开发人员专业技术丛书

HZ BOOKS

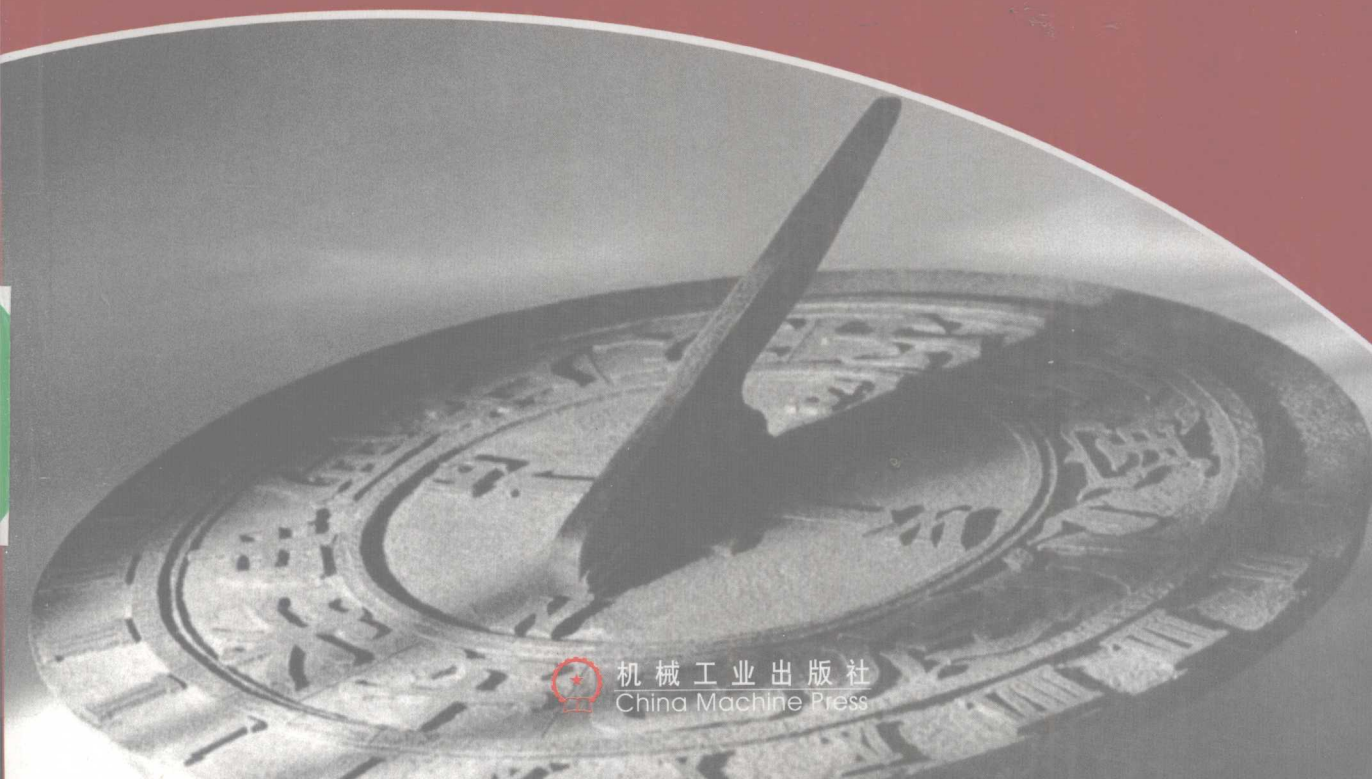
深入揭示Windows高级调试技术

Advanced Windows Debugging

Windows高级调试

（美） Mario Hewardt
Daniel Pravat 著

聂雪军 等译



机械工业出版社
China Machine Press

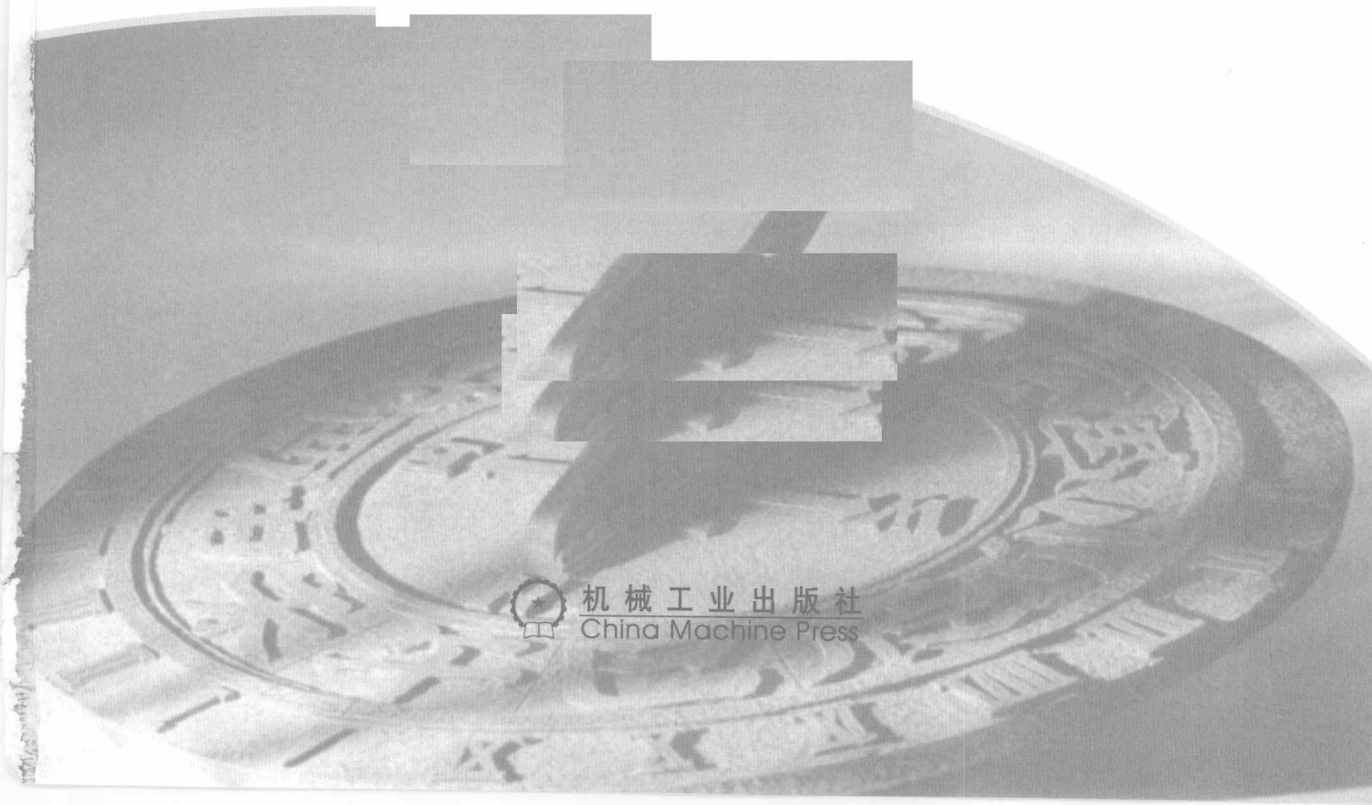
开发人员专业技术丛书

Advanced
Windows Debugging

Windows高级调试

（美） Mario Hewardt
Daniel Pravat 著

聂雪军 等译



机械工业出版社
China Machine Press

本书主要讲解Windows高级调试思想和工具，并涉及一些高级调试主题。本书内容主要包括：工具简介、调试器简介、调试器揭密、符号文件与源文件的管理、栈内存破坏、堆内存破坏、安全、进程间通信、资源泄漏、同步、编写定制的调试扩展、64位调试、事后调试、Windows Vista基础以及应用程序验证器的测试设置等。本书内容详实、条理清楚。

本书适合Windows开发人员、Windows测试人员和Windows技术支持人员等参考。

本书版权登记号：图字：01-2008-1793

图书在版编目（CIP）数据

Windows高级调试 /（美）赫瓦特（Hewardt, M.）等著；聂雪军等译. —北京：机械工业出版社，2009.5

书名原文：Advanced Windows Debugging

ISBN 978-7-111-26639-6

Ⅰ. W…　Ⅱ. ① 赫…　② 聂…　Ⅲ. 窗口软件，Windows　Ⅳ. TP316.7

中国版本图书馆CIP数据核字（2009）第041437号

机械工业出版社（北京市西城区百万庄大街22号　邮政编码　100037）

责任编辑：周茂辉

北京京北印刷有限公司印刷

2009年5月第1版第1次印刷

186mm × 240mm · 32.75印张

标准书号：ISBN 978-7-111-26639-6

定价：79.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

本社购书热线：（010）68326294

对本书的赞誉

“谁说你不能学习别人的经验？本书就包含了丰富而翔实的信息，它们非常清晰地阐述了如何通过一种逻辑的方法来找出和修复程序中的问题。对于那些在Microsoft Windows平台上开发、测试和提供技术支持的人员来说，本书绝对是一本不可或缺的参考手册。”

——Bob Wilton，资深工程师
就职于Microsoft公司中CPR（Critical Problem Resolution）小组

“我有幸与本书的作者在超高要求系统（Extremely Demanding System）领域共事了8年多的时间。本书包含了非常有价值的知识，我们曾经感叹，要是在项目开始之前就知道这些知识该有多好——这位调试大师或许只有2月29号才会告诉你这些知识，因为他只有在这一天的下午才有空；只有亲自去构建和调试那些复杂的系统项目而不是道听途说，才能获得这些宝贵的知识。

在大多数书籍中，一些高级主题总是作为‘留给读者的练习’或者‘请参阅其他高级参考书’，而这些主题似乎从来就没有出现过。本书属于那些‘其他高级参考书’。要买就买两本吧，因为你将经常要借一本给其他人。”

——Raymond McCollum，架构师
就职于Microsoft前沿安全产品（Forefront Security Product）部门

“由Microsoft的Mario和Daniel合著的这本书是一本非常棒的参考书，面向的读者包括中级调试人员和高级调试人员。本书通过对一些示例进行深入讲解来阐述如何调试各种错综复杂的问题，例如栈破坏和堆破坏等，这使得本书与目前市面上讲解Win32软件调试的众多书籍相比显得卓尔不群。本书的实用性非常高，包含了丰富的调试技巧和策略。”

——Kinshuman，开发主管
就职于Windows核心操作系统部门

“我非常高兴地看到在这本书中包含了大量非常聪明的调试技巧。它不仅介绍了如何对付那些难以诊断的问题，而且还详细解释了在这些技术后面隐藏的底层机制。本书中介绍的实用方法对于人们理解一些关键的Windows领域是非常有帮助的。”

——Adrian Marinescu，软件架构师
就职于Microsoft

本书详细描述了如何调试和修复软件中的各种问题。本书的内容是根据作者在跟踪各种软件问题时所积累的丰富经验而提炼出来的。本书不仅给出了各种问题的代表性示例，而且还介绍了在分析这些问题时所使用的工具，以及这些工具的详细使用说明。无论是软件开发人员还是软件测试人

员，在理解了这些示例后都将受益匪浅。”

——Daniel Mihai，软件设计工程师
就职于Microsoft开发人员生产率工具（Developer Productivity Tools）部门

“我编写了WinDbg符号处理器、符号服务器以及源文件服务器。即便如此，我仍然无法教会我妻子使用WinDbg。她认为这个工具非常难用，因此并不了解这个工具的强大之处。我买了这本书送给她，这样她就可以知道如何使用WinDbg。本书中关于事后调试（Postmortem Debugging）和内存破坏等方面的内容，有效地揭示了在程序出错时的运行环境和操作系统的内部状态。Mario和Daniel在调试领域积累了丰富经验，因为他们经常被要求解决陌生程序中一些莫名其妙的问题。这也是具有工业强度的调试技术的真正意义所在。”

——Pat Styles
就职于Microsoft

译 者 序

软件调试是开发人员日常工作的重要组成部分。无论软件工程的理念多么先进，开发进度的安排多么合理，或者开发人员的经验多么丰富，人们在开发软件时总是无法保证不出现任何错误。当软件出现错误时，就需要进行调试。近年来，随着软件规模和复杂性的不断增加，错误的出现频率以及调试难度也在以非线性的方式增长。要想提高调试工作的效率，采用正确的调试思路和调试工具是非常重要的。有时候，一个需要数天时间才能解决的问题，如果换一种调试思路或者借助某种特殊的调试工具，或许只需1个小时就可以解决。

软件人员通常都知道许多高效的编码方法，例如极限编程、代码自动生成框架等，但他们很少知道一些高效的调试方法。大多数软件开发人员的调试思路和所使用的调试工具都还停留在比较初级的阶段，例如仅限于通过集成开发环境自带的调试器来进行单步跟踪。事实上，随着Windows的不断发展，Windows上调试工具的数量和功能也在不断增加。但可惜的是，很少有人知道这些强大的调试工具，而了解这些工具的独特优势并且针对不同调试问题采用正确调试工具的人则更少。有些软件开发人员经常加班和熬夜，往往是因为他们使用了一些低效的方法来调试复杂的问题，他们并不知道其实存在着一些更高效的调试方式可以帮他们节约大量的时间。

本书的写作初衷正是为了将一些非常有价值的调试思路和调试工具推荐给软件开发人员，目的是提高开发人员的调试效率。本书的特点如下。

1) 系统地介绍了Windows调试的基础知识。这些知识包括两个方面：Windows中一些底层组件的知识，包括调用栈的结构、堆管理器的工作原理、安全管理机制、进程间通信协议等；一些基本的调试知识，包括调试器的工作原理、调试符号的使用、内存转储文件的结构以及栈回溯中包含的调试信息等。

2) 结合实际问题来阐述正确的调试思路以及各种调试工具的使用。本书采用的叙述方式是，首先给出某个实际的问题，例如内存破坏、同步问题、资源泄漏、安全威胁等；然后介绍分析这个问题的正确调试思路，例如如何通过程序的异常表现来了解问题的具体信息；最后根据所调试的问题来采用正确的调试工具，并详细讲解了各种调试工具的使用方法和技巧。

3) 讲解了一些高级调试主题，包括64位操作系统上的调试、Windows Vista系统上的调试、事后调试以及调试扩展的编写等。这些主题为开发人员进行复杂的调试提供了必要的基础知识。

作者Mario Hewardt和Daniel Pravat是Microsoft公司的资深工程师，他们从事的工作就是保证Windows产品的可靠性、稳定性以及安全性。他们在调试领域已经工作了十余年的时间，积累了丰富的调试实践经验，而本书正是融合了这些宝贵的经验而形成的。因此，对于Windows开发人员来说，本书是一本不可多得的调试参考手册。

本书的技术性较强，需要读者对Windows系统的底层架构和组件有一定的了解，并且具备一定的编程基础和调试基础。如果读者需要补充这些方面的知识，可以参考作者在书中介绍的相关参考书籍。

参与本书翻译工作的还有李杨、吴汉平、徐光景、童胜汉、陈军、胡凯、刘红、张玮、陈红、李斌、李勇涛、王海涛、周云波、彭敏才和张世锋等。由于译者的时间和水平有限，翻译中的疏漏和错误在所难免，请读者和同行不吝指正。

感谢妻子云兰和女儿彤彤，虽然在翻译本书的过程中减少了陪伴你们的时间，但你们依然理解和支持我的工作。感谢父母在生活上的帮助，使我可以全身心地投入到本书的翻译工作中。

聂雪军

2009年2月于武汉

序　言

软件的目标之一就是简化人们的工作。如果能够对某个工作流进行优化或者自动化，那么人们在存储数据或者处理数据时就能变得更加高效，软件的出现正是为了实现这个目标。然而，在带来简化的同时，一定不能在软件中引入更多的复杂性，这就意味着软件在安装时应该只需要很少的用户交互，能够与其他程序提供的服务和数据无缝地集成起来，并且对软件和硬件环境的变化有着很好的适应性。

然而，在努力简化用户和管理员操作的同时，软件也正在变得越来越复杂。这种复杂性可能体现在多个方面，例如需要处理的数据量、相互通信程序的数量、内部并行语义的深度或者从其他软件中导入函数的数量。在软件外表的简单性下隐藏了不同软件层次之间的许多微妙问题，例如同步、相互依赖性以及各种假设等，这些问题通常涉及不同的程序，甚至不同的计算机。软件的故障通常表现为在各个库中发生的崩溃、毫无意义的错误消息或者程序的挂起，要找出发生故障的组件（还不是要找出故障的原因）都是非常困难的。

阅读本书的理由是，当你在开发、测试或者提供技术支持时，通常会遇到各种软件故障，而你往往需要分析这些故障的产生根源，甚至还可能需要修复这些故障。如果想顺利地完成这些工作，那么就需要尽可能快速和高效地找出问题的源头，这意味着首先要知道观察软件的哪些方面，从什么地方开始观察以及如何进行观察。换句话说，你需要知道目前有哪些可用的工具，对于每种类型的故障使用何种工具最为有效，以及如何利用这些工具来快速缩小问题的查找范围。

大多数时候，在工作中学习如何分析和调试Windows程序是惟一选择。在调试某个程序故障时，如果知道通过某种工具或者某个特定的调试命令可以极大地减少工作量，那么你将迅速地使问题水落石出，而不会花了数小时甚至数天的时间还不能取得任何进展。这也是本书物超所值的原因所在。

本书不仅来源于Mario和Daniel多年的调试实践经验，更来源于Microsoft客户支持服务部门和Windows产品与工具开发团队的集体智慧。没有任何其他参考资料能够比本书更权威地介绍Windows调试领域的知识，例如Windows堆管理器如何影响缓冲区溢出的行为，或者在调试DCOM挂起问题时应该使用哪些调试扩展命令等。我在调试Windows应用程序和设备驱动程序等领域已经工作了10余年的时间，但当我阅读本书时，还是学到了许多新的技术、工具和调试命令，既包括我从未遇到过的技术，也包括我曾经使用过的技术。

我们的工作价值并不在于如何调试问题，而是在于调试问题的速度和准确度。无论你在Windows程序调试领域已经工作了数年的时间，还是刚刚开始，Mario和Daniel都为你提供了丰富的知识。祝你的调试工作进展顺利！

Mark Russinovich

Microsoft公司平台与服务部门技术专家

前　言

不久前，我们还在回忆着在工作中曾经遇到的一个极为困难的问题。质量保证小组对我们的产品进行压力测试，并且每隔4～5天，在产品中就会出现一个崩溃问题。当然，我们在调试这个问题时已经竭尽全力，并且进行了大量的代码审查工作来找出问题的原因，但可惜的是，我们还是没有获得足够的信息来找出问题。几个星期之后依然没有任何进展，因此我们开始寻求其他的方法。在一次偶然的谈话中，有人提到了一个叫做gflags的工具。我们之前从来没有听说过这个工具，因此我们首先想了解这个工具能否有助于找出问题。但遗憾的是，这个学习过程有些困难。首先，在查阅工具的一些帮助信息时就遇到了麻烦。在工具的相关参考文档中包含了大量的信息，这使得我们很难找到该从什么地方下手。我们很快意识到，如果得不到一些基本的指导信息，那么我们将无法使用这个工具。自然而然地，我们想到了去请教当初提到这个工具的人，看看他是否了解这个工具的一些知识。他首先向我们简单地介绍了这个工具，然后告诉我们哪些人曾经使用过这个工具。接下来，我们与这些人进行长时间的、富有指导性的交谈，因此在这个工具中包含的基本思想也逐渐浮现出来。

我们是否找出了崩溃问题出现的原因？是的，我们确实找出来了。事实上，通过在进行压力测试时启用gflags工具，我们只花了一个小时的代码审查时间就找出并修复了存在错误行为的代码。如果我们一开始就知道并且使用这个工具，那该有多好，我们将节约几个星期的工作时间。从那时候起，我们投入了大量的时间来深入学习这些工具，并且不断琢磨如何在调试问题代码时有效地使用它们。

多年来，Windows调试器和调试工具在不断地变得更加成熟，功能也变得更为强大。这些功能不仅节约了大量的调试时间，而且数量非常之多，令人难以置信。然而，同样令人难以置信的是，大多数开发人员都不知道这些调试器和调试工具。即使一些开发人员发现了这些工具，也不得不像我们数年前一样经历痛苦的学习过程。我们的优势在于与Microsoft工程师们（他们其中一些人编写了这些工具）一起工作，但如果没有这种优势，许多开发人员最终都将陷入死胡同，他们永远都无法感受到这些工具带来的好处。正是由于缺乏这些工具的相关学习资料，才使得我们萌发了编写本书的念头。要想使开发人员获得一些必备的知识，关键要提供一个包含各种简明信息的知识库，并且这些信息能够完整地说明在调试工具和调试过程中的各种输入与输出。本书正是这样的一个知识库，它历经了3年的写作时间，融合了15年的调试经验。

我们希望你在阅读本书时能够获得与我们在编写本书时一样多的乐趣，它将为你打开高效软件调试之门。了解如何使用本书中介绍的工具和技术是每一位计算机工作人员的重要工作之一，本书将教会你如何高效地找出软件中一些最为困难的问题。

本书的读者对象

这个问题的简单答案是：在软件开发过程中希望深入了解Windows内部机制的每一个人。本书的技术性较强，因此你或许会认为书中的内容只适合于高级系统工程师，但事实并非如此。本书的关键内容之一是揭开了调试技术的神秘面纱。出于各种原因，许多软件工程师都认为在软件和操作系统之间存在着一种神秘的关联。当某个问题需要对操作系统的一些组件（例如，RPC/COM或者Windows

堆管理器）进行分析时，这种先入为主的“神秘感”会使得开发人员不愿意对Windows进行深入研究以获得更多有助于解决问题的信息。因此，要想有效地使用本书，你首先一定要破除这种先入为主的观念，并且坚定地认为不存在任何神秘之处。Windows核心组件应该被视为对产品的一些扩展，而不是某种独立的或者神秘的层次结构。毕竟，它们也只是代码——只不过这些代码是由其他人编写的。如果你调整好了思维方式来接受这种新的观念，那么就朝着掌握Windows调试技术迈出了第一步。

软件开发人员

无论是底层系统开发人员，还是高层RAD开发人员，都能在阅读本书的过程中获得有用的东西。无论你是喜欢用汇编语言还是.NET框架来编写Windows软件，都可以从中了解到在Windows调试过程中使用的工具和技术。

多年来，我们曾经与一些高层RAD开发人员进行过多次讨论，他们认为自己没有必要了解这些底层知识。毕竟，在更高层次上编写代码的好处在于，所有的底层复杂性都被抽象和隐藏了，开发人员无需去关心它们，对此我们表示赞同。然而，我们同时也认为，虽然抽象化编程使开发人员无需关注底层的实现细节，但这与了解这些抽象机制的工作原理并不矛盾。这种观点的理由很简单：你所使用的只是一种抽象，如果将这种抽象应用在某种它并不适合的设计环境中，那么将使软件产生严重的问题；并且，如果在这些情况中不能透彻地理解抽象机制的工作原理，那么可能就意味着产品的发布时间将推迟数个月。

要掌握Windows调试器和调试工具的另一个关键理由是，我们需要调试实时生产服务器。虽然在发布产品之前，我们已经竭尽全力地修复产品中的问题，但仍然会存在一些漏网之鱼。当这些问题在产品发布之后才出现时，要找出并修复它们将是一项非常棘手的工作。如果在实时生产服务器上出现了这些问题，那么客户往往对服务器的停机时间和配置信息的变化是非常敏感的，因此我们将无法在这些服务器上安装复杂的调试软件包。然而，如果通过Windows调试工具集（Debugging Tools for Windows）来进行调试，那么将既不需要修改服务器的配置，也不需要安装任何软件。简而言之，它能够使服务器在调试过程中保持不间断地运行。

质量保证工程师

正如软件工程师们能够从本书中获得一些有助于他们日常工作的信息一样，质量保证工程师们同样可以发现一些有价值的信息。通常，质量保证工程师需要在被测试的组件上运行大量的测试案例。在此期间可能会出现一定数量的问题。无论这些问题是内存破坏、资源泄漏，还是程序挂起，如果在测试期间启用一些合适的调试工具，那么将极大地减少在问题分析过程所花费的时间。例如，假设质量保证工程师需要对某个信用卡授权服务进行压力测试，其中一个测试目标是，这个服务需要在一周时间里持续不断地处理高负荷的客户请求。当测试进行到第6天时，这个服务对所有的客户请求都给出错误信息报告。此时，负责这个服务的开发人员将接到电话要求分析这个问题。很快，开发人员发现服务器耗尽了内存，这可能是由于一个很小的内存泄漏随着时间的推移不断地累积起来而造成的。然而，如果要在内存泄漏累积了6天之后才去分析发生泄漏的源头，那么无疑是非常困难的，这可能需要耗费数天的时间来进行调试和代码审查。如果在执行测试期间启用了正确的调试工具，那么将极大地减少在分析内存泄漏上所花的时间。

产品支持工程师

产品支持工程师同样可以使用这些调试器和工具来分析问题的根源。

产品支持工程师所面对的大多数问题与质量保证工程师和软件开发工程师在日常工作中所面对的问题是一样的，只不过在他们工作环境中存在一些限制，这些限制包括：无法获得对出现问题服务器的完整访问权限，无法获得对客户源代码的完整访问权限等。

本书的内容对于产品支持工程师在处理这些棘手的问题时同样会带来非常大的帮助。如果知道如何通过最少的停机时间和最小的系统配置变化来调试客户所遇到的问题，那么产品支持工程师就能够以更有效的、非侵入式的方式来收集在分析问题时需要的数据。

有志者事竟成

本书内容的技术性较强。如果你对Windows的内部机制一无所知，那么我们无法确保你能充分感受本书带来的好处。与其他的技术书籍一样，你需要具备一定的基础知识。

好奇心与学习的愿望

在编写本书时，我们意识到自己对Windows某些领域的理解有些肤浅。在大多数时候，我们都只是知其然，但却不知其所以然。我们本可以不进行深究，但好奇心往往占据了上风（情况经常都是这样）。我们投入了大量的时间来研究这些领域并且努力将零散的片断都关联起来。长此以往，我们对Windows有了更深入的了解，而这又使得我们在调试问题时更加高效。

在学习任何知识时，很关键的就是要有学习的愿望。根据知识背景的不同，你可能会对本书中的一些高级内容感到力不从心。但不要害怕这种困难，你一定能够彻底地掌握和理解本书的内容。

如果你拥有学习的愿望，并且有着极大的好奇心，那么你就正朝着成为Windows调试专家的方向在前进。

C/C++

本书的所有示例代码都是用C/C++编写的，因此你需要较为深入地理解这些语言以及在这些语言中的对象内存布局。如果你不熟悉书中的一些语言概念，并且希望进一步提升自己的C/C++技术，那么我们推荐你阅读以下的书籍。

- *The C++ Programming Language (3rd Edition)*, by Bjarne Stroustrup, Boston: Addison-Wesley, 2000。
- 中文版：《C++程序设计语言（第3版）》，Bjarne Stroustrup著，裘宗燕译，机械工业出版社，2002。
- *Inside the C/C++ Object Model*, by Stanley B. Lippman, Reading, MA: Addison-Wesley, 1996。
- 中文版：《深度探索C++对象模型》，Stanley B.Lippman著，侯捷译，华中科技大学出版社，2001。

Windows内核

本书是关于Windows高级调试的，因此书中的部分内容是关于Windows中一些组件的内核（例如堆管理器、RPC、安全子系统等）。我们并没有全面地讲解这些组件的各个方面，而只是对组件功能与调试情况之间的关系进行了简短而又深入的归纳。如果你希望进一步了解Windows的内幕知识，那么我们推荐你阅读下面这本书：

Microsoft Windows Internals, Fourth Edition: Microsoft Windows Server 2003, Windows XP, and Windows 2000, by Mark E. Russinovich and David A. Solomon. Redmond, WA: Microsoft Pres, 2004.

内容组织

本书主要由3个部分组成。下面我们将简要介绍每一章的内容。

第一部分：概述

这部分的内容为后面两部分的内容打下基础。这部分介绍了一些工具和调试器，使你首先熟悉调试器的一些基础知识。即使你对这些Windows调试器非常熟悉，但我们还是建议你阅读这部分的内容，除非极为必要，否则不要跳过这些章节，因为它们包含了大量非常有价值的信息。

第1章“工具简介”，本章介绍了书中所使用的一些工具，详细给出了这些工具的下载地址、安装说明以及应用场合等。

第2章“调试器简介”，本章向读者介绍了Windows调试器的一些基本知识，例如目前存在哪些调试器、如何使用它们以及如何配置它们。

第3章“调试器揭密”，本章对用户态调试器进行了更深入的阐述，给出了调试器的最小功能集合，此外还介绍了一些更高级的主题，例如异常分发机制的工作原理。

第4章“符号文件与源文件的管理”，本章讨论了如何维护两种最关键的调试信息：符号文件与源文件。我们简要地介绍了符号服务器和源文件服务器，如何在调试器中使用它们，如何通过构建符号服务器来高效地管理调试符号以及如何维护源文件服务器。

第二部分：调试实践

第二部分介绍的是如何通过Windows调试器来分析一些常见的编程错误。这一部分的每一章都介绍了某一类问题，例如内存破坏、内存泄漏以及PRC/COM等。每一章的开头部分首先给出了对一些Windows组件的简要介绍，然后介绍如何对一些涉及这些组件的常见编程错误进行调试。

除了第5章和第6章之外，第二部分的其他各章都是独立的，可以单独阅读。

第5章“内存破坏之一——栈”和第6章“内存破坏之二——堆”，这两章深入分析了困扰着开发人员的一个常见问题：内存破坏。第5章介绍了栈破坏，第6章则介绍了堆破坏。每一章首先介绍了在内存破坏中包含的一些基本概念，然后对一些常见的内存破坏情况进行了分析。在每种情况中都包含了相应的示例代码，并且给出了分析问题的完整流程。

第7章“安全”，本章讨论了在软件开发中一些与安全相关的常见问题。开发人员经常会遇到这样的情况：在调用某个系统函数时返回了访问拒绝错误，并且无法获得任何进一步的相关信息，这就很难找出问题的原因或者跟踪出这个错误的来源。本章将给出一些与安全相关的示例代码，并且详细介绍如何通过调试器和一些合适的工具来找出问题的原因。

第8章“进程间通信”，本章介绍了如何对进程间通信的问题进行调试。RPC/LPC或许是在进程间通信中最常使用的协议，但它们同样也是最为神秘的协议。在大多数应用程序中，知道如何分析与这个组件相关的问题是非常重要的。本章将介绍如何通过调试器来跟踪各种标识以及分析RPC故障等。

第9章“资源泄漏”，本章讨论了软件开发中的一个常见问题：资源泄漏。资源泄漏的最常见形式就是内存泄漏，但不仅限于这种形式，例如还包括注册表项泄漏、文件句柄泄漏等形式。本章介绍了一些资源泄漏问题及相应的示例代码，并且给出了如何通过调试器和工具来分析这些问题。

第10章“同步”，本章讨论了应用程序挂起的现象，以及如何通过调试器来分析一些同步问题，例如死锁和锁竞争等。本章介绍了许多同步问题，并且给出了详尽的分析过程。

第三部分：高级主题

第三部分给出的都是一些高级主题，例如事后调试、64位调试、Windows Vista基础等。这些内容并不是对相关领域的详细描述，而只是为读者提供一些入门知识。

第11章“编写定制的调试扩展”，本章讨论了如何编写定制的调试扩展。在Windows调试器中包含了一组非常强大的命令和工具，但有时候你可能希望将调试会话中的某些步骤自动化。本章详细介绍了调试器的可扩展模型，并且给出了如何编写定制调试扩展的示例。

第12章“64位调试”，本章介绍了64位调试的基本概念，包括栈回溯、函数调用以及参数传递等，这些概念为读者在64位架构下进行调试提供了一些入门知识。

第13章“事后调试”，本章讨论了事后调试，如果在问题出现时无法立即进行调试，那么可以使用事后调试的方法。这种调试方法通常适用的情况是：产品已经发布出去，并且客户在使用过程中出现了问题。

第14章“功能强大的工具”，本章讨论了两种可以将调试过程自动化的强大工具。第一种工具是DebugDiag，它能够将资源泄漏的调试过程自动化；另一种工具是调试命令analyze，它可以将初始的错误分析过程自动化。

第15章“Windows Vista基础”，本章介绍了Windows Vista中的一些基础知识。在这个新的Windows平台中，操作系统的某些方面发生了巨大的变化，本章介绍了其中的一些关键变化。

所需工具

本书中使用的所有工具都可以免费下载。新的Windows驱动程序开发包（Windows Drivers Kit，WDK）包含了一个完整的命令行C/C++开发环境，以及丰富的相关开发工具。

示例代码

作为软件工程师，我们在大部分时间里都在寻求着编写完美的代码的终极宝典。然而，在撰写本书的过程中，我们却要面对一个相反的问题——如何编写一些不完美的代码，以说明一些常见的编程错误。

示例代码的目的是：通过最简洁的形式给出常见编程错误的示例，同时还要确保不破坏这些编程错误的基本行为。为了实现最简洁的示例，我们有时候不得不编造一些示例，而不是使用真实的示例。虽然这些示例代码是“编造”出来的，但它们很好地模拟了真实的情况，并且对于所要分析的问题有着很强的针对性。

所有的示例代码都是用C/C++编写的。我们选择这种编程语言主要是基于两个简单的原因：

- 在Windows开发中，C/C++占据主流地位；
- 为了不会淡化在高层抽象中的调试概念，我们选择了这种最常使用并最接近系统核心的语言。

所有的示例代码都在Windows驱动程序开发包中进行编译和测试。我们选择WDK的目的是为了使读者在学习Windows调试技术时无需购买完整的开发软件包。

源代码假设运行环境是Unicode环境，这样在调试器中看到的Win32 系统函数都是Unicode版本的函数。例如，示例程序可能调用了`CreateProcess`，但在调试器中看到的将是`CreateProcessW`。在调试会话中给出的函数前面还包含了这个函数所在的模块，例如CreateProcessW是在kernel32.dll中实现的，那么通常需要同时指定模块的名字和函数的名字，并且用“！”分隔（例如`kernel32!CreateProcessW`）。

所有的示例代码和二进制程序都可以从本书的网址免费下载（http://www. advanced windows-debugging.com）。除了可以下载源代码和二进制程序外，这个网址还用作本书中所有程序的符号服务器和源代码服务器。当你运行书中的调试示例时，并不需要下载所有的符号，而只需将调试符号

路径直接指向本书的符号服务器，那么在调试时可以使用这些远程符号。源文件同样可以从本书网址的源文件服务器中获得。

为了在学习过程中保持一致性，在构建本书网址上的二进制文件时没有启用针对x86架构的优化选项，并且都是checked版本。我们选择Windows XP作为讲解调试技术的平台，因为它是使用最为广泛的操作系统。如果在其他不同的操作系统上构建示例程序，那么在调试输出中可能会存在一些差异。

如果想亲自构建示例代码，可以打开WDK构建窗口，并在包含makefile的目录下键入build/ZCc。如果在编译源代码时需要额外的步骤，那么在相应的章节中将给出这些步骤。

本书假设所有的二进制程序都是从本书的网址下载并复制到本地硬盘上的C:\AWDBIN目录中（保持子文件夹的结构完整性），并且源代码下载到文件夹C:\AWD中。

编写体例

在本书的许多示例和分析过程中都给出了详细的调试信息流（Debug Spew）。简单来说，调试信息流是指调试器在执行了某个用户动作之后产生的输出信息。通常，调试信息流中的信息组织形式是非常紧凑和简洁的。为了突出调试信息流中的一些重要数据，我们使用了粗体和斜体来显示这些数据。此外，在调试信息流中的粗体文字表示正在键入的命令。在下面的示例中说明不同字体的使用。

```
0:000> ~*kb
. 0 Id: 924.a18 Suspend: 1 Teb: 7ffdf000 Unfrozen
ChildEBP RetAddr Args to Child
0007fb1c 7c93edc0 7ffdf000 7ffd4000 00000000 ntdll!DbgBreakPoint
0007fc94 7c921639 0007fd30 7c900000 0007fce0 ntdll!LdrpInitializeProcess+0xffa
0007fd1c 7c90eac7 0007fd30 7c900000 00000000 ntdll!_LdrpInitialize+0x183
00000000 00000000 00000000 00000000 00000000 ntdll!KiUserApcDispatcher+0x7
0:000> dd 0007fd30
0007fd30 00010017 00000000 00000000 00000000
0007fd40 00000000 00000000 00000000 ffffffff
0007fd50 ffffffff f735533e f7368528 ffffffff
0007fd60 f73754c8 804eddf9 8674f020 85252550
0007fd70 86770f38 f73f4459 b2f3fad0 804eddf9
0007fd80 b30dccd1 852526bc b30e81c1 855be944
0007fd90 85252560 85668400 85116538 852526bc
0007fda0 852526bc 00000000 00000000 00000000
```

在上面的示例中，你需要在调试会话中键入命令**~*kb**。执行这个命令将输出几行数据，其中最重要的信息就是*0007fd30*。接下来，你键入的命令是*dd 0007fd30*，这个命令将输出粗体数值*0007fd30*的更多信息。

在本书中使用的所有工具都假设是从安装文件夹中启动的。例如，如果Windows调试器安装在C:\Program Files\Debugging Tools for Windows文件夹中，那么启动windbg.exe的命令就是：

```
c:\windbg
```

支持的Windows版本

本书基于Windows XP或者更高的版本，所有的示例代码和调试任务都分别在Windows XP SP2

或者Windows Server 2003上运行通过。请注意，SP的构建编号，甚至一些特定的补丁，都有可能改变命令的输出结果，但这些变化不会影响我们在调试会话中所要介绍的内容。

第15章“Windows Vista基础”介绍了Windows Vista中的一些最重要变化，并且给出了一些必须在Windows Vista中运行的调试会话。

所有的示例和调试会话都是运行在32位版本的Windows上的。第12章“64位调试”使用的一些示例则是运行在64位版本的Windows XP上。

技术支持

尽管我们竭尽全力希望确保本书尽可能正确，但毫无疑问还是会有所疏漏。如果你在本书中发现了一个错误，可以通过以下邮箱告诉我们。

E-mail：marioh@advancedwindowsdebugging.com

或者daniel@advancedwindowsdebugging.com

此外，还可以通过本书的论坛http://www.advancedwindowsdebugging.com来报告错误信息。对错误的更正信息将放在网址的勘误页面中。

致谢

编写一本技术书籍需要付出大量的劳动，这远远超出了我们的预期。作为作者，我们完成了本书的初稿，但在本书的整个编辑过程中，许多人都提供了一些有益的见解和建议，这使得本书非常值得一读。

感谢Addison Wesley的所有团队成员，尤其是Elizabeth Peterson、Jana Jones、Curt Johnson、Joan Murray和Gina Kanouse。感谢Chris Zahn在本书的编辑过程中帮助我们更正了一些不规范的语法。

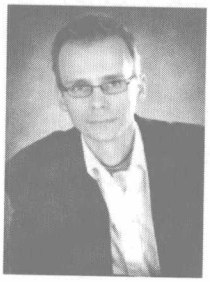
与其他的技术书籍一样，在技术方面的准确性是最为重要的。幸运的是，许多工程师（他们中的大多数人都擅长于本书的某个技术领域）都阅读了书中的内容并提供了反馈。感谢Mark Russinovich、Ivan Brugiolo、Pat Styles、Pavel Lebedynskiy、Daniel Mihai、Doug Ellis、Cristi Vlasceanu、Adrian Marinescu、Saji Abraham、Kamen Moutafov、Kinshuman Kinshumann、Bob Wilton、Raymond McCollum、Viorel Mititean、Andy Cheung、Saar Picker、Drew Bliss、Jason Cunningham、Adam Edwards、Jen-Lung Chiu、Alain Lissoir以及Brandon Jiang。

特别感谢Mark Russinovich，他不仅审阅了本书，还为本书撰写了序。Mark的一些经典著作在软件开发人员中广为流传，这些书籍对我们以及无数其他的工程师们都有着深远的影响。

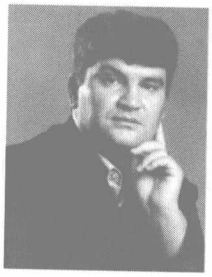
Ivan Brugiolo同样审阅了本书，并且提供了深入的反馈意见。Ivan非常乐意分享他的经验和知识，这为本书增色不少。

我们还要感谢Alexandra Hewardt设计和实现了本书的网页。

作者简介



Mario Hewardt是Microsoft公司的高级设计师，他在过去9年中主要从事Windows系统级的开发工作。在Windows的5个版本（从Windows 98开始）中，他主要在服务器和桌面管理等领域工作，负责确保产品的可靠性、稳定性以及安全性。



Daniel Pravat是Microsoft公司的高级设计师，负责Windows中的一些组件。在加入Microsoft之前，他主要为基于计算机的电话服务器开发电信软件。他希望所有的软件程序都是可靠的、可预测的和高效的。

目　录

第二部分 调试实践