华章科技
PEARSON
信息安全技术丛书

安全软件开发之道

构筑软件安全的本质方法

[美] John Viega　Gary McGraw　著
殷丽华　张冬艳　郭云川　颜子夜　译

Building Secure Software

How to Avoid Security Problems the Right Way

- 本书被誉为安全技术领域的“黄帝内经”，由安全技术大师亲力打造，畅销全球，数位安全技术专家联袂推荐。
- 综合论述如何在软件开发整个生命周期内建立安全屏障，对于如何设计安全的软件给出了高屋建瓴的指导，全面翔实，深入浅出。对于任何关注安全软件开发的人来说，都是一本必备之书。

机械工业出版社
China Machine Press

安全软件开发之道

构筑软件安全的本质方法

Building Secure Software
How to Avoid Security Problems the Right Way

[美] John Viega　Gary McGraw　著
殷丽华　张冬艳　郭云川　颜子夜　译

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

安全软件开发之道：构筑软件安全的本质方法/（美）维格（Viega, J.），（美）麦格劳（McGraw, G.）著；殷丽华等译．—北京：机械工业出版社，2014.3
（信息安全技术丛书）
书名原文：Building Secure Software: How to Avoid Security Problems the Right Way

ISBN 978-7-111-45915-6

I. 安… II. ① 维… ② 麦… ③ 殷… III. 软件开发－安全技术 IV. TP311.52

中国版本图书馆 CIP 数据核字（2014）第 032794 号

本书被誉为安全技术领域的“黄帝内经”，由安全技术大师亲力打造，畅销全球，数位安全技术专家联袂推荐。综合论述如何在软件开发整个生命周期内[illegible]关于设计安全的软件给出了高屋建瓴的指南，全面翔实，深入浅出。对于任何关注安全软件开发的人来说，都是一本必备之书。

本书分为两大部分。第一部分介绍在编写代码之前应该了解的软件安全知识，讲解如何在软件工程的实践中引入安全性，任何涉及软件开发的人都应该阅读。主要内容包括：软件安全概论、软件安全风险管理、技术的选择、开放源代码和封闭源代码、软件安全的指导原则、软件审计。第二部分涉及软件开发实现的细节，介绍如何在编程中避免一些常见的安全问题，适合编程一线的技术人员参考。主要内容包括：缓冲区溢出、访问控制、竞争条件、随机性和确定性、密码学的应用、信任管理和输入验证、口令认证、数据库安全、客户端安全、穿越防火墙等。

安全软件开发之道：构筑软件安全的本质方法

[美] 维格（Viega, J.）；麦格劳（McGraw, G.） 著

出版发行：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：吴 怡
印 刷：廊城市京瑞印刷有限公司
开 本：186mm×240mm 1/16
书 号：ISBN 978-7-111-45915-6
版 次：2014 年 3 月第 1 版第 1 次印刷
印 张：21
定 价：79.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
客服热线：（010）88378991 88361066
购书热线：（010）68326294 88379649 68995259
投稿热线：（010）88379604
读者信箱：hzjsj@hzbook.com

译 者 序

不治已病治未病，不治已乱治未乱。——《黄帝内经》

软件安全领域最能吸引大众注意力的问题大概就是网络安全或者信息安全了，各种不安全的事件充斥着人们的视听，迫使人们一知半解地接受补丁程序、病毒、木马等枯燥的名词。尤其是在刚开始翻译本书的时候，曝出了棱镜门事件，这一事件更是如同很多突发的新闻事件一样，给普通大众进行了一次关于信息安全的科普。相信更多的民众在茶余饭后的谈资中会添加这个话题。在这种大众的安全意识刚被唤醒，但尚处于懵懂状态之时，民众越是容易被各种流言和曲解所误导，此时，大众更需要对此获得科学、准确的知识。本书尽管不是直接面向一般民众的科普读物，但至少对于专业技术人员可以起到正本清源的作用。

就本书的内容而言，这是一本关于如何构建安全软件的书，在如今动辄将某书夸为圣经或者宝典的时代，如何给这本书定位呢？我觉得称之为软件安全的“黄帝内经”较为恰当。如果将软件看作生命体，安全性必然是衡量其健康水平的重要指标。一些微不足道的疾病就可能导致人的生命丧失，除了抱怨世事无常之外，更需审视我们自身机体的抵抗力，甚至要做一下基因检测。我们的软件也是如此，即便是某个细节的疏忽所造成的后果也可能是灾难性的。或者说，当前软件的生存环境更为恶劣，在意外感染疾病之外，还被各种人和机构虎视眈眈地视为技能实验的小白鼠，或者是获取不当利益的掘金场。这些人无论是仅仅出于对技术的爱好还是心怀恶意，都会对软件的安全造成严重威胁。

不幸的是，面对软件的各种问题和缺陷，人们很少去认真思考其内在的成因，而是忙于进行各种亡羊补牢式的补救，就像生病之后只是处理症状，而没有对疾病的内在成因进行深入的探究，更无从谈及疾病的预防。即便是现在那些号称是软件安全宝典或者圣经的书籍，也是给出各种治病的药方，甚少从软件的根源上来全面提供一套如何构建安全软件的可信赖体系。这使得各种软件安全事件层出不穷，人们不得不对其采取容忍的态度，或者普遍接受软件必然存在错误的观点。

但是事实上，相比较于生命体，软件更容易做好事先的预防工作。尽管现在已经有了转基因技术，但是还不能随心所欲地构造出任意的 DNA。而软件作为一种完全由人工构造出来的产品，可以从构造之初就开始努力避免那些可能出现的设计错误。本书的核心观点是，在给出具体的安全缺陷之外，更重要的是强调在软件设计的前期就开始将安全性作为软件的一种内在特质进行考虑。这种思想与《黄帝内经》的“治未病”思想不谋而合。

本书成书已经十年，这期间新的软件安全事件、新的攻击技术和防卫技术层出不穷。尽管所有这些技术的外在表现花样百出，但其内在根源都归为软件设计的缺陷。本书给出的那些指导性原则依然保有旺盛的生命力，并不因技术的发展而有所失色，本书也依然是值得仔细研读的软件安全方面的经典书籍。希望通过对本书的翻译，将原作者的观点引入国内的软件设计领域。

对于这样一本富有个性的书，译者自知才智浅薄，本着集思广益的方法，召集一批志同道合的人士共同完成原书的研读和翻译工作。在翻译的过程中，着实体会到作者对软件安全方面的深刻理解，本书也是对译者在安全方面知识的一次系统化梳理。本书的翻译和出版得益于张冬艳博士、郭云川博士的精诚合作。此外，尤其感谢颜子夜博士帮助校对了本书的大部分章节。

由于本书涉及面广，许多术语尚无固定译法，翻译难度较大。有时，对一个术语虽经反复推敲、讨论，仍然难免词不达意。此外，由于译者水平有限，译文中的不当之处也在所难免。我们真诚地希望读者不吝赐教。如有任何的意见和建议请发邮件给我们：yinlihua@iie.ac.cn、zhangdy@ustb.edu.cn、guoyunchuan@iie.ac.cn 或者 yanziye@hotmail.com，我们将不胜感激。

殷丽华

2013 年 9 月于北京

对本书的赞誉

“John 和 Gary 对计算机安全性提供了一个令人耳目一新的视角。一开始就做得正确，那么就无须再修修补补。在当今这个‘蹩脚软件’（shovelware）盛行的世界，这无疑是一种超前而又重要的观念！在这个各大软件厂商都在为产品版本的 beta 测试而伤脑筋的行业中，本书不失为醒脑之音，值得一读！”

——Marcus J. Ranum，NFR 安全公司 CTO，
《Web Security Sourcebook》的作者

“系统开发人员们：学习这本书，捍卫你们的系统，网络空间将变得更好。”

——Fred Schneider，康奈尔大学计算机科学教授，
《Trust in Cyberspace》的作者

“我们不断地遇到因软件错误引发的安全问题。系统越复杂，找到问题的难度就越大，花费也越高。当我们期望进行安全可靠的交易，并从物理身份识别领域朝数字身份识别领域迈进时，遵循本书提出的一系列原则就显得越来越重要了。本书写得非常好，对于任何关注安全软件开发的人来说，都是一本必备之书。”

——Terry Stanley，Master Card 公司芯片安全副总裁

“在入侵者肆虐之时，很多人都在想着关闭门户，但 Viega 先生和 McGraw 先生却在计算机安全的源头入手，研究如何预先在系统中构建安全性。简单来说，就是如何处理基本的安全性优先级问题。”

—— Charlie Babcock，Interactive Week

“应用安全问题是阻碍电子商务发展的最重要问题之一。本书以一种简单明了并易于理解的方式提供了复杂的应用程序安全问题的解决方案。例如缓冲区溢出、竞争条件以及实施加密的方式。

这是任何应用程序开发人员或安保专家都应读的一本书。”

——Paul Raines，巴克莱资本（Barclays Capital）信息风险管理的全球主管，
《软件杂志》（Software Magazine）专栏作家

“Viega 和 McGraw 先生终于完成了技术界期待已久的作品。贯穿全书的中心思想是，两位顶级专家关于如何建立安全系统的全新视角，以及基于风险管理的方法解决安全问题。无论是在代码中避免缓冲区溢出，还是了解元素的完整性和交互性，本书都提供了全面而详实的指南。作者阐释了理解和管理风险对于任何系统项目的重要性。本书是任何从事系统设计、系统构建或系统管理的人必读之书。”

——Aviel D. Rubin 博士，AT&T 实验室首席研究员，
《White-Hat Security Arsenal》和《Web Security Sourcebook》的作者

“本书恰逢其时！”

——Michael Howard，微软 Windows XP 团队小组安全主管

“对于信息安全，正确的操作似乎已成为一项失传的艺术。本书重新发掘了开发安全系统所需的知识、智慧、原则和纪律，同时也可将其应用在可靠性和良好的软件工程实践方面。”

——Peter G. Neumann，《Computer Related Risks》和
《Moderator of RISKS digest》的作者

“John Viega 和 Gary McGraw 先生为设计或实现软件以及关注安全性的专业人士奉献了一本相当实用的手册。本书不仅解释了编写安全软件所涉及的一系列概念和原则，也包含了如何构筑安全软件的具体方法以防范攻击者的破坏，例如，对缓冲区溢出的深入解释，大多数软件灾难的深入分析等。而且，本书还列出了许多有用工具的出处（免费软件等），从而更提高了实用性。这是一本互联网软件开发人员的必读之书。”

——Jeremy Epstein，webMethods 公司产品安全及性能部总监

“安全性其实很简单，只需运行完美的软件即可。但完美是不切实际的，必须寻求切实可行的替代办法，否则就得面临长期的安全漏洞问题。Viega 和 McGraw 先生为软件开发商提供了极好的完美软件的替代实现原则。”

——Crispin Cowan 博士，Oregon 研究所副研究员，
WireX 联合创始人和首席科学家

“大部分人都在处理各种安全性问题的症状，很少有人去探究导致大多数安全问题的根源：设

计和开发周期中的问题。在许多大学中，学生学到的是不安全的编码风格。很多人将开发单用户系统的编程概念带到了彼此关联的网络环境中，这是非常危险的。这些状况正在迅速地削弱国家的重要基础设施以及大多数的商业组织，并将个体公民置于风险当中。当务之急，是大部分软件设计人员要改掉他们的坏习惯并重新学习。因此，我衷心地希望本书能够引起对此领域的关注和重视。毕竟，这个领域是需要‘疗伤’，用户不会总是‘正确地’使用系统，恶意攻击者更是如此。因此，编写安全的代码以抵御恶劣的环境才是真正的解决办法。”

——Mudge，Stake 公司首席科学家和研发执行副总裁

“编程是很困难的，程序员非常珍贵，优秀的程序员更是罕见。要尽可能让软件变得容易使用，且物美价廉，为此，我们需要所有的帮助、所有的工具、所有的原则。我们还没能做到这点，但是本书应该有所帮助。”

——Bill Cheswick，《Firewalls and Internet Security》的作者

“这本书不错。”

——Peter Gutmann，《Cryptographic Security Architecture》的作者

摘自本书序：

“本书在理解安全软件方面是一个非常重要的工具。Viega 和 McGraw 在表述安全软件设计的理论和实践方面均做得非常出色。本书非常实用，易于理解且非常全面。它并不会奇迹般地让你摇身一变成为软件安全专家，但它会让你对软件的安全更加敏感。而对安全越是敏感，就越接近问题的最终解决。”

——Bruce Schneier，Counterpane Internet Security 创始人兼首席技术官，《应用密码学》和《网络信息安全的真相》的作者

序

如果软件本身的安全性不是这么糟糕的话，我们也就无须在网络安全方面耗费那么多的时间、金钱和精力了。想想你最近听说的那些最新的安全漏洞吧。也许是极具杀伤力的数据包，让攻击者通过发送特殊包的方式使服务器崩溃；也许是缓冲区溢出漏洞，让攻击者通过发送特定格式的不正确消息来控制某台计算机；也许是加密漏洞，让攻击者读取加密信息或欺骗认证系统。如此种种，安全问题的背后都是软件的问题。

有时候，网络安全可以抵御这些漏洞：可以通过设置防火墙来阻止特定类型的数据包或消息，或只允许来自可信源的连接（但愿攻击者没有在防火墙里面，也不是受雇于可信源）；也可以将入侵检测系统设置为在特定漏洞被利用时报警；可管理的安全监控服务可以捕获正在进行中的漏洞攻击并立即阻止入侵。但是，所有这些漏洞问题的根源都来自于软件，正是糟糕的软件导致了这些漏洞。

糟糕的软件非常普遍，超乎你的想象。通常，大型应用软件会带着数百甚至数千个与安全相关的漏洞运行着。其中一些随着人们对该程序的使用逐渐被发现，于是厂商们针对这些已知的漏洞发布补丁程序，并希望用户安装（它们也确实发挥了作用）或者通过重新配置网络安全设备来抵御这些漏洞。而软件中未被发现的漏洞则继续潜伏，甚至永远都不会被发现。但是它们确实存在，而且都有被发现和利用的可能。

劣质软件难辞其咎。

软件开发系统导致了劣质软件的产生。安全性不可能在开发过程的最后被一下子附加上，必须从一开始就进行正确的设计。遗憾的是，那些负责产品安全的人并不负责软件的开发过程，呼吁增强安全的人总是争不过那些呼吁增强功能的人，拥护和支持软件安全设计原则的人并不负责软件的发布流程。软件公司往往更追求“更多、更快”，而不是“少一些、慢一点、更安全”。

要创建安全的软件，开发人员需要理解如何能够安全地设计软件。这包含几个方面的内容。首先，需要良好的教育和培训，程序员必须学习如何将安全性融入到软件设计中，以及如何安全地编写代码。其次，需要更好的计算机语言和开发工具——这些工具可以在开发过程中捕获常见的安全漏洞，并帮助更容易地解决问题。最重要的是，我们需要一些相关的意识：风险意识、问题意识及修复意识。这些都不可能是一夜之间形成的，必须从始至终努力直到最终达成此种功力。

本书在理解安全软件方面是一个非常重要的工具。Viega 和 McGraw 在表述安全软件设计的理论和实践方面均做得非常出色。本书非常实用，易于理解且非常全面。它并不会奇迹般地让你摇身一变成为软件安全专家，但它会让你对软件的安全更加敏感。而对安全越是敏感，就越接近问题的最终解决。

我们寄希望于你们这些读者（程序员、软件架构师、软件项目经理）能找出问题的解决之道。软件产业本身并不能解决软件的安全问题，因为缺少生产安全软件的市场刺激——这是软件缺乏责任的结果。缺乏生产安全软件的市场积极性的原因是，即使产品不安全，软件厂商也不用承担任何风险。软件产业已经证明，在各种产品接连不断的情况下，即使生产漏洞百出的软件，仍然能够占有一定的市场份额。假如汽车制造商对产品也可以免责，那么我们就可能买到一部使用一氧化二氮喷油嘴的摩托车，也可以加速到超过刹车能够承受的速度。厂商可以不计后果地制造尽可能多的性能和功能。但事实上这些不可能出现，因为如果摩托车制造商销售不安全的产品的话，他们将面临法律诉讼。

软件产业没有类似的诉讼。看看你刚刚购买的软件（甚至是你帮助写的）的拆封许可证，软件只按“现状”出售，没有任何保证。甚至都不保证软件的功能与广告宣传的一样，更不用说安全性或可靠性了。它也不能保证安装后不会导致整个网络崩溃。事实上，软件厂商会明确免除任何责任，如果他们有责任的话。

多年来我一直在呼吁，计算机安全也像其他所有安全问题一样，需要从风险管理方面来考虑。我们无法通过神奇的技术和程序来避免威胁。我们需要的是管理安全风险。在这个意义上，电子空间与现实世界没有什么不同。人类四千多年以来一直在试图解决社会安全问题，但仍然没有技术可以杜绝盗窃、绑架或谋杀。我们最好的办法也只是管理风险。为什么电子空间不是这样呢？

本书就是采用风险管理方法来解决安全性问题。近年来我的关注点主要是对安全问题进行检测和响应，本书则侧重于预防。最重要的是，它在源头，也就是在软件的设计阶段防范安全问题。本书开篇就设置了恰当的目标和期望。前面几章讨论了以风险管理方式对待软件安全的重要性，在安全性和最终目标（能正常工作的软件）之间引入了技术权衡的思想，并使之根植于心。第 5 章和第 6 章则是深入安全设计的核心，提供了构建安全软件的十项指导原则，并阐述了软件审计的思想。接下来的章节就比较偏重于技术了，主要是软件安全的技术细节，范围也很广，涵盖了从恶性缓冲区溢出到开发商所面临的应对防火墙的问题等。

软件安全仍然有很长的路要走。我们不仅需要学习如何去做，更需要懂得这样做的重要性。你手中有这本书就表明你在正确的方向上迈出了一步。阅读它，学习它，并付诸实践。

希望寄托在你们身上。

Bruce Schneier

Counterpane Internet Security 创始人兼首席技术官

http://www.counterpane.com

前　言

“一本书就是一部思想的机器。”

——I.A. Richards《Principles Of Literary Criticism》

本书旨在帮助从事软件开发的相关人员学习构建安全软件所必需的原则。尽管包含了一些底层的技术细节，但本书的读者对象包括软件开发过程中的所有人，从管理人员到开发人员。本书后半部分展示了一些具体的代码实例和技术细节，而前半部分则是针对安全问题的一般性原则和通用知识，目的是营造一个安全软件的环境，包括安全目标、安全技术以及软件风险管理的概念。

涉及计算机安全的技术书籍有很多，但迄今为止，还没有一本书着重讨论开发安全的软件这一主题。如果你想学习如何建立防火墙，锁定单台主机或建立虚拟专用网络，你会找到许多其他有用的资料，本书并不包括在内。大多数有关安全的书籍是为了解决人们在网络安全方面迫切关心的问题，因此往往关注的是，在软件总是一再被攻破的网络世界中，如何提高保密程度，如何保护网络资源。

遗憾的是，许多安全人员已经习惯于这样的事实：软件中存在安全问题是很平常的，甚至是可以接受的。有些人甚至认为要求开发人员构建安全的软件太困难了，因此不会提出这个问题。于是，他们努力进行“最佳实践”的网络安全解决方案，架设防火墙，以及努力实现及时检测入侵和修补已知的安全问题。

我们认为软件的安全性问题是可以解决的，并对此保持乐观态度。但事实是，编写没有安全漏洞的程序是很难的。然而可以肯定，编写一个“足够安全”的程序比编写一个完美无错的程序要容易得多。难道因为不可能彻底剔除软件中的所有错误，就从根本上放弃对软件错误的扫荡吗？当然不能！同样，人们也不应该在没有理解这个问题之前，就自动放弃对构建安全软件的努力。

学习任何一点点知识都可能有深远的影响。这么多的产品都有安全问题，其中一个最大的原因是，许多从事开发的技术人员从来没有学习过如何编写安全的代码。另一个原因是，到目前为止，几乎没有什么地方可以提供有益的信息。本书旨在消除这种教育差距，以编写安全程序所需的基本必备技巧来武装软件开发人员。

也就是说，你不能期望仅仅通过阅读本书就可以消除软件中的所有安全问题。声称本书为安全性提供了良方，相当于忽视了这一事实：建立安全的计算机软件有多么困难。我们不能忽视现实，应该正视和接受它，并且运用风险管理的思想来对待软件安全性问题。

在现实世界中，软件几乎不可能绝对安全。首先是没有 100% 安全的事物。大多数软件都存在可被利用的安全风险，问题仅在于突破系统需要花费多少时间和精力而已。即使软件没有任何 bug，服务器也被防火墙保护着，盯着你的人也会通过内部来攻击你，或者采用“黑包”的突破手法。安全性是复杂的，是一个系统工程问题，因此，我们不仅提供了安全软件设计的一般性原则，同时也关注那些最常见的风险，以及降低这些风险的方法。

本书组织方式

本书分为两部分。第一部分侧重于在编写代码之前应该了解的软件安全相关知识内容，焦点是如何将安全性集成到软件工程实践中去，重点是在开发周期的早期降低和减少安全风险的方法和原则。在系统设计初期就考虑安全性是一种相对容易的做法，在投资和花费上远比事后修补便宜得多。本书不仅关注需求和设计，还特别强调了系统的安全性分析，这也是一项非常关键的技能。本书第一部分对涉足软件开发任何层次的人都是有益的，无论是商业层的领导还是一线的具体开发人员。

在本书的第二部分，我们深入到实现的细节中。即使是一个相当坚实的架构，在开发时也会有许多和安全问题相关的死角。我们向开发人员展示了鲜活生动的细节，以阐述如何识别和避免一些常见的编码级问题，例如缓冲区溢出、竞争条件等。本书第二部分适用于那些奋战在编程第一线的人。

我们尽量提供普遍通用的材料，也就是说，除非是安全攸关的问题，否则我们都会尽量避免那些依赖特定操作系统或程序设计语言的内容。例如，我们不讨论 POSIX 的“性能”，因为它们没有普遍性。但我们却用整整一章来讨论缓冲区溢出，因为这是一个极其重要的问题，尽管大多数缓冲区溢出都和 C 或 C++ 相关。

由于我们关注的是那些能够应用在最广泛层面的技术，因此许多有价值的技术本书没有涵盖，包括 Kerberos、可插拔认证模块（pluggable authentication module, PAM）以及移动代码（mobile code）、沙箱技术（sandboxing）等。许多技术都值得专门编写一本书（并非每种技术都有足够广泛的应用）。本书的配套站点（http://www.buildingsecuresoftware.com/）提供了这些安全技术的相关资源的链接。

代码示例

虽然本书提供的材料在很大程度上与具体程序语言无关，但我们的示例大多是用 C 语言编写

的，主要是因为 C 语言应用比较广。另外，与其他语言相比，用 C 语言来表达事件的准确性要更难一些。将示例代码移植到其他编程语言很容易，主要是找到相应的调用或数据结构即可。不过，通常在一些与用 C 语言编写差别比较显著的情况下，我们偶尔也用到了以 Python、Java 和 Perl 编写的代码示例。本书所有代码都可在 http://www.buildingsecuresoftware.com/ 上得到。

尽管坚持操作系统独立的原则，本书仍然比较侧重于 UNIX 系统。我们承认对其他操作系统（特别是 Windows 系统）特性的讨论，还不甚理想。虽然 Windows NT 对 POSIX 是松散兼容的，但实际上，Windows 系统的程序员往往并不使用 POSIX 应用编程接口（API）。比如，我们听说大多数 Windows 程序员不使用标准的 C 字符串库，而青睐于使用 Unicode 字符串处理例程。在撰写本书时，我们仍不知道 Windows API 中哪些常见函数容易受到缓冲区溢出的影响，所以无法提供一个全面的列表。如果以后有人能给出这样的列表，我们将很乐意将其发布到本书的网站上。

本书提供的代码已全部在运行 Red Hat 6.2 的机器上测试过。大部分程序也在 OpenBSD 机器上测试过。不过，这些代码仍然是在“想当然”的基础上的。尽管我们努力使 Web 站点上发布的代码尽可能具备可移植性，但事先声明，这些资源的可移植性仍然很低。我们没有足够的时间去帮助大家解决在特定系统下编译代码遇到的问题，但我们真诚地欢迎读者提供软件补丁。

联系我们

欢迎大家给我们发电子邮件，任何评论、错误修订或其他建议都让我们欣喜和感激。我们的网址是 http://www. buildingsecuresoftware.com。

致　谢

在众多人的参与下，本书得到了极大的改善。当然，对于书中出现的错误和遗漏主要责任在我们。非常感谢以下人员为本书初稿提供了细致并富有帮助的意见，他们分别是：Bill Arbaugh、Steve Bellovin、Leigh Caldwell、Pravir Chandra、Bill Cheswick、Jeremy Epstein、Ed Felten、Robert Fleck、Jim Fox、Zakk Girouard、Peter Gutmann、Brian Kernighan、Matt Messier、Mudge、Peter Neumann、Tom O'Connor、Jens Palsberg、Marcus Ranum、John Regehr、Steve Riley、Avi Rubin、Craig Sebenik、David Strom 以及 Guido Van Rossum。

Peter Gutmann 为我们提供了一些非常棒的代码实例，其中包括第 9 章中安全打开文件的代码以及第 14 章中 ASCII 到 ASCII 的加密代码，我们在使用它们时稍微做了一点修改。Radim Bacinschi 为我们提供了第 15 章中的汇编代码。在本书接近最后期限时，Ashit Gandhi 在短短的几分钟内为我们提供了一张 SecurID 的照片。

最后，我们感谢 Addison-Wesley 出版社的所有人，尤其是编辑 Karen Gettman，以及她的助手 Emily Frey。

John 的致谢

首先最重要的是，我想感谢我的整个家庭，尤其是我优秀的女儿——Emily 和 Molly，以及 Anne、Michael、妈妈和 John。我希望可以将花费在这本书上的大量时间都补偿给你们，并且还要提前为下一次的工作道歉！

我想感谢那些多年来一直在指导我的人，他们包括 Reimier Behrends、Tony Brannigan、Tutt Stapp-Harris 和 Randy Pausch。在我生命中一直指导我的人当中，我必须特别感谢 Paul Reynolds。对于在我最需要的时候他所给予我的鼓励和建议，我所给予的感谢远远不够。多年来，他实实在在地提供给我许多帮助。

Secure Software Solutions（www.securesw.com）是一个十分不错的公司。我很高兴周围有这么多天赋高且诚恳的人。我想感谢很多直接或间接帮助该公司成功的人，包括 Pravir Chandra、Jeremy Epstein、Guillaume Girard、Zachary Girouard、Dave Lapin、Josh Masseo、Doug

Maughan 和美国国防部高级研究工艺局（Defense Advanced Research Process Agency，DARPA）、Matt Messier、Mike Shinn、Scott Shinn 和 David Wheeler，以及我们的所有客户。

同样，我也得到了许多 Shmoo Group（www.shmoo.com）成员的帮助。Bruce Potter 不仅是一个很棒的朋友，他还召集了那些头脑异常灵活的人，其中包括：Dustin Andrews-Jon Callas、Craig Fell、Paul Holman、Andrew Hobbs、Tadayoshi Kohno、Preston Norvell、John McDaniel、Heidi Potter、Joel Sadler、Len Sassaman、Adam Shand、Rodney Thayer、Kristen Tsolis 以及 Brian Wotring。

我很幸运在安全社区结识了许多优秀的朋友，他们的观点一直影响着我，他们包括：Lee Badger、Nikita Borisov、Ian Brown、Crispan Cowan、Jordan Dimov、Jeremy Epstein、Dave Evans、Ed Felten、Rop Gonggriip、Peter Gutmann、Greg Hoglund、Darrell Kienzle、John Kelsey、John Knight、Carl Landwehr、Steve Lipner、Mudge、Peter Neumann、Jens Palsberg、David Wagner、David Wheeler、Chenxi Wong 和 Paul Wouters (D.H.)。

我在技术圈中的朋友，无论是过去的朋友还是现在的朋友，都值得感谢，因为是他们让我获得广泛信息并且保持思路敏捷，他们包括：Roger Alexander、J.T. Bloch、Leigh Caldwell、Larry Hiller、Tom Mutdosch、Tom O'Connor、James Patten、George Reese、John Regehr、Rob Rose、Greg Stein、Fred Tarabay、Guido Van Rossum、Scott Walters 以及 Barry Warsaw。

我还要感谢那些可能以为我在职业追求过程中早已将他们遗忘了的好朋友，包括：Marcus Arena、Jason Bredfeldt、Brian Jones、J.C. Khul、Beth Mallory、Kevin Murphy、Jeff Peskin、Chris Saady、David Thompson、Andy Waldeck、Andrew Winn、Chris Winn 和 Jimmy Wood。我并没有忘记你们，你们对我一直都很重要。我要想对那些我一时想不起来感谢的人致歉，你们知道我指的是谁。

我也很庆幸多年来能与许多优秀的人一起工作，他们每个人都为这本书做出了积极的贡献。在 Randy Pausch 的用户界面组离开弗吉尼亚大学之前，组内的每个人都教会了我很多。我也要感谢我以前的雇主——Widevine Technologies，包括 Thomas Inskip、Dan Robertson、Michael Rutman、Eric Shapiro 和 Amanda Walker，尤其是 Brad Kollmyer 和 Neal Taylor。Widevine 的支持对这本书的完成是至关重要的，我亏欠他们很多。我还必须感谢 Cigital 中那些不错的人。我还要感谢在每个我所参与的咨询项目中遇见的人，其中每一次咨询都是很好的学习经历。在咨询业务过程中，我见过的有才华的人太多了，在这里无法全部罗列出来，但我对你们所有人都有深深的感激之情。

Mike Firetti 想付款给我要求我感谢他，但后来他没有支付支票。在我拿到钱的时候，他就会得到我对他的感谢，只是现在价格已经上涨到 2.50 美元了。

我要特别感谢 Steve Bellovin、Jeremy Epstein 和 Peter Gutmann，他们想尽办法来帮助我们完成这本书。

最后，我要感谢我的合作作者 Gary McGraw，尽管他非常喜欢，但并没有使用太多的混合隐

喻、成语和其他俗语。作为回报，我尽量没有在句子开头使用“so”这个词。无论是关于本书还是本书之外的其他工作，与 Gary 的合作都非常愉快。他不仅是一个非常不错的合作者，同时也是一位非常不错的朋友。

Gary 的致谢

本书是在 Cigital（原名为可靠的软件技术（Reliable Software Technologies））的肥沃土壤中成长起来的。Cigital（www.cigital.com）一直是一个不错的工作地点，在那里，使软件运行起来始终是件有趣的挑战。特别要感谢我在管理团队中的同事：Jeff Payne、Jeff Voas、John Bowman、Charlie Crew、Jen Norman、Ed McComas、Anup Ghosh 以及 Rich Leslie，是他们让我有时间来写这本书。Cigital 的软件安全组（Software Security Group，SSG）于 1999 年由 John Viega、我和 Brad Arkin 创建。现在由 Mark McGovern 运作的 SSG 已跻身于世界最优秀的软件安全从业团队。感谢所有使软件安全远景的关键部分成为现实的 SSG 成员。还要特别感谢容忍我的古怪行为和过度出游行为的 Stacy Norton。

我的合著者 John 也值得大力称赞，因为是他促使我开始写这本书的（他用星巴克的 double short dry cap 贿赂了我）。

James Bach 也值得称赞，因为他让我为《IEEE Computer》写了一篇名为《Software Assurance for Security》的文章 [McGraw 1999b]。正是那篇文章使我一发不可收拾。接下来，IBM 的 DeveloperWorks 门户网站也帮助促使我开始写这本书，在那里我和 John 写下一些后来成为本书内容的原始材料。那篇原始文章仍然可以访问 http://www.ibm.com/developer/ security 获得。

正如我所有的书，这本书也是合作的结果。我在安全社区中的朋友以各种各样的方式提供了帮助，他们包括：Ross Anderson、Matt Bishop、Steve Bellovin、Bill Cheswick、Crispin Cowan、Drew Dean、Jeremy Epstein、Dave Evans、Ed Felten、Anup Ghosh、Li Gong、Peter Honeyman、Mike Howard、Carl Landwehr、Steve Kent、Paul Kocher、Greg Morrisett、Mudge、Peter Neumann、Marcus Ranum、Avi Rubin、Fred Schneider、Bruce Schneier、Gene Spafford、Kevin Sullivan、Roger Thompson 以及 Dan Wallach。感谢 DARPA、美国国家科学基金会（National Science Foundation）和美国国家标准与技术高级技术项目机构（National Institute of Standards and Technology's Advanced Technology Program），感谢他们多年来对我研究的支持。特别感谢那些教授了我宝贵安全课程的 Cigital 客户，包括 Ken Ayer（Visa）、Lance Johnson（DoubleCredit）和 Paul Raines（巴克莱）。

最后，也是最重要的，感谢我的家人。我想他们最终还是不再幻想写作本书只需要一小段时间。Amy Barley、Jack 和 Eli，我爱你们。对农场里的动物们我要发出特别的呼喊：猪腊肉、鼠尾草、willy 和 sally、walnut 和 ike、winston、craig。也同样感谢 rhine、april、cyn、heather 以及 1999 年的 penny。

目　录