

C语言实现MD5加密，竟如此简单！

编程学习基地 昨天

以下文章来源于一口Linux，作者土豆居士



一口Linux

15年嵌入式开发经验古董级老鸟。曾任职中兴通讯，某研究所，华清远见教学总监。Linux...



点击蓝字 ··· 关注我们

本文详细讲解视频已经上传到B站：

<https://www.bilibili.com/video/BV1uy4y1p7on/>

公众号后台回复【md5】即可获得本文所有源码。

一、摘要算法

摘要算法又称哈希算法。

它表示输入任意长度的数据，输出固定长度的数据，它的主要特征是加密过程不需要密钥，并且经过加密的数据无法被解密。

目前可以被解密逆向的只有CRC32算法，只有输入相同的明文数据经过相同的消息摘要算法才能得到相同的密文。

消息摘要算法不存在密钥的管理与分发问题，适合于分布式网络上使用。由于其加密计算的工作量相当巨大，所以以前的这种算法通常只用于数据量有限的情况下的加密。

消息摘要算法分为三类：

- MD(Message Digest)：消息摘要
- SHA(Secure Hash Algorithm)：安全散列

- MAC(Message Authentication Code): 消息认证码

这三类算法的主要作用：**验证数据的完整性**

二、MD5简介

MD5即Message-Digest Algorithm 5（信息-摘要算法）。

属于摘要算法，是一个不可逆过程，就是无论多大数据，经过算法运算后都是生成固定长度的数据，结果使用16进制进行显示的128bit的二进制串。通常表示为32个十六进制数连成的字符串。

MD5有什么用？

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。更多用在文档校验上，用来生成密钥检测文档是否被篡改。

三、在线MD5加密

有很多在线进行MD5加密的网站，如下：

<http://www.metools.info/code/c26.html>

举例: 给字符串 **12334567** 加密成。

在线加密解密 DES加密解密 3DES加密解密 AES加密解密 SHA1加密 MD5加密 HMAC计算

转换前：


12334567

MD5加密(32位)> MD5加密(16位)> ☒ 大写

转换后：

32135A337F8DC8E2BB9A9B80D86BDFD0


如图结果为：



```
32135A337F8DC8E2BB9A9B80D86BDFD0
```

四、C语言实现MD5算法

源文件如下：md5.h



```
#ifndef MD5_H
#define MD5_H

typedef struct
{
    unsigned int count[2];
    unsigned int state[4];
    unsigned char buffer[64];
}MD5_CTX;

#define F(x,y,z) ((x & y) | (~x & z))
#define G(x,y,z) ((x & z) | (y & ~z))
#define H(x,y,z) (x^y^z)
#define I(x,y,z) (y ^ (x | ~z))
#define ROTATE_LEFT(x,n) ((x << n) | (x >> (32-n)))
#define FF(a,b,c,d,x,s,ac) \
    { \
        a += F(b,c,d) + x + ac; \
        a = ROTATE_LEFT(a,s); \
        a += b; \
    }
#define GG(a,b,c,d,x,s,ac) \
    { \
        a += G(b,c,d) + x + ac; \
        a = ROTATE_LEFT(a,s); \
        a += b; \
    }
#define HH(a,b,c,d,x,s,ac) \
```

```

    { \
    a += H(b,c,d) + x + ac; \
    a = ROTATE_LEFT(a,s); \
    a += b; \
    }

#define II(a,b,c,d,x,s,ac) \
    { \
    a += I(b,c,d) + x + ac; \
    a = ROTATE_LEFT(a,s); \
    a += b; \
    }

void MD5Init(MD5_CTX *context);

void MD5Update(MD5_CTX *context,unsigned char *input,unsigned int inputlen);

void MD5Final(MD5_CTX *context,unsigned char digest[16]);

void MD5Transform(unsigned int state[4],unsigned char block[64]);

void MD5Encode(unsigned char *output,unsigned int *input,unsigned int len);

void MD5Decode(unsigned int *output,unsigned char *input,unsigned int len);

#endif

```

md5.c

```

#include <memory.h>
#include "md5.h"

unsigned char PADDING[]={0x80,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
                        0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
                        0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
                        0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};

void MD5Init(MD5_CTX *context)
{
    context->count[0] = 0;
    context->count[1] = 0;
    context->state[0] = 0x67452301;
    context->state[1] = 0xEFCDAB89;
    context->state[2] = 0x98BADCFE;

```

```
    context->state[3] = 0x10325476;
}

void MD5Update(MD5_CTX *context,unsigned char *input,unsigned int inputlen)
{
    unsigned int i = 0,index = 0,partlen = 0;
    index = (context->count[0] >> 3) & 0x3F;
    partlen = 64 - index;
    context->count[0] += inputlen << 3;
    if(context->count[0] < (inputlen << 3))
        context->count[1]++;
    context->count[1] += inputlen >> 29;

    if(inputlen >= partlen)
    {
        memcpy(&context->buffer[index],input,partlen);
        MD5Transform(context->state,context->buffer);
        for(i = partlen;i+64 <= inputlen;i+=64)
            MD5Transform(context->state,&input[i]);
        index = 0;
    }
    else
    {
        i = 0;
    }
    memcpy(&context->buffer[index],&input[i],inputlen-i);
}

void MD5Final(MD5_CTX *context,unsigned char digest[16])
{
    unsigned int index = 0,padlen = 0;
    unsigned char bits[8];
    index = (context->count[0] >> 3) & 0x3F;
    padlen = (index < 56)?(56-index):(120-index);
    MD5Encode(bits,context->count,8);
    MD5Update(context,PADDING,padlen);
    MD5Update(context,bits,8);
    MD5Encode(digest,context->state,16);
}

void MD5Encode(unsigned char *output,unsigned int *input,unsigned int len)
{
    unsigned int i = 0,j = 0;
    while(j < len)
```

```

{
    output[j] = input[i] & 0xFF;
    output[j+1] = (input[i] >> 8) & 0xFF;
    output[j+2] = (input[i] >> 16) & 0xFF;
    output[j+3] = (input[i] >> 24) & 0xFF;
    i++;
    j+=4;
}
}

void MD5Decode(unsigned int *output,unsigned char *input,unsigned int len)
{
    unsigned int i = 0,j = 0;
    while(j < len)
    {
        output[i] = (input[j]) |
                    (input[j+1] << 8) |
                    (input[j+2] << 16) |
                    (input[j+3] << 24);

        i++;
        j+=4;
    }
}

void MD5Transform(unsigned int state[4],unsigned char block[64])
{
    unsigned int a = state[0];
    unsigned int b = state[1];
    unsigned int c = state[2];
    unsigned int d = state[3];
    unsigned int x[64];
    MD5Decode(x,block,64);

    FF(a, b, c, d, x[ 0], 7, 0xd76aa478); /* 1 */
    FF(d, a, b, c, x[ 1], 12, 0xe8c7b756); /* 2 */
    FF(c, d, a, b, x[ 2], 17, 0x242070db); /* 3 */
    FF(b, c, d, a, x[ 3], 22, 0xc1bdcee); /* 4 */
    FF(a, b, c, d, x[ 4], 7, 0xf57c0faf); /* 5 */
    FF(d, a, b, c, x[ 5], 12, 0x4787c62a); /* 6 */
    FF(c, d, a, b, x[ 6], 17, 0xa8304613); /* 7 */
    FF(b, c, d, a, x[ 7], 22, 0xfd469501); /* 8 */
    FF(a, b, c, d, x[ 8], 7, 0x698098d8); /* 9 */
    FF(d, a, b, c, x[ 9], 12, 0x8b44f7af); /* 10 */

```

```
FF(c, d, a, b, x[10], 17, 0xffff5bb1); /* 11 */
FF(b, c, d, a, x[11], 22, 0x895cd7be); /* 12 */
FF(a, b, c, d, x[12], 7, 0x6b901122); /* 13 */
FF(d, a, b, c, x[13], 12, 0xfd987193); /* 14 */
FF(c, d, a, b, x[14], 17, 0xa679438e); /* 15 */
FF(b, c, d, a, x[15], 22, 0x49b40821); /* 16 */

/* Round 2 */
GG(a, b, c, d, x[ 1], 5, 0xf61e2562); /* 17 */
GG(d, a, b, c, x[ 6], 9, 0xc040b340); /* 18 */
GG(c, d, a, b, x[11], 14, 0x265e5a51); /* 19 */
GG(b, c, d, a, x[ 0], 20, 0xe9b6c7aa); /* 20 */
GG(a, b, c, d, x[ 5], 5, 0xd62f105d); /* 21 */
GG(d, a, b, c, x[10], 9, 0x2441453); /* 22 */
GG(c, d, a, b, x[15], 14, 0xd8a1e681); /* 23 */
GG(b, c, d, a, x[ 4], 20, 0xe7d3fbc8); /* 24 */
GG(a, b, c, d, x[ 9], 5, 0x21e1cde6); /* 25 */
GG(d, a, b, c, x[14], 9, 0xc33707d6); /* 26 */
GG(c, d, a, b, x[ 3], 14, 0xf4d50d87); /* 27 */
GG(b, c, d, a, x[ 8], 20, 0x455a14ed); /* 28 */
GG(a, b, c, d, x[13], 5, 0xa9e3e905); /* 29 */
GG(d, a, b, c, x[ 2], 9, 0xfcefa3f8); /* 30 */
GG(c, d, a, b, x[ 7], 14, 0x676f02d9); /* 31 */
GG(b, c, d, a, x[12], 20, 0x8d2a4c8a); /* 32 */

/* Round 3 */
HH(a, b, c, d, x[ 5], 4, 0xfffa3942); /* 33 */
HH(d, a, b, c, x[ 8], 11, 0x8771f681); /* 34 */
HH(c, d, a, b, x[11], 16, 0x6d9d6122); /* 35 */
HH(b, c, d, a, x[14], 23, 0xfde5380c); /* 36 */
HH(a, b, c, d, x[ 1], 4, 0xa4beea44); /* 37 */
HH(d, a, b, c, x[ 4], 11, 0x4bdecfa9); /* 38 */
HH(c, d, a, b, x[ 7], 16, 0xf6bb4b60); /* 39 */
HH(b, c, d, a, x[10], 23, 0xbebfbcb70); /* 40 */
HH(a, b, c, d, x[13], 4, 0x289b7ec6); /* 41 */
HH(d, a, b, c, x[ 0], 11, 0xeeaa127fa); /* 42 */
HH(c, d, a, b, x[ 3], 16, 0xd4ef3085); /* 43 */
HH(b, c, d, a, x[ 6], 23, 0x4881d05); /* 44 */
```

```

HH(a, b, c, d, x[ 9], 4, 0xd9d4d039); /* 45 */
HH(d, a, b, c, x[12], 11, 0xe6db99e5); /* 46 */
HH(c, d, a, b, x[15], 16, 0x1fa27cf8); /* 47 */
HH(b, c, d, a, x[ 2], 23, 0xc4ac5665); /* 48 */

/* Round 4 */
II(a, b, c, d, x[ 0], 6, 0xf4292244); /* 49 */
II(d, a, b, c, x[ 7], 10, 0x432aff97); /* 50 */
II(c, d, a, b, x[14], 15, 0xab9423a7); /* 51 */
II(b, c, d, a, x[ 5], 21, 0xfc93a039); /* 52 */
II(a, b, c, d, x[12], 6, 0x655b59c3); /* 53 */
II(d, a, b, c, x[ 3], 10, 0x8f0ccc92); /* 54 */
II(c, d, a, b, x[10], 15, 0xffeff47d); /* 55 */
II(b, c, d, a, x[ 1], 21, 0x85845dd1); /* 56 */
II(a, b, c, d, x[ 8], 6, 0x6fa87e4f); /* 57 */
II(d, a, b, c, x[15], 10, 0xfe2ce6e0); /* 58 */
II(c, d, a, b, x[ 6], 15, 0xa3014314); /* 59 */
II(b, c, d, a, x[13], 21, 0x4e0811a1); /* 60 */
II(a, b, c, d, x[ 4], 6, 0xf7537e82); /* 61 */
II(d, a, b, c, x[11], 10, 0xbd3af235); /* 62 */
II(c, d, a, b, x[ 2], 15, 0x2ad7d2bb); /* 63 */
II(b, c, d, a, x[ 9], 21, 0xeb86d391); /* 64 */

    state[0] += a;
    state[1] += b;
    state[2] += c;
    state[3] += d;
}

```

五、MD5加密实例

MD5加密步骤如下：

1. 定义

```
MD5_CTX md5c;
```


2. 初始化

```

/*****
* 名    称: MD5Init()
* 功    能: 初始化MD5结构体
* 入口参数:
    context : 要初始化的MD5结构体
* 出口参数: 无
*****/
MD5Init(MD5_CTX *context);

```

3. MD5值计算

实现MD5值的计算及结构体的更新:

```

/*****
* 名    称: MD5Update()
* 功    能: 将要加密的信息传递给初始化过的MD5结构体·无返回值
* 入口参数:
    context : 初始化过的MD5结构体
    input   : 需要加密的信息·可以任意长度
    inputLen: 指定input的长度
* 出口参数: 无
*****/
MD5Update(MD5_CTX *context,(unsigned char *)input,inputLen);

```

4. 输出转换

```

/*****
* 名    称: MD5Update()
* 功    能: 将加密结果存储到·无返回值
* 入口参数:
    context : 初始化过的MD5结构体
    digest  : 加密过的结果
* 出口参数: 无

```

```
*****/  
MD5Final(MD5_CTX *context,unsigned char digest[16]);
```

5. 格式整理

转换成32位的16进制字符串。

实例1 字符串加密

对字符串进行加密：

```
1 #include <stdio.h>  
2 #include <stdlib.h>  
3 #include "md5.h"  
4 #include <sys/types.h>  
5 #include <sys/stat.h>  
6 #include <fcntl.h>  
7 #include <string.h>  
8  
9 void main( void )  
10 {  
11     int read_len;  
12     int i ;  
13     char temp[8]={0};  
14     unsigned char digest[16]; //存放结果  
15     char hexbuf[128]="12334567";  
16     unsigned char decrypt[16]={0};  
17     unsigned char decrypt32[64]={0};  
18  
19     MD5_CTX md5c;  
20  
21     MD5Init(&md5c); //初始化  
22     read_len = strlen(hexbuf);  
23     MD5Update(&md5c,(unsigned char *)hexbuf,read_len);  
24  
25     MD5Final(&md5c,decrypt);
```

```
26     strcpy((char *)decrypt32,"");
27
28     for(i=0;i<16;i++)
29     {
30         sprintf(temp,"%02x",decrypt[i]);
31         strcat((char *)decrypt32,temp);
32     }
33     printf("md5:%s\n",decrypt32);
34
35     return;
36 }
```

执行结果如下：

```
zh@ubuntu:~/yikou/MD5$ ./a.out
md5:32135a337f8dc8e2bb9a9b80d86bdfd0
```

本例对字符串**12334567**进行加密，结果和在线加密结果一致。

实例2 文件加密

对文件进行加密

```
#include <stdio.h>
#include <stdlib.h>
#include "md5.h"
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <string.h>

#define FORWORD_FW "123.c"

int calc_md5(char*filename,char*dest)
{
    int i;
    int filelen = 0;
```

```
int read_len;

char temp[8]={0};
char hexbuf[128]={0};
unsigned char decrypt[16]={0};
unsigned char decrypt32[64]={0};
MD5_CTX md5;
char fw_path[128];

int fdf;

fdf = open(filename,O_RDWR);
if(fdf<0)
{
    printf("%s not exist\n",FORWORD_FW);
    return -1;
}

MD5Init(&md5);
while(1)
{
    read_len = read(fdf, hexbuf,sizeof(hexbuf));

    if (read_len <0) {
        close(fdf);
        return -1;
    }

    if(read_len==0)
    {
        break;
    }
    filelen += read_len;
    MD5Update(&md5,(unsigned char *)hexbuf,read_len);
}

MD5Final(&md5,decrypt);
strcpy((char *)decrypt32,"");

for(i=0;i<16;i++)
{
    sprintf(temp,"%02x",decrypt[i]);
    strcat((char *)decrypt32,temp);
}

strcpy(dest,decrypt32);
```

```
printf("md5:%s len=%d\n",dest,filelen);
close(fdf);

return filelen;
}

int main(int argc, char *argv[])
{
    int ret;
    int filelen;
    char md5_str[64]={0};
    char cmd[256]={0};

    filelen = calc_md5(FORWORD_FW,md5_str);
    if(filelen<0)
    {
        printf("calc_md5 fail\n");
        return -1;
    }

    return 0;
}
```

运行结果：

```
zhh@ubuntu:~/yikou/MD5$ ./run
md5:b8fdaa19d9fad56810253ba652081e67 len=11
```

在线验证结果对比：

<http://www.metools.info/other/o21.html>

MD5加密

文件MD5计算

SHA1加密

文件SHA1/SHA256校验

选择文件

C:\fakepath\123.c

MD5计算

文件MD5值：

执行过程：

计算耗时：24ms

计算成功，MD5值：b8fdaa19d9fad56810253ba652081e67

加载数据：第1部分，总1部分

开始计算，文件名：(123.c)

结果

||||

.....END.....

你们的在看就是对我最大的肯定，
点个在看好吗~



编程学习基地
常回基地看看

[阅读原文](#)

喜欢此内容的人还喜欢

C语言函数执行成功时，返回1和返回0，究竟哪个好？

C语言编程



为什么MySQL不推荐使用子查询和join

我是程序汪

