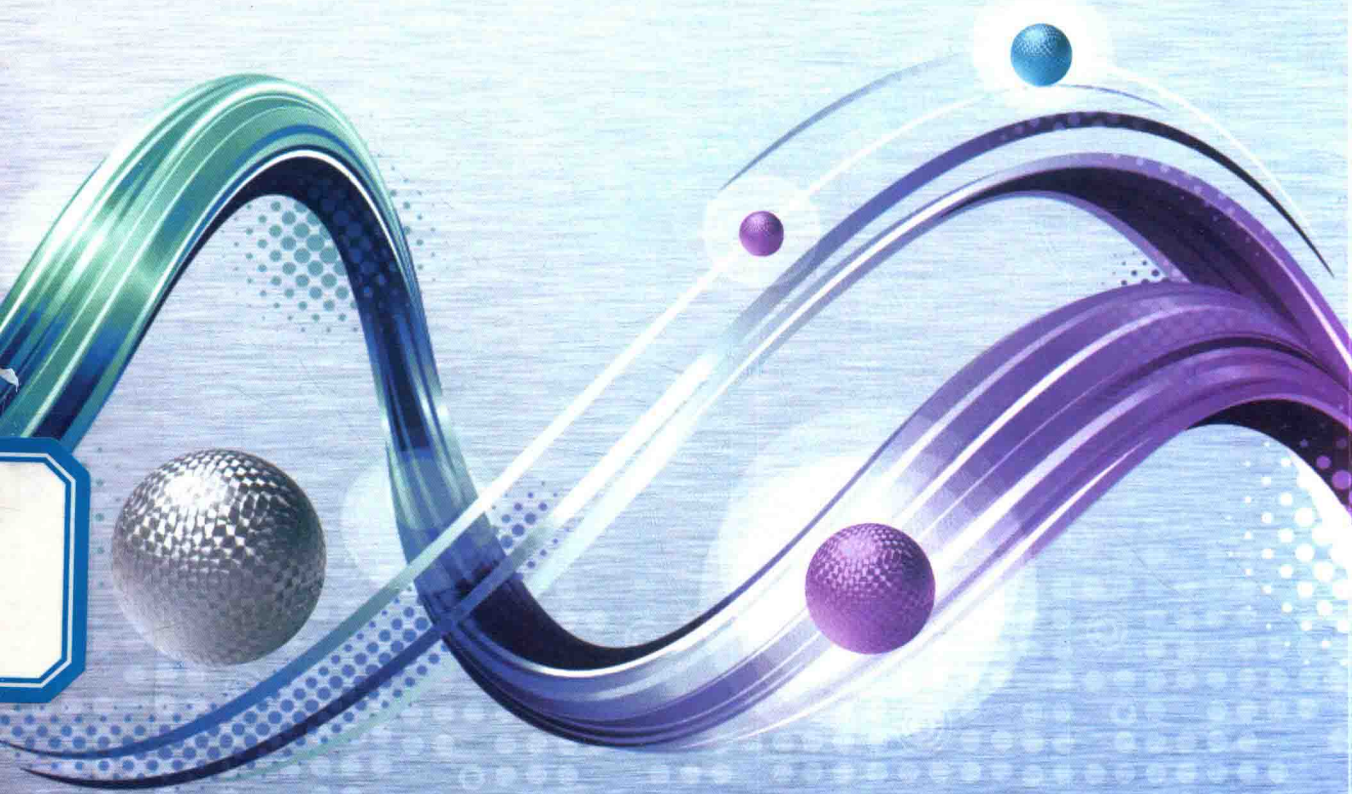
中国电子教育学会高教分会推荐
高等学校网络与信息安全类“十三五”规划教材

软件逆向工程
原理与实践

孙聪　李金库　马建峰　编著



西安电子科技大学出版社
http://www.xduph.com

中国电子教育学会高教分会推荐

高等学校网络与信息安全类“十三五”规划教材

软件逆向工程原理与实践

孙　聪　李金库　马建峰　编著

西安电子科技大学出版社

内 容 简 介

本书系统介绍了软件逆向工程基本原理和常见技术工具，并以主流的硬件架构和操作系统为背景，介绍了常见的软件逆向工程方法。本书的主要内容包括软件逆向工程概述、x86 与 x64 体系结构、ARM 体系结构、PE 文件格式、DLL 注入、API 钩取、代码混淆技术、Android 应用程序逆向分析和 ROP 攻击等。本书注重突出实用性和实践性。

本书适合作为高等院校信息安全、计算机等相关专业本科生或研究生的教材，也可供计算机软件相关技术领域的研究人员和工程人员参考使用。

图书在版编目(CIP)数据

软件逆向工程原理与实践/孙聪，李金库，马建峰编著. —西安：西安电子科技大学出版社，2018.2
ISBN 978-7-5606-2497-6

Ⅰ. ① 软… Ⅱ. ① 孙… ② 李… ③ 马 Ⅲ. ① 软件工程 Ⅳ. ① TP311.5

中国版本图书馆 CIP 数据核字(2018)第 016894 号

策　　划　刘玉芳　李惠萍
责任编辑　祝婷婷　李惠萍
出版发行　西安电子科技大学出版社(西安市太白南路 2 号)
电　　话　(029)88242885　88201467　　邮　　编　710071
网　　址　www.xduph.com　　电子邮箱　xdupfxb001@163.com
经　　销　新华书店
印刷单位　陕西华沐印刷科技有限责任公司
版　　次　2018 年 2 月第 1 版　2018 年 2 月第 1 次印刷
开　　本　787 毫米×1092 毫米　1/16　印 张 10
字　　数　231 千字
印　　数　1～2000 册
定　　价　22.00 元
ISBN 978 - 7 - 5606 - 2497 - 6/TP
XDUP 2789001-1

前　言

随着互联网的飞速发展和社会各领域信息化水平的大幅提高，网络空间安全已成为与广大人民群众息息相关的关键问题。2016 年 12 月，我国颁布了《国家网络空间安全战略》。该战略指出，建立和完善安全技术支撑体系和基础设施，实施网络安全人才工程，加强网络安全学科专业建设，是未来网络空间安全建设的战略任务之一。该战略还强调，应重视软件安全，加快安全可信产品的推广应用。软件逆向工程技术作为软件安全的核心技术之一，在网络空间安全保障中发挥着不可替代的作用。为了适应网络安全人才培养的需要，我们编写了这本适合网络空间安全专业教学要求的软件逆向工程教材。

长期以来，软件逆向工程就像是从脾气古怪的技师的工具箱中掏出的神奇工具一样，被人们看作是功能强大的“黑盒”。虽然国外在软件逆向工程方面有一些先进技术书籍，但大多内容庞杂，或者难度太大，不适宜作为基础性的教材使用。本书的编写思路，就是寻找当前各种先进的软件逆向分析方法所涉及的基础知识和共性技术，辅以简明的示例，为读者提供解决典型软件逆向工程问题的基本方法，并适当地介绍 Android 应用逆向分析、ROP 攻击等相关的前沿内容。

软件逆向工程是一门对计算机系统结构、操作系统内核编程、汇编语言、编译技术等方面的基础知识要求很高的课程。如何能够在紧密围绕软件逆向工程核心技术的前提下，对所涉及的关键性基础知识进行有针对性的介绍，是我们编写本书时所考虑和关注的问题。本书较系统地阐述了软件逆向工程（特别是 Windows 操作系统环境下的软件逆向工程）的基本内容和核心技术。全书共 9 章，分别介绍了软件逆向工程的基本概念、x86 和 x64 体系结构、ARM 体系结构、PE 文件格式、DLL 注入、API 钩取、代码混淆技术、Android 应用程序逆向分析和 ROP 攻击等内容。

本书具有以下特点：

(1) 实用性。软件逆向工程不是一门单纯的理论，它需要在学习过程中进行大量的编程和分析实践。本书在主要章节给出了很多可运行且有代表性的示例程序，以及分析工具和分析结果截图，引导读者动手实现一些典型的逆向分析过程。

(2) 选择的用例尽量简洁，减少学生理解和动手操作的负担，容易引起学生的兴趣。

(3) 提供的练习题力求少而精，并强调实践性，其中多数的练习题可以作为实际课程教学过程中的上机习题来使用。

(4) 提及了其他教材和技术书籍中不常涉及的内容，如 ROP 攻击、Windows 系统编程等，力图使我们的教材兼顾前沿性和自包含性。

参与本书编写的老师均常年在教学第一线担任相关的教学工作，对计算机系统安全、程序设计、计算机网络等课程有着非常丰富的教学经验。不仅如此，他们还从事计算机软件理论、系统和网络安全领域的科研工作，对于软件逆向工程的作用以及在相关研究领域的使用方法有较深的理解，能够利用在科研和工程实践中的实际经验对软件逆向工程技术进行有针对性的介绍。

本书编写人员的具体分工如下：孙聪编写第 1～8 章，李金库编写第 9 章，马建峰总体指导全书编写。

本书的编写得到西安电子科技大学教材建设基金资助。

由于软件逆向工程技术发展很快，而本书作者的水平有限，因此书中难免会有不足之处，恳请读者提出宝贵的建议。

编　者

2017 年 12 月

目　录

第 1 章　软件逆向工程概述

逆向工程(Reverse Engineering)的概念早在计算机之前就已经存在，它是从任何人造的东西中提取知识或者设计规划的过程，现在这一过程被越来越广泛地应用于软件逆向分析和攻击上，其目的就是要打开程序的“外壳”，获得其内部信息。本章首先介绍软件逆向工程的基本概念、基本过程和方法，然后介绍软件逆向工程的主要应用领域及相关的合法性问题，并在介绍 Windows 应用程序逆向分析工具 OllyDbg 的基本用法基础上，用一个例子说明如何用该工具进行简单的逆向分析和破解。

1.1　逆向工程的概念和基本方法

什么是逆向工程？简单地说，就是理解一个系统的过程。这个被理解的系统，可以是硬件设备、软件程序、协议、物理/化学过程等。在现实中，理解的目的往往不仅仅只是获取信息，更在于构建特定的事物。因此，逆向工程也可以看作一个从对象中提取知识或设计信息，并重建对象或基于对象信息的事物的过程。

对于软件逆向工程而言，被理解的系统即软件。这一对象往往以可执行程序的形式存在。具体地讲，软件逆向工程即指通过构建通用等级的静态和动态模型来理解已知软件的结构和行为。因此，软件逆向工程的过程可以大致分为两个阶段，如图 1-1 所示。第一阶段是收集信息，被收集的信息可以是静态信息或动态信息。常用于收集静态信息的工具包括语法分析器、静态分析工具等，常用于收集动态信息的工具包括调试器、事件监控器等。第二阶段是信息抽象，以获得更高层的可理解的模型。此阶段得到的模型可构成不同的视图，包括动态视图、静态视图和混合视图。这些视图从不同的侧面反映了软件产品的功能、结构、处理流程、界面设计等要素。

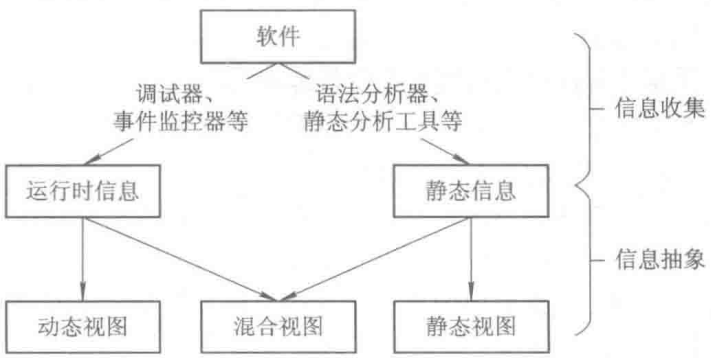


图 1-1　软件逆向工程的两阶段

从软件工程的一般过程来看，通常正向过程的先后顺序是需求分析、概要设计、详细设计和软件实现及测试。软件逆向工程的过程恰好相反，是从软件的实现起始的。但软件逆向工程的过程一般不会延伸到需求分析的层面，而是通过软件逆向工程的抽象过程得到抽象的系统模型，然后在该系统模型的基础上进行正向工程，实现一个新的系统。

而从更细粒度的程序实现的角度看，程序编译的过程与软件逆向工程的过程相反。图1-2给出了编译和软件逆向工程的过程流。可以看到，编译的过程是将源代码通过语法分析得到语法树，再通过中间代码生成得到控制流图和中间代码，然后通过最终代码生成得到汇编代码，再通过汇编得到机器码，最后链接为可执行程序。而软件逆向工程则是从程序的机器代码恢复出程序的高级语言结构和语义的过程，具体包括反汇编和反编译等步骤。反汇编是从机器码得到汇编语言代码，而反编译是从汇编语言代码得到高级语言结构和语义。

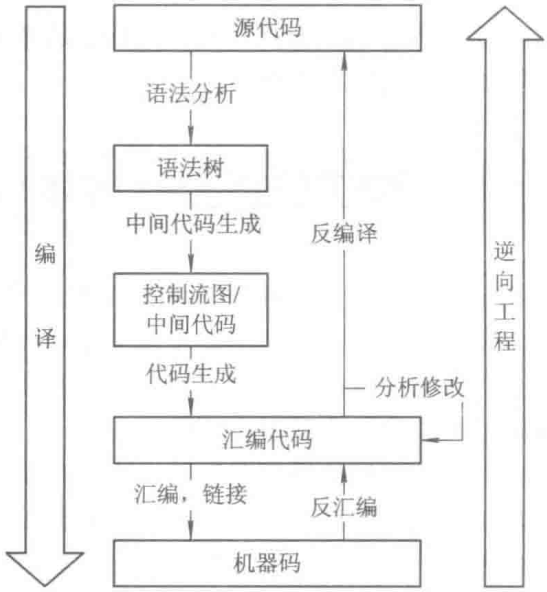


图 1-2　编译和软件逆向工程的过程流

不同的场景决定了软件逆向工程的过程是否需要在反汇编的基础上进行反编译。对于有经验的逆向分析人员而言，反编译不是必需的。如果软件逆向工程的目的是得到一个新的程序，那么在反汇编的基础上，还需要对汇编程序进行分析和必要的修改(破解)。从这个角度看，软件逆向工程的主要工作即反汇编、汇编程序分析和程序修改(破解)。对软件进行逆向工程通常需要具备计算机系统结构、汇编语言、操作系统等多方面专业知识。

软件逆向工程的实现方法可以分为静态方法和动态方法两类。所谓静态方法，是指分析但不运行代码的方法，相比动态方法而言更为安全。常见的反汇编器 IDA Pro、objdump 等都采用的是静态方法。而动态方法则是指通过在虚拟环境或实际系统环境中运行和操作进程，检查进程执行过程中寄存器和内存值的实时变化的方法，常见的调试器如 WinDbg、Immunity、OllyDbg、GDB 等都采用的是动态方法。较为复杂的动态方法可能会将程序的二进制代码置于可控的虚拟环境中，通过虚拟环境中的 CPU 得到其执行轨迹，然后利用条件跳转指令泄漏路径约束信息，使用符号执行技术从执行轨迹中收集逻辑谓词，进而通过

约束求解准确地推断出程序的内部逻辑。

不同的反汇编引擎会采用不同的反汇编算法。反汇编算法首先确定需要进行反汇编的代码区域，然后读取特定地址的二进制代码并执行表查找，将二进制操作码转换为汇编语言助记符，此后对汇编语言进行格式化并输出汇编代码。如何选定下一条被反汇编的指令，不同的算法采取的策略不同。常见的策略包括线性扫描(Linear Sweep)和递归下降(Recursive Traversal)两种。GDB、WinDbg、objdump 等均采用线性扫描的方法，而 IDA Pro 则采用的是递归下降方法。

1.2 软件逆向工程的应用

恶意代码分析是软件逆向工程的主要应用之一。与软件逆向工程的实现方法相对应，恶意代码分析也可以分为动态分析和静态分析。动态分析是指在严格控制的沙箱环境中执行恶意程序，记录并报告程序出现的所有行为，从中识别出恶意行为。而静态分析则是指通过分析反汇编后得到的汇编程序代码或通过反编译得到的高层代码来理解程序行为。

软件逆向工程需解决的另一个重要问题是闭源软件的漏洞分析。这也说明了软件逆向分析不仅能应用于“恶意”软件，也能应用于“良性”软件。“良性”软件的开发过程中可能由于开发者的经验、编译器的配置等因素而引入很多漏洞，这些漏洞有时容易被攻击者利用。为了发现和分析这些潜在的漏洞，我们可以采用模糊测试(fuzzing test)或静态分析，而静态分析通常就以反汇编为基础。对“良性”软件的修改在二进制层面也有“良性”和“恶意”之分，通过反汇编和调试等逆向技术，我们可以开发出相应的补丁程序或破解程序(Exploit)。

软件逆向工程的另一个典型应用是闭源软件的互操作性分析。如果我们只能获得一个软件的二进制码，而又要开发与其互操作的软件和插件，或者开发适用于其他硬件平台的程序(属于软件移植的范畴)，那么往往需要通过逆向工程相关的技术来界定该软件的行为和接口。

此外，我们还可以利用反编译等技术来衡量编译器的正确性和安全性，验证编译器是否符合规范，寻找优化编译器输出的方法，并检查由此编译器生成的代码中是否能插入后门，从而判定编译器本身是否有安全问题。

1.3 软件逆向工程的合法性

与病毒和网络攻击类似，软件逆向工程并不总是合法的。美国和欧盟等主要经济体在软件逆向工程合法性和软件版权保护方面都制定了各自的法律法规。美国法律认为，对人工制品和过程的逆向工程权，随该制品/过程的合法获得而被法律许可。但同时，对计算机软件的逆向工程受合同法保护，由于大多数终端用户许可证(End-User License Agreement，EULA)都禁止逆向工程，因而对合法取得的软件进行逆向工程也会因违反终端用户许可证

而违反合同法。欧盟在 1991 年制定了计算机程序合法保护版权法令(Copyright Directive on Legal Protection of Computer Programs)，明确了对计算机程序进行非授权的复制、翻译、改编和转换构成对作者的侵权。而对代码的复制和翻译如果对于实现该程序与其他程序的互操作而言不可避免，那么有权使用该软件的人可以复制和翻译该软件而无需获得权利人授权。

通常，当软件版权所有者无法进行软件错误修正时，我们可以通过逆向工程对软件错误进行修正和破解。而在不违反专利权或商业秘密保护的前提下，可以通过逆向工程确定软件中不受版权保护的部分(如一些算法)。一般针对软件逆向工程的法规通常声明如下合法性限制：① 逆向工程人员为合法用户；② 逆向工程以互操作为目的，仅对实现互操作程序所必要的那部分程序进行逆向工程；③ 需获取的“必要信息”不能从其他途径取得；④ 通过逆向工程获得的信息不能用于实现互操作程序以外的目的，不能扩散给不必要的第三人；⑤ 不能用于开发形式类似或有其他著作权侵权因素的程序；⑥ 不得不合理地损害权利人的正当利益或妨碍计算机程序的正常使用。

我国第一部有关计算机软件保护的法律条例《计算机保护条例》(1991)，为计算机软件的使用和版权保护做出了相关规定，但条例未直接涉及软件逆向工程相关的问题。自 2002 年该条例修订后，对软件的“合理使用”进行了更为严格的规定。2007 年 2 月施行的《最高人民法院关于审理不正当竞争民事案件应用法律若干问题的解释》第十二条规定，通过反向工程方式获得的商业秘密，不认定为反不正当竞争法规定的侵犯商业秘密行为。这是我国法律首次对逆向工程进行相关规定。与欧美等发达国家的相对完善的法律相比，我国的相关法律法规依然相对落后。

与此同时，国外已经出现了一些与软件逆向工程相关的法律案例，例如 Connectix 公司对 SONY 公司 PlayStation 的逆向侵权案例[①]。PlayStation 是 SONY 公司斥巨资研发的一款电视游戏系统，1995 年向全球市场发行并在商业上取得了巨大成功。1998 年，Connectix 公司开始对 PlayStation 的软件进行仿制，以使 PlayStation 系统上的游戏能运行于个人电脑。该公司对 SONY 已经申请专利保护的 PlayStation 系统的 BIOS 进行逆向工程，从而开发出和 PlayStation 功能相似的可应用于个人电脑上的程序 VGS。1999 年 1 月，SONY 公司向美国加州法院提起诉讼，状告 Connectix 公司侵犯其著作权。加州法院一审认为，尽管 VGS 中没有包含 PlayStation 源程序，但 VGS 的使用妨害了 PlayStation 系统的销售，因而该法院对 Connectix 公司发布了发售禁令。而在 Connectix 的后续上诉过程中，美国联邦第九巡回上诉法院则判定 Connectix 公司属于正当使用，解除了加州法院的禁令。联邦第九巡回法院的判决思想认为：为了解 PlayStation 的设计思想和功能概念这些不受版权保护的程序侧面，对该程序的目标代码进行反汇编是必需而不可绕过的，因而这种反汇编属于合理使用。

1.4 初识工具

本节将介绍的第一个逆向分析工具是 OllyDbg。OllyDbg 是一种具有可视化界面的 32

① https://en.wikipedia.org/wiki/Sony_Computer_Entertainment,_Inc._v._Connectix_Corp.

位汇编分析调试器。作为一种动态追踪工具，OllyDbg 适用于 Windows 环境。该调试器融合了 IDA 和 SoftICE 的思想，支持 Ring 3 级调试。同时，该调试器还支持插件扩展功能，是目前最强大的调试工具之一。

OllyDbg 的主界面样式如图 1-3 所示，其中编号的界面区域所具有的功能分别说明如下：

(1) 汇编指令地址；

(2) 汇编指令对应的十六进制机器码；

(3) 反汇编代码；

(4) 反汇编代码对应的注释信息；

(5) 当前执行到的反汇编代码信息；

(6) 当前程序执行状态下的各寄存器值；

(7) 数据所处的内存地址；

(8) 数据对应的十六进制编码；

(9) 数据对应的 ASCII 编码；

(10) 栈地址；

(11) 栈数据；

(12) 栈数据对应的说明信息。

其中，汇编代码窗口包括(1)～(5)；寄存器窗口包括(6)；数据窗口包括(7)～(9)；栈窗口包括(10)～(12)。

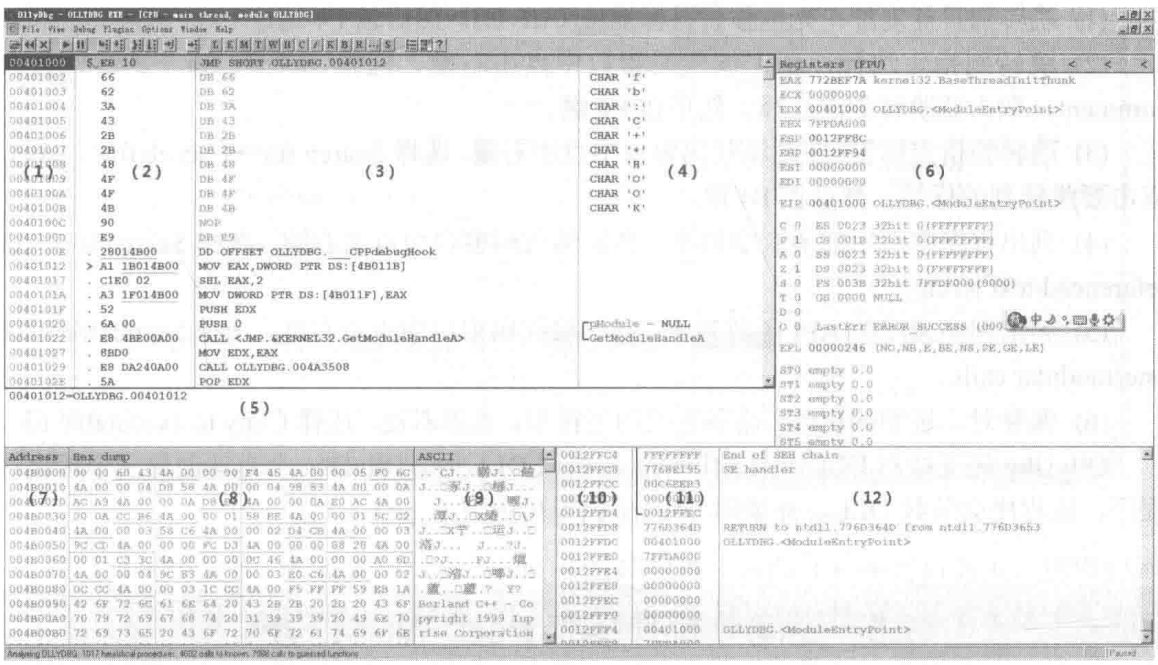


图 1-3　OllyDbg 的主界面样式

直接用 OllyDbg 打开具体的应用程序，即可发起对该应用程序的调试过程。OllyDbg 的操作方便，表 1-1 中的快捷键能够帮助我们高效地调试目标应用程序。

表 1-1 OllyDbg 的主要快捷键及其功能

快捷键	功　能
F7	Step Into。执行一句操作码，若遇调用命令 CALL，则进入被调用函数代码内部
F8	Step Over。执行一句操作码，若遇调用命令 CALL，则直接执行函数，不进入函数内部
Ctrl+F2	重新开始调试(终止正在调试的进程后再次运行)
Ctrl+F9	运行到函数的 RETN 命令处，用于跳出函数
F2	设置断点
F9	程序运行到下一个断点处暂停
Alt+F9	运行到用户代码处，用于快速跳出系统函数
Alt+B	查看当前的断点列表
Alt+M	打开内存映射窗口
Ctrl+G	输入十六进制的地址，让光标移动到指定地址
Ctrl+E	打开编辑窗口，编辑已选内容
F4	让调试流运行到光标所在位置
;	添加注释
:	在指定地址添加特定标签

在调试具体程序时，我们还常会用到一些典型的操作方法，具体包括：

(1) 跳转到目标地址：光标移到目标地址(Ctrl+G)，然后按 F4 键。

(2) 跳转到指定注释：在汇编代码窗口中点击右键，选择 Search for→User-defined comment，双击要跳转到的注释，然后按 F4 键。

(3) 跳转到指定标签：在汇编代码窗口中点击右键，选择 Search for→User-defined label，双击要跳转到的标签，然后按 F4 键。

(4) 列出程序中引用的所有字符串：在汇编代码窗口中点击右键，选择 Search for→All referenced text string。

(5) 列出程序调用的 API 函数列表：在汇编代码窗口中点击右键，选择 Search for→All intermodular calls。

(6) 保存对二进制的更改：选中更改的字符串，点击右键，选择 Copy to executable file。

OllyDbg 还支持对 DLL 文件的调试。在调试 DLL 时，OllyDbg 会自动创建一个可执行程序，该程序会装载 DLL，并调用 DLL 中的导出函数。

1.5 逆向分析并修改 Hello World 程序

在本节中，我们将尝试编写一个 Hello World 可执行程序，通过 OllyDbg 对其进行逆向分析，并依据分析结果对其二进制程序进行改写。

我们所用的 Hello World 程序的源代码如图 1-4 所示。

```
#include "windows.h"
#include "tchar.h"
int _tmain(int argc, TCHAR *argv[]){
        MessageBox(NULL, _T("Hello World!"), _T("www.xidian.edu.cn"), MB_OK);
        return 0;
}
```

图 1-4　Hello World 程序的源代码

编译该程序时，选择 Visual Studio 的 Release 模式，这样可以避免编译器插入大量调试信息，方便我们进行调试和修改。编译出的 HelloWorld.exe 的运行效果如图 1-5 所示。

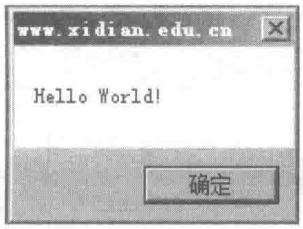


图 1-5　HelloWorld 程序的运行效果

在得到可执行程序 HelloWorld.exe 后，用 OllyDbg 将其打开，初始效果如图 1-6 所示。

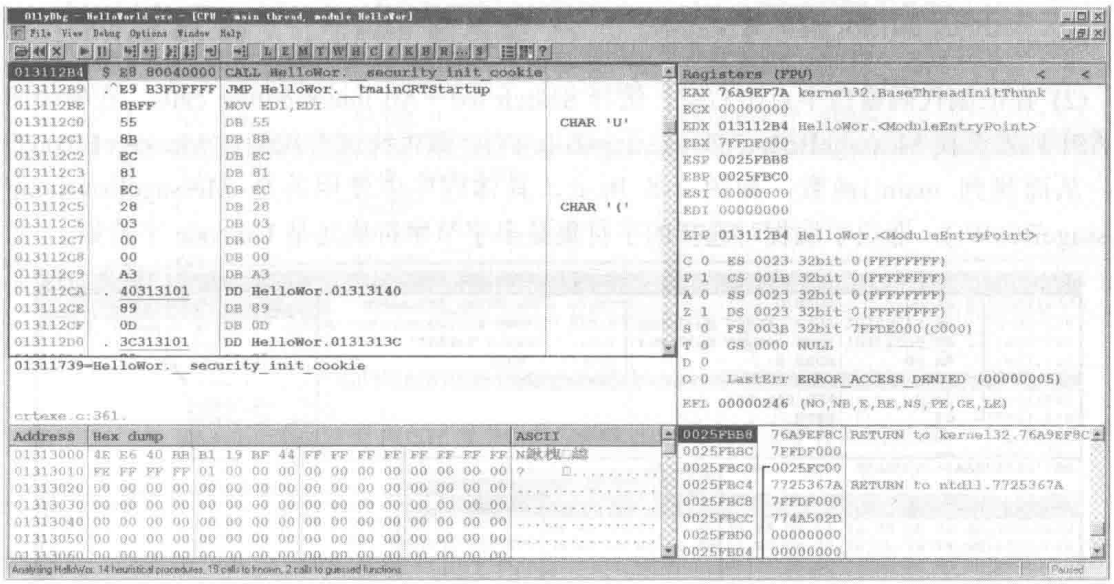


图 1-6　使用 OllyDbg 调试 HelloWorld 的初始效果

调试该程序的第一步是如何找到 main()函数的主体。从图 1-6 中可以看出，程序初始执行时，并非直接从 main()函数的第一句开始执行，而是执行一些编译器和系统要求做的初始化及准备工作。这些代码又称为启动代码。进行逆向分析时，不需要仔细分析启动代码，但应能区分出启动代码和用户代码。

那么，如何迅速地找到 main()函数主体呢？我们从执行 HelloWorld.exe 的效果看，能够判断出该程序使用了两个字符串“www.xidian.edu.cn”和“Hello World!”，有经验的程序员还能观察到我们是调用了 MessageBox()函数将对话框显示出来的。因此，除了耐心地

通过点击快捷键 F7 和 F8 单步执行到 main()函数中这一方法之外，还可以使用上一节介绍的两种方法：

(1) 在汇编代码窗口中点击右键，选择 Search for→All referenced text string，在被引用字符串的列表中选择“www.xidian.edu.cn”或“Hello World!”字符串，即可快速找到对这两个字符串参数的 PUSH 压栈操作，从而找到 main()函数，如图 1-7 所示。

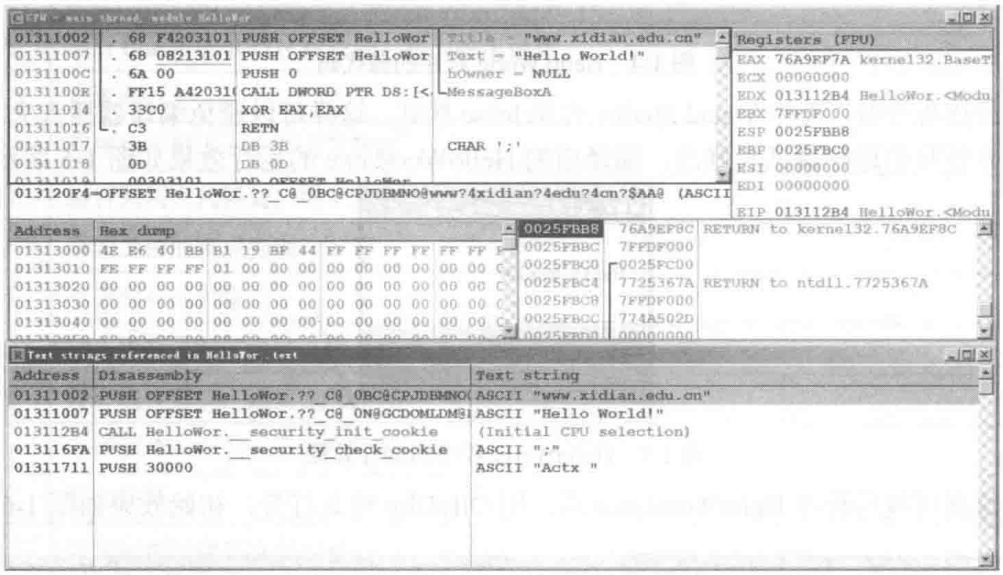


图 1-7 通过查找被引用字符串的方式查找目标程序片段

(2) 在汇编代码窗口中点击右键，选择 Search for→All intermodular calls，在模块间调用 API 列表查找 MessageBoxA()或 MessageBoxW()，即可快速查找到对 MessageBox()的调用，从而找到 main()函数，如图 1-8 所示。具体程序中使用的是 MessageBoxA()还是 MessageBoxW()，取决于编程时选择的字符集是多字节字符集还是 Unicode 字符集。

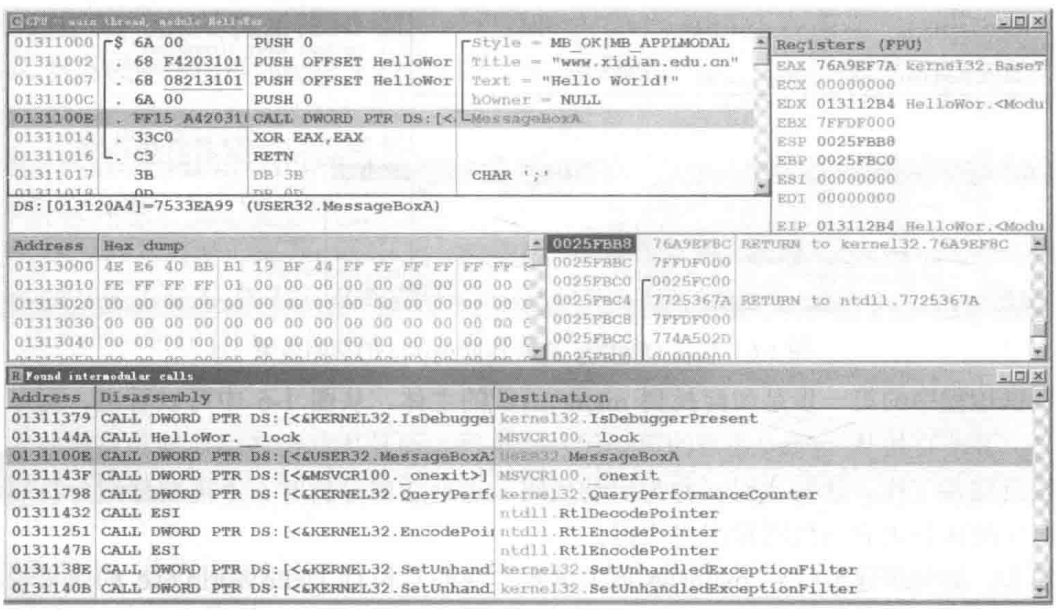
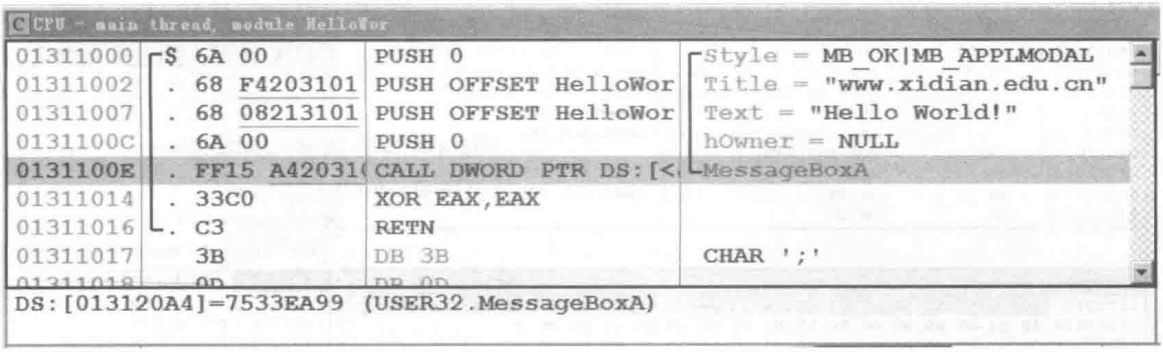
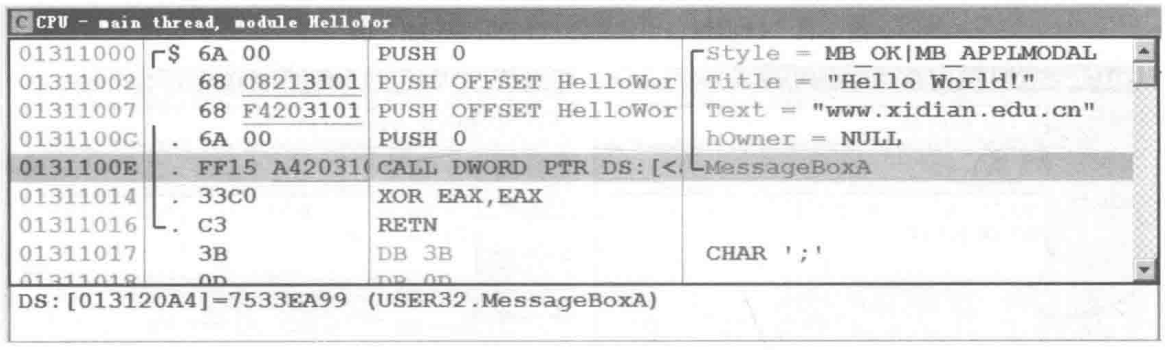


图 1-8 通过查找模块间调用 API 的方式查找目标程序片段

找到 main()函数以后，就可以在逆向分析汇编程序的基础上，对特定的指令或数据进行修改了。第一个修改是：能否让界面上“www.xidian.edu.cn”和“Hello World!”两个字符串的位置对调。一个简单的做法是针对图 1-9(a)中对这两个字符串压栈的 PUSH 指令，将两条指令的地址(013120F4H 和 01312108H)对调，对调的结果如图 1-9(b)所示。对调完成后，通过点击右键，选择 Copy to executable file→All modifications，弹出十六进制文件窗口，再在十六进制文件窗口中通过点击右键，选择 Save file，保存为新的 EXE 文件。新的可执行文件的运行效果如图 1-10 所示。



(a) 地址对调前



(b) 地址对调后

图 1-9　通过对调 PUSH 指令中的字符串地址实现界面字符串的对调

图 1-10　字符串对调后程序的运行效果

第二个修改是：将原来 HelloWorld.exe 界面上的“Hello World!”字符串，改为“Hello Students!”字符串。此时，首先要做的是找到字符串对应的数据内容并对其进行修改。从指令中我们可以看出，“Hello World!”字符串在数据段的地址是 01312108H，在数据窗口