编号：

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 实验 | 一 | 二 | 三 | 四 | 五 | 六 | 七 | 八 | 总评 | 教师签名 |
| 成绩 |  |  |  |  |  |  |  |  |  |  |

武汉大学国家网络安全学院

课程实验(设计)报告

题 目： 实验二：PE文件格式分析

专业(班)： 2022级信安5班

学 号： 2022302181161

姓 名： 王亚鹏

课程名称： 软件安全实验

任课教师： 赵磊

2024年 10 月 15日

**目 录**

[实验2 PE文件格式分析(模板) 1](#_Toc179902489)

[1.1实验名称 1](#_Toc179902490)

[1.2实验目的 1](#_Toc179902491)

[1.3实验步骤及内容 1](#_Toc179902492)

[1.4实验关键过程、数据及其分析 3](#_Toc179902493)

[1.4.1 PE文件格式初步分析与调试 3](#_Toc179902494)

[1.4.2 函数的引入与引出机制 5](#_Toc179902495)

[1.4.3 在目标程序中新增功能代码 7](#_Toc179902496)

[1.4.4 图标资源替换与软件手工汉化 8](#_Toc179902497)

[1.5打造最小PE文件 10](#_Toc179902498)

[1.6实验体会和拓展思考 11](#_Toc179902499)

## 实验2 PE文件格式分析(模板)

### 1.1实验名称

PE文件格式分析

### 1.2实验目的

1. 熟悉各种PE编辑查看工具，详细了解PE文件格式
2. 重点分析PE文件文件头、引入表、引出表，以及资源表
3. 自己打造一个尽可能小的PE文件

### 1.3实验步骤及内容

**第一阶段：****PE文件格式初步分析与调试**

* 分析例子程序hello-2.5.exe
  + 使用二进制编辑工具观察PE文件例子程序hello-2.5.exe的16进制数据
  + 使用Ollydbg对该程序进行初步调试，了解该程序功能结构，在内存中观察该程序的完整结构
  + 使用PE编辑工具修改该程序，使得该程序仅弹出第二个对话框

**第二阶段：函数的引入与引出机制**

* 熟悉各类PE文件格式查看和编辑工具：
  + 结合hello-2.5.exe熟悉PE文件头部、引入表的结构
* 熟悉函数导入的基本原理
* 手工修改hello-2.5.exe程序，使得其可以弹出第三个对话框（提示框标题为“武汉大学信安病毒实验”，内容为：你的姓名+学号）
* 找到系统System32目录下的user32.dll文件，用二进制编辑器打开并分析该文件引出表，找出函数MessageBoxA的地址，并验证该地址是否正确。【视频中采用的方法是从文件中定位查找，在实验报告中请直接从实际进程（如test.exe）内存空间进行定位查找】

**第三阶段：在目标程序中新增功能代码**

* 用二进制编辑工具修改hello-2.5.exe程序的引入表，使该程序仅可以从kernel32.dll中引入LoadLibrary和GetProcAddress函数，而不从user32.dll导入任何函数。
* 在代码节中写入部分代码利用这两个函数获取MessageBoxA的函数地址，使hello-2.5.exe程序原有功能正常。

**第四阶段：图标资源替换与软件手工汉化**

* 资源表资源操作实践
  + 利用PEview.exe分析PEview.exe程序
  + 用二进制编辑工具修改PEview.exe，使得该文件的图标变成csWhu.ico
  + 熟悉eXeScope工具的实用，并利用该工具汉化PEview.exe程序

**课后习题思考：**

* 如何打造最小的PE文件
  + 修改WHU\_PE-2.5.exe文件，保持该文件的功能不变，使得该文件大小尽可能小
  + 本文件的最小极限可能是多少？结合tinyPE一文进行描述。<http://www.phreedom.org/research/tinype/>
* 如何编码实现PE程序中对应资源的提取与替换？涉及到哪些关键API函数。
* 当目标程序的图标资源为多个时，每个图标资源分别对应着哪里？此时图标替换策略应该如何调整？
* 资源节与恶意代码有何关联？
* 什么是HOOK？其与本章学习有何关系？

### 1.4实验关键过程、数据及其分析

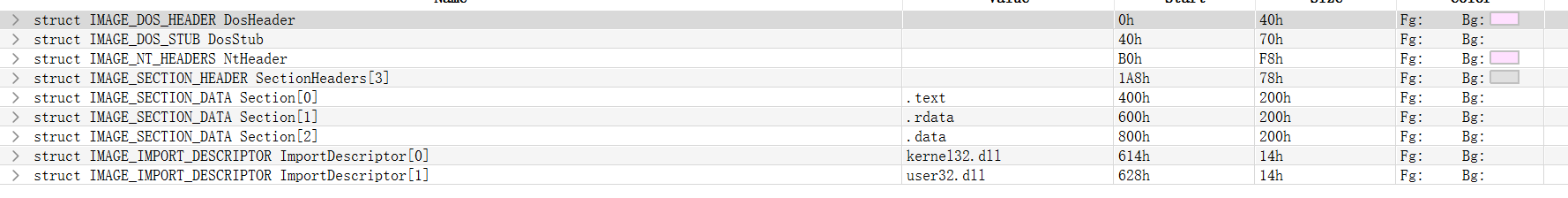
#### 1.4.1 PE文件格式初步分析与调试

**（1）PE文件结构的查看分析**



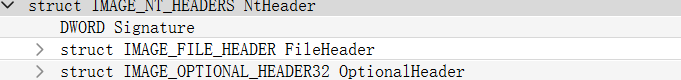
这是PE文件的基本格式

我们使用010editor来打开hello-2.5.exe,显示结果如下

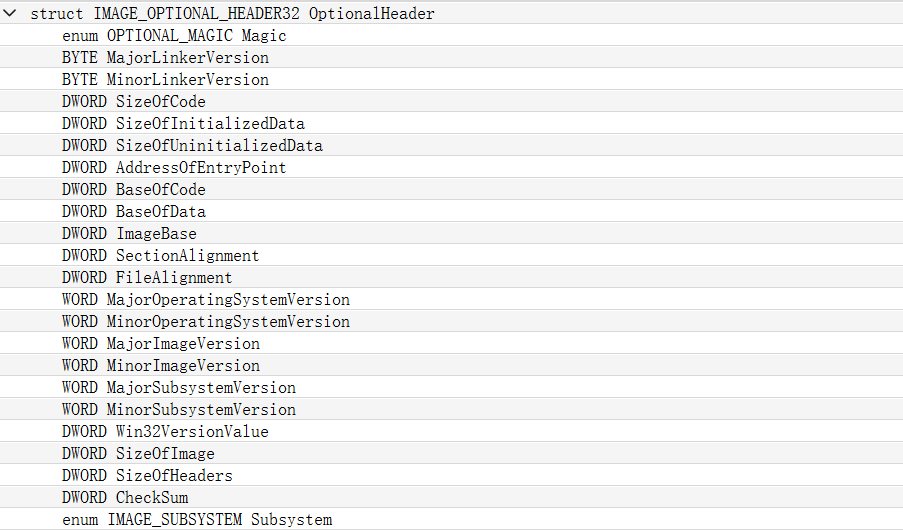


可以点开对应项目详细查看内容

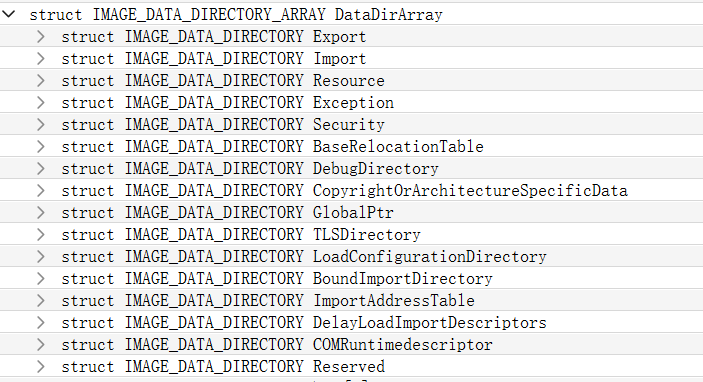
**NT头: 包括PE字段、映像文件头和可选文件头**



可选头文件包括东西众多



可选文件头的最后是数据目录表



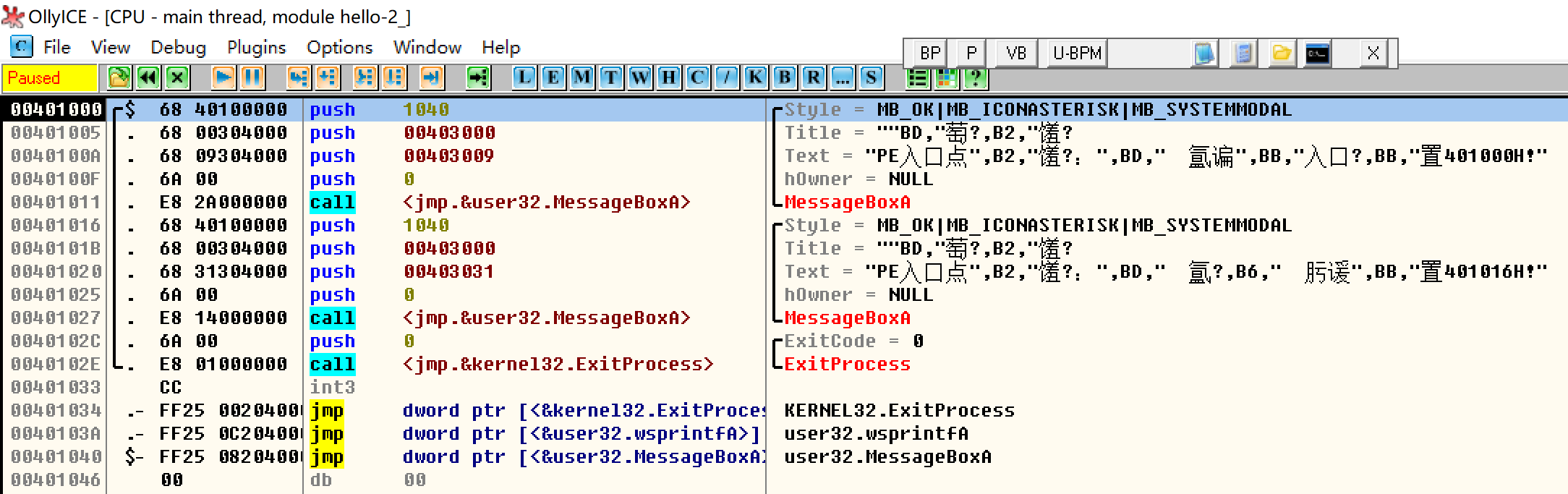
也可以找到引入函数表如下：



以上就是使用程序查看PE文件结构，可以进一步加深我们对PE文件的认识

**（2）PE文件调试**

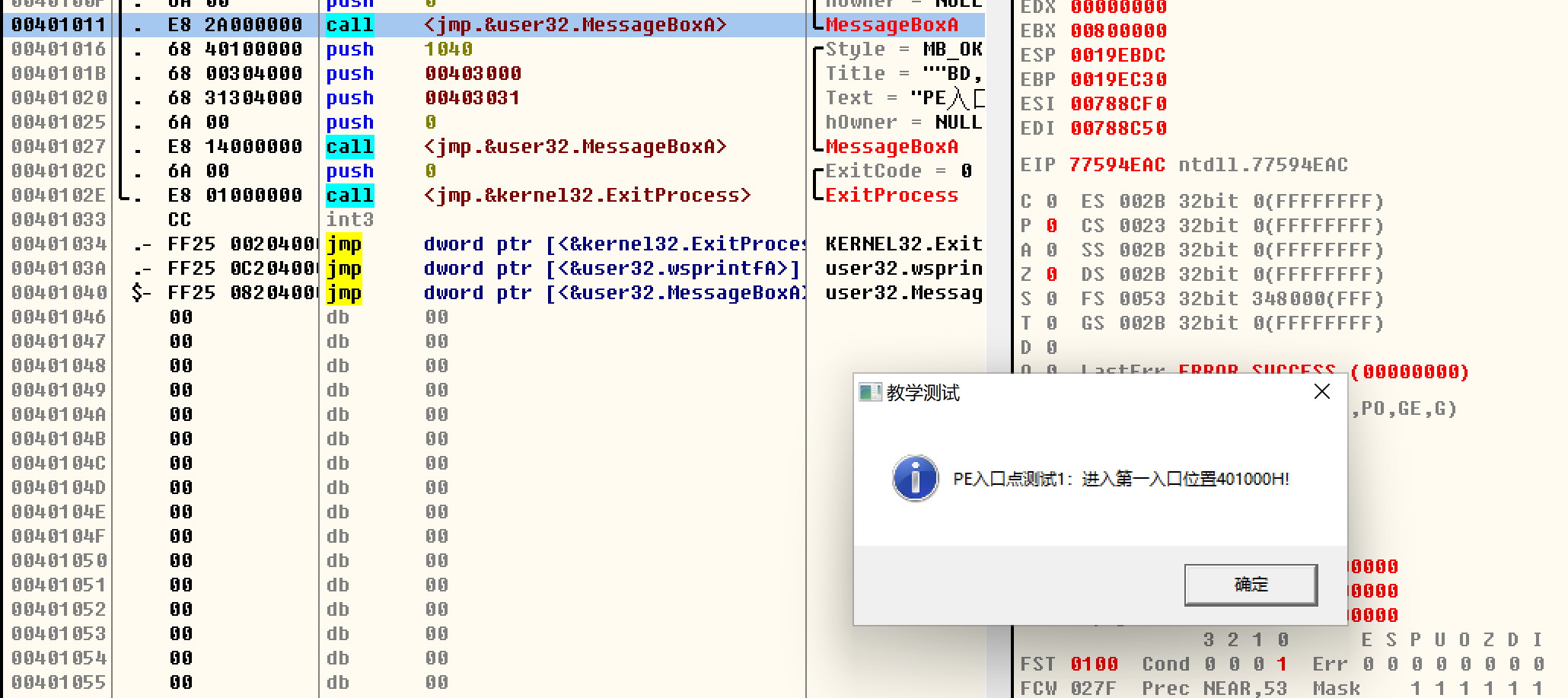
使用ollydbg打开hello-2.5.exe



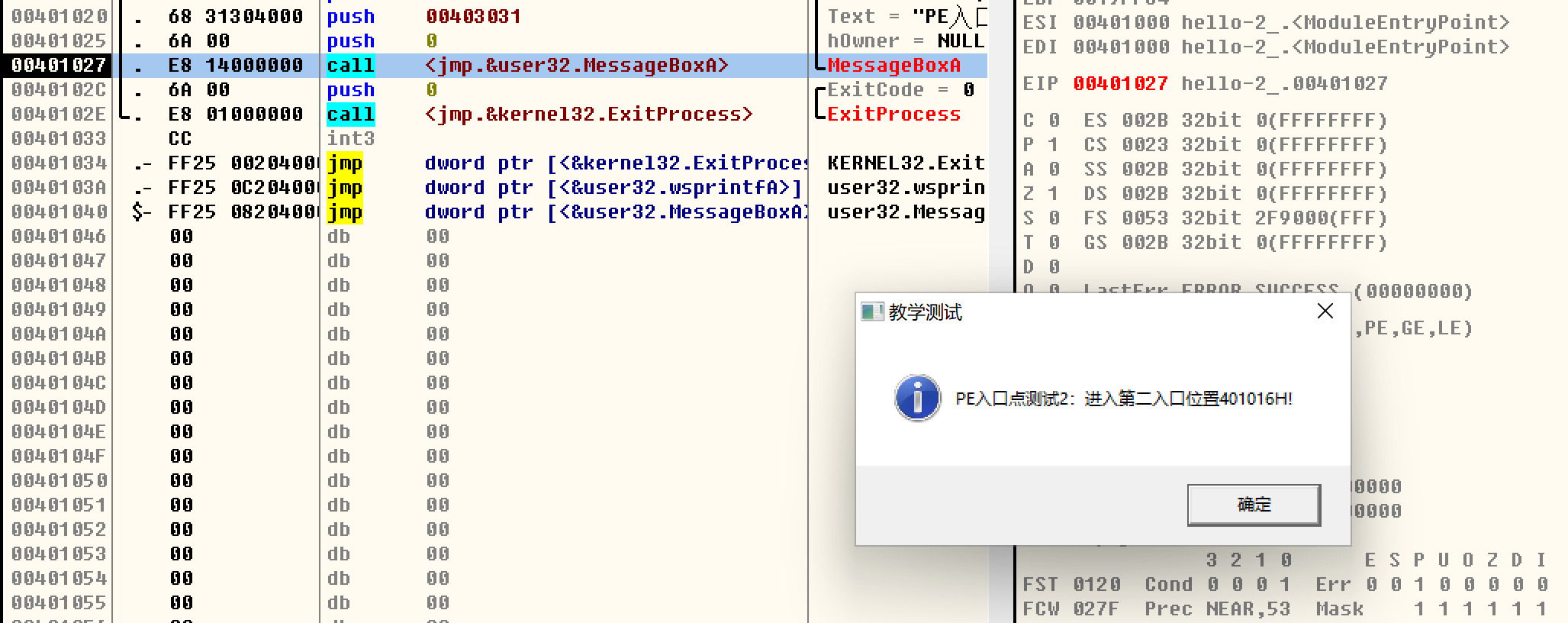
程序断点停在程序入口点，地址401000，这与我们在010中看到的完全一致，基地址+偏移正好等于401000

对程序进行单步调试（双击下断点）

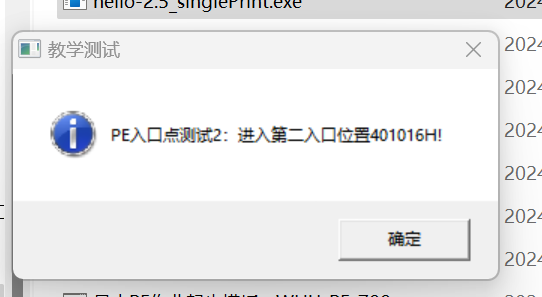
当进入到如图所示语句时，会弹出一个框



走到下一个call指令弹出第二个窗口



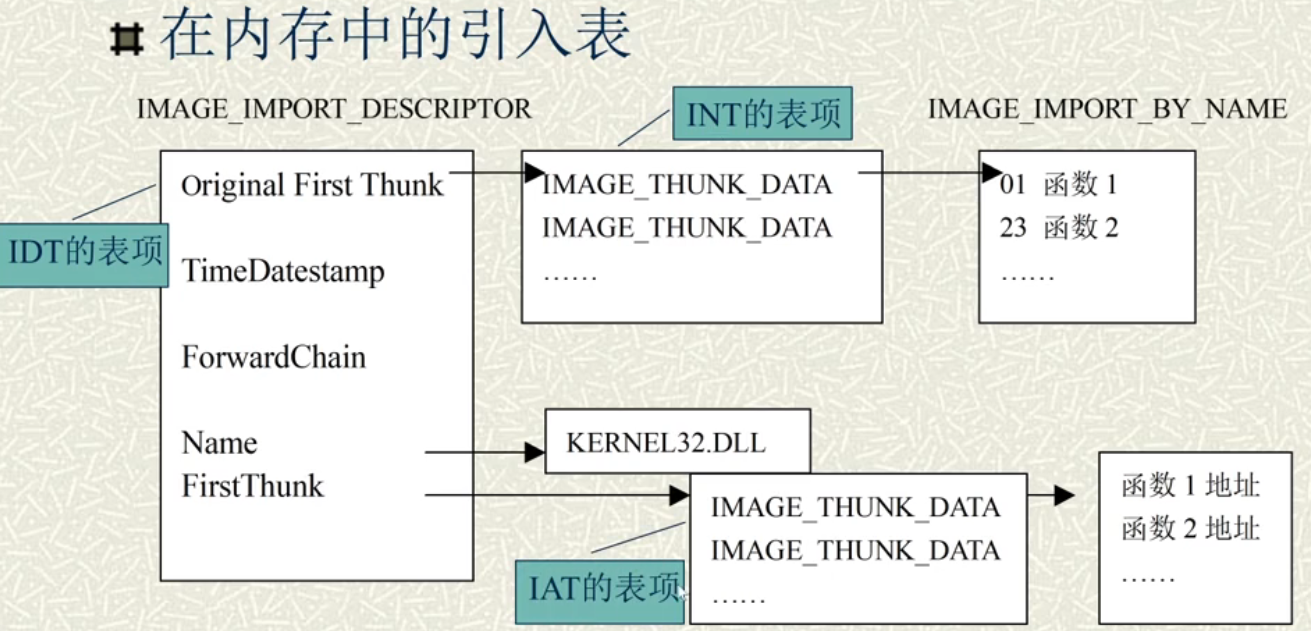
根据视频讲解，我们要想仅有一个窗口弹出，就应该修改程序的入口，使其直接跳到第二个窗口开始之处，修改后的文件为**hello-2.5\_singlePrint.exe**。运行结果如下：



#### 1.4.2 函数的引入与引出机制

**（1）函数引入表**

函数引入表的结构如下：



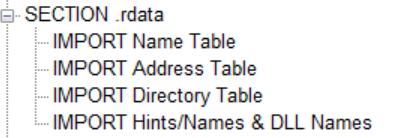
**IAT (Import Address Table)**: 存储被导入函数地址，程序运行时通过IAT访问导入函数，地址由操作系统加载时填充。

**IDT (Import Descriptor Table)**: 包含导入函数的描述信息，如名称、序号和在DLL中的位置，帮助程序定位函数。

**INT (Import Name Table)**: 存储导入函数的名称，程序通过IDT获取序号，再通过INT获取函数名称。

**DLL文件名**: 指包含导入函数的动态链接库名称，程序通过该名称加载并定位DLL文件中的函数。

程序运行时根据这些表定位并调用所需函数

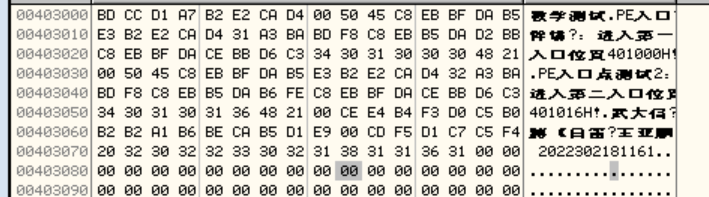


使用PEview查看exe文件可显示引入表结构

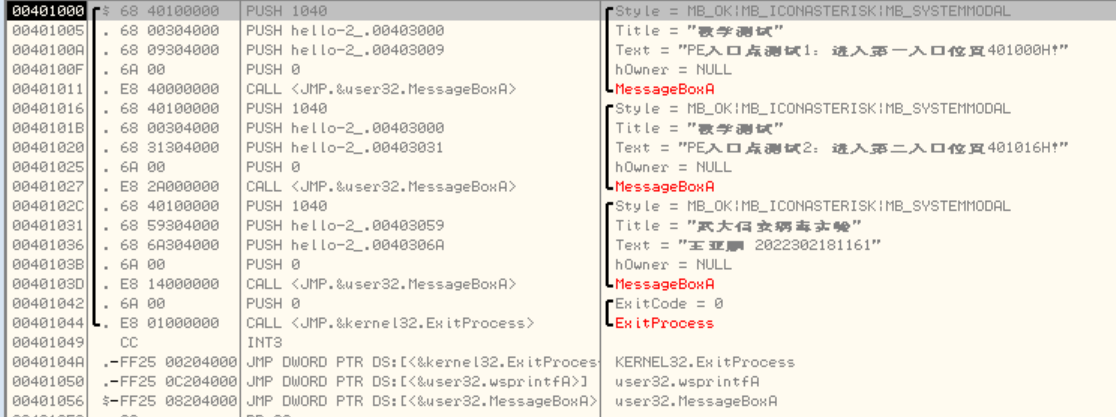
**（2）修改hello-2.5.exe程序，使得其可以弹出第三个对话框**

主要思路如下：

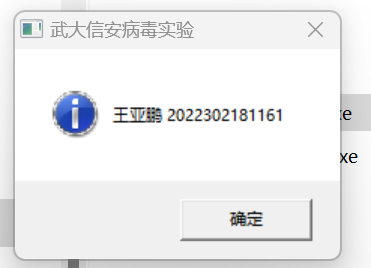
1. 修改.text段：添加一个MessageBoxA的函数调用
2. 修改.data段：添加一个MessageBoxA的参数信息
3. 修改.text和.data段的节表信息



数据段加入自己的相关信息



修改代码段使得新加入的内容被打印



运行结果如图所示

修改后的代码是**hello-2.5.addString.exe**

(3) 找到系统System32目录下的user32.dll文件，用二进制编辑器打开并分析该文件引出表，找出函数MessageBoxA的地址，并验证该地址是否正确

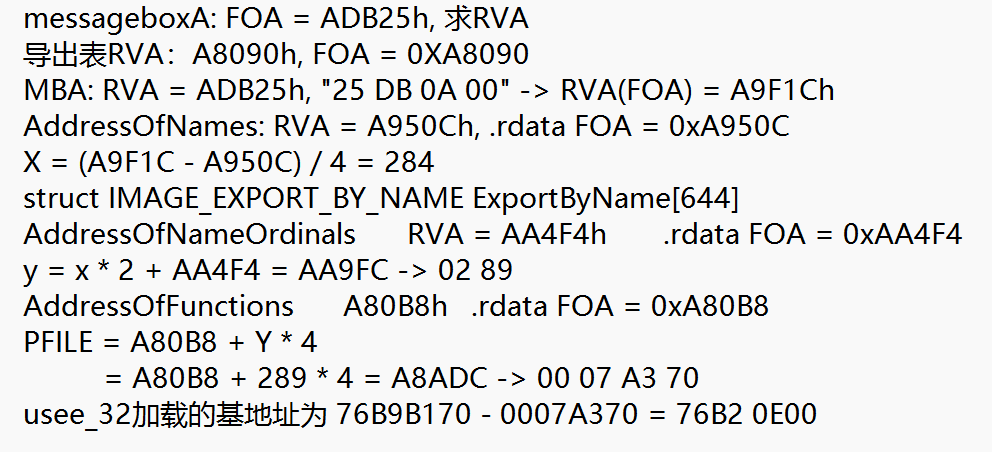
主要步骤如下：

从AddressOfNames指向的指针数组中找到“MessageBoxA”字符串，并记下该数组序号x

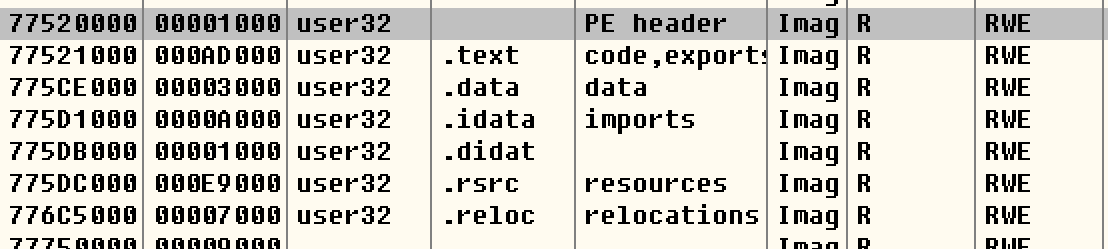
从AddressofNameOrinals指向的数组中，定位第x项成员，得到一个序号y，y为MessageBoxA函数地址在AddressOfFunction中的索引号。

从AddressOfFunction指向的数组中定位第y项，获得函数的RVA地址

首先搜索MesssageboxA



进行如上方法的验证，可以知道这种方法是正确的



#### 1.4.3 在目标程序中新增功能代码

**（1）用二进制编辑工具修改hello-2.5.exe程序的引入表，使该程序仅可以从kernel32.dll中引入LoadLibrary和GetProcAddress函数**

具体步骤如下：

1. 修改IMPORT Hints/Names&DLL Names

将kernel32的ExitProcess的Hint改为0

将Kernel32改为LoadLibraryA,最后用00截断

再引入GetProcAddress，最后用00截断

将kernel32放到最后，并将多余的位置清零

2. 修改INT表

要对FirstThunk进行修改，我们找到600的位置

由于data节中RVA=2000，文件偏移为600

因此计算得到600处的2064-2000+600 = 664，因此第一个导入函数在文件中的偏移是664

以此类推可以发现第二个trunk的值为672，第三个为681，因此修改如下

本例只引入三个函数，因此第四个字段要置零

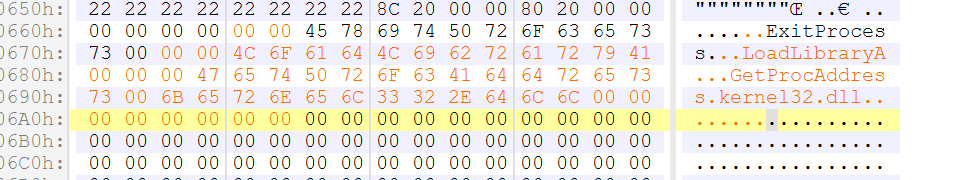
3. 修改IDT表

由于我只引入了一个dll，因此我要将原来的usr32.dll清空

另外由于kernel32更改了位置，因此我要将name处的地址更改

接下来要与本部分第二阶段联系起来，请看恢复原有功能

在010editor中修改对应内容

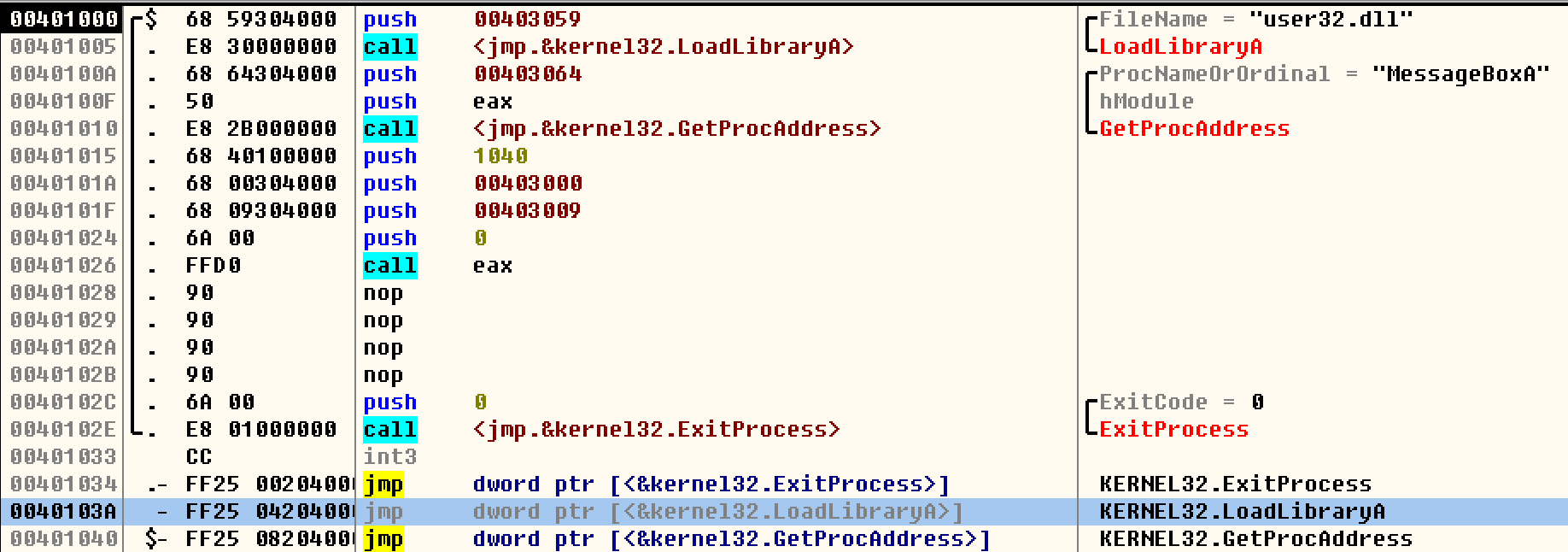


**（2）在代码节中写入部分代码利用这两个函数获取MessageBoxA的函数地址，使hello-2.5.exe程序原有功能正常**

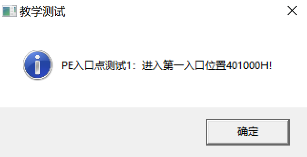
修改.data段，添加两个字符串



修改.text段



运行结果如下：



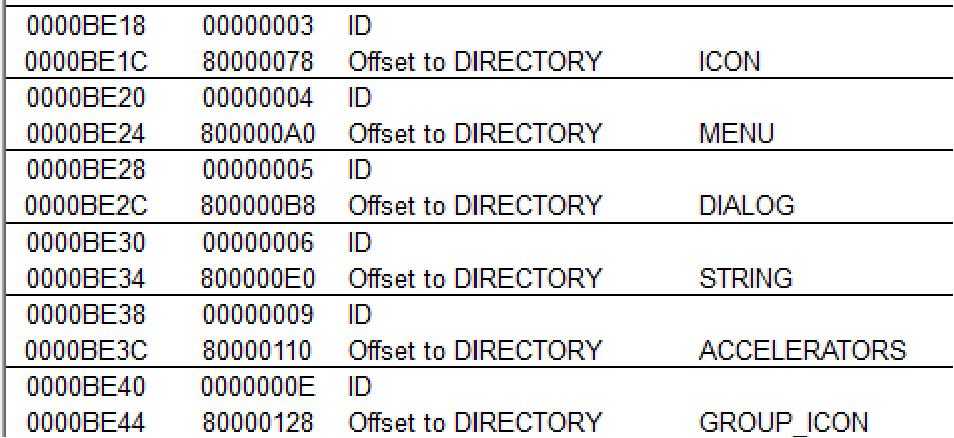
是和原版程序一样的运行结果，代码保存到hello-2.5same.exe

#### 1.4.4 图标资源替换与软件手工汉化

**（1）图标资源替换**

将PEview.exe的图标替换为whu.ico

首先使用PEview.exe查看一个PEview.exe的副本，结构如下

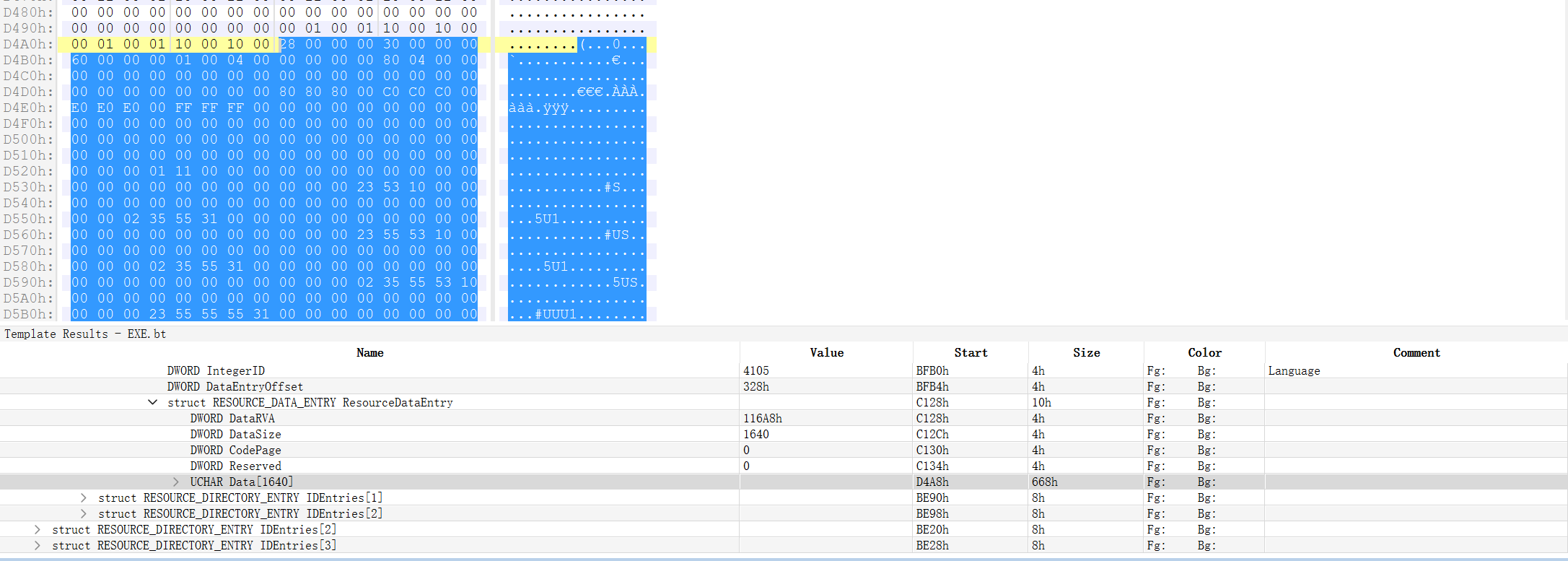


其图标在ID=3的位置处，GROUP\_ICON在ID=E处

**ICON结构：图标资源的图像数据，对应ICO文件中的ICONIMAGE结构：**

我们首先要定位到图标资源

在010中查看ID为3的部分，可以看到它有三个图标资源（图中阴影部分上方）



可以看到大小为668h，大于我们要进行替换的目标图标大小，因此我们直接复制2E8大小的图标部分，粘贴到PEview中

GROUP\_ICON:与ICO文件的ICONDIR结构相同，除了ICONIRENTRY最后一个word字段，ICONDIR结构最后是一个双字DWORD dwlmageOffset，GROUP\_ICON最后一个字段为word nID

所以我们还要修改GROUP\_ICON

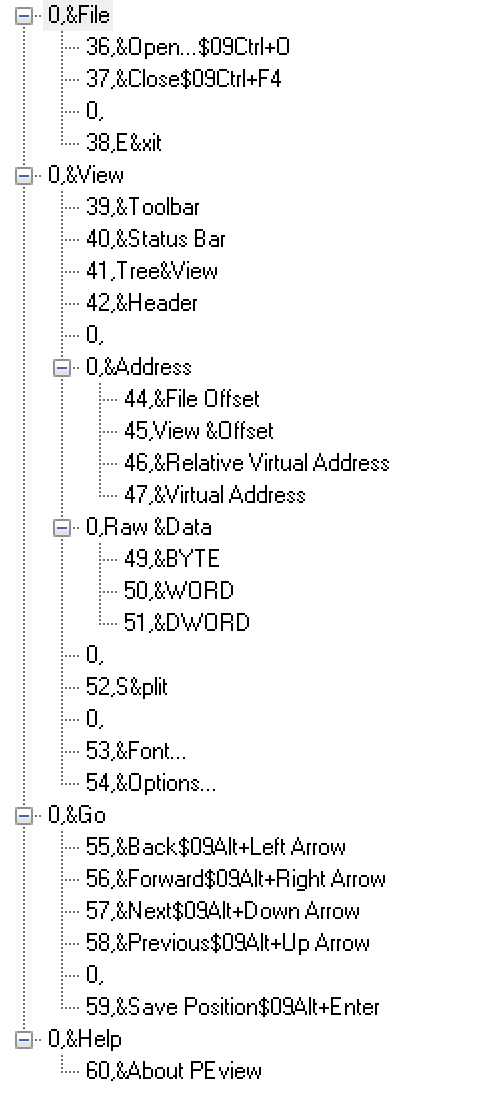
保存后可以看见修改的图标

替换后的软件为PEview\_修改图标.exe，效果如下图所示

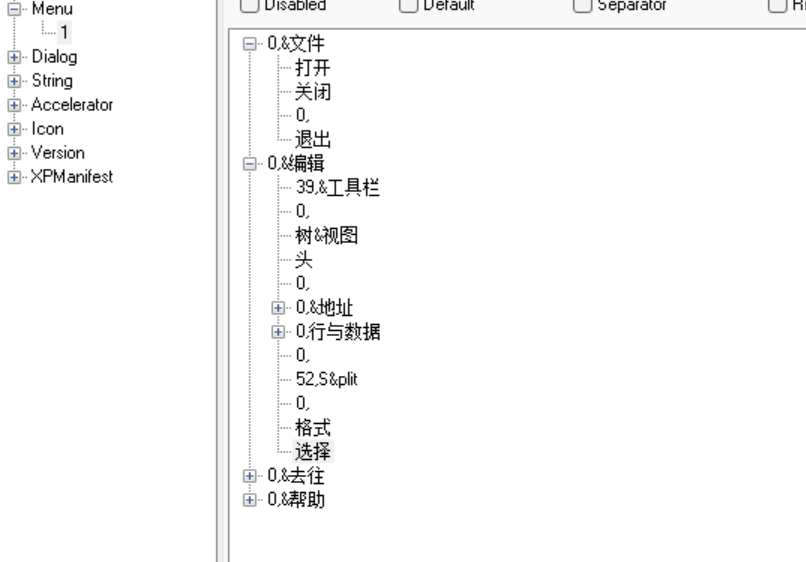


**（2）手工汉化**

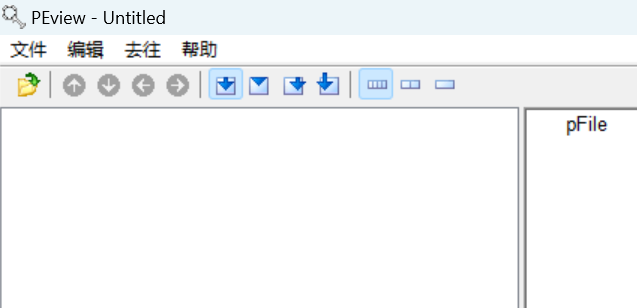
使用exeScope查看PEview



点击menu，在视图的左边选择想要修改的部分直接修改即可



以下是汉化结果：



汉化后的软件为PE汉化版.exe

### 1.5打造最小PE文件

* 如何打造最小的PE文件
  + 修改WHU\_PE-2.5.exe文件，保持该文件的功能不变，使得该文件大小尽可能小

**主要思路**

（1）删除冗余节：保持代码和数据节的数量最小化。通常我们只需要 .text 节（代码）和 .rdata 节（常量）。

（2）缩小PE头部：PE头部信息是可调整的，可以通过最小化各节的表项、去除不必要的表项来减少大小。

（3）修改对齐方式：PE文件的节对齐值决定了每个节的最小单位大小，适当减小对齐数可以减少PE文件中空余的填充数据

* + 本文件的最小极限可能是多少？结合tinyPE一文进行描述。<http://www.phreedom.org/research/tinype/>

我认为是大概200B（原来的1/5），tinyPE通过手动压缩可以使压缩效率非常高

* 如何编码实现PE程序中对应资源的提取与替换？涉及到哪些关键API函数。

在编码中实现PE程序中资源的提取与替换，通常涉及到Windows API函数，这些函数能够帮助我们访问和修改PE文件的资源。关键步骤和API包括：

* **加载PE文件资源**：使用 LoadLibrary() 和 FindResource() 函数来加载PE文件并找到所需的资源。
* **获取资源数据**：LoadResource() 和 LockResource() 用于加载和锁定资源，从而获取其数据。
* **更新资源**：BeginUpdateResource(), UpdateResource(), EndUpdateResource() 是用来修改PE文件资源的核心API。

例如，提取某个资源时可以先使用 FindResource() 定位资源，通过 LoadResource() 获取数据指针。修改时则需要打开资源句柄并使用 UpdateResource() 进行替换，确保在修改后调用 EndUpdateResource() 完成更新

* 当目标程序的图标资源为多个时，每个图标资源分别对应着哪里？此时图标替换策略应该如何调整？

当目标程序包含多个图标资源时，这些图标通常存储在 .rsrc 节中，以资源的 RT\_GROUP\_ICON 和 RT\_ICON 形式存在。每个图标资源包含一个图标组资源（RT\_GROUP\_ICON），指向若干个具体的图标图像（RT\_ICON），这些图像通常具有不同的分辨率和颜色深度。

**图标替换策略**：

* **定位所有图标组**：使用 FindResource() 找到所有 RT\_GROUP\_ICON 类型的资源，了解这些组资源如何与具体的 RT\_ICON 关联。
* **匹配替换**：为了替换图标，需要确保新图标的分辨率和颜色深度与原图标一致，以避免程序加载失败。对于多个图标，需要分别替换每一个 RT\_ICON，并更新对应的 RT\_GROUP\_ICON。

通过 UpdateResource() 函数可以对每个图标进行替换，确保对齐原有结构

* 资源节与恶意代码有何关联？

资源节（.rsrc）在恶意软件中常常被用于隐藏恶意代码或数据。这是因为资源节通常用于存储图片、图标、对话框等用户界面元素，容易被忽略。恶意代码可能会将真正的恶意载荷存储在资源节中，然后在程序运行时提取和执行这些代码。

一些恶意程序通过隐藏在 .rsrc 节中的数据和在运行时解密和执行恶意代码，从而绕过静态分析工具的检测。对资源节的修改和分析对于恶意软件逆向工程非常重要

* 什么是HOOK？其与本章学习有何关系？

**资源节（.rsrc）在恶意软件中常常被用于隐藏恶意代码或数据。这是因为资源节通常用于存储图片、图标、对话框等用户界面元素，容易被忽略。恶意代码可能会将真正的恶意载荷存储在资源节中，然后在程序运行时提取和执行这些代码。**

**一些恶意程序通过隐藏在 .rsrc 节中的数据和在运行时解密和执行恶意代码，从而绕过静态分析工具的检测。对资源节的修改和分析对于恶意软件逆向工程非常重要**

### 1.6实验体会和拓展思考

在这次实验中，我通过对PE文件的深入学习和实践，系统地掌握了其结构和各部分的功能。通过使用二进制编辑工具和Ollydbg对示例程序hello-2.5.exe进行了观察和调试，我逐步了解了PE文件的整体结构和程序的功能运作。首先，通过对PE文件头部和导入表进行分析，我深入理解了函数的导入和导出机制，特别是在程序运行时如何通过引入表获取外部函数，明确了各个表在程序运行中的作用和协作方式。

此外，我还进行了手动修改hello-2.5.exe的尝试，使其能够弹出特定的目标对话框。这一步让我不仅强化了对PE文件结构的理解，也增强了对二进制文件直接编辑的操作能力。在资源修改实践中，我利用PEview.exe和eXeScope工具进一步操作PE文件，成功更改了PEview.exe的图标为csWhu.ico，同时使用eXeScope对其进行了汉化操作，使界面更加直观、易懂。这些实践不仅让我熟悉了多种实用工具的操作，还帮助我建立了PE文件分析的基本技能框架。整个过程特别有趣，通过动手实践，我切实感受到工具的使用对理解底层系统结构的重要性，这也为我后续的学习奠定了坚实的基础。

**注意：实验报告书写完毕之后请更新目录**