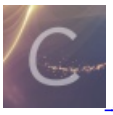
IDA pro简介

原创

SpongeB0B 于 2019-10-30 16:47:44 发布 9221 收藏 49

分类专栏： 工具使用教程 文章标签： IDA ida pro 逆向工具 REVERSE

本文链接：https://blog.csdn.net/john_david_/article/details/102822253



IDA pro介绍

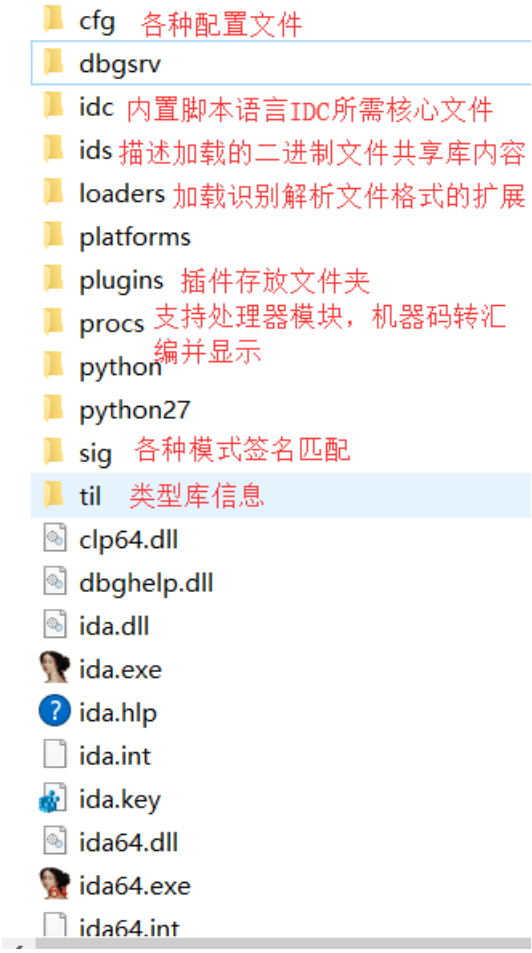
本周分享的工具是IDA Pro 7.0。IDA Pro全称是交互式反汇编器专业版（Interactive Disassembler Professional），简称IDA，它是一种典型的递归下降反汇编器。IDA并非免费软件，但Hex-Rays公司提供了一个功能有限的免费版本。IDA是Windows，Linux或Mac OS X托管的多处理器反汇编程序 和调试程序，它提供了许多功能，是一款很强大的静态反编译工具。支持很多插件和python，利用一些插件可以提供很多方便的功能大大减少工作量，在CTF中，逆向和pwn都少不了它，更多强大的功能等待童鞋们自己去学习挖掘，三言两语讲不完。它支持数十种CPU指令集其中包括Intel x86，x64，MIPS，PowerPC，ARM，Z80，68000，c8051等等。

`IDA pro 7.0（绿色英文版）`和`部分插件`+`《IDA Pro权威指南 第2版 》`已经上传至群文件，来源于：`吾爱破解论坛`。论坛也有汉化版，英文原版本习惯了都一样。`看雪`有一个`IDA pro插件收集区`，大家有需要也可以去那找https://bbs.pediy.com/forum-53.htm

IDA pro常用功能介绍

IDA pro安装很简单，没什么难度，这个还是绿色版直接解压运行即可，就是大家需要什么插件自己装，这样比较干净整洁，挺好的。

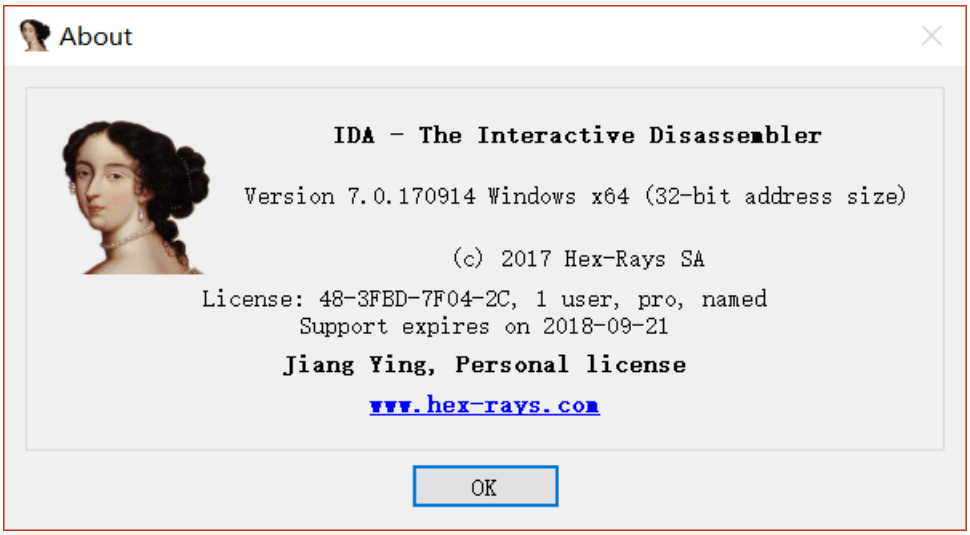
IDA文件目录简介

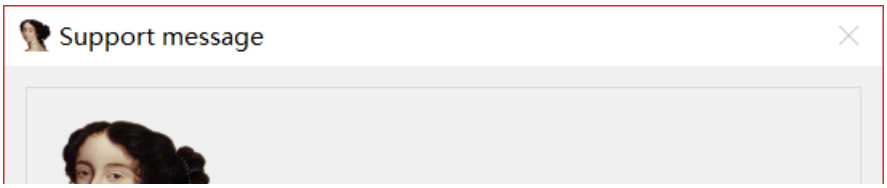


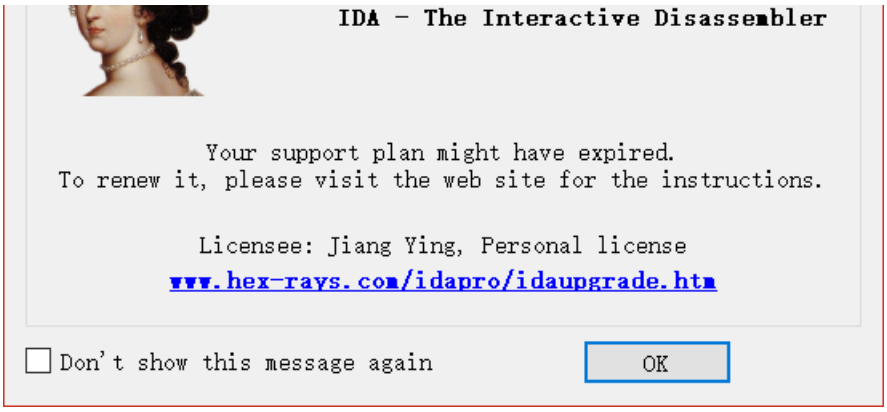
`ida.exe`是`32位`的，大部分CTF中`windows`上可执行程序逆向用这个就可以了;
`ida64.exe`顾名思义是`64位`，一般CTF中`ELF文件`中都是用这个打开，`32位版本`支持不好的都可以用这个。

启动

打开以后首先显示IDA的介绍和注册信息等



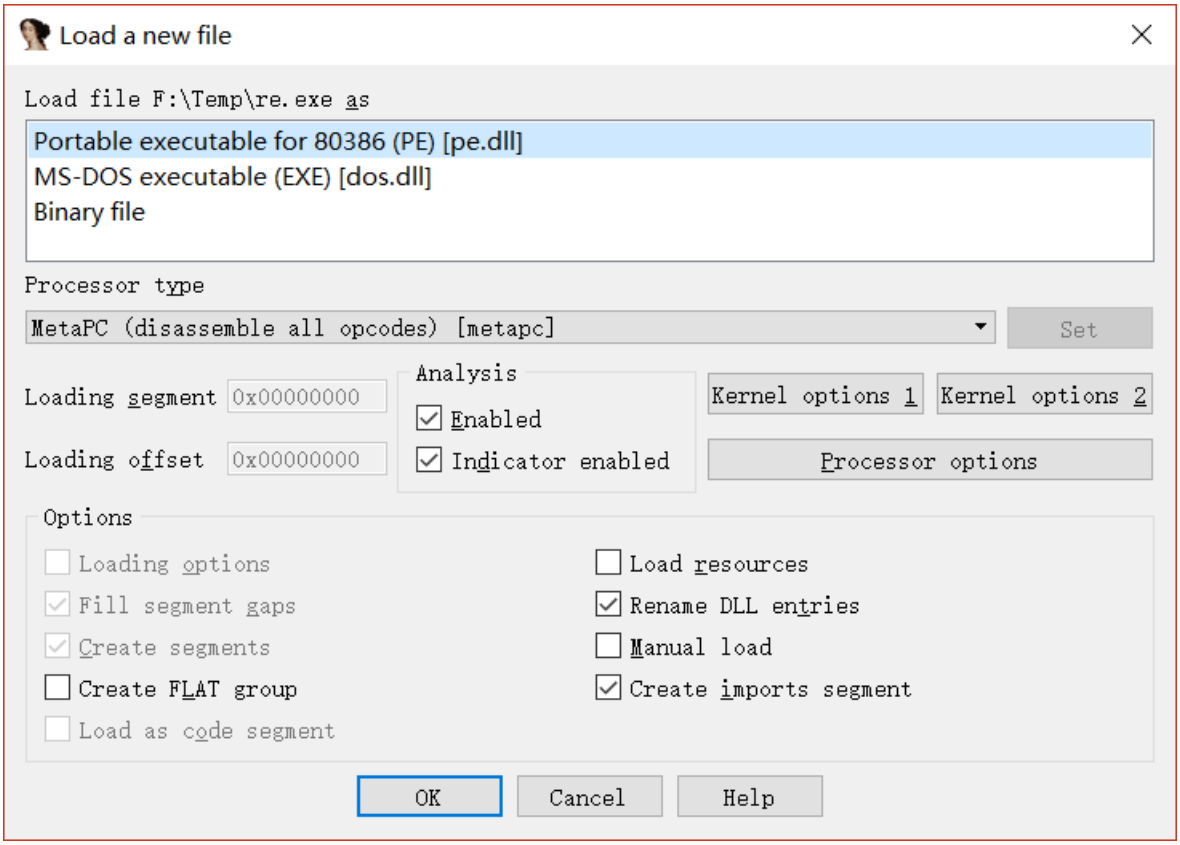




New（新建空白的工程），Go（运行工程），Previous（运行过的工程记录）

选择`File菜单`下的`Open`，打开要逆向的文件，弹出一个`Load a new file`的界面。可以选择:

1. 程序的类型；
2. 处理器的类型；
3. 加载的段地址和偏移量；
4. 是否允许分析；
5. 一些加载选项；
6. 内核和处理器的一些选项；
7. windows系统dll所在的目录。



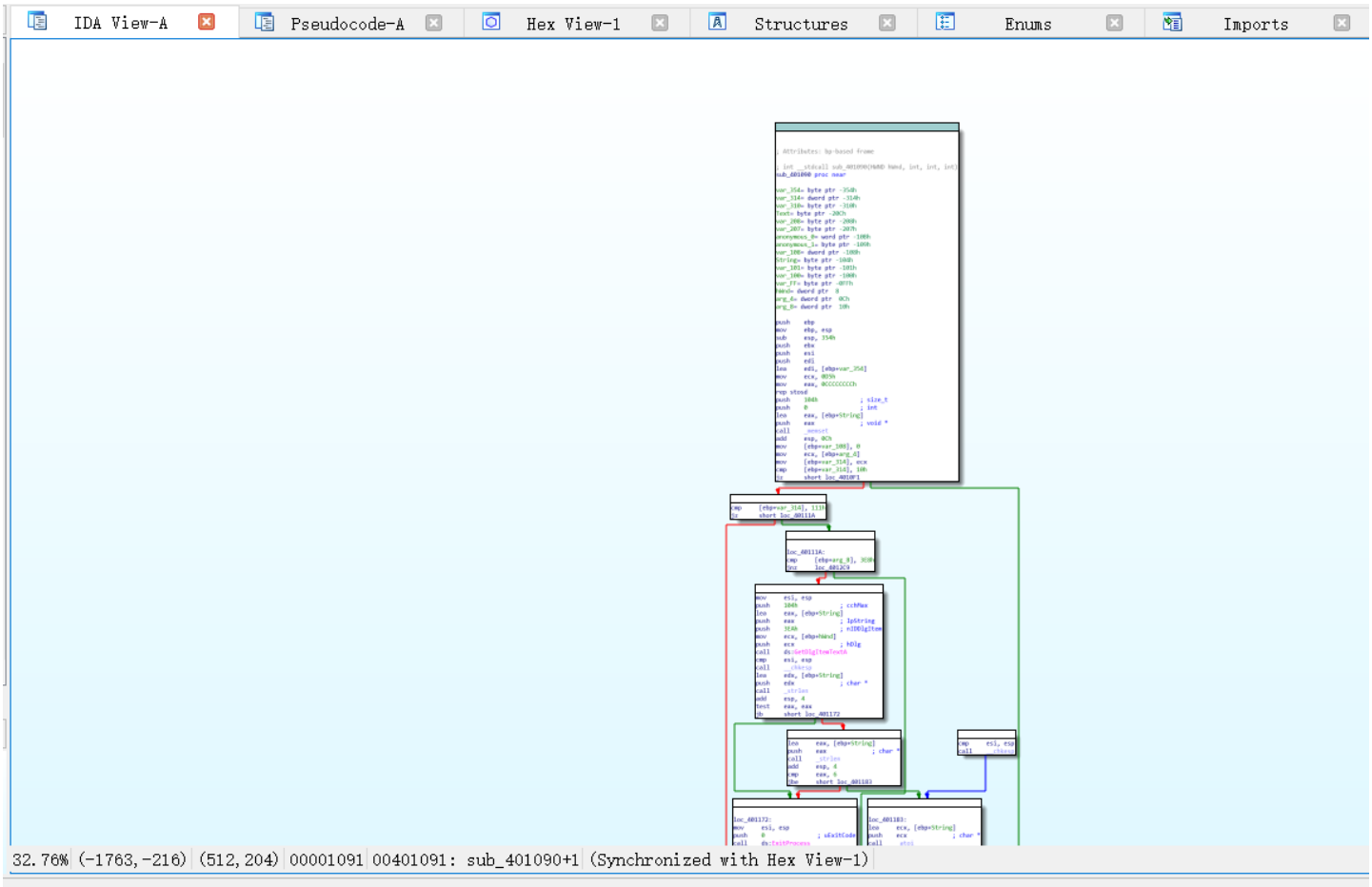
默认选择第一个`PE文件`就可以，一些其他格式的文件可以使用第三个选项`Binary file`以二进制文件的形式记载，自己解析。
[外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-3mRkASp1-1572424879647)(https://i.loli.net/2019/02/20/5c6d2463a8dc2.png)]
各窗口的功能：

1. IDA view: 定位要修改的代码段在哪里。
2. Hex view: 用来修改我们的数据
3. exports: 导出函数表窗口
4. import: 导入函数表窗口
5. names: 函数和参数的命名列表
6. functions: 样本的所有函数窗口
7. strings: 字符串显示窗口，会列出程序中的所有字符串

顶部是工具栏其中包括一些逆向分析过程中经常使用的工具，基本工具栏能满足普通用户的需要，当然还可以通过`菜单栏`的`View`->`Toolbats`->`Advanced mode`选项切换到`高级模式工具栏`，高级模式工具栏包含更加丰富的工具。切换回基本模式方法一样，在上面有个`Basic mode`。

工具栏下面是导航栏，加载文件地址空间的线性视图，不同的颜色代表不同类型的文件内容，在导航栏下方列出了不同颜色所代表的文件内容。可以放大或缩小导航带，点击导航栏可以在反汇编窗口或十六进制窗口中跳转到对应选中的位置。

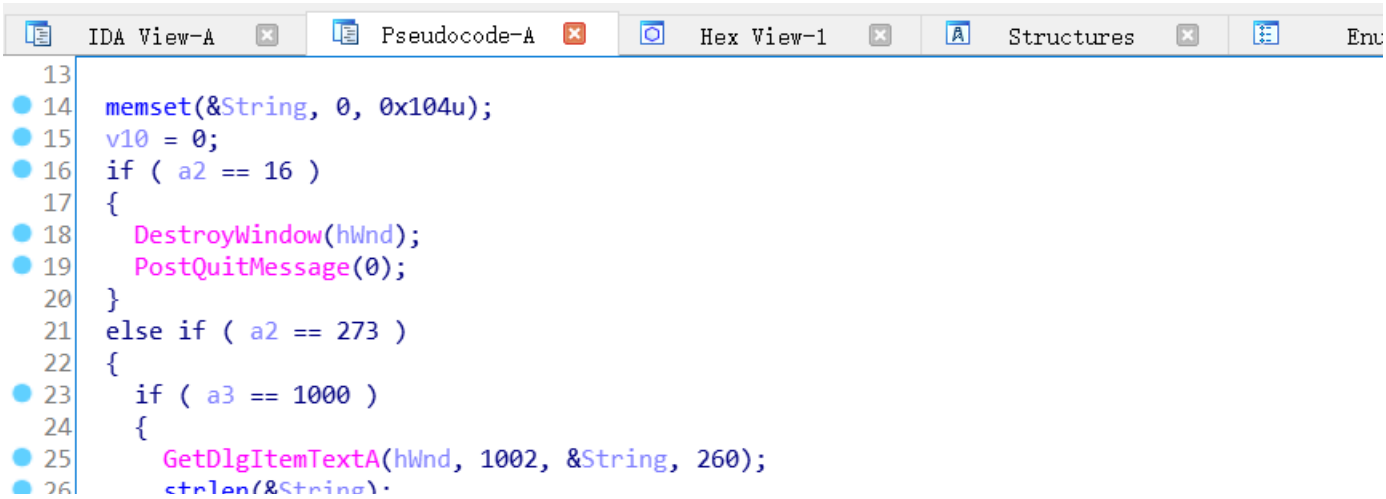
反汇编窗口也称`IDA view`窗口，显示了被加载文件的反汇编代码，是我们静态分析过程中最主要的窗口。该窗口有两种显示格式，分别为文本视图和图形视图。很多时候我们会根据分析的需要在文本视图和图形视图之间切换可以使用快捷键：`空格键`。



图形视图能够清晰地显示一个函数的控制流程，可以通过"CTRL +鼠标滑轮"来缩放图形显示的大小。但对于大型或者复杂的函数可能会导致图形视图变得极为杂乱，此时可以通过IDA在图形视图下默认打开的图形概况窗口来定位需要查看的图形区域。
[外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-05TnQrYe-1572424879650)(https://i.loli.net/2019/02/20/5c6d2eb7f17fc.png)]
文本视图中，左边显示了所在段名称+文件地址，如图中光标所在处为`.text:00401091`表示地址`0x00401091`处代码位于`.text`段，右边显示反汇编出来的汇编指令。

把鼠标移到某些标识符上`IDA Pro`会有一些提示，双击能自动跳到相应的位置。把一个函数逆向的方法很简单，只要光标在函数区域按`F5键`（需要安装`Hex-Rays Decompiler`插件，这里安装包已经自带，这是IDA最nb的插件）就转换显示出伪代码。

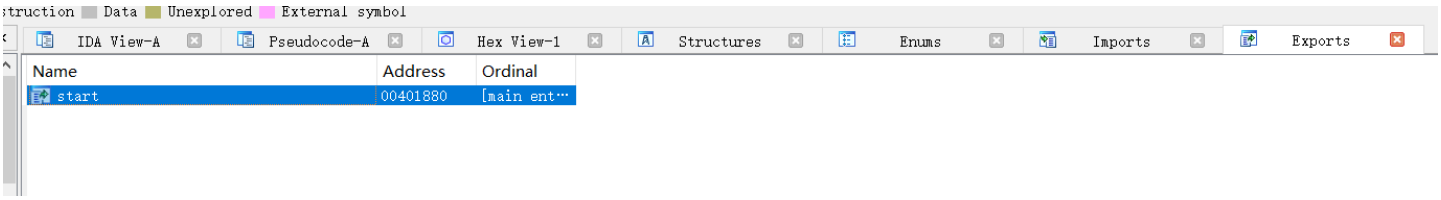
用户还可以通过`菜单栏`的`View`->`Open Subviews`选项打开需要的其他窗口。
下面将逐一介绍在静态分析过程中经常使用的一些次要显示窗口。



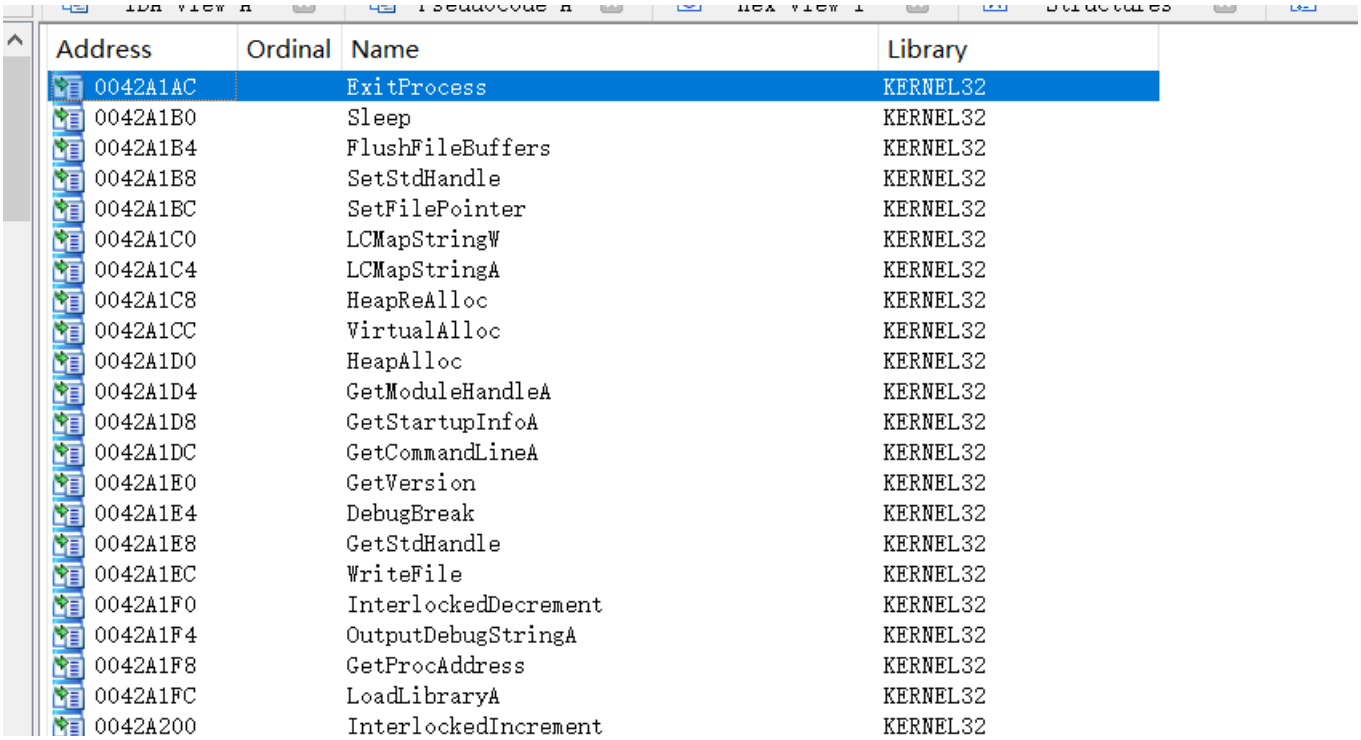
```
27      if ( strlen(&String) > 6 )
28        ExitProcess(0);
29      v10 = atoi(&String) + 1;
30      if ( v10 == 123 && v12 == 120 && v14 == 122 && v13 == 121 )
31      {
32        strcpy(Text, "flag");
33        memset(&v7, 0, 0xFCu);
34        v8 = 0;
35        v9 = 0;
36        _itoa(v10, &v5, 10);
37        strcat(Text, "{");
38        strcat(Text, &v5);
39        strcat(Text, "_");
40        strcat(Text, "Buff3r_0v3rf|0w");
41        strcat(Text, "}");
42        MessageBoxA(0, Text, "well done", 0);
43      }
44      SetTimer(hWnd, 1u, 0x3E8u, TimerFunc);
45    }
46    if ( a3 == 1001 )
00001090 sub_401090:13 (401090)
```

这是通过F5转换出的伪代码，方便用户更快速理解分析，大大降低了逆向工程的门槛，不再让人看汇编代码看到头秃。
[外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-7L5qEOUh-1572424879651)(https://i.loli.net/2019/02/20/5c6d30633bf47.png)]

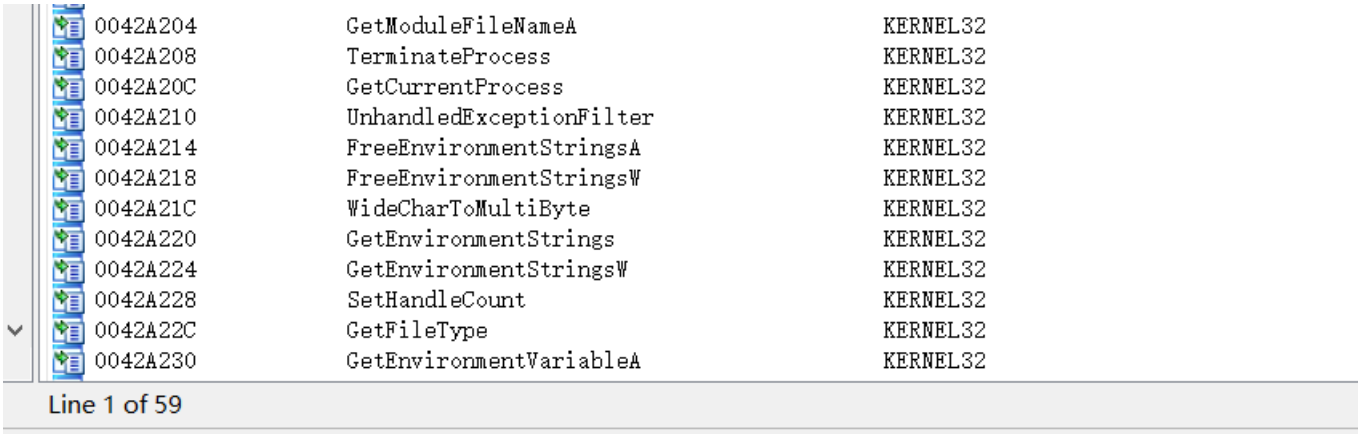
`Hex_View`窗口也称十六进制窗口，相当于一个十六进制的编辑器，可以直接对代码和数据进行修改，用户可以同时打开多个十六进制窗口。



`Exports`窗口为`导出函数窗口`，列出了被载入文件的所有导出函数，若载入文件没有抹去符号，很多时候用户可以直接在导出函数列表中找到入口函数。



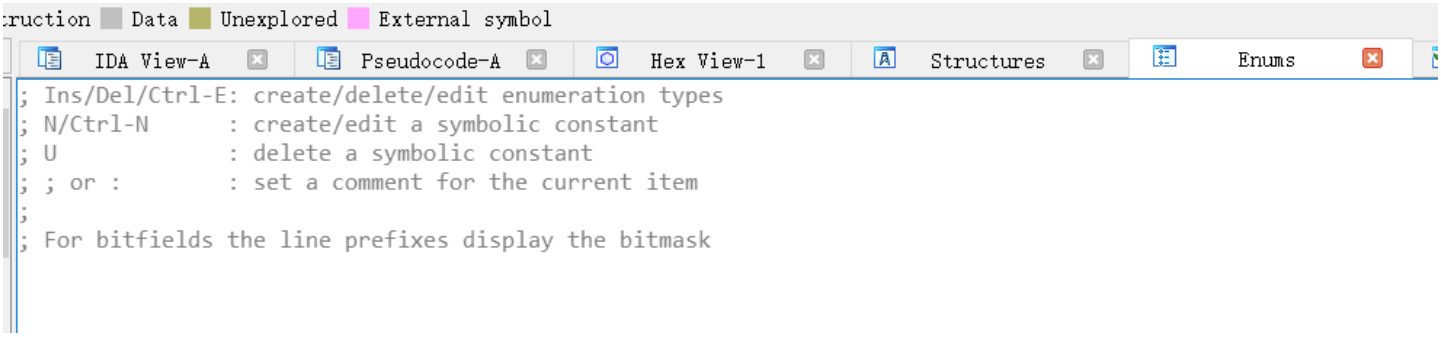
Address	Ordinal	Name	Library
0042A1AC		ExitProcess	KERNEL32
0042A1B0		Sleep	KERNEL32
0042A1B4		FlushFileBuffers	KERNEL32
0042A1B8		SetStdHandle	KERNEL32
0042A1BC		SetFilePointer	KERNEL32
0042A1C0		LCMapStringW	KERNEL32
0042A1C4		LCMapStringA	KERNEL32
0042A1C8		HeapReAlloc	KERNEL32
0042A1CC		VirtualAlloc	KERNEL32
0042A1D0		HeapAlloc	KERNEL32
0042A1D4		GetModuleHandleA	KERNEL32
0042A1D8		GetStartupInfoA	KERNEL32
0042A1DC		GetCommandLineA	KERNEL32
0042A1E0		GetVersion	KERNEL32
0042A1E4		DebugBreak	KERNEL32
0042A1E8		GetStdHandle	KERNEL32
0042A1EC		WriteFile	KERNEL32
0042A1F0		InterlockedDecrement	KERNEL32
0042A1F4		OutputDebugStringA	KERNEL32
0042A1F8		GetProcAddress	KERNEL32
0042A1FC		LoadLibraryA	KERNEL32
0042A200		InterlockedIncrement	KERNEL32



`Imports` 窗口为导入函数窗口，它会列出被分析的二进制文件导入的所有函数。

[外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-LtSkDTlk-1572424879652)(https://i.loli.net/2019/02/20/5c6d32a9f2595.png)]

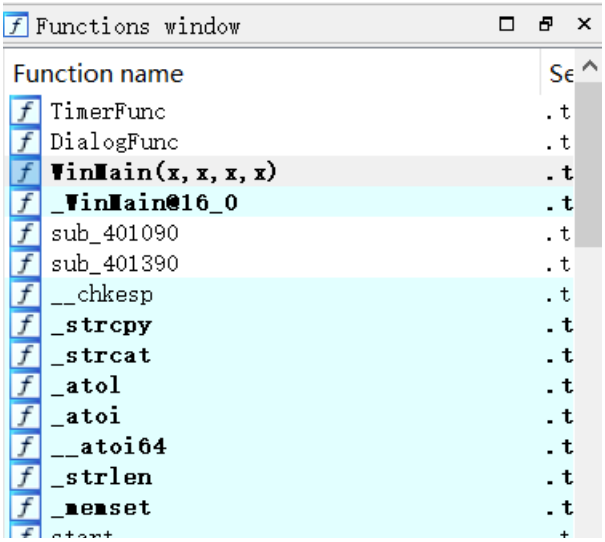
`Structures` 窗口，在分析阶段，IDA会查询它的函数类型签名扩展库，设法将函数的参数类型和程序使用的内存匹配起来。但在实际分析过程中经常会遇到IDA无法自动识别的数据结构，此时就需要用户自己判断内存结构的类型，若该数据为结构体，可以在 `Structures窗口` 创建一个 `自定义结构体`，然后将对应内存数据解析成该结构体，方便后续地分析。如下图所示为 `Structures窗口` 中列出的一些快捷键用来创建结构体。在创建结构体后，可以在需要应用该结构体的数据地址处，使用ALT + Q快捷键列出 创建的自定义的结构体。更加详细地创建和使用结构体 的方法可以参见 `《IDA Pro权威指南 第2版 》` 的第8章。

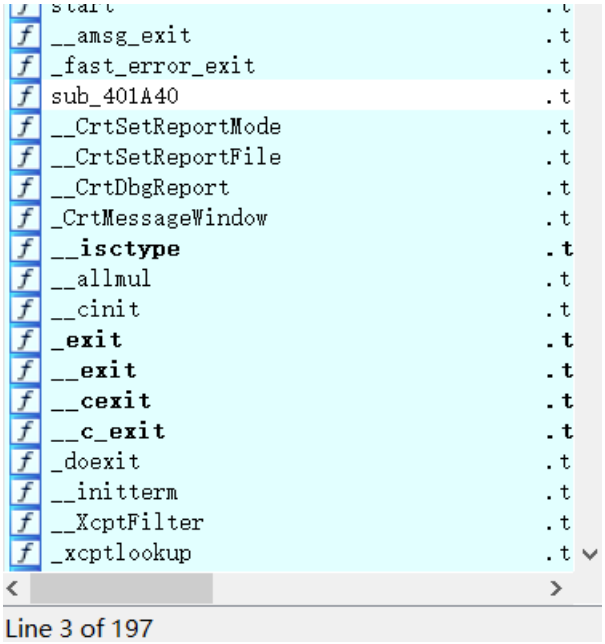


`Enums` 窗口，枚举窗口与结构体窗口类似，用户同样可以创建自定义联合体。如下图所示为Enums 窗口中列出的一些快捷键用来创建联合体。

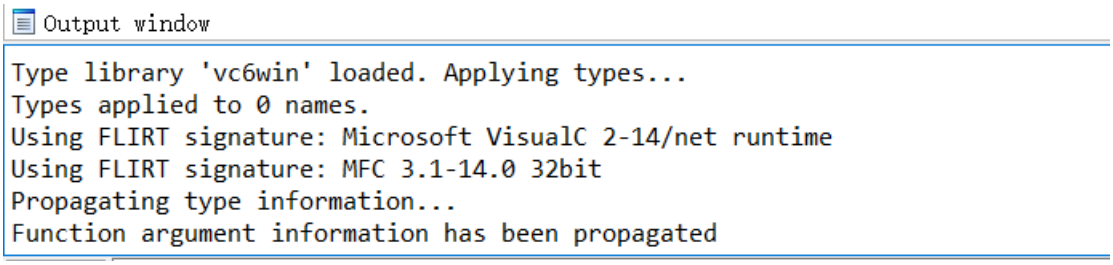
[外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-mQFAS5ft-1572424879653)(https://i.loli.net/2019/02/20/5c6d33635a15b.png)]

`Strings` 窗口显示了从被分析的二进制文件中提取出的字符串以及字符串所在的地址。在静态分析过程中，分析人员经常使用的一种方法就是字符串定位法，通过在Strings窗口中搜索一些特定的字符串，然后通过字符串的引用能够快速定位到关键的代码逻辑。





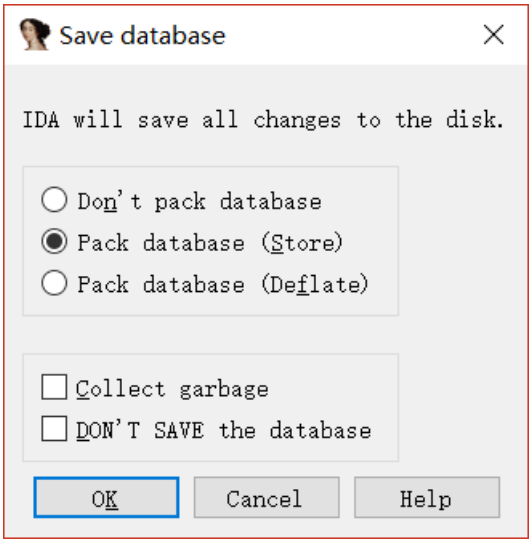
`functions`窗口列举出了IDA识别出的所有函数，双击选定函数条目，反汇编窗口会跳转到选定函数所在的起始地址处，在分析过程中能够快速定位到指定函数。



输出信息窗口，输出窗口将输出你所执行的各种操作的状态。

输出窗口下面的输入框在旧版本的`IDA pro`里是用于执行IDC脚本命令，新版开始支持`python`，如果你想用`IDC语法`，点击一下左下角`python按钮`切换到`IDC模式`即可。

在关闭IDA程序或者打开其他数据库时，IDA都会弹出下图所示的保存数据库的对话框。



若勾选`Pack database (Store)`，会将之前分析过程中生成的四个数据库组件文件打包生成一个idb文件，并且删除四个数据库组件文件，若原目录下已有该分析文件对应的idb文件，会直接覆盖。在下次继续分析该文件时可以直接用IDA打开该`idb文件`。当用户不需要保存数据库时，可以勾选`DON'T SAVE the database`选项，此时仍然会删除四个数据库组件文件，但不会生成或者覆盖`idb文件`。

IDA常用快捷键功能

`空格键`:反汇编窗口切换文本跟图形
`Esc`：在反汇编窗口中使用为后退到上个操作的地址处
`Shift +F5`:打开签名窗口
`shift+F12`:自动分析出参考字符串
`ALT+T`:搜索字符串(文本搜索)
`ALT+L`:标记(Lable)
`ALT+M`:设置标签(mark)
`ALT+G`:转换局部变量为结构体
`ALT+Enter`:跳转到新的窗口
`Alt+B`:快捷键用于搜索十六进制字节序列，通常在分析过程中可以用来搜索opcode
`CTRL+M`:列举出当前已经添加的标签
`CTRL+S`列举出二进制程序的段的开始地址、结束地址、权限等信息
`F9`:动态调试程序(其实IDA主要用作静态分析用的)
`F5`:将一个函数逆向出来(生成c伪代码)
`G`:跳转到指定地址
`A`:将选择的信息转换成ASCII(转换成可读性跟强的字符串)
`X(ctrl+X)`:`交叉引用`,类似于OD中的栈回溯操作
`N`:对符号重命名
`:&;(冒号&分号)`:光标所在位置添加常规注释和可重复注释
`P`:创建函数
`T`:解析结构体偏移
`M`:转换为枚举类型常量
`Y`:设置变量类型
`H`:转换16进制
`C`:光标所在地址处的内容解析成代码
`D`:光标所在地址处的内容解析成数据
`A`:光标所在地址处的内容解析成ascll码字符串
`U`:光标所在地址处的内容解析成未定义内容。

总结

IDA是一款功能很丰富很复杂很强大的工具，应用十分广泛。逆向工程的难点除了工具使用，更多的是分析方法和调试技巧。童鞋们如果想进一步深入学习，建议可以先了解常见的汇编指令和C语言，可以多尝试自己写写程序并锻炼动态调试能力。对于分析方法的训练，建议大家可以自己对照着源码和反汇编以后的代码去理解。更深层次的就是需要了解微机原理和程序设计方法，熟悉win api这些，这两年在CTF中linux `ELF` 文件的逆向居多，win可执行程序较少，对于脱壳等也较少，更多倾向于算法逆向分析对于逆向技巧减少了很多。但是实际应用中，这些都是必备技能。希望大家能在2019学到更多知识，变的越来越牛逼。相关工具资源大家可以去 `看雪` 、 `吾爱破解` 论坛去找，也欢迎各位大佬在评论区留言分享经验技巧和资源。

这篇文章是之前写的，给小白简单介绍，大概对IDA有个认识，之后会把IDA进阶的使用发出来，如何快速定位关键函数地址，静态分析恶意代码。

博客原文地址