                    //将一组字符编码成一个字节序列
```

收稿日期:2009-04-07

```
char[] asciiChars = new char[ascii.GetCharCount(asciiBytes,
0, asciiBytes.Length)];          // 进行字节到字符再到串的转换
ascii.GetChars(asciiBytes, 0, asciiBytes.Length,
asciiChars, 0);                  // 由密钥 m_sEncryptionKey 扩展成加密密钥扩展数组 m_nKeyEncryptBox
for (j = 0; j < 8; j++)          //加密子密钥[j]循环左移 8+[j+1]位
……
```

2.2 加密解密算法的实现

IDEA加密算法和解密算法是相同的，其处理为：将 64 比特输入分成 4 个 16 比特的子块 A、B、C、D，然后进行 8 轮循环，在第 j 轮循环中，子密钥为：key[6*i]–key[6*i+5]，其中，i=0……7。部分代码如下。

```
public static void IdeaCipher(byte[] inbuf, byte[] outbuf, ushort[] key)
{
ushort A, B, C, D, E, F;
int i;
//明文[0]和明文[2]，明文[4]明文[6]……循环左移 8 位后相或
A = (ushort)((inbuf[0] & 0xFF) | ((ushort)(inbuf[2] & 0xFF)) << 8);
B = (ushort)((inbuf[4] & 0xFF) | ((ushort)(inbuf[6] & 0xFF)) << 8);
C = (ushort)((inbuf[8] & 0xFF) | ((ushort)(inbuf[10] & 0xFF)) << 8);
D = (ushort)((inbuf[12] & 0xFF) | ((ushort)(inbuf[14] & 0xFF)) << 8);
for (i = 0; i < 8; i++)
{
A = Mul(A, key[6*i]);
B += key[6*i+1];          //B 的值加上密钥 key[……]后返回给 B
……
```

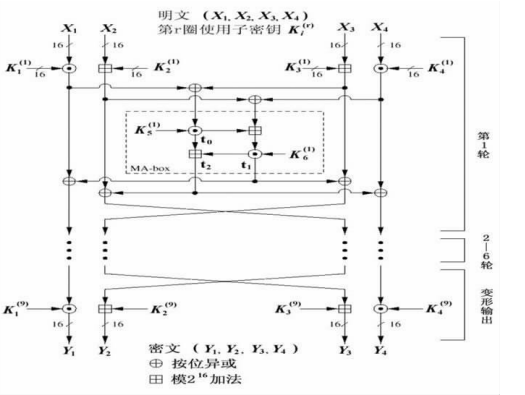

图 3　IDEA 算法结构图

2.3 解密密钥扩展算法的实现

解密密钥也需要扩展，不过，它与加密密钥扩展不同，它不需要从密钥串中变换而来。IDEA 解密密钥扩展数组 m_nKeyDecryptBox[52]是由加密密钥数组 m_nKeyEncryptBox 处理而来的。其具体处理步骤为：首先由 m_nKeyEncryptBox[0] 到 m_nKeyEncryptBox[3] 四个加密密钥数组元素处理成四个解密密钥数组元素 m_nKeyDecryptBox[51] 到 m_nKeyDecryptBox[48]，然后进行 8 轮迭代。在第 m 轮中，由六个加密密钥数组元素 m_nKeyEncryptBox[4+6*m]到 m_nKeyEncryptBox[9+6*m]生成六个解密密钥数组元素 m_nKeyDecryptBox[47–6*m]到 m_nKeyDecryptBox[42–6*m]。最终生成 832 比特的加密密钥数组 m_nKeyDecryptBox[52]。部分代码如下。

```
A = MulInv(m_nKeyEncryptBox[0]);      //调用 MulInv 处理子密钥[0]后赋给 A
B = (ushort)(–m_nKeyEncryptBox[1]);          //子密钥[1]赋值给 B
C = (ushort)(–m_nKeyEncryptBox[2]);          //子密钥[2]赋值给 C
m_nKeyDecryptBox[51] = MulInv(m_nKeyEncryptBox[3]);  // 调用 MulInv 处理子密钥[3]后赋给解密子密钥[51]
……
A = m_nKeyEncryptBox[4 + m * 6];          // 调用 MulInv 处理子密钥[4+m*6]后赋给 A
m_nKeyDecryptBox[47 – m * 6] = m_nKeyEncryptBox[5 + m * 6]; //子密钥[5+m*6]赋值给[47–m*6]
m_nKeyDecryptBox[46 – m * 6] = A;          //子密钥 A 赋值给[47–m*6]
A = MulInv(m_nKeyEncryptBox[6 + m * 6]);      // 调用 MulInv 处理子密钥[6+m*6]后赋给 A
B = (ushort)(–m_nKeyEncryptBox[7 + 6 * m]);   //子密钥[7 + 6 * m]赋值给 B
C = (ushort)(–m_nKeyEncryptBox[8 + 6 * m]);   //子密钥[8 + 6 * m]赋值给 C
m_nKeyDecryptBox[45 – 6 * m] = MulInv(m_nKeyEncryptBox[9 + 6 * m]); // 调用 MulInv 处理子密钥[9 + 6 * m]后赋给解密子密钥[45 – 6 * m]
……
```

2.4 模(216+1)乘法算法

在 IDEA 算法的加密和解密过程中，每一轮迭代都是由分组数据之间，分组数据和密钥间进行模加、模乘和逐位异或运算组成的。其中，模乘的核心代码如下。

```
public static ushort Mul(ushort x, ushort y)   //明文 x,密钥 y 进行模(216+1)乘法运算
{ int temp;                                    //设一个中间变量 temp
if (x == 0)                                   //若 x=0,x=65537–y
x = (ushort)(0x10001–y);
else
if (y == 0)                                   //若 y=0,x=65537–x
x = (ushort)(0x10001–x);
else
{ temp = x * y;                               //否则 temp=x*y
y = Low16(temp);                              //y 取 temp 的低 16 位
x = (ushort)(temp>>16);                       //x 取 temp 的高 16 位
x = (ushort)((y–x)+((y<x)?1:0));              //若 y<x,x=(y–x)+1,否则 x=y–x
}
return (ushort)(x&0xFFFF);
……
```

2.5 模(216+1)乘逆元算法

模乘逆元算法是在解密密钥扩展时的主要算法，解密密钥扩展数组 m_nKeyDecryptBox[52]是由加密密钥数组 m_nKeyEncryptBox 通过加法逆或乘法逆处理而来的。模乘逆元算法的核心代码如下。（下转第 5178 页）

图 1　人物行走动画

图 2　爆炸动画

以爆炸动画为例说明动画的具体实现：Step 用来记录当前帧；FPS 用来控制动画速度，数值越大，则动画速度也就越快；C*C 为图块的大小；X、Y 为爆炸位置的坐标。爆炸动画代码可以由线程执行，也可以放在 TimerTask 的派生类的 run 函数中，主要代码如下：

```
Step = (ActionFPS /FPS) % 3;              //控制动画的循环，由 3 帧组成
g.setClip(X, Y, C, C);
g.drawImage(IMG, X – Step * C ,Y , g.TOP | g.LEFT);      //由 Step 来控制更换图片
ActionFPS++;
```

3 结束语

本文对游戏实现中的关键技术进行简要介绍，并论述了游戏实现的要点，包括状态转换处理、声音处理、地图场景处理、图形绘制、动画的实现等。游戏设计并移植到手机上运行，效果良好。实践证明，J2ME 技术适合应用于手机游戏的开发，具有技术入门容易、移植性强、开发速度快等优势。

参考文献：

[1] 詹建飞.J2ME 开发精解[M].北京:电子工业出版社,2006:49–116.
[2] 郎锐.J2ME 手机程序 Eclipse 开发基础[M].北京:机械工业出版社,2006:155–240.

张元巩(1983–)，男，主要研究方向：J2ME 开发应用；

冯永晋(1977–)，男，讲师，硕士，主要研究方向：分布式数据库，信息管理系统。

(上接第 5156 页)

```
public static ushort MulInv(ushort x)          //密钥 x 进行模乘逆元运算
{
ushort t0, t1, q, y;              //定义短整型变量 t0,t1,q,y
if (x <= 1)                       //若 x 小于 1,返回 x
return x;
t1 = (ushort)(0x10001 / x);          //65537 除 x 后,把商赋给 t1,余数赋给 y
y = (ushort)(0x10001 % x);
if (y == 1)                       //若 y=1,返回 65537 减 t1 后的值的低 16 位
return Low16(0x10001–t1);
t0 = 1;                           //位 t0 赋初值 1
do
……
```

2.6 演示界面

演示界面如图 4。

图 4　IDEA 加解密实现界面

5 结束语

IDEA 算法加密速度快，密钥产生方法简单。该算法若采用搜索破译，需要 2128 大约 1038 试探，8 轮迭代使得没有任何捷径破译。对于差分和线性攻击是安全的。若将字节有 16byte 增至 32byte，密钥长度相应长 256byte。采用模加和模乘，则更加可以进一步强化 IDEA 算法。目前，正将该算法作为独立的加解密芯片嵌入军用数码保护器系统。本文主要是对 IDEA 算法的设计思想和原理进行深入的研究和运用所学的知识进行设计和实现其主要加解密功能。

参考文献：

[1] 叶树华,高志红.网络编程实用教程[M].北京:人民邮电出版社,2006.
[2] 张天津,于志平.电子邮件传输中加密与解密的实现[J].计算机工程,2003,29(4).
[3] 王衍波,薛通.应用密码学[M].北京:机械工业出版社,2003.
[4] 胡志远.口令破解与加密技术[M].北京:机械工业出版社,2003.
[5] 段钢.加密与解密[M].北京:电子工业出版社,2003.

杨建武(1972–)，男，黑龙江鸡西人，研究方向：网络安全。