

逆向工程核心原理



[逆向工程核心原理 下载链接1](#)

著者:[韩] 李承远

出版者:人民邮电出版社

出版时间:2014-4-25

装帧:平装

isbn:9787115350183

本书十分详尽地介绍了代码逆向分析的核心原理。作者在Ahnlab

研究所工作多年，书中不仅包括其以此经验为基础亲自编写的大量代码，还包含了逆向工程研究人员必须了解的各种技术和技巧。彻底理解并切实掌握逆向工程这门技术，就能在众多IT 相关领域进行拓展运用，这本书就是通向逆向工程大门的捷径。

想成为逆向工程研究员的读者或正在从事逆向开发工作的开发人员一定会通过本书获得很大帮助。同时，想成为安全领域专家的人也可从本书轻松起步。

作者介绍:

李承远，目前在Ahnlab研究所从事恶意代码分析工作，并运营逆向工程的专门博客网站。

为把逆向工程技术广泛传播而不懈努力中并积极地的参加各种领域的活动，乐于与读者进行知识共享。

目录: 目 录

第一部分 代码逆向技术基础

第1章 关于逆向工程 2

1.1 逆向工程 2

1.2 代码逆向工程 2

1.2.1 逆向分析法 2

1.2.2 源代码、十六进制代码、汇编代码 4

1.2.3 “打补丁”与“破解” 5

1.3 代码逆向准备 5

1.3.1 目标 5

1.3.2 激情 6

1.3.3 谷歌 6

1.4 学习逆向分析技术的禁忌 6

1.4.1 贪心 6

1.4.2 急躁 7

1.5 逆向分析技术的乐趣 7

第2章 逆向分析Hello World!程序 8

2.1 Hello World!程序 8

2.2 调试HelloWorld.exe程序 9

2.2.1 调试目标 9

2.2.2 开始调试 9

2.2.3 入口点 10

2.2.4 跟踪40270C函数 10

2.2.5 跟踪40104F跳转语句 12

2.2.6 查找main()函数 12

2.3 进一步熟悉调试器 14

2.3.1 调试器指令 14

2.3.2 “大本营” 15

2.3.3 设置“大本营”的四种方法 15

2.4 快速查找指定代码的四种方法 17

2.4.1 代码执行法 18

2.4.2 字符串检索法 19

2.4.3 API检索法 (1)：在调用代码中设置断点 20

2.4.4 API检索法 (2)：在API代码中设置断点 21

2.5 使用“打补丁”方式修改“Hello World!”字符串 23

2.5.1 “打补丁” 23

2.5.2 修改字符串的两种方法 24

2.6 小结	28
第3章 小端序标记法	31
3.1 字节序	31
3.1.1 大端序与小端序	32
3.1.2 在OllyDbg中查看小端序	32
第4章 IA-32寄存器基本讲解	34
4.1 什么是CPU寄存器	34
4.2 IA-32寄存器	34
4.3 小结	40
第5章 栈	41
5.1 栈	41
5.1.1 栈的特征	41
5.1.2 栈操作示例	41
第6章 分析abex' crackme#1	44
6.1 abex' crackme #1	44
6.1.1 开始调试	45
6.1.2 分析代码	45
6.2 破解	47
6.3 将参数压入栈	47
6.4 小结	48
第7章 栈帧	49
7.1 栈帧	49
7.2 调试示例：stackframe.exe	49
7.2.1 StackFrame.cpp	50
7.2.2 开始执行main()函数&生成栈帧	51
7.2.3 设置局部变量	52
7.2.4 add()函数参数传递与调用	53
7.2.5 开始执行add()函数&生成栈帧	54
7.2.6 设置add()函数的局部变量 (x,y)	55
7.2.7 ADD运算	55
7.2.8 删除函数add()的栈帧&函数执行完毕 (返回)	56
7.2.9 从栈中删除函数add()的参数 (整理栈)	57
7.2.10 调用printf()函数	58
7.2.11 设置返回值	58
7.2.12 删除栈帧&main()函数终止	58
7.3 设置OllyDbg选项	59
7.3.1 Disasm选项	59
7.3.2 Analysis1选项	60
7.4 小结	61
第8章 abex' crackme #2	62
8.1 运行abex' crackme #2	62
8.2 Visual Basic文件的特征	63
8.2.1 VB专用引擎	63
8.2.2 本地代码和伪代码	63
8.2.3 事件处理程序	63
8.2.4 未文档化的结构体	63
8.3 开始调试	63
8.3.1 间接调用	64
8.3.2 RT_MainStruct结构体	64
8.3.3 ThunRTMain()函数	65
8.4 分析crackme	65
8.4.1 检索字符串	65
8.4.2 查找字符串地址	66
8.4.3 生成Serial的算法	68

- 8.4.4 预测代码 69
- 8.4.5 读取Name字符串的代码 69
- 8.4.6 加密循环 70
- 8.4.7 加密方法 70
- 8.5 小结 72
- 第9章 Process Explorer——最优秀的进程管理工具 74
- 9.1 Process Explorer 74
- 9.2 具体有哪些优点呢 75
- 9.3 sysinternals 75
- 第10章 函数调用约定 76
- 10.1 函数调用约定 76
- 10.1.1 cdecl 76
- 10.1.2 stdcall 77
- 10.1.3 fastcall 78
- 第11章 视频讲座 79
- 11.1 运行 79
- 11.2 分析 79
- 11.2.1 目标 (1) : 去除消息框 79
- 11.2.2 打补丁 (1) : 去除消息框 81
- 11.2.3 目标 (2) : 查找注册码 83
- 11.3 小结 85
- 第12章 究竟应当如何学习代码逆向分析 86
- 12.1 逆向工程 86
- 12.1.1 任何学习都应当有目标 86
- 12.1.2 拥有积极心态 86
- 12.1.3 要感受其中的乐趣 86
- 12.1.4 让检索成为日常生活的一部分 87
- 12.1.5 最重要的是实践 87
- 12.1.6 请保持平和的心态 87
- 第二部分 PE文件格式
- 第13章 PE文件格式 90
- 13.1 介绍 90
- 13.2 PE文件格式 90
- 13.2.1 基本结构 91
- 13.2.2 VA&RVA 92
- 13.3 PE头 92
- 13.3.1 DOS头 93
- 13.3.2 DOS存根 94
- 13.3.3 NT头 94
- 13.3.4 NT头: 文件头 95
- 13.3.5 NT头: 可选头 97
- 13.3.6 节区头 101
- 13.4 RVA to RAW 104
- 13.5 IAT 105
- 13.5.1 DLL 105
- 13.5.2 IMAGE_IMPORT_DESCRIPTOR 107
- 13.5.3 使用notepad.exe练习 108
- 13.6 EAT 112
- 13.6.1 IMAGE_EXPORT_DIRECTORY 113
- 13.6.2 使用kernel32.dll练习 114
- 13.7 高级PE 116
- 13.7.1 PView.exe 116
- 13.7.2 Patched PE 117
- 13.8 小结 118

- 第14章 运行时压缩 121
 - 14.1 数据压缩 121
 - 14.1.1 无损压缩 121
 - 14.1.2 有损压缩 121
 - 14.2 运行时压缩器 122
 - 14.2.1 压缩器 122
 - 14.2.2 保护器 123
 - 14.3 运行时压缩测试 123
- 第15章 调试UPX压缩的notepad程序 127
 - 15.1 notepad.exe的EP代码 127
 - 15.2 notepad_upx.exe的EP代码 127
 - 15.3 跟踪UPX文件 129
 - 15.3.1 OllyDbg的跟踪命令 129
 - 15.3.2 循环 #1 129
 - 15.3.3 循环 #2 130
 - 15.3.4 循环 #3 131
 - 15.3.5 循环 #4 131
 - 15.4 快速查找UPX OEP的方法 132
 - 15.4.1 在POPAD指令后的JMP指令处设置断点 132
 - 15.4.2 在栈中设置硬件断点 133
 - 15.5 小结 133
- 第16章 基址重定位表 135
 - 16.1 PE重定位 135
 - 16.1.1 DLL/SYS 135
 - 16.1.2 EXE 136
 - 16.2 PE重定位时执行的操作 136
 - 16.3 PE重定位操作原理 138
 - 16.3.1 基址重定位表 138
 - 16.3.2 IMAGE_BASE_RELOCATION结构体 139
 - 16.3.3 基址重定位表的分析方法 139
 - 16.3.4 练习 141
- 第17章 从可执行文件中删除.reloc节区 142
 - 17.1 .reloc节区 142
 - 17.2 reloc.exe 142
 - 17.2.1 删除.reloc节区头 142
 - 17.2.2 删除.reloc节区 143
 - 17.2.3 修改IMAGE_FILE_HEADER 143
 - 17.2.4 修改IMAGE_OPTIONAL_HEADER 144
 - 17.3 小结 145
- 第18章 UPack PE文件头详细分析 146
 - 18.1 UPack说明 146
 - 18.2 使用UPack压缩notepad.exe 146
 - 18.3 使用Stud_PE工具 148
 - 18.4 比较PE文件头 148
 - 18.4.1 原notepad.exe的PE文件头 149
 - 18.4.2 notepad_upack.exe运行时压缩的PE文件头 149
 - 18.5 分析UPack的PE文件头 150
 - 18.5.1 重叠文件头 150
 - 18.5.2 IMAGE_FILE_HEADER.SizeOfOptionalHeader 150
 - 18.5.3 IMAGE_OPTIONAL_HEADER.NumberOfRvaAndSizes 152
 - 18.5.4 IMAGE_SECTION_HEADER 153
 - 18.5.5 重叠节区 155
 - 18.5.6 RVA to RAW 156
 - 18.5.7 导入表 (IMAGE_IMPORT_DESCRIPTOR array) 158

- 18.5.8 导入地址表 160
- 18.6 小结 161
- 第19章 UPack调试? 查找OEP 162
 - 19.1 OllyDbg运行错误 162
 - 19.2 解码循环 163
 - 19.3 设置IAT 165
 - 19.4 小结 166
- 第20章 “内嵌补丁” 练习 167
 - 20.1 内嵌补丁 167
 - 20.2 练习: Patchme 168
 - 20.3 调试: 查看代码流 168
 - 20.4 代码结构 172
 - 20.5 “内嵌补丁” 练习 173
 - 20.5.1 补丁代码要设置在何处呢 173
 - 20.5.2 制作补丁代码 175
 - 20.5.3 执行补丁代码 176
 - 20.5.4 结果确认 177
- 第三部分 DLL注入
- 第21章 Windows消息钩取 180
 - 21.1 钩子 180
 - 21.2 消息钩子 180
 - 21.3 SetWindowsHookEx() 181
 - 21.4 键盘消息钩取练习 182
 - 21.4.1 练习示例HookMain.exe 182
 - 21.4.2 分析源代码 185
 - 21.5 调试练习 187
 - 21.5.1 调试HookMain.exe 188
 - 21.5.2 调试Notepad.exe进程内的KeyHook.dll 190
 - 21.6 小结 192
- 第22章 恶意键盘记录器 194
 - 22.1 恶意键盘记录器的目标 194
 - 22.1.1 在线游戏 194
 - 22.1.2 网上银行 194
 - 22.1.3 商业机密泄露 194
 - 22.2 键盘记录器的种类与发展趋势 195
 - 22.3 防范恶意键盘记录器 195
 - 22.4 个人信息 195
- 第23章 DLL注入 197
 - 23.1 DLL注入 197
 - 23.2 DLL注入示例 198
 - 23.2.1 改善功能与修复Bug 198
 - 23.2.2 消息钩取 198
 - 23.2.3 API钩取 198
 - 23.2.4 其他应用程序 199
 - 23.2.5 恶意代码 199
 - 23.3 DLL注入的实现方法 199
 - 23.4 CreateRemoteThread() 199
 - 23.4.1 练习示例myhack.dll 199
 - 23.4.2 分析示例源代码 203
 - 23.4.3 调试方法 208
 - 23.5 Applnit_DLLs 210
 - 23.5.1 分析示例源码 211
 - 23.5.2 练习示例myhack2.dll 212
 - 23.6 SetWindowsHookEx() 214

- 23.7 小结 214
- 第24章 DLL卸载 216
 - 24.1 DLL卸载的工作原理 216
 - 24.2 实现DLL卸载 216
 - 24.2.1 获取进程中加载的DLL信息 219
 - 24.2.2 获取目标进程的句柄 220
 - 24.2.3 获取FreeLibrary() API地址 220
 - 24.2.4 在目标进程中运行线程 220
 - 24.3 DLL卸载练习 220
 - 24.3.1 复制文件及运行notepad.exe 220
 - 24.3.2 注入myhack.dll 221
 - 24.3.3 卸载myhack.dll 222
- 第25章 通过修改PE加载DLL 224
 - 25.1 练习文件 224
 - 25.1.1 TextView.exe 224
 - 25.1.2 TextView_patched.exe 225
 - 25.2 源代码 - myhack3.cpp 227
 - 25.2.1DllMain() 227
 - 25.2.2 DownloadURL() 228
 - 25.2.3 DropFile() 229
 - 25.2.4 dummy() 230
 - 25.3 修改TextView.exe文件的准备工作 231
 - 25.3.1 修改思路 231
 - 25.3.2 查看IDT是否有足够空间 231
 - 25.3.3 移动IDT 233
 - 25.4 修改TextView.exe 235
 - 25.4.1 修改导入表的RVA值 235
 - 25.4.2 删除绑定导入表 235
 - 25.4.3 创建新IDT 235
 - 25.4.4 设置Name、INT、IAT 236
 - 25.4.5 修改IAT节区的属性值 238
 - 25.5 检测验证 240
 - 25.6 小结 241
- 第26章 PE Tools 242
 - 26.1 PE Tools 242
 - 26.1.1 进程内存转储 243
 - 26.1.2 PE编辑器 245
 - 26.2 小结 245
- 第27章 代码注入 247
 - 27.1 代码注入 247
 - 27.2 DLL注入与代码注入 247
 - 27.3 练习示例 249
 - 27.3.1 运行notepad.exe 249
 - 27.3.2 运行CodeInjection.exe 249
 - 27.3.3 弹出消息框 250
 - 27.4 CodeInjection.cpp 250
 - 27.4.1 main()函数 251
 - 27.4.2 ThreadProc()函数 251
 - 27.4.3 InjectCode()函数 254
 - 27.5 代码注入调试练习 256
 - 27.5.1 调试notepad.exe 256
 - 27.5.2 设置OllyDbg选项 256
 - 27.5.3 运行CodeInjection.exe 257
 - 27.5.4 线程开始代码 258

27.6 小结	259
第28章 使用汇编语言编写注入代码	260
28.1 目标	260
28.2 汇编编程	260
28.3 OllyDbg的汇编命令	260
28.3.1 编写ThreadProc()函数	262
28.3.2 保存文件	265
28.4 编写代码注入程序	266
28.4.1 获取ThreadProc()函数的二进制代码	266
28.4.2 CodeInjection2.cpp	267
28.5 调试练习	270
28.5.1 调试notepad.exe	270
28.5.2 设置OllyDbg选项	270
28.5.3 运行CodeInjection2.exe	271
28.5.4 线程起始代码	272
28.6 详细分析	272
28.6.1 生成栈帧	272
28.6.2 THREAD_PARAM结构体指针	273
28.6.3 “User32.dll” 字符串	274
28.6.4 压入 “user32.dll” 字符串参数	274
28.6.5 调用LoadLibraryA(“user32.dll”)	275
28.6.6 “MessageBoxA” 字符串	276
28.6.7 调用GetProcAddress(hMod, “MessageBoxA”)	276
28.6.8 压入MessageBoxA()函数的参数 1 - MB_OK	277
28.6.9 压入MessageBoxA()函数的参数 2 - “ReverseCore”	277
28.6.10 压入MessageBoxA()函数的参数 3 - “www.reversecore.com”	278
28.6.11 压入MessageBoxA()函数的参数 4 -NULL	279
28.6.12 调用MessageBoxA()	279
28.6.13 设置ThreadProc()函数的返回值	280
28.6.14 删除栈帧及函数返回	280
28.7 小结	280
第四部分 API钩取	
第29章 API钩取：逆向分析之“花”	282
29.1 钩取	282
29.2 API是什么	282
29.3 API钩取	283
29.3.1 正常调用API	283
29.3.2 钩取API调用	284
29.4 技术图表	284
29.4.1 方法对象（是什么）	285
29.4.2 位置（何处）	285
29.4.3 技术（如何）	286
29.4.4 API	286
第30章 记事本WriteFile() API钩取	288
30.1 技术图表 - 调试技术	288
30.2 关于调试器的说明	289
30.2.1 术语	289
30.2.2 调试器功能	289
30.2.3 调试器的工作原理	289
30.2.4 调试事件	289
30.3 调试技术流程	290
30.4 练习	291
30.5 工作原理	293
30.5.1 栈	293

- 30.5.2 执行流 295
- 30.5.3 “脱钩” & “钩子” 295
- 30.6 源代码分析 295
 - 30.6.1 main() 296
 - 30.6.2 DebugLoop() 296
 - 30.6.3 EXIT_PROCESS_DEBUG_EVENT 298
 - 30.6.4 CREATE_PROCESS_DEBUG_EVENT-OnCreateProcess-DebugEvent() 298
 - 30.6.5 EXCEPTION_DEBUG_EVENT-OnException-DebugEvent() 300
- 第31章 关于调试器 305
 - 31.1 OllyDbg 305
 - 31.2 IDA Pro 305
 - 31.3 WinDbg 306
- 第32章 计算器显示中文数字 308
 - 32.1 技术图表 308
 - 32.2 选定目标API 309
 - 32.3 IAT钩取工作原理 312
 - 32.4 练习示例 314
 - 32.5 源代码分析 316
 - 32.5.1DllMain() 316
 - 32.5.2 MySetWindowTextW() 317
 - 32.5.3 hook_iat() 319
 - 32.6 调试被注入的DLL文件 322
 - 32.6.1 DllMain() 325
 - 32.6.2 hook_iat() 325
 - 32.6.3 MySetWindowTextW() 327
 - 32.7 小结 328
- 第33章 隐藏进程 329
 - 33.1 技术图表 329
 - 33.2 API代码修改技术的原理 329
 - 33.2.1 钩取之前 330
 - 33.2.2 钩取之后 330
 - 33.3 进程隐藏 332
 - 33.3.1 进程隐藏工作原理 332
 - 33.3.2 相关API 332
 - 33.3.3 隐藏技术的问题 333
 - 33.4 练习 #1 (HideProc.exe, stealth.dll) 333
 - 33.4.1 运行notepad.exe、procexp.exe、taskmgr.exe 334
 - 33.4.2 运行HideProc.exe 334
 - 33.4.3 确认stealth.dll注入成功 334
 - 33.4.4 查看notepad.exe进程是否隐藏成功 335
 - 33.4.5 取消notepad.exe进程隐藏 336
 - 33.5 源代码分析 336
 - 33.5.1 HideProc.cpp 336
 - 33.5.2 stealth.cpp 338
 - 33.6 全局API钩取 344
 - 33.6.1 Kernel32.CreateProcess() API 344
 - 33.6.2 Ntdll.ZwResumeThread() API 345
 - 33.7 练习#2 (HideProc2.exe,Stealth2.dll) 345
 - 33.7.1 复制stealth2.dll文件到%SYSTEM%文件夹中 345
 - 33.7.2 运行HideProc2.exe -hide 346
 - 33.7.3 运行ProcExp.exe¬epad.exe 346
 - 33.7.4 运行HideProc2.exe -show 347
 - 33.8 源代码分析 348
 - 33.8.1 HideProc2.cpp 348

- 33.8.2 stealth2.cpp 348
- 33.9 利用“热补丁”技术钩取API 350
 - 33.9.1 API代码修改技术的问题 350
 - 33.9.2 “热补丁”（修改7个字节代码） 350
- 33.10 练习 #3: stealth3.dll 353
- 33.11 源代码分析 353
- 33.12 使用“热补丁”API钩取技术时需要考虑的问题 356
- 33.13 小结 357
- 第34章 高级全局API钩取：IE连接控制 359
 - 34.1 目标API 359
 - 34.2 IE进程结构 361
 - 34.3 关于全局API钩取的概念 362
 - 34.3.1 常规API钩取 363
 - 34.3.2 全局API钩取 363
 - 34.4 ntdll!ZwResumeThread() API 364
 - 34.5 练习示例：控制IE网络连接 368
 - 34.5.1 运行IE 368
 - 34.5.2 注入DLL 369
 - 34.5.3 创建新选项卡 369
 - 34.5.4 尝试连接网站 370
 - 34.5.5 卸载DLL 371
 - 34.5.6 课外练习 372
 - 34.6 示例源代码 372
 - 34.6.1 DllMain() 372
 - 34.6.2 NewInternetConnectW() 373
 - 34.6.3 NewZwResumeThread() 374
 - 34.7 小结 375
- 第35章 优秀分析工具的五种标准 376
 - 35.1 工具 376
 - 35.2 代码逆向分析工程师 376
 - 35.3 优秀分析工具的五种标准 376
 - 35.3.1 精简工具数量 377
 - 35.3.2 工具功能简单、使用方便 377
 - 35.3.3 完全掌握各种功能 377
 - 35.3.4 不断升级更新 377
 - 35.3.5 理解工具的核心工作原理 377
 - 35.4 熟练程度的重要性 377
- 第五部分 64位&Windows内核6
- 第36章 64位计算 380
 - 36.1 64位计算环境 380
 - 36.1.1 64位CPU 380
 - 36.1.2 64位OS 381
 - 36.1.3 Win32 API 381
 - 36.1.4 WOW64 381
 - 36.1.5 练习：WOW64Test 384
 - 36.2 编译64位文件 385
 - 36.2.1 Microsoft Windows SDK（Software Development Kit） 386
 - 36.2.2 设置Visual C++ 2010 Express环境 386
- 第37章 x64处理器 389
 - 37.1 x64中新增或变更的项目 389
 - 37.1.1 64位 389
 - 37.1.2 内存 389
 - 37.1.3 通用寄存器 389

- 37.1.4 CALL/JMP指令 390
- 37.1.5 函数调用约定 391
- 37.1.6 栈 & 栈帧 392
- 37.2 练习：Stack32.exe & Stack64.exe 392
- 37.2.1 Stack32.exe 392
- 37.2.2 Stack64.exe 394
- 37.3 小结 397
- 第38章 PE32+ 398
- 38.1 PE32+ (PE+、PE64) 398
- 38.1.1 IMAGE_NT_HEADERS 398
- 38.1.2 IMAGE_FILE_HEADER 398
- 38.1.3 IMAGE_OPTIONAL_HEADER 399
- 38.1.4 IMAGE_THUNK_DATA 401
- 38.1.5 IMAGE_TLS_DIRECTORY 403
- 第39章 WinDbg 405
- 39.1 WinDbg 405
- 39.1.1 WinDbg的特征 405
- 39.1.2 运行WinDbg 406
- 39.1.3 内核调试 407
- 39.1.4 WinDbg基本指令 409
- 第40章 64位调试 411
- 40.1 x64环境下的调试器 411
- 40.2 64位调试 411
- 40.3 PE32：WOW64Test_x86.exe 413
- 40.3.1 EP代码 414
- 40.3.2 Startup代码 414
- 40.3.3 main()函数 415
- 40.4 PE32+：WOW64Test_x64.exe 416
- 40.4.1 系统断点 416
- 40.4.2 EP代码 417
- 40.4.3 Startup代码 418
- 40.4.4 main()函数 420
- 40.5 小结 423
- 第41章 ASLR 424
- 41.1 Windows内核版本 424
- 41.2 ASLR 424
- 41.3 Visual C++ 424
- 41.4 ASLR.exe 425
- 41.4.1 节区信息 426
- 41.4.2 IMAGE_FILE_HEADER\Characteristics 427
- 41.4.3 IMAGE_OPTIONAL_HEADER\DLL Characteristics 428
- 41.5 练习：删除ASLR功能 428
- 第42章 内核6中的会话 430
- 42.1 会话 430
- 42.2 会话0隔离机制 432
- 42.3 增强安全性 432
- 第43章 内核6中的DLL注入 433
- 43.1 再现DLL注入失败 433
- 43.1.1 源代码 433
- 43.1.2 注入测试 435
- 43.2 原因分析 436
- 43.2.1 调试 #1 436
- 43.2.2 调试 #2 438
- 43.3 练习：使CreateRemoteThread()正常工作 440

- 43.3.1 方法 #1: 修改CreateSuspended参数值 440
- 43.3.2 方法 #2: 操纵条件分支 441
- 43.4 稍作整理 443
- 43.5 InjectDll_new.exe 443
 - 43.5.1 InjectDll_new.cpp 443
 - 43.5.2 注入练习 446
- 第44章 InjDll.exe: DLL注入专用工具 448
 - 44.1 InjDll.exe 448
 - 44.1.1 使用方法 448
 - 44.1.2 使用示例 449
 - 44.1.3 注意事项 450
- 第六部分 高级逆向分析技术
- 第45章 TLS回调函数 452
 - 45.1 练习 #1: HelloTls.exe 452
 - 45.2 TLS 453
 - 45.2.1 IMAGE_DATA_DIRECTORY[9] 453
 - 45.2.2 IMAGE_TLS_DIRECTORY 454
 - 45.2.3 回调函数地址数组 454
 - 45.3 TLS回调函数 455
 - 45.4 练习 #2: TlsTest.exe 456
 - 45.4.1 DLL_PROCESS_ATTACH 457
 - 45.4.2 DLL_THREAD_ATTACH 457
 - 45.4.3 DLL_THREAD_DETACH 457
 - 45.4.4 DLL_PROCESS_DETACH 457
 - 45.5 调试TLS回调函数 458
 - 45.6 手工添加TLS回调函数 459
 - 45.6.1 修改前的原程序 460
 - 45.6.2 设计规划 460
 - 45.6.3 编辑PE文件头 461
 - 45.6.4 设置IMAGE_TLS_DIRECTORY结构体 463
 - 45.6.5 编写TLS回调函数 464
 - 45.6.6 最终完成 464
 - 45.7 小结 465
- 第46章 TEB 466
 - 46.1 TEB 466
 - 46.1.1 TEB结构体的定义 466
 - 46.1.2 TEB结构体成员 466
 - 46.1.3 重要成员 469
 - 46.2 TEB访问方法 470
 - 46.2.1 Ntdll.NtCurrentTeb() 470
 - 46.2.2 FS段寄存器 471
 - 46.3 小结 472
- 第47章 PEB 473
 - 47.1 PEB 473
 - 47.1.1 PEB访问方法 473
 - 47.1.2 PEB结构体的定义 474
 - 47.1.3 PEB结构体的成员 475
 - 47.2 PEB的重要成员 477
 - 47.2.1 PEB.BeingDebugged 478
 - 47.2.2 PEB.ImageBaseAddress 478
 - 47.2.3 PEB.Ldr 479
 - 47.2.4 PEB.ProcessHeap & PEB.NtGlobalFlag 480
 - 47.3 小结 480
- 第48章 SEH 481

- 48.1 SEH 481
- 48.2 SEH练习示例 #1 481
 - 48.2.1 正常运行 481
 - 48.2.2 调试运行 482
- 48.3 OS的异常处理方法 484
 - 48.3.1 正常运行时的异常处理方法 484
 - 48.3.2 调试运行时的异常处理方法 484
- 48.4 异常 485
 - 48.4.1 EXCEPTION_ACCESS_VIOLATION(C0000005) 486
 - 48.4.2 EXCEPTION_BREAKPOINT(80000003) 486
 - 48.4.3 EXCEPTION_ILLEGAL_INSTRUCTION(C000001D) 488
 - 48.4.4 EXCEPTION_INT_DIVIDE_BY_ZERO(C0000094) 488
 - 48.4.5 EXCEPTION_SINGLE_STEP(80000004) 489
- 48.5 SEH详细说明 489
 - 48.5.1 SEH链 489
 - 48.5.2 异常处理函数的定义 489
 - 48.5.3 TEB.NtTib.ExceptionList 491
 - 48.5.4 SEH安装方法 492
- 48.6 SEH练习示例 #2 (seh.exe) 492
 - 48.6.1 查看SEH链 493
 - 48.6.2 添加SEH 493
 - 48.6.3 发生异常 494
 - 48.6.4 查看异常处理器参数 494
 - 48.6.5 调试异常处理器 496
 - 48.6.6 删除SEH 498
- 48.7 设置OllyDbg选项 499
 - 48.7.1 忽略KERNEL32中发生的内存非法访问异常 500
 - 48.7.2 向被调试者派送异常 500
 - 48.7.3 其他异常处理 500
 - 48.7.4 简单练习 500
- 48.8 小结 501

第49章 IA-32指令 502

- 49.1 IA-32指令 502
- 49.2 常用术语 502
 - 49.2.1 反汇编器 503
 - 49.2.2 反编译器 504
 - 49.2.3 反编译简介 504
- 49.3 IA-32指令格式 506
 - 49.3.1 指令前缀 507
 - 49.3.2 操作码 507
 - 49.3.3 ModR/M 507
 - 49.3.4 SIB 508
 - 49.3.5 位移 508
 - 49.3.6 立即数 509
- 49.4 指令解析手册 509
 - 49.4.1 下载IA-32用户手册 509
 - 49.4.2 打印指令解析手册 509
- 49.5 指令解析练习 510
 - 49.5.1 操作码映射 510
 - 49.5.2 操作数 511
 - 49.5.3 ModR/M 512
 - 49.5.4 Group 514
 - 49.5.5 前缀 516
 - 49.5.6 双字节操作码 518

- 49.5.7 移位值&立即数 519
- 49.5.8 SIB 520
- 49.6 指令解析课外练习 524
- 49.7 小结 524
- 第七部分 反调试技术
- 第50章 反调试技术 526
 - 50.1 反调试技术 526
 - 50.1.1 依赖性 526
 - 50.1.2 多种反调试技术 526
 - 50.2 反调试破解技术 526
 - 50.3 反调试技术的分类 527
 - 50.3.1 静态反调试技术 528
 - 50.3.2 动态反调试技术 528
- 第51章 静态反调试技术 529
 - 51.1 静态反调试的目的 529
 - 51.2 PEB 529
 - 51.2.1 BeingDebugged(+0x2) 531
 - 51.2.2 Ldr(+0xC) 531
 - 51.2.3 Process Heap(+0x18) 532
 - 51.2.4 NtGlobalFlag(+0x68) 533
 - 51.2.5 练习：?StaAD_PEB.exe 534
 - 51.2.6 破解之法 534
 - 51.3 NtQueryInformationProcess() 537
 - 51.3.1 ProcessDebugPort(0x7) 538
 - 51.3.2 ProcessDebugObjectHandle(0x1E) 539
 - 51.3.3 ProcessDebugFlags(0x1F) 539
 - 51.3.4 练习：StaAD_NtQIP.exe 540
 - 51.3.5 破解之法 540
 - 51.4 NtQuerySystemInformation() 542
 - 51.4.1 SystemKernelDebugger-Information(0x23) 544
 - 51.4.2 练习：StaAD_NtQSI.exe 545
 - 51.4.3 破解之法 545
 - 51.5 NtQueryObject() 545
 - 51.6 ZwSetInformationThread() 549
 - 51.6.1 练习：StaAD_ZwSIT.exe 549
 - 51.6.2 破解之法 550
 - 51.7 TLS回调函数 550
 - 51.8 ETC 551
 - 51.8.1 练习：StaAD_FindWindow.exe 551
 - 51.8.2 破解之法 551
 - 51.9 小结 553
- 第52章 动态反调试技术 554
 - 52.1 动态反调试技术的目的 554
 - 52.2 异常 554
 - 52.2.1 SEH 554
 - 52.2.2 SetUnhandledException-Filter() 558
 - 52.3 Timing Check 562
 - 52.3.1 时间间隔测量法 562
 - 52.3.2 RDTSC 563
 - 52.4 陷阱标志 565
 - 52.4.1 单步执行 566
 - 52.4.2 INT 2D 569
 - 52.5 0xCC探测 572
 - 52.5.1 API断点 573

- 52.5.2 比较校验和 575
- 第53章 高级反调试技术 577
 - 53.1 高级反调试技术 577
 - 53.2 垃圾代码 577
 - 53.3 扰乱代码对齐 578
 - 53.4 加密/解密 581
 - 53.4.1 简单的解码示例 581
 - 53.4.2 复杂的解码示例 582
 - 53.4.3 特殊情况：代码重组 584
 - 53.5 Stolen Bytes (Remove OEP) 584
 - 53.6 API重定向 587
 - 53.6.1 原代码 588
 - 53.6.2 API重定向示例 #1 588
 - 53.6.3 API重定向示例#2 589
 - 53.7 Debug Blocker (Self Debugging) 593
 - 53.8 小结 595
- 第八部分 调试练习
- 第54章 调试练习1：服务 598
 - 54.1 服务进程的工作原理 598
 - 54.1.1 服务控制器 598
 - 54.1.2 服务启动过程 599
 - 54.2 DebugMe1.exe示例讲解 600
 - 54.2.1 安装服务 600
 - 54.2.2 启动服务 602
 - 54.2.3 源代码 604
 - 54.3 服务进程的调试 606
 - 54.3.1 问题在于SCM 606
 - 54.3.2 调试器无所不能 606
 - 54.3.3 常用方法 606
 - 54.4 服务调试练习 606
 - 54.4.1 直接调试：强制设置EIP 606
 - 54.4.2 服务调试的常用方法：“附加”方式 609
 - 54.5 小结 615
- 第55章 调试练习2：自我创建 616
 - 55.1 自我创建 616
 - 55.2 工作原理 617
 - 55.2.1 创建子进程（挂起模式） 617
 - 55.2.2 更改EIP 618
 - 55.2.3 恢复主线程 618
 - 55.3 示例程序源代码 618
 - 55.4 调试练习 620
 - 55.4.1 需要考虑的事项 620
 - 55.4.2 JIT调试 621
 - 55.4.3 DebugMe2.exe 622
 - 55.5 小结 626
- 第56章 调试练习3：PE映像切换 627
 - 56.1 PE映像 627
 - 56.2 PE映像切换 628
 - 56.3 示例程序：Fake.exe、Real.exe、DebugMe3.exe 628
 - 56.4 调试1 631
 - 56.4.1 Open? 输入运行参数 631
 - 56.4.2 main()函数 632
 - 56.4.3 SubFunc_1() 634
 - 56.4.4 CreateProcess(“fake.exe”，CREATE_SUSPENDED) 635

56.4.5 SubFunc_2() 635
56.4.6 SubFunc_3() 641
56.4.7 ResumeThread() 644
56.5 调试2 644
56.5.1 思考 645
56.5.2 向EP设置无限循环 645
56.6 小结 647
第57章 调试练习4: Debug Blocker 648
57.1 Debug Blocker 648
57.2 反调试特征 648
57.2.1 父与子的关系 649
57.2.2 被调试进程不能再被其他调试器调试 649
57.2.3 终止调试进程的同时也终止被调试进程 649
57.2.4 调试器操作被调试者的代码 649
57.2.5 调试器处理被调试进程中发生的异常 649
57.3 调试练习: DebugMe4.exe 650
57.4 第一次调试 650
57.4.1 选定调试的起始位置 650
57.4.2 main() 650
57.5 第二次调试 651
57.6 第三次调试 653
57.7 第四次调试 656
57.8 第五次调试 658
57.8.1 系统断点 658
57.8.2 EXCEPTION_ILLEGAL_INSTRUCTION(1) 659
57.8.3 EXCEPTION_ILLEGAL_INSTRUCTION(2) 660
57.9 第六次调试 661
57.9.1 40121D (第一个异常) 661
57.9.2 401299 (第二个异常) 665
57.10 第七次调试 667
57.10.1 静态方法 668
57.10.2 动态方法 669
57.11 小结 673
结束语 674
索引 676
• • • • • (收起)

[逆向工程核心原理 下载链接1](#)

标签

逆向工程

计算机

信息安全

编程

安全

黑客

程序设计

C++

评论

我的逆向第二本教程，也是令我开窍的一本。2016.2

作为入门书很不错…不过说实话逆向真的有点繁琐，很实战派，很不优雅

非常非常好的书，对Windows开发人员，中高级C++程序员是非常好的进阶书籍。

入门级好书

虽然有些杂乱，看起来是博客的集合。

见过。

零基础也可以手把手把你带进门，好长时间没读过这么痛快的好书了！

10分好书，事无巨细，结语很象征。

讲的比较细致

相当好，有些章节不免杂乱，内容很充实，深度广度都照顾到了

非常好！！！！

我非常喜欢这本书，章节短小精悍，注重实际操作，一般而言，这类书籍会从PE开始讲起，但是这本书不同，始终是以实际操作的角度来展开，十分注重读者的心理感受，为了读者能够理解并掌握逆向工程的原理，作者可谓煞费苦心。这本书是最值得推荐的逆向工程书籍之一，特别适合入门者，尽管这本书有600多页。

诚意之作

注定成为经典的书！

@.@

让我开窍的一本书

入门好书

windows类的逆向，不太深入和广泛。

棒子写的书，翻译的还不错，内容也挺好，逆向工程里面最经典的著作了。

没见过这么详细,循序渐进的技术书...

[逆向工程核心原理 下载链接1](#)

书评

非常非常好的书，对Windows开发人员，中高级C++程序员是非常好的进阶书籍。
这部书里面的每个章节细细挖掘都有非常丰富的内容。
这是我这两三年内读到的最好的技术书籍之一，作者功力深厚，翻译的也非常到位，人品推荐。

不错的书，我看了1/3了，看这本书最好有点汇编的基础知识，和C++还有windows编程等。否则苦果难咽。
.....

[逆向工程核心原理 下载链接1](#)