# 逆向分析技术

## 实验三注册码分析

实验内容：

请各位同学利用OllyDbg，参照课程中的示例对EXP3.exe中的注册码进行分析。

实验要求：

请各位同学通过QQ作业于4月12日前在线提交如下内容：

描述用OllyDbg对EXP3.exe进行分析的过程，回答EXP3中注册码的每一位应当是什么，并解释原因。（不用截很多图，例如对指令的解释用文字描述即可）

1. 拖入PEID中查壳分析，无壳，直接用ollydbg打开程序exp3, F9运行

2.右键，查找，所有参考文本字串，发现0040143D处有ASCII "Serial is Correct!!!"

3. 再往上翻，找到

000013E7 83F8 08 cmp eax,0x8

000013EA 75 5D jnz short 00001449

cmp指令为第一个操作减去第二个操作数,但不影响第两个操作数的值,它影响flag的CF，ZF，OF，AF，PF若执行指令后:ZF=1,则说明两个数相等，因为zero为1说明结果为0。jnz：ZF不等于0则跳转。

所以意思是把eax寄存器中的值与8做比较，如果不等于8则跳转到00001449。推测出要比较注册码长度是否为8位。

为了验证，我们在接下来的004013EC处打上断点。输入8位注册码，停在了断点处。输入非八位注册码，在断点处没有停止，验证了我们的猜想。

004013EC . 807C24 09 2D cmp byte ptr ss:[esp+0x9],0x2D

004013F1 . 75 56 jnz short EXP3.00401449

4.通过观察，我们发现接下来是一系列 cmp，jnz比较跳转。计算机每一个地址代表一个字节，总共比较从esp+0x4（低地址）到esp+0xB（高地址）这8个字节的内容。而我们计算机的小端模式，是指数据的高位保存在内存的高地址中，而数据的低位保存在内存的低地址中。所以esp+0x4保存注册码的第一个字节；esp+0xB保存注册码的第八个字节。

接下来具体分析8个字节的判断。

5．之前比较完注册码位数后，如果是8位，则下一段指令是

004013EC . 807C24 09 2D cmp byte ptr ss:[esp+0x9],0x2D

004013F1 . 75 56 jnz short EXP3.00401449

注释：将esp+9 （第六位）与2D（十进制45）比较，不同跳转，所以第六位即45，作为ascill码对应的字符也就是“-”

6. 下一段指令是

004013F3 . 0FBE4C24 04 movsx ecx,byte ptr ss:[esp+0x4]

004013F8 . D1E1 shl ecx,1

004013FA . 83F9 64 cmp ecx,0x64

004013FD . 75 4A jnz short EXP3.00401449

注释：movsx:带符号扩展。将esp+0x4作为有符号数存入ecx。

 ecx左移一位与64做比较，如果不等于64则跳转。推出第一位是32，作为ascill码对应的字符也就是“2”

7. 004013FF . 8A4424 0B mov al,byte ptr ss:[esp+0xB]

00401403 . 84C0 test al,al

00401405 . /74 42 je short EXP3.00401449

注释：把esp+0xB（第八位）存入al。

Test命令将两个操作数进行逻辑与运算，al&&al，je是 当运算结果为0时则跳转。所以al不为0即可。

8.

00401407 . 807C24 08 2B cmp byte ptr ss:[esp+0x8],0x2B

0040140C . 75 3B jnz short EXP3.00401449

注释：esp+8 （第五位）与2B（43）作比较，不同则跳转，所以第五位即43，也就是“+”

9.

0040140E . 0FBE5424 05 movsx edx,byte ptr ss:[esp+0x5]

00401413 . 83C2 0A add edx,0xA

00401416 . 83FA 44 cmp edx,0x44

00401419 . 75 2E jnz short EXP3.00401449

注释：movsx:带符号扩展，将esp+0x5（第二位）作为有符号数存入edx，edx加上10，edx与0x44（68）比较，不等于则跳转。所以第二位为58，对应字符“：”

10.

0040141B . 0FBE4424 07 movsx eax,byte ptr ss:[esp+0x7]

00401420 . 83E8 2E sub eax,0x2E

00401423 . 75 24 jnz short EXP3.00401449

注释：movsx:带符号扩展，将esp+0x7（第4位）作为有符号数存入eax，eax减去46，eax与0作比较，不同则跳转，所以第四位就是46，即“.”

11． 00401425 . 807C24 0A 4D cmp byte ptr ss:[esp+0xA],0x4D

0040142A . 75 1D jnz short EXP3.00401449

注释： esp+A （第七位）与4D（77）作比较，不同则跳转，所以第五位即77，也就是“M”

12.

0040142C . 0FBE4C24 06 movsx ecx,byte ptr ss:[esp+0x6]

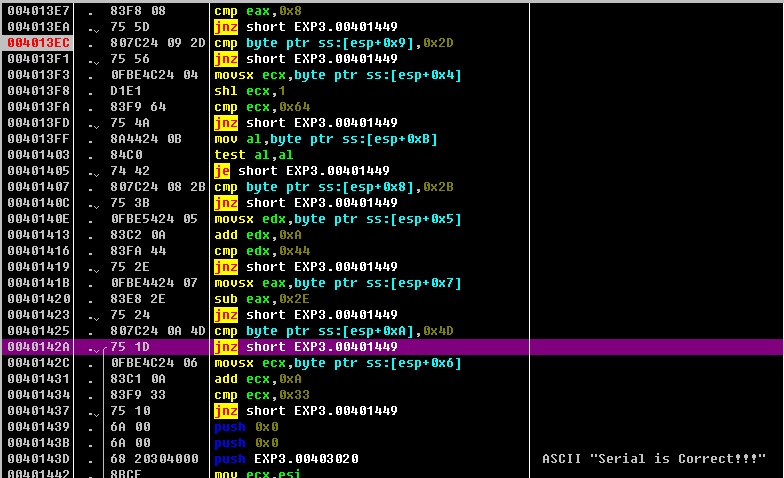
00401431 . 83C1 0A add ecx,0xA

00401434 . 83F9 33 cmp ecx,0x33

00401437 . 75 10 jnz short EXP3.00401449

注释：movsx:带符号扩展，将esp+0x6（第三位）作为有符号数存入ecx，ecx加上10，ecx与0x33（51）作比较，不同则跳转，所以第四位就是41，即“)”

综上，注册码为2:).+-M0



相关代码段

004013E7 . 83F8 08 cmp eax,0x8

004013EA . 75 5D jnz short EXP3.00401449

004013EC . 807C24 09 2D cmp byte ptr ss:[esp+0x9],0x2D

004013F1 . 75 56 jnz short EXP3.00401449

004013F3 . 0FBE4C24 04 movsx ecx,byte ptr ss:[esp+0x4]

004013F8 . D1E1 shl ecx,1

004013FA . 83F9 64 cmp ecx,0x64

004013FD . 75 4A jnz short EXP3.00401449

004013FF . 8A4424 0B mov al,byte ptr ss:[esp+0xB]

00401403 . 84C0 test al,al

00401405 . 74 42 je short EXP3.00401449

00401407 . 807C24 08 2B cmp byte ptr ss:[esp+0x8],0x2B

0040140C . 75 3B jnz short EXP3.00401449

0040140E . 0FBE5424 05 movsx edx,byte ptr ss:[esp+0x5]

00401413 . 83C2 0A add edx,0xA

00401416 . 83FA 44 cmp edx,0x44

00401419 . 75 2E jnz short EXP3.00401449

0040141B . 0FBE4424 07 movsx eax,byte ptr ss:[esp+0x7]

00401420 . 83E8 2E sub eax,0x2E

00401423 . 75 24 jnz short EXP3.00401449

00401425 . 807C24 0A 4D cmp byte ptr ss:[esp+0xA],0x4D

0040142A . 75 1D jnz short EXP3.00401449

0040142C . 0FBE4C24 06 movsx ecx,byte ptr ss:[esp+0x6]

00401431 . 83C1 0A add ecx,0xA

00401434 . 83F9 33 cmp ecx,0x33

00401437 . 75 10 jnz short EXP3.00401449

00401439 . 6A 00 push 0x0

0040143B . 6A 00 push 0x0

0040143D . 68 20304000 push EXP3.00403020 ; ASCII "Serial is Correct!!!"

00401442 . 8BCE mov ecx,esi

00401444 . E8 0D020000 call <jmp.&MFC42.#?MessageBoxA@CWnd@@QAE>

00401449 > 5E pop esi