MD5算法

一、MD5简介

MD5即Message-Digest Algorithm 5（信息-摘要算法5），用于确保信息传输完整一致。是计算机广泛使用的杂凑算法之一（又译摘要算法、哈希算法），主流编程语言普遍已有MD5实现。

MD5算法具有以下特点：

1、压缩性：任意长度的数据，算出的MD5值长度都是固定的。

2、容易计算：从原数据计算出MD5值很容易。

3、抗修改性：对原数据进行任何改动，哪怕只修改1个字节，所得到的MD5值都有很大区别。

4、强抗碰撞：已知原数据和其MD5值，想找到一个具有相同MD5值的数据（即伪造数据）是非常困难的。

MD5的作用是让大容量信息在用数字签名软件签署私人密钥前被"压缩"成一种保密的格式（就是把一个任意长度的字节串变换成一定长的16进制数字串）。

大家都知道，地球上任何人都有自己独一无二的指纹，这常常成为司法机关鉴别罪犯身份最值得信赖的方法；与之类似，MD5就可以为任何文件（不管其大小、格式、数量）产生一个同样独一无二的MD5“数字指纹”，如果任何人对文件做了任何改动，其MD5也就是对应的“数字指纹”

都会发生变化。

二、MD5加密原理步骤

1.填充

在MD5算法中，首先需要对信息进行填充，使其位长对512求余的结果等于448，**并且填充必须进行，即使其位长对512求余的结果等于448**。因此，信息的位长（Bits Length）将被扩展至N*512+448，N为一个非负整数，N可以是零。

填充的方法如下：

1) 在信息的后面填充一个1和无数个0，直到满足上面的条件时才停止用0对信息的填充。

2) 在这个结果后面附加一个以64位二进制表示的填充前信息长度（单位为Bit），如果二

进制表示的填充前信息长度超过64位，则取低64位。

经过这两步的处理，信息的位长=N*512+448+64=(N+1)*512，即长度恰好是512的整数倍。这样做的原因是为满足后面处理中对信息长度的要求。

2. 初始化变量

初始的128位值为初试链接变量，这些参数用于第一轮的运算，以大端字节序来表示，他们分别为：A=0x01234567，B=0x89ABCDEF，C=0xFEDCBA98，D=0x76543210。

（每一个变量给出的数值是高字节存于内存低地址，低字节存于内存高地址，即大端字节序。在程序中变量A、B、C、D的值分别为0x67452301，0xEFCDAB89，0x98BADCFE，0x10325476）

3. 处理分组数据

每一分组的算法流程如下：

第一分组需要将上面四个链接变量复制到另外四个变量中：A到a，B到b，C到c，D到d。从第二分组开始的变量为上一分组的运算结果，即A = a， B = b， C = c， D = d。

主循环有四轮（MD4只有三轮），每轮循环都很相似。第一轮进行16次操作。每次操作对a、b、c和d中的其中三个作一次非线性函数运算，然后将所得结果加上第四个变量，文本的一个子分组和一个常数。再将所得结果向左**环移**一个不定的数，并加上a、b、c或d中之一。最后用该结果取代a、b、c或d中之一。

以下是每次操作中用到的四个非线性函数（每轮一个）。

F(X ,Y ,Z) = (X & Y) | ((~X) & Z)

G(X ,Y ,Z) = (X & Z) | (Y & (~Z))

H(X ,Y ,Z) =X ^ Y ^ Z

I(X ,Y ,Z) =Y ^ (X | (~Z))

（&是与（And），|是或（Or），~是非（Not），^是异

或（Xor））

这四个函数的说明：如果X、Y和Z的对应位是独立和均匀的，那么结果的每一位也应是独立和均匀的。

F是一个逐位运算的函数。即，如果X，那么Y，否则Z。函数H是逐位奇偶操作符。

假设Mj表示消息的第j个子分组（从0到15），常数ti是4294967296*abs(sin(i) ）的整数部分，i 取值从1到64，单位是弧度。（4294967296=2的32次方）

现定义：

FF(a ,b ,c ,d ,Mj ,s ,ti) 操作为 a = b + ((a + F(b,c,d) + Mj + ti) << s)

GG(a ,b ,c ,d ,Mj ,s ,ti) 操作为 a = b + ((a + G(b,c,d) + Mj + ti) << s)

HH(a ,b ,c ,d ,Mj ,s ,ti) 操作为 a = b + ((a + H(b,c,d) + Mj + ti) << s)

II(a ,b ,c ,d ,Mj ,s ,ti) 操作为 a = b + ((a + I(b,c,d) + Mj + ti) << s)

现定义：

FF(a ,b ,c ,d ,Mj ,s ,ti) 操作为 a = b + ((a + F(b,c,d) + Mj + ti) << s)

GG(a ,b ,c ,d ,Mj ,s ,ti) 操作为 a = b + ((a + G(b,c,d) + Mj

+ ti) << s)

HH(a ,b ,c ,d ,Mj ,s ,ti) 操作为 a = b + ((a + H(b,c,d) + Mj + ti) << s)

II(a ,b ,c ,d ,Mj ,s ,ti) 操作为 a = b + ((a + I(b,c,d) + Mj + ti) << s)

注意：“<<”表示循环左移位，不是左移位。

这四轮（共64步）是：

第一轮

FF(a ,b ,c ,d ,M0 ,7 ,0xd76aa478)

FF(d ,a ,b ,c ,M1 ,12 ,0xe8c7b756)

FF(c ,d ,a ,b ,M2 ,17 ,0x242070db)

FF(b ,c ,d ,a ,M3 ,22 ,0xc1bdceee)

FF(a ,b ,c ,d ,M4 ,7 ,0xf57c0faf)

FF(d ,a ,b ,c ,M5 ,12 ,0x4787c62a)

FF(c ,d ,a ,b ,M6 ,17 ,0xa8304613)

FF(b ,c ,d ,a ,M7 ,22 ,0xfd469501)

FF(a ,b ,c ,d ,M8 ,7 ,0x698098d8)

FF(d ,a ,b ,c ,M9 ,12 ,0x8b44f7af)

FF(c ,d ,a ,b ,M10 ,17 ,0xffff5bb1)

FF(b ,c ,d ,a ,M11 ,22 ,0x895cd7be)

FF(a ,b ,c ,d ,M12 ,7 ,0x6b901122)

FF(d ,a ,b ,c ,M13 ,12 ,0xfd987193)

FF(c ,d ,a ,b ,M14 ,17 ,0xa679438e)

FF(b ,c ,d ,a ,M15 ,22 ,0x49b40821)

第二轮

GG(a ,b ,c ,d ,M1 ,5 ,0xf61e2562)

GG(d ,a ,b ,c ,M6 ,9 ,0xc040b340)

GG(c ,d ,a ,b ,M11 ,14 ,0x265e5a51)

GG(b ,c ,d ,a ,M0 ,20 ,0xe9b6c7aa)

GG(a ,b ,c ,d ,M5 ,5 ,0xd62f105d)

GG(d ,a ,b ,c ,M10 ,9 ,0x02441453)

GG(c ,d ,a ,b ,M15 ,14 ,0xd8a1e681)

GG(b ,c ,d ,a ,M4 ,20 ,0xe7d3fbc8)

GG(a ,b ,c ,d ,M9 ,5 ,0x21e1cde6)

GG(d ,a ,b ,c ,M14 ,9 ,0xc33707d6)

GG(c ,d ,a ,b ,M3 ,14 ,0xf4d50d87)

GG(b ,c ,d ,a ,M8 ,20 ,0x455a14ed)

GG(a ,b ,c ,d ,M13 ,5 ,0xa9e3e905)

GG(d ,a ,b ,c ,M2 ,9 ,0xfcefa3f8)

GG(c ,d ,a ,b ,M7 ,14 ,0x676f02d9)

GG(b ,c ,d ,a ,M12 ,20 ,0x8d2a4c8a)

第三轮

HH(a ,b ,c ,d ,M5 ,4 ,0xfffa3942)

HH(d ,a ,b ,c ,M8 ,11 ,0x8771f681)

HH(c ,d ,a ,b ,M11 ,16 ,0x6d9d6122)

HH(b ,c ,d ,a ,M14 ,23 ,0xfde5380c)

HH(a ,b ,c ,d ,M1 ,4 ,0xa4beea44)

HH(d ,a ,b ,c ,M4 ,11 ,0x4bdecfa9)

HH(c ,d ,a ,b ,M7 ,16 ,0xf6bb4b60)

HH(b ,c ,d ,a ,M10 ,23 ,0xbebfbc70)

HH(a ,b ,c ,d ,M13 ,4 ,0x289b7ec6)

HH(d ,a ,b ,c ,M0 ,11 ,0xeaa127fa)

HH(c ,d ,a ,b ,M3 ,16 ,0xd4ef3085)

HH(b ,c ,d ,a ,M6 ,23 ,0x04881d05)

HH(a ,b ,c ,d ,M9 ,4 ,0xd9d4d039)

HH(d ,a ,b ,c ,M12 ,11 ,0xe6db99e5)

HH(c ,d ,a ,b ,M15 ,16 ,0x1fa27cf8)

HH(b ,c ,d ,a ,M2 ,23 ,0xc4ac5665)

第四轮

II(a ,b ,c ,d ,M0 ,6 ,0xf4292244)

II(d ,a ,b ,c ,M7 ,10 ,0x432aff97)

II(c ,d ,a ,b ,M14 ,15 ,0xab9423a7)

II(b ,c ,d ,a ,M5 ,21 ,0xfc93a039)

II(a ,b ,c ,d ,M12 ,6 ,0x655b59c3)

II(d ,a ,b ,c ,M3 ,10 ,0x8f0ccc92)

II(c ,d ,a ,b ,M10 ,15 ,0xffeff47d)

II(b ,c ,d ,a ,M1 ,21 ,0x85845dd1)

II(a ,b ,c ,d ,M8 ,6 ,0x6fa87e4f)

II(d ,a ,b ,c ,M15 ,10 ,0xfe2ce6e0)

II(c ,d ,a ,b ,M6 ,15 ,0xa3014314)

II(b ,c ,d ,a ,M13 ,21 ,0x4e0811a1)

II(a ,b ,c ,d ,M4 ,6 ,0xf7537e82)

II(d ,a ,b ,c ,M11 ,10 ,0xbd3af235)

II(c ,d ,a ,b ,M2 ,15 ,0x2ad7d2bb)

II(b ,c ,d ,a ,M9 ,21 ,0xeb86d391)

所有这些完成之后，将a、b、c、d分别在原来基础上再加上A、B、C、D。

即a = a + A，b = b + B，c = c + C，d = d + D

然后用下一分组数据继续运行以上算法。

4. 输出

最后的输出是a、b、c和d的级联。

当你按照我上面所说的方法实现MD5算法以后，你可以用以下几个信息对你做出来的程序作一个简单的测试，看看程序有没有错误。

MD5 ("") = d41d8cd98f00b204e9800998ecf8427e

MD5 ("a") = 0cc175b9c0f1b6a831c399e269772661

MD5 ("abc") = 900150983cd24fb0d6963f7d28e17f72

MD5 ("message digest") =

f96b697d7cb7938d525a2f31aaf161d0

MD5 ("abcdefghijklmnopqrstuvwxyz") =
c3fcd3d76192e4007dfb496cca67e13b

MD5
("ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz") =

f29939a25efabaef3b87e2cbfe641315

MD5 ("8a683566bcc7801226b3d8b0cf35fd97")
=cf2cb5c89c5e5eeebef4a76becddfcfd

MD5加密字符串实例

现以字符串“jklmn”为例。

该字符串在内存中表示为：6A 6B 6C 6D 6E（从左到右为低地址到高地址，后同），信息长度为40 bits， 即0x28。

对其填充，填充至448位，即56字节。结果为：

6A 6B 6C 6D 6E 80 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

剩下64位，即8字节填充填充前信息位长，按小端字节序填充剩下的8字节，结果为。

6A 6B 6C 6D 6E 80 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 28 00 00 00 00 00 00 00

（64字节，512 bits）

初始化A、B、C、D四个变量。

将这64字节填充后数据分成16个小组（程序中对应为16个数组），即：

M0：6A 6B 6C 6D （这是内存中的顺序，按照小端字节序原则，对应数组M(0)的值为0x6D6C6B6A，下同）

M1：6E 80 00 00

M2：00 00 00 00

.....

M14：28 00 00 00

M15：00 00 00 00

经过“**3. 分组数据处理**”后，a、b、c、d值分别为0xD8523F60、0x837E0144、0x517726CA、0x1BB6E5FE

在内存中为a：60 3F 52 D8

b：44 01 7E 83

c：CA 26 77 51

d：FE E5 B6 1B

a、b、c、d按内存顺序输出即为最终结果：603F52D844017E83CA267751FEE5B61B。这就是字符串“jklmn”的MD5值。

三、代码实现MD5转换器

主程序：

```
package com.oop.lhw0524;

import java.awt.BorderLayout;
import java.awt.Color;

import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPasswordField;
import javax.swing.JTextField;

public class MD5 {
        /*
         * 四个链接变量
         */
        private final int A = 0x67452301;
        private final int B = 0xefcdab89;
        private final int C = 0x98badcfe;
        private final int D = 0x10325476;
        /*
```

```
     * ABCD的临时变量
     */
    private int Atemp, Btemp, Ctemp, Dtemp;

    /*
     * 常量ti公式:floor(abs(sin(i+1))×(2pow32)
     */
    private final int K[] = { 0xd76aa478, 0xe8c7b756, 0
                    0xf57c0faf, 0x4787c62a, 0xa8304613,
                    0x8b44f7af, 0xffff5bb1, 0x895cd7be,
                    0xa679438e, 0x49b40821, 0xf61e2562,
                    0xe9b6c7aa, 0xd62f105d, 0x02441453,
                    0x21e1cde6, 0xc33707d6, 0xf4d50d87,
                    0xfcefa3f8, 0x676f02d9, 0x8d2a4c8a,
                    0x6d9d6122, 0xfde5380c, 0xa4beea44,
                    0xbebfbc70, 0x289b7ec6, 0xeaa127fa,
                    0xd9d4d039, 0xe6db99e5, 0x1fa27cf8,
                    0x432aff97, 0xab9423a7, 0xfc93a039,
                    0xffeff47d, 0x85845dd1, 0x6fa87e4f,
                    0x4e0811a1, 0xf7537e82, 0xbd3af235,
    /*
     * 向左位移数,计算方法未知
     */
    private final int s[] = { 7, 12, 17, 22, 7, 12, 17,
                    12, 17, 22, 5, 9, 14, 20, 5, 9, 14,
                    4, 11, 16, 23, 4, 11, 16, 23, 4, 11
                    15, 21, 6, 10, 15, 21, 6, 10, 15, 2

    /*
     * 初始化函数
     */
    private void init() {
            Atemp = A;
            Btemp = B;
```

```
        Ctemp = C;
        Dtemp = D;
    }

    /*
     * 移动一定位数
     */
    private int shift(int a, int s) {
        return (a << s) | (a >>> (32 - s));// 右移的
    }

    /*
     * 主循环
     */
    private void MainLoop(int M[]) {
        int F, g;
        int a = Atemp;
        int b = Btemp;
        int c = Ctemp;
        int d = Dtemp;
        for (int i = 0; i < 64; i++) {
            if (i < 16) {
                F = (b & c) | ((~b) & d);
                g = i;
            } else if (i < 32) {
                F = (d & b) | ((~d) & c);
                g = (5 * i + 1) % 16;
            } else if (i < 48) {
                F = b ^ c ^ d;
                g = (3 * i + 5) % 16;
            } else {
                F = c ^ (b | (~d));
                g = (7 * i) % 16;
            }
```

```
                int tmp = d;
                d = c;
                c = b;
                b = b + shift(a + F + K[i] + M[g],
                a = tmp;
        }
        Atemp = a + Atemp;
        Btemp = b + Btemp;
        Ctemp = c + Ctemp;
        Dtemp = d + Dtemp;

}

/*
 * 填充函数处理后应满足bits≡448(mod512),字节就是bytes≡5
 * 最后加上64位的原来长度
 */
private int[] add(String str) {
        int num = ((str.length() + 8) / 64) + 1;//
        int strByte[] = new int[num * 16];// 64/4=1
        for (int i = 0; i < num * 16; i++) {// 全部补
                strByte[i] = 0;
        }
        int i;
        for (i = 0; i < str.length(); i++) {
                strByte[i >> 2] |= str.charAt(i) <<
        }
        strByte[i >> 2] |= 0x80 << ((i % 4) * 8);//
        /*
         * 添加原长度，长度指位的长度，所以要乘8，然后是小
         */
        strByte[num * 16 - 2] = str.length() * 8;
        return strByte;
}
```

```
/*
 * 调用函数
 */
public String getMD5(String source) {
        init();
        int strByte[] = add(source);
        for (int i = 0; i < strByte.length / 16; i+
                int num[] = new int[16];
                for (int j = 0; j < 16; j++) {
                        num[j] = strByte[i * 16 + j
                }
                MainLoop(num);
        }
        return changeHex(Atemp) + changeHex(Btemp)
                        + changeHex(Dtemp);

}

/*
 * 整数变成16进制字符串
 */
private String changeHex(int a) {
        String str = "";
        for (int i = 0; i < 4; i++) {
                str += String.format("%2s",
                                Integer.toHexString
        }
        return str;
}

/*
 * 单例
 */
```

```
private static MD5 instance;
public static MD5 getInstance() {
        if (instance == null) {
                instance = new MD5();
        }
        return instance;
}

private MD5() {
};
public static void main(String[] args) {
        JFrame jf = new JFrame();
        jf.setTitle("MD5");
        jf.setSize(400, 400);
        jf.setLayout(new BorderLayout());
        jf.setLocationRelativeTo(null);
        jf.setDefaultCloseOperation(3);
        jf.setResizable(false);
        jf.setBackground(Color.DARK_GRAY);
    java.awt.FlowLayout f1=new java.awt.FlowLayout(
    jf.setLayout(f1);
        JLabel  input=new JLabel("输入");
        jf.add(input);
        JTextField inputcontent=new JTextField(30);
        jf.add(inputcontent);
        JLabel output=new JLabel("输出");
        jf.add(output);
        JTextField outputcontent=new JTextField(30)
        jf.add(outputcontent);
        JButton change=new JButton("加密");
        jf.add(change);
        MD5Listener md5=new MD5Listener(inputconten
        change.addActionListener(md5);
        jf.setVisible(true);
```

```
        }
```

监听程序：

```
package com.oop.lhw0524;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.JTextField;

public class MD5Listener implements ActionListener {
        private JTextField inputcontent;
        private JTextField outputcontent;

        public MD5Listener(JTextField f1, JTextField f2) {
                inputcontent = f1;
                outputcontent = f2;
        }

        public void actionPerformed(ActionEvent e) {
                String str = MD5.getInstance().getMD5(input
                outputcontent.setText(str);
                System.out.println(str);
        }
}
```

编辑于 2018-05-24