初识exe程序反汇编小感

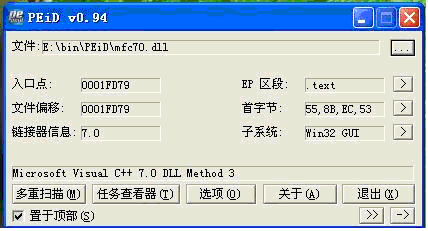
近期由于一个项目的需要，我初步接触了exe程序反汇编。折腾了好几天终于见到了那个程序中急需的算法的真面目了。回顾整个过程，小有感悟。为记念初识程序反汇编的心路历程，特记录下对程序反汇编过程的粗浅感受。

**一、工作环境：**　　**1、**查壳工具：PEiD（用于初步确定exe程序是否加壳以及使用何种开发软件编写）；  
　　**2、**反编译工具：DeDe（经过查壳工具检查，将要被反汇编的程序是用“BorlandC++ 1999”编写的，所以选用专门反编译Delphi程序的DeDe；用于初步确定程序模块、函数方法划分对应exe程序相应汇编指令地址的位置）；  
　　**3、**反汇编动态调试工具：OllyICE（即，OD；用于动态反汇编调试程序的相关代码逻辑）；  
　　**4、**开发环境：VS.NET 2005；  
　　**5、**其他工具：EditPlus、记事本、计算器等若干；

**二、基础知识：**　　**1、**汇编程序设计：主要深入了解程序数据存放、堆栈使用、子程序调用等技巧；不同类型数据的地址跳转规则（一般都是跳转该类型长度个位置）；  
　　**2、**c++程序设计：基本语法结构；指针、数据类型的使用和控制；  
　　**3、**相关工具软件的基本使用方法；

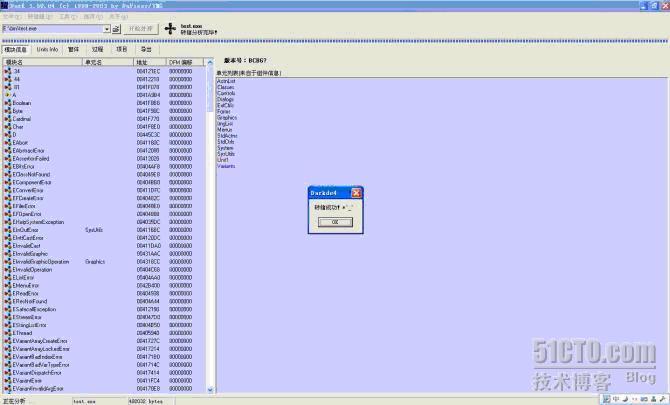
**三、工作过程：**  我的反汇编大概过程是：  
**查壳**->脱壳、判断开发程序使用的环境->**反编译**->初步定位主要功能模块或函数的汇编指令起始、终止地址->**反汇编准备**->查找程序中的特征字符串或特征值，进一步确定需要反汇编的代码逻辑的位置->**开始反汇编调试运行**，分析需要反汇编的代码逻辑的具体过程->将分析出来的逻辑流程**使用c++语言实现**。

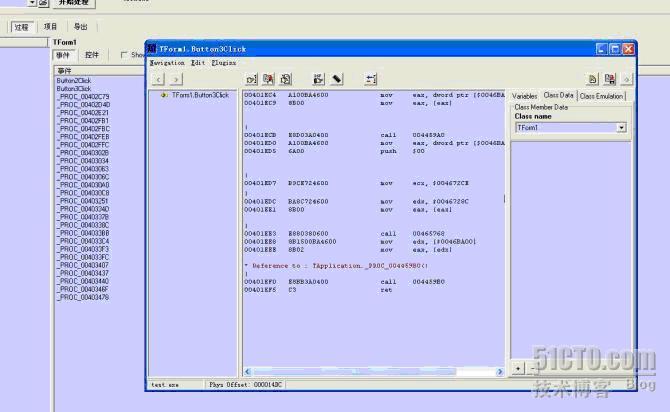
**1、 查壳：**运行PeiD，打开要反汇编的exe程序，在这里姑且称之为：test.exe吧。  
如示意图1：

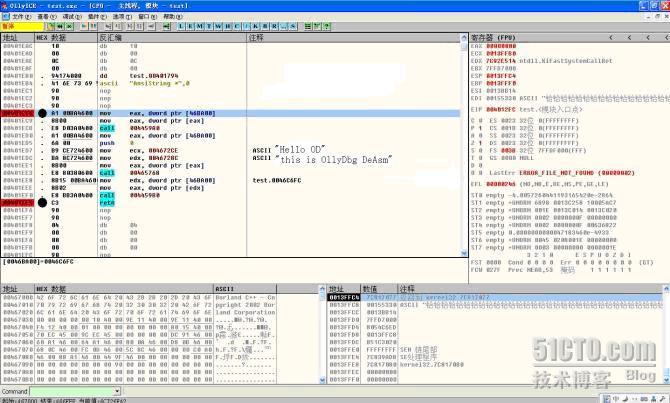
  
“多项扫描”上面那行“Microsoft Visual C++ 7.0 Dll Method 3”就是该程序的开发环境了。  
接下来电击窗口右下角的“>>”按钮，再来看看该程序是否加壳，如示意图2：

  
分别点击“熵值”、“EP校验”、“快速校验”这三行最右侧的“-”按钮，就可以看到是否加壳的提示了。  
哈哈，test.exe程序太干净了。没有任何加壳，可以选用相应的反编译工具进行下一步啦。  ^\_^

**2、 反编译：**“1、”中只是示意图，我实际反编译的程序是由“BorlandC++ 1999”编写的，所以选择DeDe 3.5作为反编译工具；DeDe3.5的具体使用方法，请自行在网上查阅。  
示意图3：

  
   
**3、 初步定位主要模块、函数对应的汇编指令地址：**根据反编译出来的部分类和方法的代码框架，确定了两组可疑的汇编指令地址段。  
如示意图4：

  
 注意选择“过程”按钮，如上图所示。然后双击右下侧列表中的Button3Click，就可以查看该方法对应的函数体了。当然，这样看到的函数体不会是我们一般意义上的函数体，这函数体只有一个框架是高级语言的样子，函数体内部都是汇编代码，不过这已经足够了，后面我们还要进行详细的反汇编调试分析呢。嘿嘿。函数体示意图5如下：  
   
好，记下这段函数的首地址、尾地址，开始尝试反汇编操作。

**4、 反汇编准备：**使用OD（OllyICE）打开test.exe程序，如示意图6：  
   
找到刚才记录下的首地址、尾地址。如上图中两个黑圆点处的红条，分别是首地址：00401EC4、尾地址：00401EF5。

**5、 反汇编调试：**根据步骤4中定位的汇编指令地址，再结合示意图6中[ASCII”Hello OD”]、[ASCII”this is OllyDbg DeAsm”]等关键字符串，进行所有匹配字符串的检索，就可以开始使用OD进行调试了。具体使用方法请参阅OD自带的help文件。  
逐句分析所需部分的汇编代码逻辑流程。其间可以用其他一些辅助工具来帮助记录动态分析过程中的数值传递轨迹及变换方法。

**6、 C++实现：**

根据分析出的汇编代码逻辑，使用C++实现同样的功能。

**备注：在定位反汇编所需的代码位置时，除了一些ASCII字符串的定位方法，还能通过程序中声明的一些常量字符串、映射控制矩阵、计数器等常量来进行更为精确的定位。**