三、逆向分析工具

动态调试

哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

- 动态分析技术最重要的工具是调试器
- 用户模式：OllyDbg，x64dbg, Visual C++等，调试应用程序
- 内核模式：WinDbg，调试操作系统内核

OllyDbg 简介

哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

- OllyDbg简称OD，是一款具有可视化界面的用户模式调试器
- 结合了动态调试和静态分析，具有强大的反汇编引擎
- 能够识别数千个被C和Windows所使用的函数，并能将其参数注释出，能自动分析函数过程、循环语句、代码中的字符串等
- 开放式设计赋予了强大的生命力

OllyDbg 特性

哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

- **支持的处理器：** 支持所有 80x86、奔腾、MMX、3DNOW！、Athlon 扩展指令集、SSE指令集以及相关的数据格式，但是不支持SSE2指令集
- **配置：** 有多达百余个选项用来设置外观和运行
- **数据格式：**能够显示的所有数据格式：HEX、ASCII、UNICODE、 16/32位有/无符号/HEX整数、32/64/80位浮点数、地址、反汇编（MASM、IDEAL或是HLA）、PE文件头或线程数据块
- **调试DLLs：** 可以调试标准动态链接库 (DLLs)。会自动运行一个可执行程序。这个程序会加载链接库，并允许调用链接库的输出函数
- **源码级调试：** 可以识别所有 Borland 和 Microsoft 格式的调试信息。这些信息包括源代码、函数名、标签、全局变量、静态变量。有限度的支持动态（栈）变量和结构

OllyDbg 特性

哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

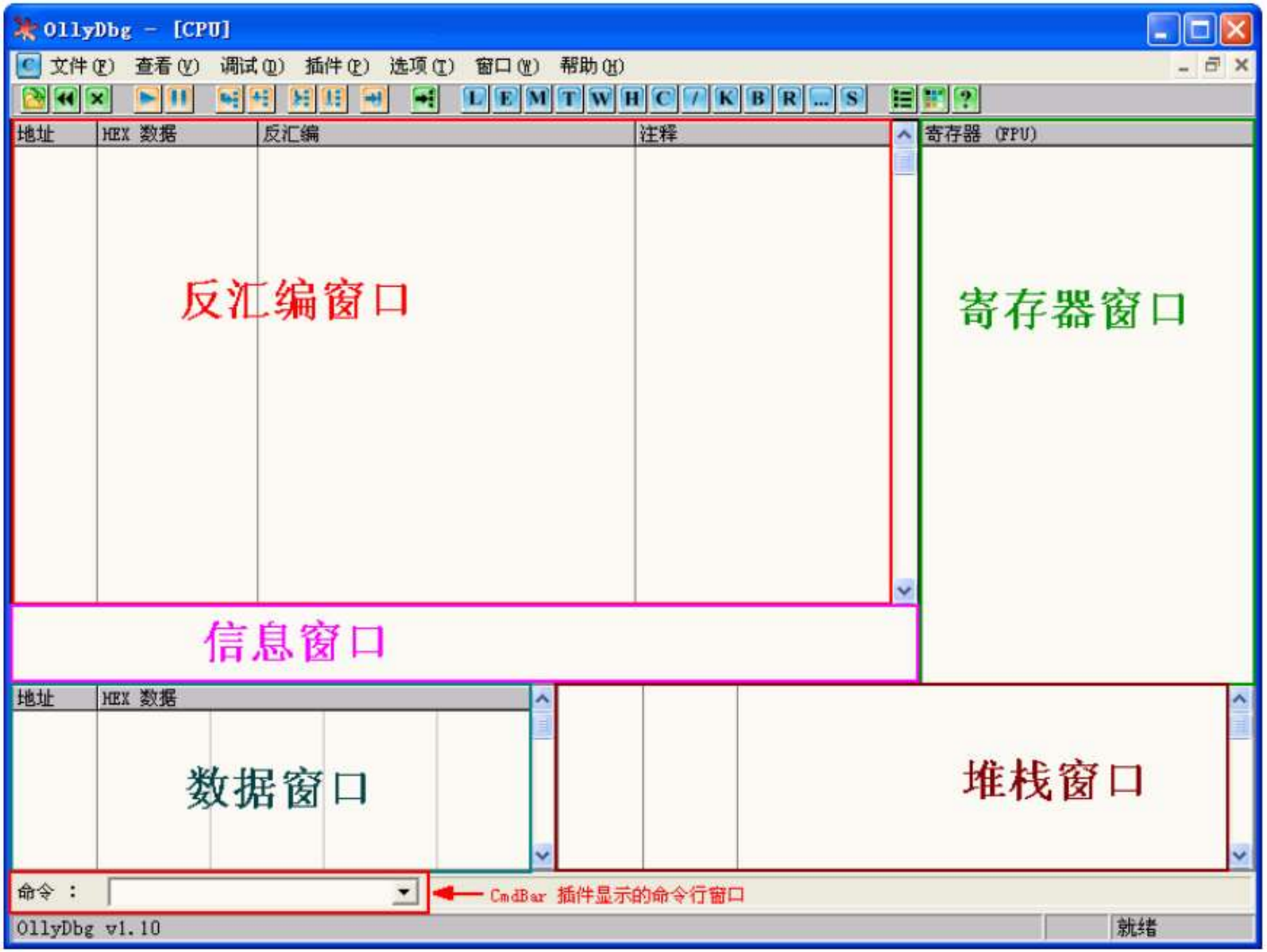
- **代码高亮：** 可以高亮不同类型的指令（如：跳转、条件跳转、入栈、出栈、调用、返回、特殊的或是无效的指令）和不同的操作数（常规、FPU/SSE、段/系统寄存器、在栈或内存中的操作数，常量），可以定制个性化高亮方案
- **分析：** 分析函数过程、循环语句、选择语句、表、常量、代码中的字符串、欺骗性指令、API调用、函数中参数的数目，import表等。这些分析增加了二进制代码的可读性，减少了出错的可能性，使得调试工作更加容易
- **名称：** 可以根据 Borland 和 Microsoft 格式的调试信息，显示输入/输出符号及名称。可以识别库函数。其中的名称和注释可任意添加。如果DLL中的某些函数是通过索引号输出的，则可通过挂接输入库来恢复原来的函数名称。不仅如此，OllyDbg还能识别大量的常量符号名（如：窗口消息、错误代码、位域...）并能够解码为已知的函数调用

OllyDbg 特性

哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

- **已知函数：**可以识别 2300 多个 C 和 Windows API 中的常用函数及其使用的参数。可以添加描述信息、预定义解码。还可以在已知函数设定 Log 断点并可以对参数进行记录
- **栈：**在栈窗口中，能智能识别返回地址和栈框架。并会留下一些先前的调用。如果程序停在已知函数上，堆栈窗口将会对其参数进行分析解码
- **窗口：**能够列出关于调试程序中的各种窗口，并且可以在窗口、类甚至选定的消息上设置断点
- **断点：** 支持各种断点：一般断点、条件断点、记录断点（比如记录函数参数到记录窗口）、内存读写断点、硬件断点（只适用于ME/NT/2000）等。在Hit跟踪情况下，可以在模块的每条命令上都设置INT3断点
- **句柄：**在基于NT的系统中，OllyDbg 可列出被调试程序的所有系统句柄

OllyDbg 主界面


OllyDbg - [CPU]
文件(F) 查看(V) 调试(D) 插件(P) 选项(T) 窗口(W) 帮助(H)
地址 HEX 数据 反汇编 注释
寄存器 (FPU)
反汇编窗口
寄存器窗口
信息窗口
地址 HEX 数据
数据窗口
堆栈窗口
命令 :
CmdBar 插件显示的命令行窗口
OllyDbg v1.10
就绪

OllyDbg 主界面

哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

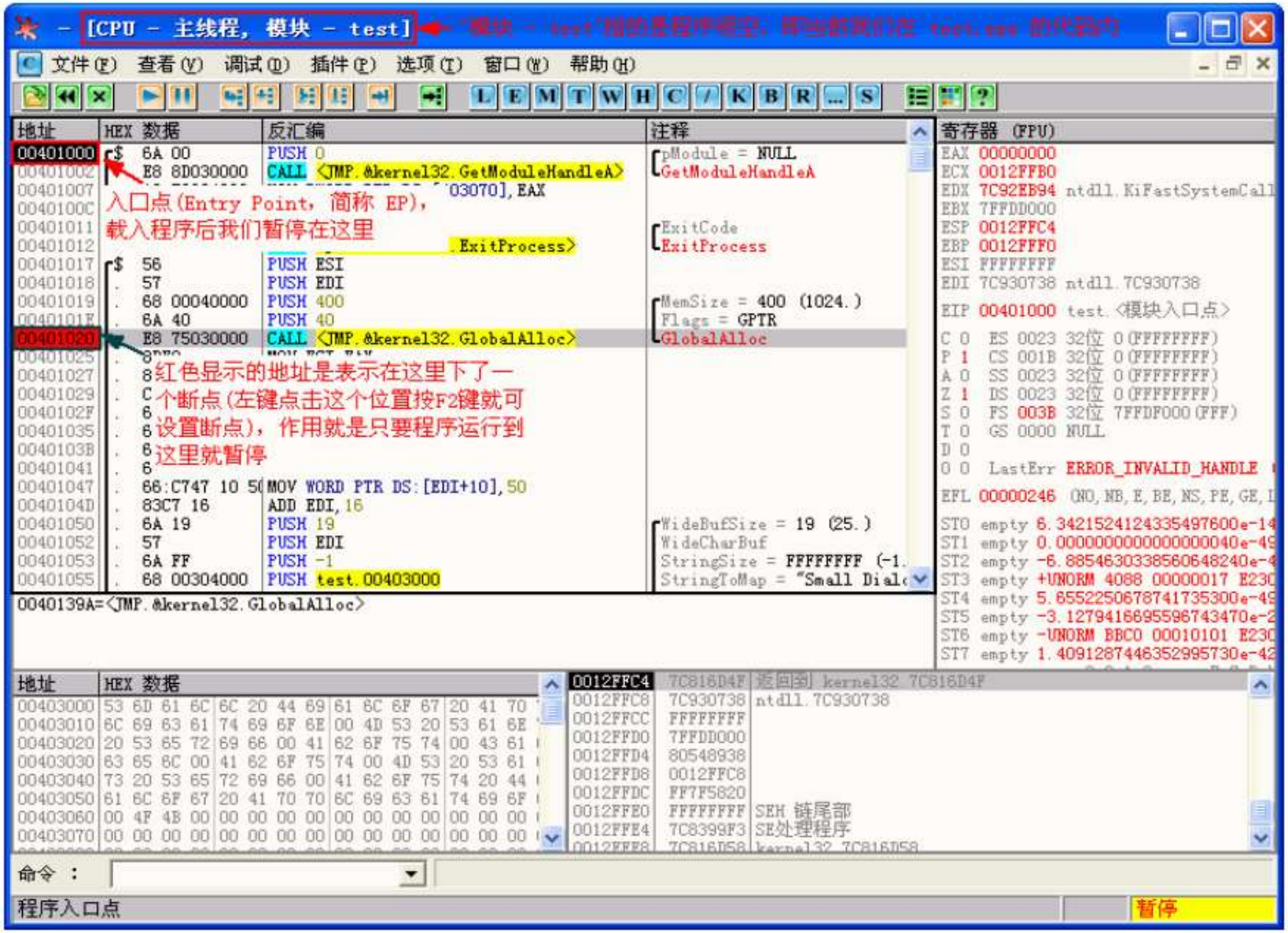
- **反汇编窗口**：显示被调试程序的反汇编代码，标题栏上的地址、HEX 数据、反汇编、注释可以通过在窗口中右击出现的菜单。用鼠标左键点击注释标签可以切换注释显示的方式。
  - 地址列：显示相对被单击地址的地址，再次双击返回到标准地址模式；
  - Hex数据列：设置或取消无条件断点，对应的快捷键是F2键；
  - 反汇编列：调用汇编器，可直接修改汇编代码；
  - 注释列：允许增加或编辑注释，对应快捷键是“；“键
- **寄存器窗口**：显示当前所选线程的 CPU 寄存器内容。同样点击标签寄存器 (FPU) 可以切换显示寄存器的方式
- **信息窗口**：显示反汇编窗口中选中的第一个命令的参数及一些跳转目标地址、字串等
- **数据窗口**：显示内存或文件的内容。右键菜单可用于切换显示方式
- **堆栈窗口**：显示当前线程的堆栈

OllyDbg 快捷键

哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

Function	Menu	Hotkey	Button
Run/Play	Debug ▸ Run	F9	
Pause	Debug ▸ Pause	F12	
Run to selection	Breakpoint ▸ Run to Selection	F4	
Run until return	Debug ▸ Execute till Return	CTRL-F9	
Run until user code	Debug ▸ Execute till User Code	ALT-F9	
Single-step/step-into	Debug ▸ Step Into	F7	
Step-over	Debug ▸ Step Over	F8	

基本调试方法


[CPU - 主线程, 模块 - test]
文件(F) 查看(V) 调试(D) 插件(P) 选项(T) 窗口(W) 帮助(H)
地址 HEX 数据 反汇编 注释
00401000 $ 6A 00 PUSH 0
00401002 E8 8D030000 CALL <JMP.&kernel32.GetModuleHandleA>
00401007 03070],EAX
0040100C
00401011
00401012 .ExitProcess>
00401017 $ 56 PUSH ESI
00401018 . 57 PUSH EDI
00401019 . 68 00040000 PUSH 400
0040101E . 6A 40 PUSH 40
00401020 E8 75030000 CALL <JMP.&kernel32.GlobalAlloc>
00401025
00401027
00401029
0040102F
00401035
0040103B
00401041
00401047 . 66:C747 10 50 MOV WORD PTR DS:[EDI+10],50
0040104D . 83C7 16 ADD EDI,16
00401050 . 6A 19 PUSH 19
00401052 . 57 PUSH EDI
00401053 . 6A FF PUSH -1
00401055 . 68 00304000 PUSH test.00403000
pModule = NULL
GetModuleHandleA
ExitCode
ExitProcess
MemSize = 400 (1024.)
Flags = GPTR
GlobalAlloc
WideBufSize = 19 (25.)
WideCharBuf
StringSize = FFFFFFFF (-1.
StringToMap = "Small Dial
入口点(Entry Point，简称 EP)，
载入程序后我们暂停在这里
红色显示的地址是表示在这里下了一
个断点(左键点击这个位置按F2键就可
设置断点)，作用就是只要程序运行到
这里就暂停
0040139A=<JMP.&kernel32.GlobalAlloc>
寄存器 (FPU)
EAX 00000000
ECX 0012FFB0
EDX 7C92EB94 ntdll.KiFastSystemCall
EBX 7FFDD000
ESP 0012FFC4
EBP 0012FFF0
ESI FFFFFFFF
EDI 7C930738 ntdll.7C930738
EIP 00401000 test.<模块入口点>
C 0 ES 0023 32位 0(FFFFFFFF)
P 1 CS 001B 32位 0(FFFFFFFF)
A 0 SS 0023 32位 0(FFFFFFFF)
Z 1 DS 0023 32位 0(FFFFFFFF)
S 0 FS 003B 32位 7FFDF000(FFF)
T 0 GS 0000 NULL
D 0
O 0 LastErr ERROR_INVALID_HANDLE
EFL 00000246 (NO,NB,E,BE,NS,PE,GE,L
ST0 empty 6.342152412433549760e-14
ST1 empty 0.00000000000000000040e-49
ST2 empty -6.885463033856064824e-4
ST3 empty +UNORM 4088 00000017 E230
ST4 empty 5.655225067874173530e-49
ST5 empty -3.127941669559674347e-2
ST6 empty -UNORM BBC0 00010101 E230
ST7 empty 1.409128744635299573e-42
地址 HEX 数据
00403000 53 6D 61 6C 6C 20 44 69 61 6C 6F 67 20 41 70
00403010 6C 69 63 61 74 69 6F 6E 00 4D 53 20 53 61 6E
00403020 20 53 65 72 69 66 00 41 62 6F 75 74 00 43 61
00403030 63 65 6C 00 41 62 6F 75 74 00 4D 53 20 53 61
00403040 73 20 53 65 72 69 66 00 41 62 6F 75 74 20 44
00403050 61 6C 6F 67 20 41 70 70 6C 69 63 61 74 69 6F
00403060 00 4F 4B 00 00 00 00 00 00 00 00 00 00 00 00
00403070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0012FFC4 7C816D4F 返回到 kernel32 7C816D4F
0012FFC8 7C930738 ntdll.7C930738
0012FFCC FFFFFFFF
0012FFD0 7FFDD000
0012FFD4 80548938
0012FFD8 0012FFC8
0012FFDC FF7F5820
0012FFE0 FFFFFFFF SEH 链尾部
0012FFE4 7C8399F3 SE处理程序
0012FFE8 7C816D58 kernel32.7C816D58
命令 :
程序入口点
暂停

设置断点

Function	Right-click menu selection	Hotkey
Software breakpoint	Breakpoint ▸ Toggle	F2
Conditional breakpoint	Breakpoint ▸ Conditional	SHIFT-F2
Hardware breakpoint	Breakpoint ▸ Hardware, on Execution	
Memory breakpoint on access (read, write, or execute)	Breakpoint ▸ Memory, on Access	F2 (select memory)
Memory breakpoint on write	Breakpoint ▸ Memory, on Write	

基本调试方法

哈爾濱工業大學 HARBIN INSTITUTE OF TECHNOLOGY

设置断点

Address	Hex dump	Disassembly	
004013A0	$ 55	push	ebp
004013A1	. 8BEC	mov	ebp, esp
004013A3	. 6A FF	push	-1
[illegible]	. 68 D0404000	push	004040D0
004013AA	. 68 D41E4000	push	00401ED4

← 在此行按“F2”键设置断点

Paused L E M T W H

Address	Module	Active	Disassembly	Comment
004013AA	TraceMe	Always	push 00401ED4	

基本调试方法

哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

- **找到断点位置 （GetDlgItemTextA）**
  - Ctrl + G
  - Ctrl + N
  - Bp命令

常用断点

- **INT 3 断点（F2）**
  - CC
  - 设置无数个断点
  - 改变了原程序的机器码
  - 检测是否下了断点

```
FARPROC Uaddr ;
BYTE Mark = 0;
(FARPROC&) Uaddr =GetProcAddress ( LoadLibrary("user32.dll"), "MessageBoxA");
Mark = *((BYTE*)Uaddr);          //取 MessageBoxA 函数的第 1 个字节
if(Mark ==0xCC)                  //如果该字节为“CC”，则认为 MessageBoxA 函数被下断了
    return TRUE                  //发现断点
```

常用断点

哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

硬件断点

- 最多设置4个断点
- F4
- 更难检测
- 速度快

内存断点

- 对所设的地址赋予不可访问/不可写属性
- 可以对数据的内存访问下内存断点
- 也可以对代码下内存断点

```
mov   dword ptr  [405528], edx    ;对[405528]处的内存进行写入
mov   dword ptr  edx,[405528]     ;对[405528]处的内存进行读取
```

```
004013D0  8915 28554000  mov   dword ptr [405528], edx
```

- IDA

IDA Pro 简介

哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

- 设计哲学
  - Nothing beats the Human brain...
  - Time is the essence...
  - Static Disassembly is limited...
  - You know better...
  - Innovation must be cherished...
  - Security is of high priority for us...

IDA Pro 简介

哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

- IDA Pro是一款功能强大、操作复杂的交互式反汇编工具
- IDA最主要的特性是交互和多处理器
- IDA不是机器人，而是把锋利的匕首
- 支持常见处理器平台上的软件产品

IDA Pro 简介

哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

- **IDA是按照区块装载PE文件的，例如.text（代码块），.data（数据块），.rsrc（资源块），.idata（输入表），.edata（输出表）**

- **IDA的分析需要两个阶段：**
  - 第一阶段：程序与代码分离，标记函数及其参数，分析跳转、调用等指令
  - 第二阶段：识别文件的编译类型，装载编译器特征文件，函数赋名等

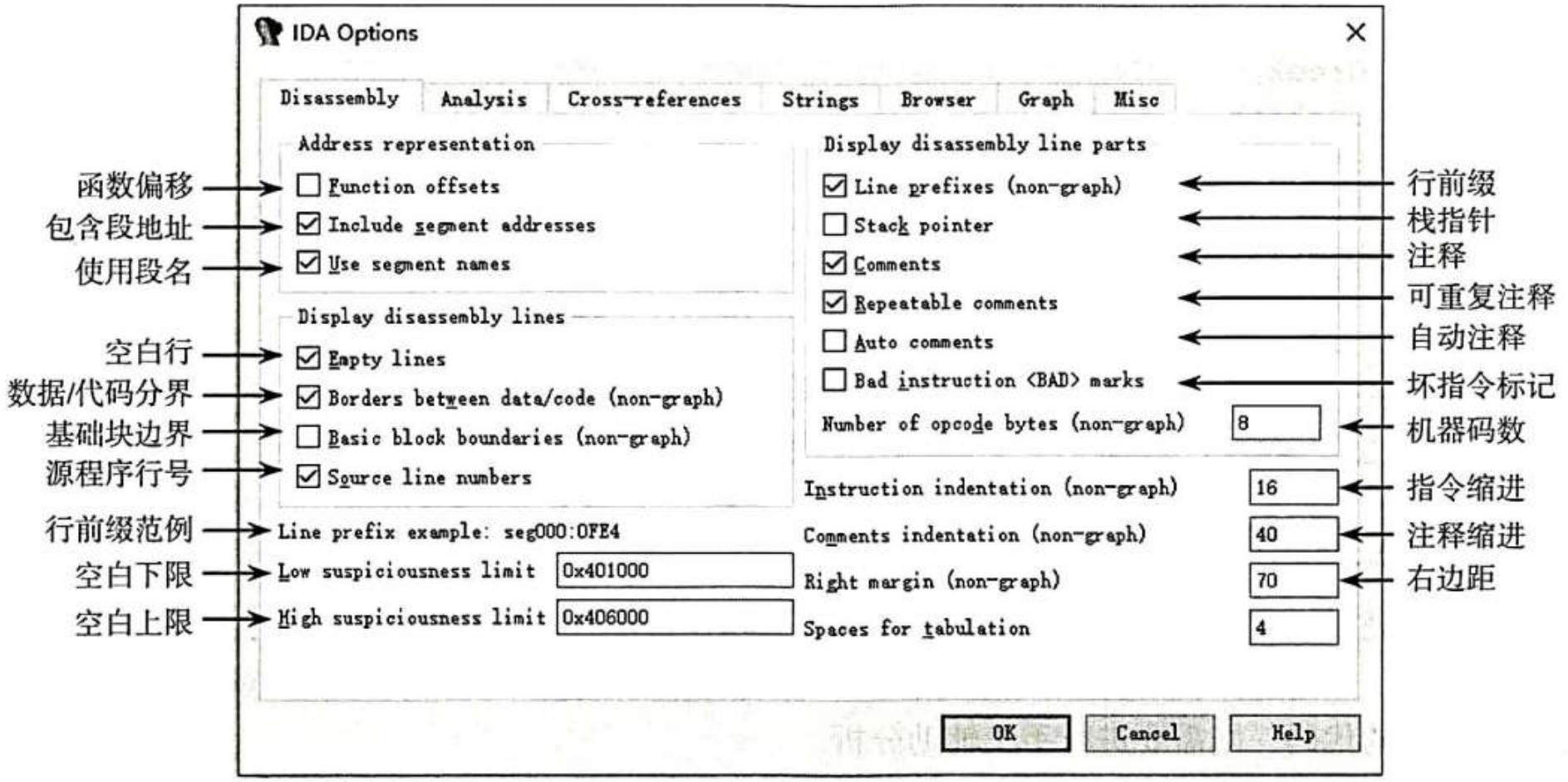
IDA配置

哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

- 合理配置IDA可以大大提供工作效率
- 通常，在软件内进行的配置仅对当前项目有效
- 要形成默认配置，需要修改ida.cfg文件

```
OPCODE_BYTES            = 6         ;机器码字节数，默认值是 0
INDENTION               = 0         ;指令缩进，默认值是 16，若设为 0 代码会整洁一些
COMMENTS_INDENTION      = 30        ;注释缩进，默认值是 40
MAX_TAIL                = 16        ;交叉参考的深度，默认值是 16
MAX_XREF_LENGTH         = 80        ;交叉参考显示的右边距
MAX_DATALINE_LENGTH     = 100       ;主窗口代码右边距，默认值是 70
SHOW_AUTOCOMMENTS       = NO        ;自动注释
SHOW_BAD_INSTRUCTIONS   = NO        ;坏指令标记
SHOW_BORDERS            = YES       ;数据与代码的分界
```

IDA配置


IDA Options
Disassembly
Analysis
Cross-references
Strings
Browser
Graph
Misc
Address representation
Function offsets
Include segment addresses
Use segment names
Display disassembly lines
Empty lines
Borders between data/code (non-graph)
Basic block boundaries (non-graph)
Source line numbers
Line prefix example: seg000:0FE4
Low suspiciousness limit
0x401000
High suspiciousness limit
0x406000
Display disassembly line parts
Line prefixes (non-graph)
Stack pointer
Comments
Repeatable comments
Auto comments
Bad instruction <BAD> marks
Number of opcode bytes (non-graph)
8
Instruction indentation (non-graph)
16
Comments indentation (non-graph)
40
Right margin (non-graph)
70
Spaces for tabulation
4
OK
Cancel
Help
函数偏移
包含段地址
使用段名
空白行
数据/代码分界
基础块边界
源程序行号
行前缀范例
空白下限
空白上限
行前缀
栈指针
注释
可重复注释
自动注释
坏指令标记
机器码数
指令缩进
注释缩进
右边距

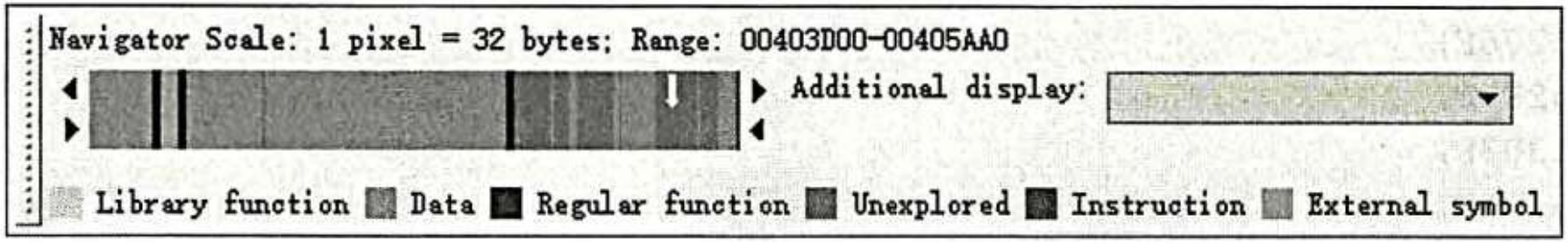
ASCII字符串与符号

```
ASCII_GENNAMES          = YES           ;生成符号名
ASCII_TYPE_AUTO         = YES           ;自动产生符号名
ASCII_PREFIX            = "a"           ;符号名前缀

#define ASCII_STYLE_C         0x00000000   ;ASCII 字符串
#define ASCII_STYLE_UNICODE   0x00000003   ;Unicode 字符串
```

IDA窗口

- **反汇编窗口**
  - 图形视图：流程图形式显示代码
  - 文本视图
  - 空格切换
  - 多窗口
- **导航栏**



IDA窗口

哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

- **注释**
  - ：单一注释
  - ；交叉注释
- **提示窗口**
- **字符串窗口**
- **输入窗口**
- **跳转到地址窗口**

IDA窗口

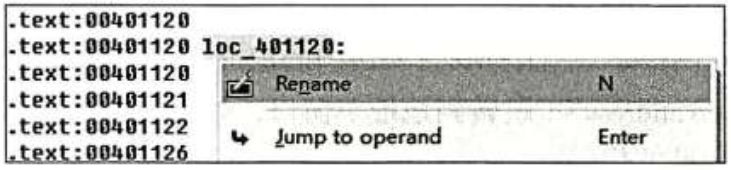
哈爾濱工業大學 HARBIN INSTITUTE OF TECHNOLOGY

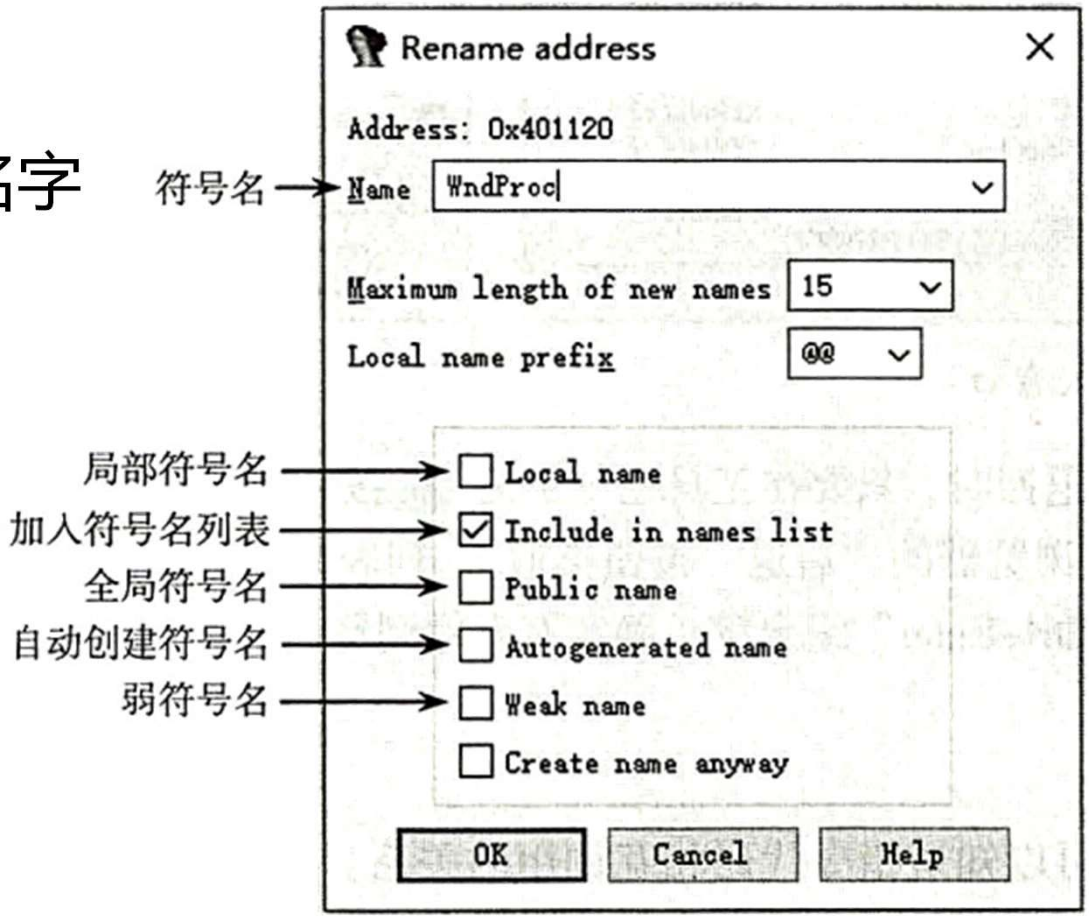
交叉参考

- 指令代码间相互调用的关系

参考重命名

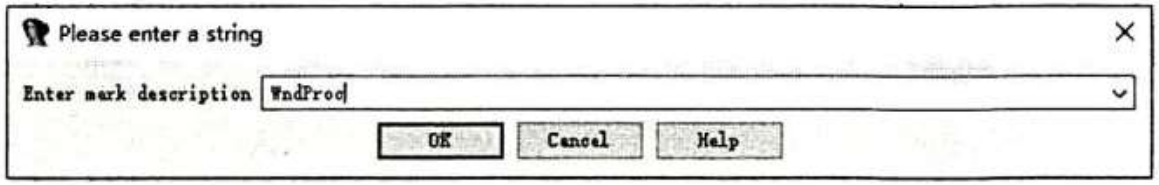
- 修改为更有意义的名字
- 提供可读性

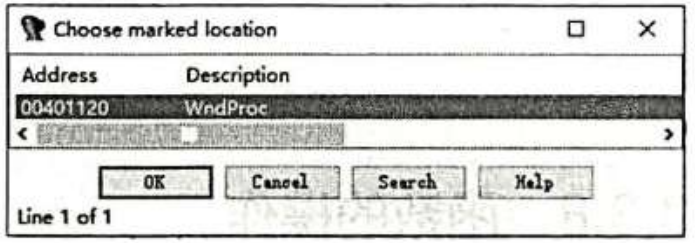




IDA窗口

标签


Please enter a string
Enter mark description
WndProc
OK
Cancel
Help


Choose marked location
Address
Description
00401120
WndProc
OK
Cancel
Search
Help
Line 1 of 1

格式化指令操作数

```
push    0                       2147483648                              H
push    esi                     20000000000o
push    0                       10000000000000000000000000000000b       B
push    0
push    80000000h               ; nHeight
push    80000000h               ; nWidth
push    80000000h               ; Y
push    80000000h               ; X
push    0CF0000h                ; dwStyle
push    offset WindowName ; "静态分析技术实例"
push    offset ClassName ; "chap231"
push    0                       ; dwExStyle
call    ds:CreateWindowExA
```

❖ 代码和数据转换

```
.text:00401000                          _WinMain@16:
.text:00401000 6A 00                                    push    0
.text:00401002 68 50 50 40 00                           push    offset Caption
.text:00401007 68 30 50 40 00                           push    offset Text
.text:0040100C 6A 00                                    push    0
.text:0040100E FF 15 94 40 40 00                        call    ds:MessageBoxA
.text:00401014 33 C0                                    xor     eax, eax
.text:00401016 C2 10 00                                 retn    10h
```

```
.text:00401000 6A                       _WinMain@16     db  6Ah ; j
.text:00401001 00                                       db    0
.text:00401002 68                                       db  68h ; h
.text:00401003 50                                       db  50h ; P
.text:00401004 50                                       db  50h ; P
.text:00401005 40                                       db  40h ; @
.text:00401006 00                                       db    0
.text:00401007 68                                       db  68h ; h
```