# 第1章 基础知识

在Windows NT/2000/XP中，Win16函数调用通过一个转换层被转化为Win32函数调用，然后被操作系统处理。在Windows 9x中，该操作正好相反：Win32函数调用通过转换层转换为Win32调用，再由操作系统处理。

常用Win32 API：GetWindowText、GetDlgItem、GetDlgItemText、GetDlgItemInt、MessageBoxEx 、MessageBox。

在Windows 9x核心是使用ANSI字符串来进行所有操作的，而Windows NT/2000/XP是使用UNICODE从头进行开发的，其系统核心完全是用UNICODE函数工作的。在Windows 9x上，使用UNICODE版本的函数如MessageBoxW，其内部会用WideCharToMultiByte()把UNICODE字符串转换成ANSI，再调用MessageBoxExA()。同样，在Windows NT/2000/XP上使用MessageBoxA，其内部会用MBToWCSEx() 把ANSI字符串转换成UNICODE字符串，再调用MessageBoxExW()。

因此在9x上用ANSI版本的API效率高，在XP上用Unicode版本的API效率高。

常用Windows 消息：WM\_COMMAND、WM\_DESTROY、WM\_GETTEXT、WM\_LBUTTONDOWN、WM\_LBUTTONUP、WM\_QUIT。

用Win32 API操作注册表的基本步骤如下：

（1）用RegOpenKeyEx()打开想要操作的主键，获得一个句柄。

（2）将句柄传递给RegQueryValueEx()和RegSetValueEx()来读写相应的键值。

（3）操作完毕之后用RegCloseKey()关闭先前获得的句柄。

Win32提供了RegOpenKey、RegQueryValue、RegSetValue仅仅是为了兼容16位版本的Windows，应用程序应该使用Ex函数。

# 第2章 代码分析技术

Entry Point

File Offset RAW Offset

Virtual Address Memory Offset

保护模式下虚拟地址的形式为：段选择子：偏移地址，例如，“0167:00401000”就是这种表示方式其中：

0167：这是段选择子，其数据保存在CS段选择器里。同一程序在不同系统下，此值可能不同，一般也不需要关心此值。

00401000：此处表示内存的虚拟地址，一般来说，同一程序的同一条指令在不同系统环境下，此值相同。

ImageBase

文件执行时将被加载到内存的首地址，这个值是由PE文件本身设定的。按照默认设置，用Visual C++建立的EXE文件基地址是0x00400000，DLL文件基地址是0x10000000。可以在创建PE文件时改变这个地址，方法是在链接是使用链接器的/BASE选项。

Relative Virtual Address = VA - ImageBase

表2-2 转移指令的条件与机器码

表2-3 条件设置指令

表2-4 常用指令修改技巧

浮点指令

# 第3章 静态分析技术