《软件安全》第五章实验

姓名：何叶 学号:2313487 班级：范玲玲班

实验名称：shellcode编码解码及提取shellcode代码

一、实验要求

复现第五章实验三，并将产生的编码后的shellcode在示例5-4中进行验证，阐述shellcode编码的原理、shellcode提取的思想。

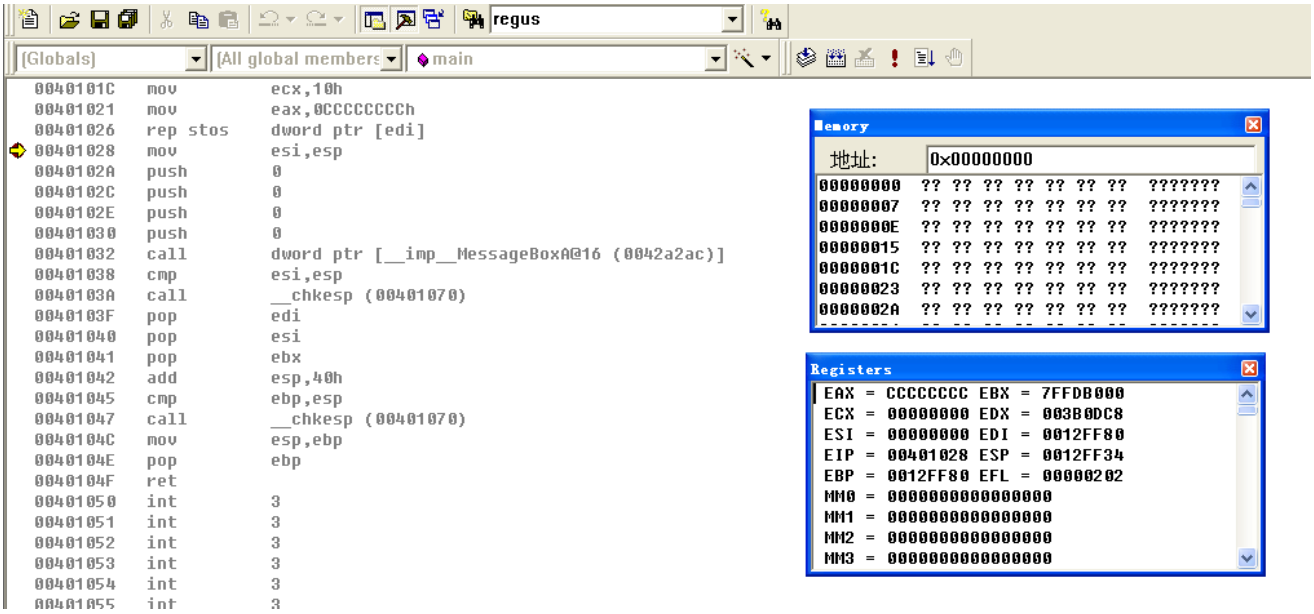
二、实验过程

1、提取

1.1先写C程序，打断点

```
#include <windows.h>
#include <stdio.h>
void main()
{
        MessageBox(NULL,NULL,NULL,0);

         return;
}
```

1.2反汇编获取代码



1.3通过_asm写汇编代码

```
#include <windows.h>
#include <stdio.h>
void main()
{
        MessageBox(NULL,NULL,NULL,0);
        _asm

        {

                xor ebx,ebx
                push ebx
                push ebx
                push ebx
                push ebx
                mov eax,77d507eah
                call eax

        }
         return;
}
```

1.4删掉MessageBox,再调试

```
#include <windows.h>
#include <stdio.h>
void main()
{
        _asm
        {
                xor ebx,ebx
                push ebx
                push ebx
                push ebx
                push ebx
                mov eax,77d507eah
                call eax

        }
         return;
}
```

```
"C:\Microsoft Visual Studio\MyProjects\shellcode\Debug\shellcode.exe"
Press any key to continue
```

1.5打断点。进入反汇编

```
[Globals]        [All global members]        main
00401021   mov        eax,0CCCCCCCCh
00401026   rep stos   dword ptr [edi]
00401028   xor        ebx,ebx
0040102A   push       ebx
0040102B   push       ebx
0040102C   push       ebx
0040102D   push       ebx
0040102E   mov        eax,77D507EAh
00401033   call       eax
00401035   pop        edi
00401036   pop        esi
00401037   pop        ebx
00401038   add        esp,40h
0040103B   cmp        ebp,esp
0040103D   call       __chkesp (00401070)
00401042   mov        esp,ebp
00401044   pop        ebp
00401045   ret
00401046   int        3
00401047   int        3
00401048   int        3
```

1.6复制地址跳转00401028

```
00401021   mov        eax,0CCCCCCCCh
00401026   rep stos   dword ptr [edi]
00401028   xor        ebx,ebx
0040102A   push       ebx
0040102B   push       ebx
0040102C   push       ebx
0040102D   push       ebx
0040102E   mov        eax,77D507EAh
00401033   call       eax
00401035   pop        edi
00401036   pop        esi
00401037   pop        ebx
00401038   add        esp,40h
0040103B   cmp        ebp,esp
0040103D   call       __chkesp (00401070)
00401042   mov        esp,ebp
00401044   pop        ebp
00401045   ret
00401046   int        3
00401047   int        3
00401048   int        3
00401049   int        3
0040104A   int        3
0040104B   int        3
```

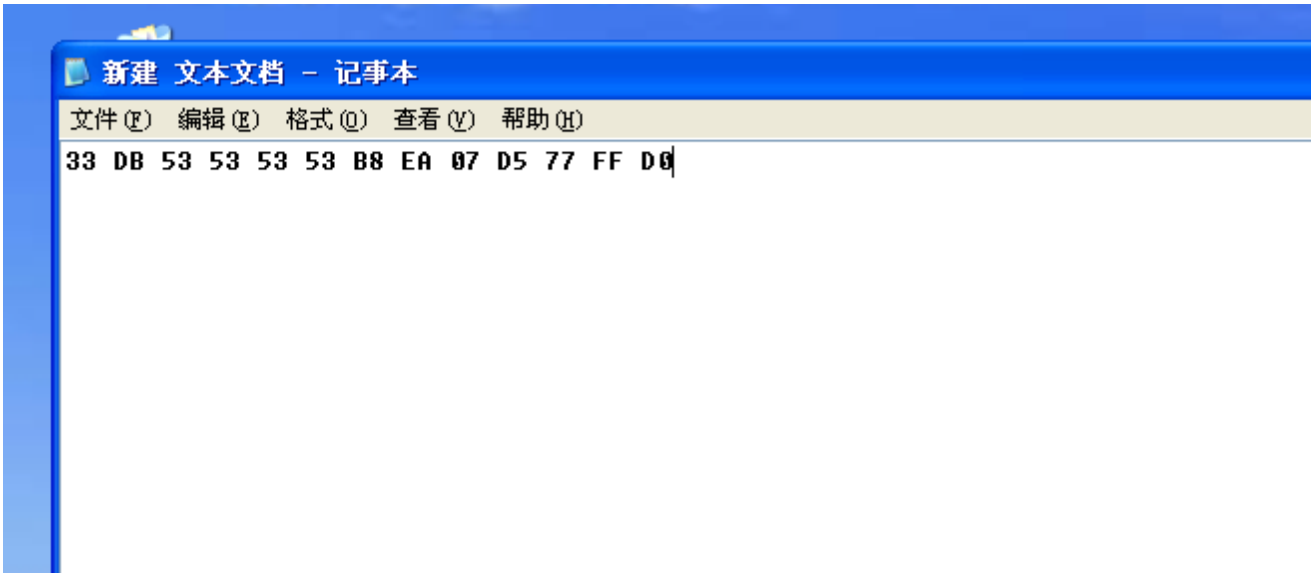
```
Memory
地址:  0x00401028
00401028  33 DB 53 53 53 53 B8  3箜SSS.
0040102F  EA 07 D5 77 FF D0 5F  ..諧.峈
00401036  5E 5B 83 C4 40 3B EC  ^[葩@;.
0040103D  E8 2E 00 00 00 8B E5  .....嬪
00401044  5D C3 CC CC CC CC CC  ]锰烫烫
0040104B  CC CC CC CC CC CC CC  烫烫烫.
00401052  CC CC CC CC CC CC CC  烫烫烫.
```

```
Registers
EAX = CCCCCCCC EBX = 7FFDD000
ECX = 00000000 EDX = 003B0DC8
ESI = 00000000 EDI = 0012FF80
EIP = 00401028 ESP = 0012FF34
EBP = 0012FF80 EFL = 00000202
MM0 = 0000000000000000
MM1 = 0000000000000000
MM2 = 0000000000000000
MM3 = 0000000000000000
```

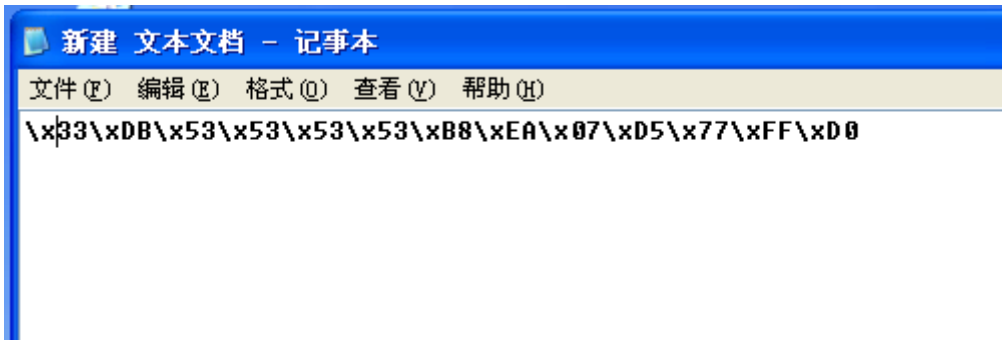
1.7找到所需代码

```
Memory
地址:  0x00401028
00401028  33 DB 53 53 53 53 B8 EA 07 D5 77 FF D0 5F  3箜SSS戈.諧.峈
00401036  5E 5B 83 C4 40 3B EC E8 2E 00 00 00 8B E5  ^[葩@;宣....嬪
00401044  5D C3 CC CC CC CC CC CC CC CC CC CC CC CC  ]锰烫烫烫烫烫.
00401052  CC CC CC CC CC CC CC CC CC CC CC CC CC CC  烫烫烫烫烫烫烫
00401060  FF 25 AC A2 42 00 CC CC CC CC CC CC CC CC  .%  B.烫烫烫烫
0040106E  CC CC 75 01 C3 55 8B EC 83 EC 00 50 52 53  烫u.肌蹙诚.PRS
0040107C  56 57 68 30 20 42 00 68 2C 20 42 00 6A 2A  VWh0 B.h, B.j*
```

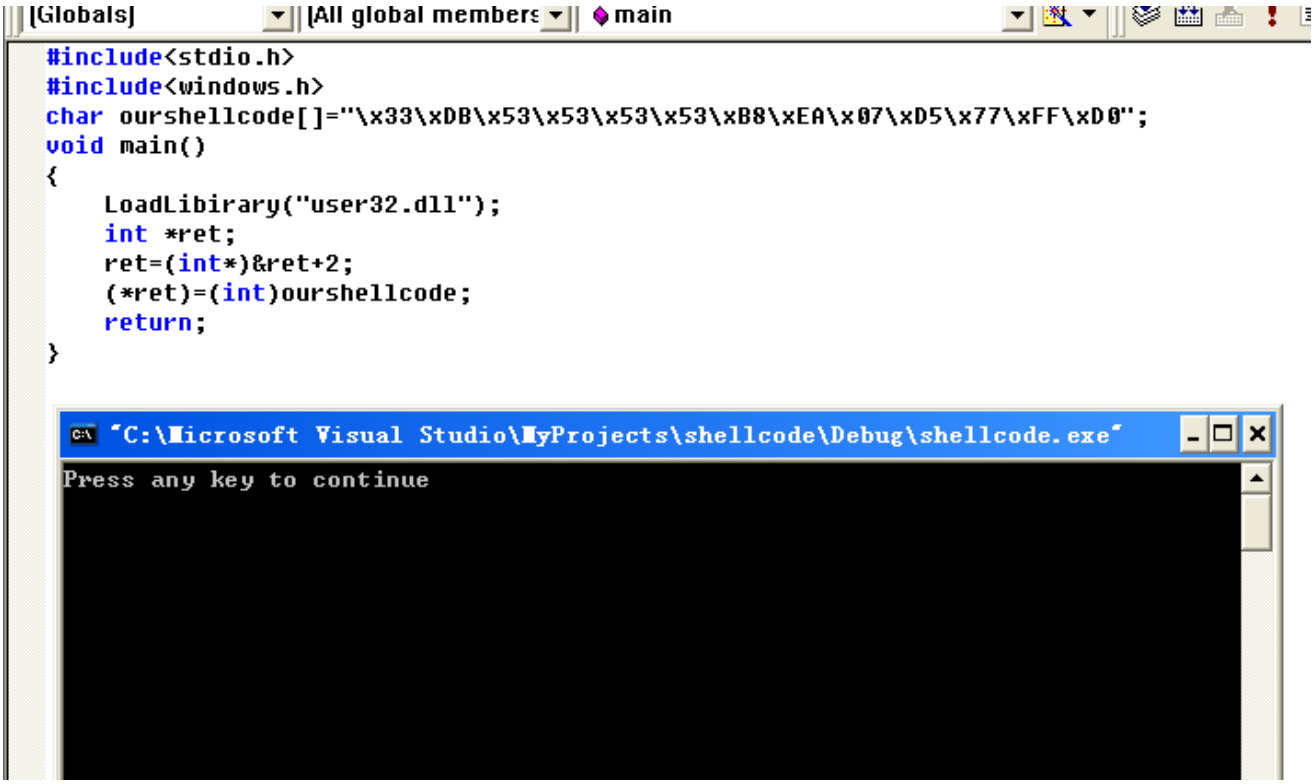
1.8新建文本储存



1.9空格替换为\x



1.10在示例5-4验证



2.编写

2.1编写hello world

```
#include <stdio.h>
#include <windows.h>
void main()
{
    LoadLibrary("user32.dll");
_asm
{
    xor ebx,ebx
    push ebx
    push 20646C72h
    push 6F77206Fh
    push 6C6C6568h
    mov eax, esp
    push ebx
    push eax
    push eax
    push ebx
    mov eax, 77d507eah
    call eax
}
return;
}
```

2.2进入反汇编

```
0040103C   xor     ebx,ebx
0040103E   push    ebx
0040103F   push    20646C72h
00401044   push    6F77206Fh
00401049   push    6C6C6568h
0040104E   mov     eax,esp
00401050   push    ebx
00401051   push    eax
00401052   push    eax
00401053   push    ebx
00401054   mov     eax,77D507EAh
00401059   call    eax
0040105B   pop     edi
0040105C   pop     esi
0040105D   pop     ebx
0040105E   add     esp,40h
00401061   cmp     ebp,esp
00401063   call    __chkesp (00401090)
00401068   mov     esp,ebp
0040106A   pop     ebp
0040106B   ret
0040106C   int     3
0040106D   int     3
0040106E   int     3
```

```
Memory
地址:  0x00401028
00401028  8B F4 68 1C 20 42 00 FF 15 48 A1 42 00 3B
00401036  F4 E8 54 00 00 00 33 DB 53 68 72 6C 64 20
00401044  68 6F 20 77 6F 68 68 65 6C 6C 8B C4 53 50
00401052  50 53 B8 EA 07 D5 77 FF D0 5F 5E 5B 83 C4
00401060  40 3B EC E8 28 00 00 00 8B E5 5D C3 CC CC
0040106E  CC CC CC CC CC CC CC CC CC CC CC CC CC CC
0040107C  CC CC CC CC CC CC CC CC FF 25 48 A1 42 00
```

```
Registers
EAX = 77D10000 EBX = 7FFDA000
ECX = 7C801BFA EDX = 00140608
ESI = 0012FF34 EDI = 0012FF80
EIP = 0040103C ESP = 0012FF34
EBP = 0012FF80 EFL = 00000246
MM0 = 0000000000000000
MM1 = 0000000000000000
MM2 = 0000000000000000
MM3 = 0000000000000000
```

2.3跳转至0040103C

```
Memory
地址:  0x0040103C
0040103C  33 DB 53 68 72 6C 64 20 68 6F 20 77 6F 68  3?hrld ho woh
0040104A  68 65 6C 6C 8B C4 53 50 50 53 B8 EA 07 D5  hell嬾SPPS戈..
00401058  77 FF D0 5F 5E 5B 83 C4 40 3B EC E8 28 00  w.峈^[瓸@;扈(.
00401066  00 00 8B E5 5D C3 CC CC CC CC CC CC CC CC  ..嬪]锰烫烫烫.
00401074  CC CC CC CC CC CC CC CC CC CC CC CC CC CC  烫烫烫烫烫烫烫烫
00401082  CC CC FF 25 48 A1 42 00 CC CC CC CC CC CC  烫.%H  .烫烫烫
00401090  75 01 C3 55 8B EC 83 EC 00 50 52 53 56 57  u.胱蘼波.PRSVW
```

```
Registers
EAX = 77D10000 EBX = 7FFDA000
ECX = 7C801BFA EDX = 00140608
ESI = 0012FF34 EDI = 0012FF80
EIP = 0040103C ESP = 0012FF34
EBP = 0012FF80 EFL = 00000246
MM0 = 0000000000000000
MM1 = 0000000000000000
MM2 = 0000000000000000
MM3 = 0000000000000000
```

2.4得到机器码

```
Memory
地址:  0x0040103C
00400FFC  00 00 00 00 CC CC CC CC CC E9 06 00 00 00 CC CC CC CC CC CC 55 8B EC 83 EC 40 53 56 57 8D 7D C0  ....
0040101C  B9 10 00 00 00 B8 CC CC CC CC F3 AB 8B F4 68 1C 20 42 00 FF 15 48 A1 42 00 3B F4 E8 54 00 00 00  ....
0040103C  33 DB 53 68 72 6C 64 20 68 6F 20 77 6F 68 68 65 6C 6C 8B C4 53 50 50 53 B8 EA 07 D5 77 FF D0 5F  3?h
0040105C  5E 5B 83 C4 40 3B EC E8 28 00 00 00 8B E5 5D C3 CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ^[瓸
0040107C  CC CC CC CC CC CC CC CC FF 25 48 A1 42 00 CC CC CC CC CC CC 75 01 C3 55 8B EC 83 EC 00 50 52 53  烫烫
0040109C  56 57 68 40 20 42 00 68 3C 20 42 00 6A 2A 68 2C 20 42 00 6A 01 E8 BA 02 00 00 83 C4 14 83 F8 01  VWh@
004010BC  75 01 CC 5F 5E 5B 5A 58 8B E5 5D C3 CC CC CC CC CC CC CC CC 55 8B EC 6A FF 68 20 21 42 00 68 E0  u.蕨
```

```
新建 文本文档 - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
\x33\xDB\x53\x68\x72\x6C\x64\x20\x68\x6F\x20\x77\x6F\x68\x68\x65\x6C\x6C\x8B\xC4\x53\x50\x50\x53\xB8\xEA\x07\xD5\x77\xFF\xD0
```

3.编码

3.1用xor编码

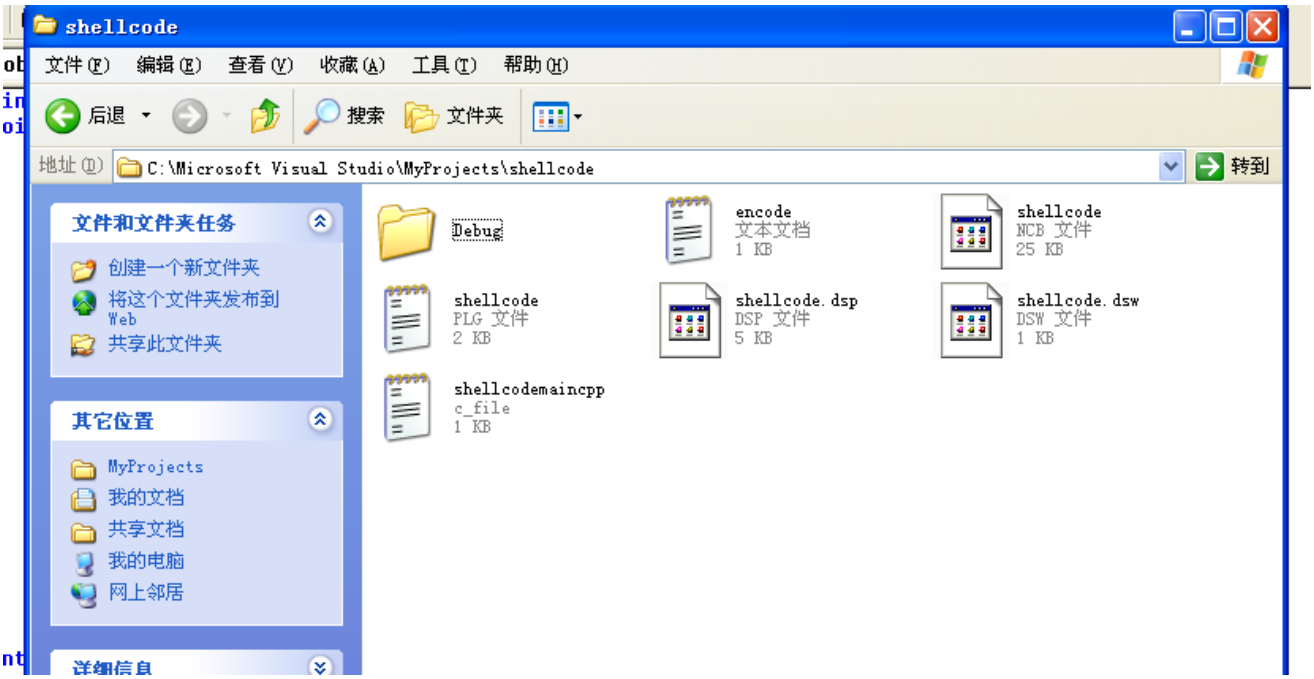
```
(Globals)  (All global members)  main

#include <stdio.h>
void encoder(char* input, unsigned char key)
{
    int i = 0, len = 0;
    FILE * fp;
    len = strlen(input);
    unsigned char * output = (unsigned char *)malloc(len + 1);
    for (i = 0; i<len; i++)
        output[i] = input[i] ^ key;
    fp = fopen("encode.txt", "w+");
    fprintf(fp, "\"");
    for (i = 0; i<len; i++)
    {
        fprintf(fp, "\\x%0.2x", output[i]);
        if ((i + 1) % 16 == 0)
            fprintf(fp, "\"\n\"");
    }
    fprintf(fp, "\"");
    fclose(fp);
    printf("dump the encoded shellcode to encode.txt OK!\n");
    free(output);
}
int main()
{
    char sc[] =         "\x33\xDB\x53\x68\x72\x6C\x64\x20\x68\x6F\x20\x77\x6F\x68\x68\x65\x6C\x6C\x8B\xC4\x53\x50\x50\x53\xB8\xEA\x07\xD5\x77\xFF\xD0\x90";
    encoder(sc, 0x44);
    getchar();
    return 0;
}
```

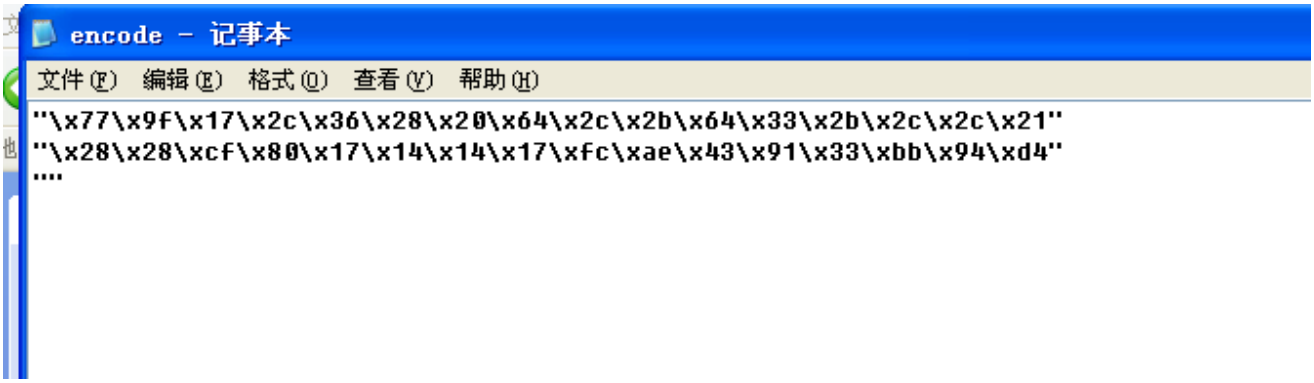
3.2运行后输出

```
#include <stdio.h>
void encoder(char* input, unsigned char key)

    int i = 0, len = 0;
    FILE * fp;
    len = strlen(input);
    unsigned char * output = (unsigned char *
    for (i = 0; i<len; i++)
        output[i] = input[i] ^ key;
    fp = fopen("encode.txt", "w+");
    fprintf(fp, "\"");
    for (i = 0; i<len; i++)
    {
        fprintf(fp, "\\x%0.2x", output[i]);
        if ((i + 1) % 16 == 0)
            fprintf(fp, "\"\n\"");
    }
    fprintf(fp, "\"");
    fclose(fp);
    printf("dump the encoded shellcode to enc
    free(output);

int main()

    char sc[] =          "\x33\xDB\x53\x68\x72
    encoder(sc, 0x44);
    getchar();
    return 0;
```

```
"C:\Microsoft Visual Studio\MyProjects\shellcode\Debug\shellcode.exe"
dump the encoded shellcode to encode.txt OK!
```

3.3生成encode文件





4.解码

4.1解码程序

```
(Globals)    (All global members    encoder

#include <stdlib.h>
#include <string.h>
#include <stdio.h>

int main()
{
    __asm
    {
        call lable;
     lable: pop eax;
        add eax, 0x15              ;越过decoder记录shellcode起始地址
        xor ecx, ecx
    decode_loop:
        mov bl, [eax + ecx]
        xor bl, 0x44               ;用0x44作为key
        mov [eax + ecx], bl
        inc ecx
        cmp bl, 0x90               ;末尾放一个0x90作为结束符
        jne decode_loop
    }

    return 0;
}
```

4.2反汇编看call lable

```
(Globals)    (All global members    main

00401038   call      lable (0040103d)
lable:
0040103D   pop       eax
0040103E   add       eax,15h
00401041   xor       ecx,ecx
decode_loop:
00401043   mov       bl,byte ptr [eax+ecx]
00401046   xor       bl,44h
00401049   mov       byte ptr [eax+ecx],bl
0040104C   inc       ecx
0040104D   cmp       bl,90h
00401050   jne       decode_loop (00401043)
00401052   xor       eax,eax
00401054   pop       edi
00401055   pop       esi
00401056   pop       ebx
00401057   add       esp,40h
0040105A   cmp       ebp,esp
0040105C   call      __chkesp (004035a0)
00401061   mov       esp,ebp
00401063   pop       ebp
00401064   ret
00401065   int       3
```

4.3运行lable得到地址

```
00401038   call        lable (0040103d)
lable:
0040103D   pop         eax
0040103E   add         eax,15h
00401041   xor         ecx,ecx
decode_loop:
00401043   mov         bl,byte ptr [eax+ecx]
00401046   xor         bl,44h
00401049   mov         byte ptr [eax+ecx],bl
0040104C   inc         ecx
0040104D   cmp         bl,90h
00401050   jne         decode_loop (00401043)
00401052   xor         eax,eax
00401054   pop         edi
00401055   pop         esi
00401056   pop         ebx
00401057   add         esp,40h
0040105A   cmp         ebp,esp
0040105C   call        __chkesp (004035a0)
00401061   mov         esp,ebp
00401063   pop         ebp
00401064   ret
00401065   int         3
00401066   int         3
```

```
Memory
地址:   0x0040103C
0040103C  00 58 83 C0 15 33 C9 8A 1C 08 80 F3 44 88
0040105C  E8 3F 25 00 00 8B E5 5D C3 CC CC CC CC CC
0040107C  CC CC CC CC CC CC CC CC CC CC CC CC CC CC
0040109C  CC CC CC CC CC CC CC CC CC CC CC CC CC CC
004010BC  CC CC CC CC CC CC CC CC CC CC CC CC CC CC
004010DC  CC CC CC CC CC CC CC CC CC CC CC CC CC CC
004010FC  CC CC CC CC CC CC CC CC CC CC CC CC CC CC
```

```
Registers
EAX = CCCCCCCC EBX = 7FFDF000
ECX = 00000000 EDX = 00380DC8
ESI = 00000000 EDI = 0012FF80
EIP = 0040103D ESP = 0012FF30
EBP = 0012FF80 EFL = 00000202
MM0 = 0000000000000000
MM1 = 0000000000000000
MM2 = 0000000000000000
MM3 = 0000000000000000
```

4.4提取机器码

```
Memory
地址:   0x00401028
00401008  00 00 CC CC CC CC CC CC 55 8B EC 83 EC 40 53 56 57 8D 7D C0 B9 10 00 00 00 B8 CC CC CC CC F3 AB
00401028  E8 00 00 00 00 58 83 C0 15 33 C9 8A 1C 08 80 F3 44 88 1C 08 41 80 FB 90 75 F1 33 C0 5F 5E 5B 83
00401048  C4 40 3B EC E8 1F 00 00 00 8B E5 5D C3 CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC
00401068  CC CC CC CC CC CC CC CC 75 01 C3 55 8B EC 83 EC 00 50 52 53 56 57 68 30 20 42 00 68 2C 20 42 00
00401088  6A 2A 68 1C 20 42 00 6A 01 E8 BA 02 00 00 83 C4 14 83 F8 01 75 01 CC 5F 5E 5B 5A 58 8B E5 5D C3
004010A8  CC CC CC CC CC CC CC CC 55 8B EC 6A FF 68 10 21 42 00 68 C0 2E 40 00 64 A1 00 00 00 00 50 64 89
004010C8  25 00 00 00 00 83 C4 F0 53 56 57 89 65 E8 FF 15 4C A1 42 00 A3 6C 7C 42 00 A1 6C 7C 42 00 C1 E8
```

```
Registers
```

4.5加encode组成完整shellcode

encode - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
"\x77\x9f\x17\x2c\x36\x28\x20\x64\x2c\x2b\x64\x33\x2b\x2c\x2c\x21"
"\x28\x28\xcf\x80\x17\x14\x14\x17\xfc\xae\x43\x91\x33\xbb\x94\xd4"
""
```

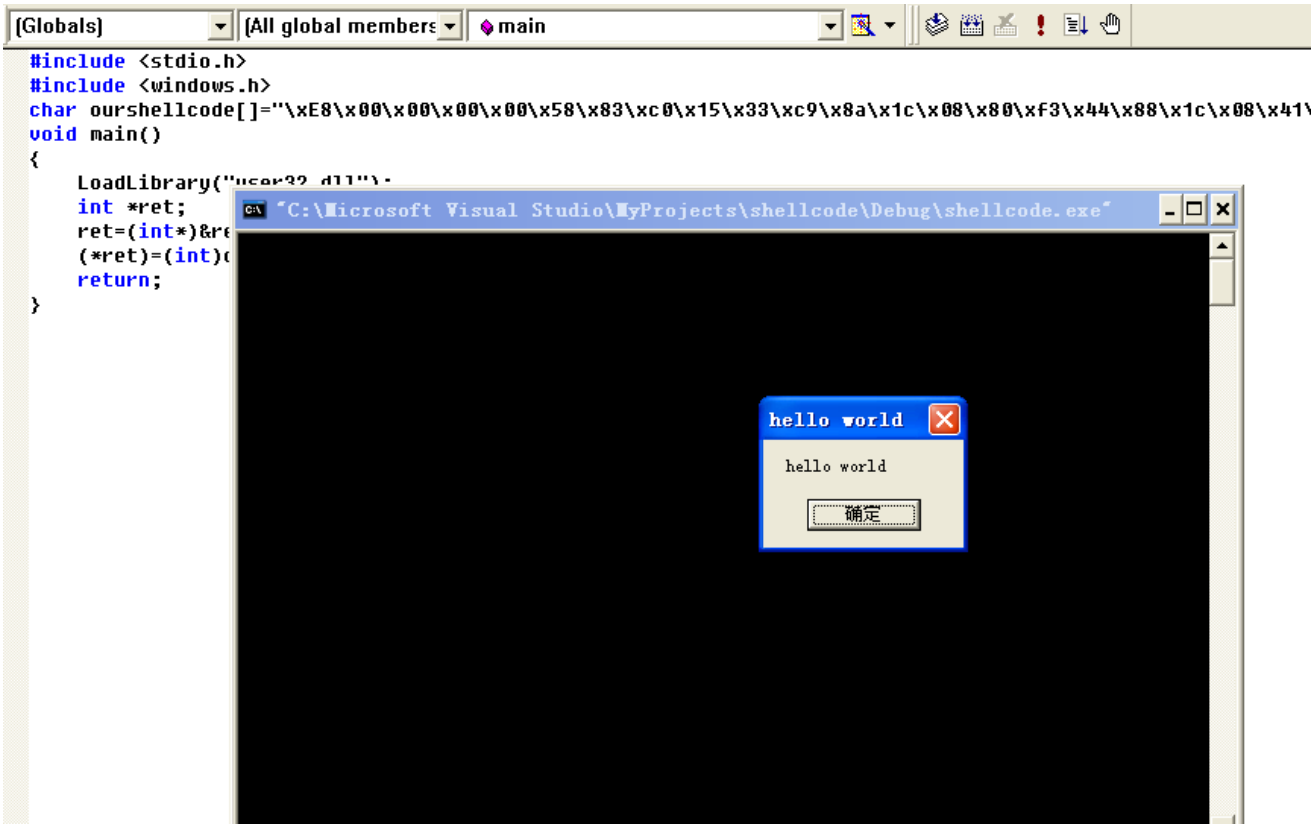
4.6放入万能验证程序验证

regus

[Globals] [All global members] main

```
#include <stdio.h>
#include <windows.h>
char ourshellcode[]="\xE8\x00\x00\x00\x00\x58\x83\xc0\x15\x33\xc9\x8a\x1c\x08\x80\xf3\x44\x88\x1c\x08\x41\x80\xfb\x90\x75\xf1\x77\x9f\x
void main()
{
    LoadLibrary("user32.dll");
    int *ret;
    ret=(int*)&ret+2;
    (*ret)=(int)ourshellcode;
    return;
}
```

4.6验证成功



三、心得体会

1.Shellcode编码原理

Shellcode编码是一种将恶意代码转换成另一种形式以规避安全机制的技术。其原理是通过改变代码的字节序列，使其在功能不变的情况下，外观与原始代码不同，从而绕过入侵检测系统。常见的编码技术包括十六进制、八进制、Base64编码，或利用特殊字符和转义序列。

2.Shellcode提取的思想

Shellcode提取是从编码数据中恢复原始shellcode的过程。这通常涉及字符串解析、模式匹配和解码算法，需要识别并解码编码技术，将shellcode还原成可读形式。在网络安全领域，Shellcode编码和提取技术使攻击者能更隐蔽地执行恶意操作，但也推动了安全社区开发更先进的检测和防御技术。