

MD5 加密

2015 级计应

15331151

李佳

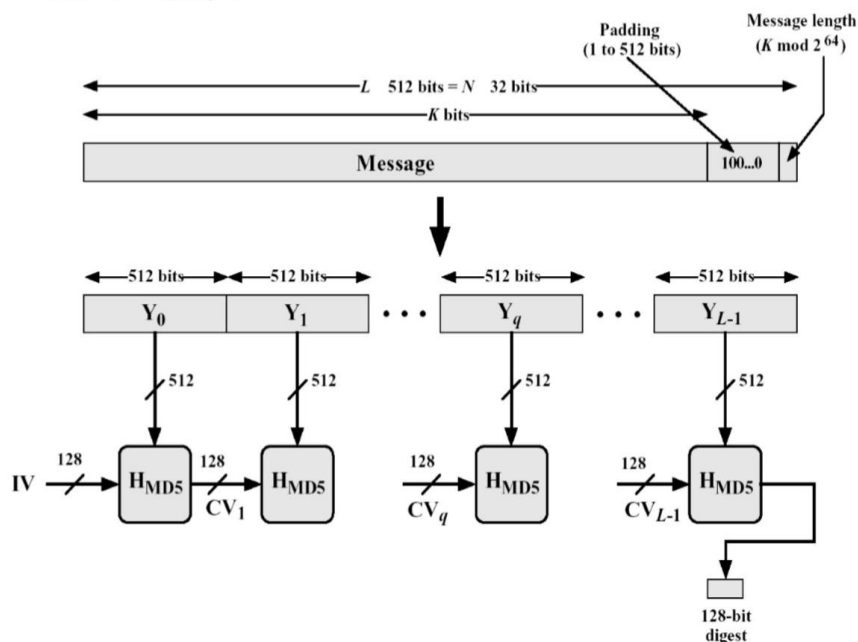
一. MD5 简介

MD5 消息摘要算法(MD5 Message-Digest-Algorithm)，是一种被广泛使用的密码散列函数，可以产生出一个 128 位(16 字节)的散列值(hash value)，用于确保信息传输完整一致。其典型应用就是对一段信息产生信息摘要，以防止被篡改。MD5 算法以任意长度的信息作为输入进行计算，产生一个 128bits 的报文摘要，而两个不同的信息所产生的报文摘要是不相同的。因此，MD5 常被用于进行数字签名、消息的完整性检测、消息源认证、操作系统的登录认证上等。它的算法是不可逆的，即从原数据转化为信息摘要后，不能从信息摘要中得到原数据；此算法产生碰撞的概率很低，即数据经过此算法后，基本上得到的信息摘要不同。

MD5 的主要应用有：第一，产生数字签名，即用 MD5 对一份数据处理，产生的信息摘要——签名，然后发送者将数据和签名一并交给接收者后，接收者可以再对数据进行 MD5 算法的处理，比对产生的签名和接收到的签名，以达到数据的完整性的判断；第二，作为密码保护，用登录网站账号的过程作为例子，首先用户输入完帐号和密码后，在发送至服务端后，会对用户密码进行 MD5 算法的处理，产生信息摘要，然后服务器再匹配数据库中帐号和密码。换言之服务器端所存储的用户密码其实也是用户原密码经过 MD5 处理过后的。所以用 MD5 作密码保护后，对管理员而言，其密码也是保密的。不仅在网站上，在系统中也会用到 MD5 作为密码保护。

二. MD5 算法原理及流程

• MD5 的基本流程



MD5 以 512 位分组来处理输入的信息，且每一分组又被划分为 16 个 32 位子分组，经过了一系列的处理后，算法的输出由四个 32 位分组组成，将这四个 32 位分组级联后将生成一个 128 位散列值。

具体过程如下：

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。填充方法为在信息的后面填充一个 1 和数个 0，直到满足条件。经以上处理，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。

2. 添加原始信息长度

在结果后面附加一个以 64 位二进制表示填充前信息的长度(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。

经以上处理，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。

3. 初始化变量

信息已被分为 512bits 一组，每次运算都由前一轮的 128 位结果和第 i 块的 512bits 的值进行运算。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 $0x67452301$ ， $0xEFCDAB89$ ， $0x98BADCFE$ ， $0x10325476$)

4. 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）。

$F(X,Y,Z)=(X \& Y) | ((\sim X) \& Z)$

$G(X,Y,Z)=(X \& Z) | (Y \& (\sim Z))$

$H(X,Y,Z)=X \wedge Y \wedge Z$

$I(X,Y,Z)=Y \wedge (X | (\sim Z))$

(&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 M_j 表示消息的第 j 个子分组（从 0 到 15），常数 t_i 是 $4294967296 * \text{abs}((\sin(i)))$ 的整数部分，i 取值从 1 到 64，单位是弧度($4294967296=2^{32}$)

现定义：

FF(a ,b ,c ,d ,Mj ,s ,ti) 操作为 $a = b + ((a + F(b,c,d) + Mj + ti) \ll s)$
GG(a ,b ,c ,d ,Mj ,s ,ti) 操作为 $a = b + ((a + G(b,c,d) + Mj + ti) \ll s)$
HH(a ,b ,c ,d ,Mj ,s ,ti) 操作为 $a = b + ((a + H(b,c,d) + Mj + ti) \ll s)$
II(a ,b ,c ,d ,Mj ,s ,ti) 操作为 $a = b + ((a + I(b,c,d) + Mj + ti) \ll s)$
注意：“ \ll ”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 ,0xbebfbcb70 )
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

```

ll(a ,b ,c ,d ,M0 ,6 ,0xf4292244 )
ll(d ,a ,b ,c ,M7 ,10 ,0x432aff97 )
ll(c ,d ,a ,b ,M14 ,15 ,0xab9423a7 )
ll(b ,c ,d ,a ,M5 ,21 ,0xfc93a039 )
ll(a ,b ,c ,d ,M12 ,6 ,0x655b59c3 )
ll(d ,a ,b ,c ,M3 ,10 ,0x8f0ccc92 )
ll(c ,d ,a ,b ,M10 ,15 ,0xffeff47d )
ll(b ,c ,d ,a ,M1 ,21 ,0x85845dd1 )
ll(a ,b ,c ,d ,M8 ,6 ,0x6fa87e4f )
ll(d ,a ,b ,c ,M15 ,10 ,0xfe2ce6e0 )
ll(c ,d ,a ,b ,M6 ,15 ,0xa3014314 )
ll(b ,c ,d ,a ,M13 ,21 ,0x4e0811a1 )
ll(a ,b ,c ,d ,M4 ,6 ,0xf7537e82 )
ll(d ,a ,b ,c ,M11 ,10 ,0xbd3af235 )
ll(c ,d ,a ,b ,M2 ,15 ,0x2ad7d2bb )
ll(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。

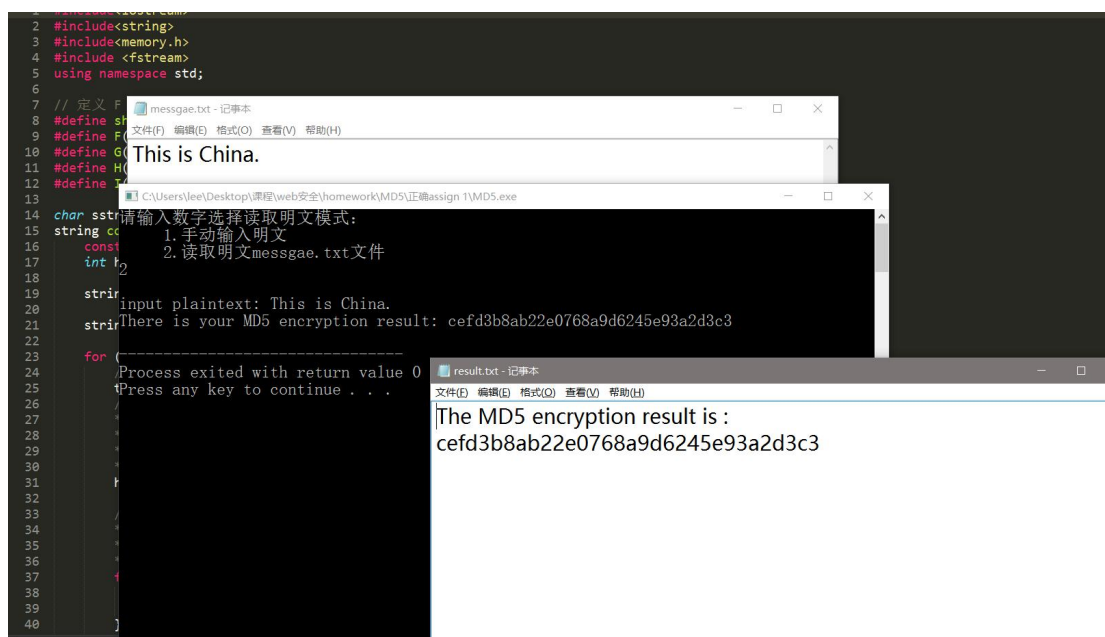
5.输出

最后的输出是 a、b、c 和 d 的级联。(a+b+c+d)

三、加密过程实现效果

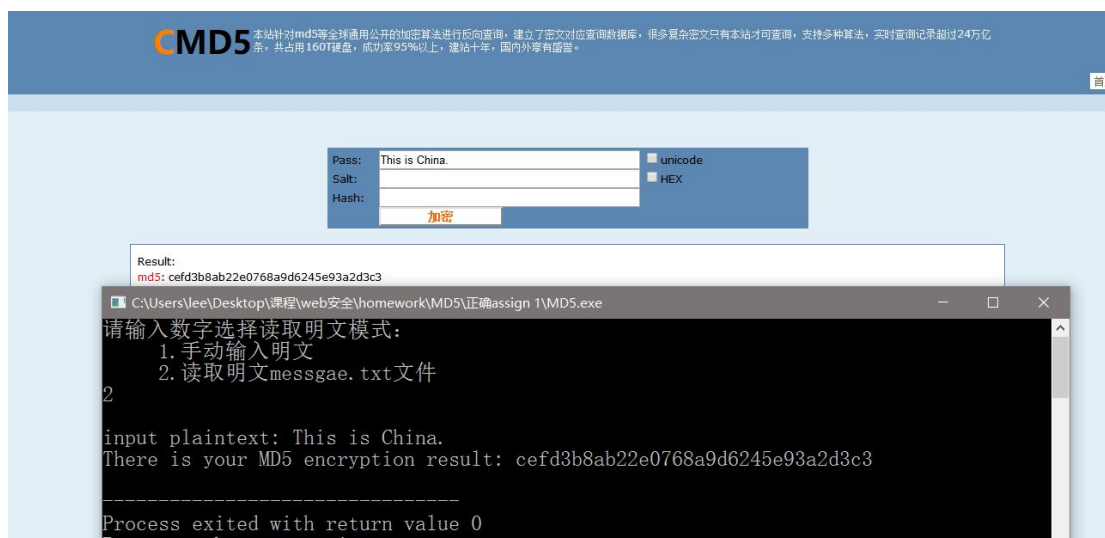
1.首先是运行的初始界面，可以选择两种方式之一进行信息加密，1 是手动输入明文进行加密，2 是读取 message.txt 中的明文。

3. 读取 message.txt 中的明文进行加密并将结果输出到 result.txt 中。



```
1 // 定义 F
2 #include<string>
3 #include<memory.h>
4 #include <fstream>
5 using namespace std;
6
7 // 定义 F
8 #define str
9 #define F
10 #define G
11 #define H
12 #define I
13
14 char sstr;
15 string cc;
16 const
17 int t2;
18
19 string input plaintext: This is China.
20 string There is your MD5 encryption result: cefd3b8ab22e0768a9d6245e93a2d3c3
21
22
23
24 Process exited with return value 0
25 Press any key to continue . . .
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
```

4. 这是加密结果与在线加密标准结果的对比，加密结果正确。



四、实验后感

作为一个安全的摘要算法在设计时必须满足两个要求——其一是寻找两个输入得到相同的输出值在计算上是不可行的，这就是我们通常所说的抗碰撞的；其二是找一个输出，能得到给定的输入在计算上是不可行的，即不可从结果推导出它的初始状态——MD5 已经无法满足第一个条件，也就是说 MD5 已经不安全了。但是碰撞概率很低，而且也没有特别高效的找出碰撞的算法，所以对其的破解，据网上搜索的结果，主流的都不是针对算法本身，而是针对算法的使用方式。比如，对未经加密的一个密码库运用 MD5 产生对应的 MD5 密码库 a，然后如果能够查询到网站服务器上的经过 MD5 处理的密码库数据 b，那么就可以用 b 和 a 中数据直接进行比对，找到 a 中对应 MD5 密码后，对应回其原密码就好了。其关键在于首先 a 要足够的大，可能包含到所需的密码，第二是要得到 b 中的数据。网络上有一些 MD5 查询破解网站，其就是有这样的一个密码库 a。提交一串 MD5 后，如果其数据库中刚好也有这个数据，即可破解成功。我觉得 MD5 如今在日常运用中还是足够安全的。