Professional Java Security

Java 安全性编程指南

〔美〕 Jess Garms
Daniel Somerfield 著

庞 南 管和昌 陈立志 等译

wrox 乐思
PROGRAMMER TO PROGRAMMER™

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

Professional Java Security

Java安全性编程指南

〔美〕 Jess Garms
Daniel Somerfield 著

庞 南 管和昌 陈立志 等译

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

内 容 提 要

本书深入讲述了使用密码学技术和Java安全模块保护系统安全的课题，并把重点放在应用程序的开发过程上。本书首先介绍了在开发安全的计算机系统时需要考虑的各种因素以及实现技术，并通过实例讲述了使用密码进行数据加密和实体认证的基本概念。然后重点介绍了Java安全模块以及安全的企业级应用程序的开发方法。

图书在版编目（CIP）数据

Java安全性编程指南/（美）盖母（Garms, J.）等著；庞南等译. - 北京：电子工业出版社，2002. 1
书名原文：Professional Java Security
ISBN 7-5053-7312-9

Ⅰ. J… Ⅱ. ①盖… ②庞… Ⅲ. JAVA语言 - 程序设计 - 安全技术 Ⅳ. TP312

中国版本图书馆CIP数据核字（2001）第089877号

书　　名：**Java安全性编程指南**
著　　者：〔美〕Jess Garms　Daniel Somerfield
译　　者：庞　南　管和昌　陈立志　等
责任编辑：刘　娟　冯　欣
印 刷 者：北京天竺颖华印刷厂
装 订 者：三河金马印装有限公司
出版发行：电子工业出版社　http://www.phei.com.cn
　　北京市海淀区万寿路173信箱　邮编：100036　电话：68279077
　　北京市海淀区翠微东里甲2号　邮编：100036　电话：68252397
经　　销：各地新华书店
开　　本：787×1092　1/16　印张：30.75　字数：780千字
版　　次：2002年1月第1版　2002年1月第1次印刷
书　　号：ISBN 7-5053-7312-9
　　TP·4202
定　　价：50.00元

版权贸易合同登记号　图字：01-2001-2138

作 者 简 介

Jess Garms

Jess Garms是ISNetworks公司的共同创始人和首席技术官。ISNetworks是一个提供基于Java的安全软件的公司。Jess Garms在该公司中负责开发过程的协调工作以及新技术的调研。

Jess Garms对以下各位表示感谢：

感谢Dave Rueter和Joe Mobley帮助进行本书的组织并帮助了解读者对本书的需求。

感谢Josh Eckels开发了本书第11章的大部分代码，并不厌其烦地编辑了那些.policy文件。

最后，感谢Lisa在过去一年中能够支持我写这本书。

Daniel Somerfield

Daniel Somerfield是ISNetworks公司的首席执行官和共同创始人。他负责公司的战略策划和软件系统结构，以及客户的管理。Daniel Somerfield为多家不同的公司（既有国内公司也有国际公司）做项目管理。

Daniel Somerfield对以下各位表示感谢：

感谢Josh Eckels、Dave Rueter和Joe Mobley在开发本书代码中所给予的帮助。你们是和我一起工作过的最好的开发团队。

引　言

安全永远是开发人员关心的问题——在所有IT会议上，关于这个问题的研讨会总是人满为患。互联网上电子商务的巨大增长使公众更多地意识到网络安全的重要性，以及在这个领域的失败将导致的问题。对于一个电子商务公司，计算机安全上的漏洞不仅使该公司内部感到尴尬，更重要的是打击了公众对于这个公司，甚至于对于整个正处于雏形的电子商务消费领域的信心。同样，对于B2B公司，这样的漏洞将严重损害工作关系和伙伴关系。

计算机安全涉及众多的课题——一个安全应用程序不仅要抵御某种破坏性的攻击（如拒绝服务攻击（denial of service），或病毒），还要抵制窃听、未授权的访问、对数据的篡改，等等。因此，我们将考虑如下领域：数据加密、认证和非否认。

本书实际上可以加一个子标题：使用密码技术和Java安全模块保护你的系统，因为本书的主要焦点是应用程序的开发。计算机安全是一个广泛的领域，在快速浏览第1章的应用安全问题的时候，我们并不准备花太多的时间考虑网络层和操作系统级别的课题。但完整地说，在考虑安全问题的时候，你要考虑端到端的应用，以及在整个系统结构的所有点的弱点。系统需要层次化，应用层和底层系统结构都必须是安全的。当然，安全性必须集成到整个开发过程中，而不应该是事后修补。

在本书开始的时候，我们需要对加密方法和编程方法有一个综合的理解，特别是针对Java语言和Java开发平台。为了达到这个目的，我们需要了解核心的Java安全特性。通过本书中获取的知识，在结束本书学习的时候，我们将能够设计和构建一个从浏览器或客户程序，通过中间层，到数据存贮层的安全应用程序。

本书为谁而写

本书讲述了一个Java程序员在设计和编写安全的应用程序时需要了解的所有事情。尽管我们在开始时给出的例子和描述比较简单，但到本书结束的时候，这些思想组合到一起将给出一个综合的解决方案，以构建一个端到端安全的应用程序。

本书的目标是中级和高级Java程序员，熟悉分布式应用程序开发的基础概念，如Java Sockets、RMI、JDBC和J2EE技术（如Servlet和JSP）。因此本书将集中讲述提高安全性的途径，并将详细解释如何使用关键的Java密码组件。如果你对我们提到的核心Java技术有兴趣，并想加强这些背景知识的话，我们推荐Wrox出版社出版的两本书：Professional Java Programming（Java专业编程指南，该书中译本已由电子工业出版社出版）和Professional Java Server Programming - J2EE Edition（Java服务器编程指南（J2EE版），该书中译本即将由电子工业出版社出版）。

本书包含的内容

在本书的第1章中，我们简要介绍了在开发一个安全的计算机系统时需要考虑的不同领域，并快速讨论了这些系统中使用的多种技术。第1章暂时撇开了本书的核心主题——开发基于Java的安全应用程序。既然本书讲述的“安全”这个词覆盖了如此多的主题，第2章将讲述编写安全的代码的通用途径，以避免在应用程序中引入“不受欢迎”的特性。

第3章介绍了密码。通过一个广为人知的莎士比亚的戏剧，我们给出了一个使用密码进行数据加密和实体认证的、容易理解的、全面的描述。不仅如此，我们还引入了Java密码架构（Java Cryptography Architecture，JCA）和Java密码扩展（Java Cryptography Extension，JCE），以支持Java加密。对于加密的主题，我们在第4章和第5章中分别讨论了对称和非对称加密，并分别给出了实际的例子。第6章讨论了安全认证的问题，并给出了消息摘要，数字签名和数字认证的使用样例。

在第7章中我们将关注Java 2安全模块和Applet安全性。尽管许多人在不同程度上对这个领域比较熟悉，前几章中对密码学的论述将使我们对这个模块的实现有更好的评判。第8章转到了企业级应用程序的开发。我们将从更多方面介绍与Servlet、企业级JavaBeans（Enterprise JavaBeans，简称EJB）、Java认证和授权服务（Java Authentication and Authorization Service，简称JAAS）相关的Java平台安全性。为了在HTTP协议上构建更安全的应用程序，在前面介绍的加密原理的基础上，人们开发了SSL（Secure Sockets Layer，安全套接层）。这种技术促进了网站上的电子商务的发展。在第9章中我们将介绍如何在应用程序中使用SSL。

接下来的两章将把前面讲述的所有概念放到一起。第10章将介绍维护数据库安全的途径。主要从两方面进行论述：一是如何使数据库连接更难以被攻击，另一方面是如何在同一数据库内对数据进行加密。然后，在第11章中，我们将演示如何把加密和认证的概念应用到一个成熟的企业型应用程序中。

在本书的主要内容之外，我们还提供了大量的附录和详细补充材料，如加强应用程序安全性的其他技术，包括E-mail安全，时间戳技术等，还有本书使用的某些软件的安装细节。

使用本书需要的条件

每个示例程序需要的软件要求详见每章的具体内容，这里把本书的代码示例程序需要的软件列举如下：

- Java 2平台，Standared Edition SDK（JDK 1.3），即Java 2平台，标准版SDK。
- Java 2平台，Enterprise Edition SDK 1.2.1 Reference Implementation，即Java 2平台，企业版SDK 1.2.1参考实现。

本书还使用了如下扩展平台：

- Java Authentication and Authorization Service（JAAS），即Java认证和安全服务。
- Java Secure Sockets Extension（JSSE），即Java安全套接字扩展。
- A Java Cryptography Extension（JCE），即Java加密扩展。

所有这些软件都可以从http://java.sun.com网站下载。我们使用的是JAAS v1.0版和JSSE 1.0.2。出于本书第3章给出的理由，尽管已经有JCE 1.2.1可用，本书还是采用了Bouncy Castle Cryptography API 1.04版并进行了扩展，该软件可以从http://www.bouncycastle.org下载。这是一个免费的网站，并且还有更完整的实现。

我们已经意识到未来的Java版本中可能包含这个领域的新变化。然而，我们只想提供纯粹的技术信息，所以只专注于写作本书时的情况。在适当的时候，我们将把你的注意力引到未来可能发生变化的地方。

在讨论安全性的不同地方，我们还使用了如下软件：

- JavaSoft的HTML Converter——这个软件可以从Sun公司的网站免费下载：http://java.sun.com/products/plugin/1.2/features.html。它可以把Web网页中对Java Applet的调用转换为可以在多个浏览器中使用的格式。
- Internet Explorer——可以从http://www.microsoft.com/downloads/下载。
- Netscape和Netscape的Capabilities API——浏览器可以从http://home.netscape.com/download/index.html?cp=djuc1下载。Netscape的Java安全模型和标准1.1或1.2模型有所不同，Netscape称之为Capabilities API。如果要在Netscape中运行applets，我们必须安装Capabilities API。这些classes可以从如下网站下载：http://developer.netscape.com/docs/manuals/signedobj/capsapi.html 。
- Microsoft的SDK for Java——为了在Internet Explorer中创建并注册.cab文件，从而系统可以运行applets，你必须安装Microsoft的SDK for Java。你可以从http://www.microsoft.com/java/下载。本书写作时候的最新版本是4.0。
- 一个数据库和相应的JDBC驱动程序——我们选的是MySQL（从http://www.mysql.com下载）。本书中讨论的代码都是设计并运行在这个数据库上的。其他与SQL92兼容的数据库，例如Oracle, SQL Server或PostgreSQL也同样可以使用，但代码需要做修改。MySQL的JDBC驱动程序可以从很多网站获取，例如：http://mmmysql.sourceforge.net。附录B中讨论了数据库安装和JDBC驱动程序方面的问题。
- 一个XML解析器——我们采用的是Xerces XML解析器来运行本书的例子。Xerces可以从如下网站下载：http://xml.apache.org/xerces-j/index.html。我们使用了1.3.1版来测试程序。
- 一个Web服务器——为了能使用第11章介绍的安全应用程序，我们采用了直接支持SSL的Tomcat 3.2.1服务器。它可以从 http://jakarta.apache.org/tomcat/index.html下载。也可以使用其他servlet引擎，如Sun和Netscape的iPLanet，或者IBM的WebSphere。但是和数据库一样，我们只提供Tomcat的安装信息。

源代码

本书的完整源代码可以从如下网站下载：http://www.wrox.com 。

由于安全类型代码和这个领域本身的特点，我们希望你注意到：

作者和出版商已经竭尽全力使本书表达的信息尽量准确。然而，本书提供的信息一经

售出后，不论是明文说明的还是隐含的，都不做任何保证。不论是作者、出版社，还是经销商和分销商，都不对声称直接和间接由本书引起的任何损害承担责任。

尽管是面向网络的，本书的源代码可以在单独一台计算机上运行，也就是说，它可以通过本地浏览器来访问http://localhost。惟一的例外是第10章的SSL-tunneling的例子，它需要两台计算机才可以运行。

第11章的应用程序已经设计为可以在多计算机系统上运行，但为了简单起见，供下载的代码被配置为单机运行。

使用惯例

为了帮助你更好地理解本书的内容，我们使用了大量图例来帮助说明，另外我们还使用了一些样式。

这些样式包括：

- 在介绍的时候，我们把重要的词语着重显示。
- 键盘按键使用如下样式显示：Ctrl-A
- 文中的文件名和代码使用如下样式显示：doGet( )
- 用户界面中的文本以及URL显示为：Menu

我们使用几种不同方式来显示代码，命令行和终端输出显示为：

```
C:\> java showStyle
When the command line is shown, it's shown in the above style, while terminal
output
is in this style.
Output needing a: response
is shown like this
```

方法和属性显示为：

```
protected void doGet (HttpServletRequest req, HttpServletResponse resp)
        throws ServletException, IOException
```

示例程序代码显示方式如下：

```
In our code examples, the code foreground style shows new, important,
  pertinent code
while code background shows code that's less important in the present context,
  or has been seen before.
```

客户支持

我们非常愿意知道你对本书的想法：哪些喜欢，哪些不喜欢，哪些下次可以做得更好。你可以把你的意见通过E-mail发送给我们。E-mail地址是：feedback@wrox.com。请务必注明信件的标题。

P2P.WROX.COM

要获取作者和同行的支持，请加入我们的Java邮件组。我们拥有独一无二的系统，它的邮件列表、论坛和新闻组可以提供程序员到程序员的支持（programmer-to-programmer™ support），以补充一对一的邮件系统的不足。你的问题不仅会被我们的专业技术支持人员审阅，还将被很多Wrox的作者和其他在邮件组中的业界专家看到。在p2p.wrox.com网站,你将看到许多针对Java编程的不同列表，以及对安全问题感兴趣的开发人员。他们不仅会在你阅读本书的时候提供支持，而且会在开发应用程序的时候提供支持。适合本书的有一些安全方面的邮件组，但在Java目录下还有许多其他相关的组。

要获取邮件组支持，请按如下步骤操作：

1. 访问p2p.wrox.com。
2. 点击Java或Security按钮。
3. 点击你想加入的邮件组。
4. 填入你的E-mail地址和口令（至少4个字母），并给我们发E-mail。

为什么本系统提供了最好的支持

你可以选择加入邮件组，或者每周接收一份文摘。如果你没有时间或条件接收邮件列表，你也可以检索我们的在线历史文档。垃圾邮件和无关的邮件将被删除，并且我们特有的Lyris系统将保护你的邮件地址。关于加入或退出邮件组，或其他关于邮件组的问题，请发E-mail到：listsupport@p2p.wrox.com

目　录

第1章 安全性考虑

很难对计算机安全下一个定义。尽管包含了很多的领域，大部分情况下，安全主要与对资源的访问控制相关。当你在写一个应用程序的时候，如果发现自己要解决如下问题，你就在考虑计算机安全了：

- 如何传输敏感信息，如信用卡号码
- 如何存储敏感信息
- 如何确保代码的来源是可靠的
- 如何保证只有经过授权的用户才能访问系统

还有很多其他问题，但它们都是围绕如何对信息和资源进行保护。

对于计算机安全，你首先需要了解的是世界上没有安全的系统。所有系统都可以被攻破，这个规则没有例外。在最极端的情况下，一个简单的物理攻击，例如在服务器端拔掉网线，就可以危害你的网络。作为一个计算机程序员，你所能做的是增加系统被破解的难度，以及在系统被攻破时能更容易地恢复系统。本书将讨论很多方法来提高你的应用程序，请把这些目标牢记在心。

本书的焦点是教你如何通过密码学和Java安全模型来保护你的Java应用程序，但在进入编码和加密细节之前，我们还要为你的工作提供一些背景知识。所以本章的主要内容是：

- 在设计安全的应用程序时，在高层次上需要考虑的事情
- 一些关于构建安全系统的实用的想法
- 关于Java语言和平台的基本安全知识
- 一个关于哈姆雷特的故事，这个故事可以帮助我们讨论安全问题

我们首先来讨论安全应用程序的目标。

安全哲学

在写应用程序时我们应该把注意力集中在如下三个安全性目标：

1. 保护敏感数据——这个目标是显而易见的，也是大部分人在考虑计算机安全时想到的问题。最常见的敏感数据是信用卡号码，当然也有很多别的数据，如口令、金融数据、医疗历史记录等等。一个好的应用程序应该安全地存储这些数据，通常通过加密来实现。计算机安全领域常讲的一个规则是：如果破解系统的代价比系统本身内容的价值高，则系统是安全的。
2. 控制对资源的访问——对于存储好的敏感数据，我们通常会使用它们。重要的是，这种访问必须处于严格控制之下。通过某种形式的身份认证可以做到这一点。大部分系统采用的是用户名和口令，但本书将研究更多的安全方法，如数字认证。访问控制的一个用途是抵制拒绝服务（denial of service，DoS）的攻击。DoS攻击通过虚

假的请求阻塞资源，导致合法用户不能访问系统。通过对资源的访问控制，你可以最大限度地减少这类攻击。

3. 记录日志——这个目标也是显然的，在什么地方什么人进行了什么动作，应该有一个记录。在危害发生的时候，日志能帮助查明危害的来源，某些情况下甚至能恢复丢失的数据。然而，在一些系统中，日志本身也被视为敏感数据而加以保护。因为日志是一个相对简单的任务，并且不需要特别的Java API，所以这部分本书不作为重点。你只需要时刻记住，对于任何产品，记录访问日志是非常重要的。

为了帮助你达到这些安全性目标，制订安全策略将非常有用。

安全策略

安全策略指的是对一个组织里，允许何种行为、禁止何种行为的一个表述。它将帮我们对应用程序中应该保护的内容做一个界定，也可以帮我们识别系统中哪些是重要资源，以便做计划维护它们的安全。安全策略定义的是如下内容：可接受的公共资源的许可的使用方式、远程访问的策略以及用户特权。它应该用简明的语言来表述，并且组织中每个需要使用系统的人都能看到。一个公司的安全策略完全不同于Java安全策略文件，后者将在以后的章节中讲到。

维护应用程序安全的第一步是阅读你所在的组织的安全策略。如果组织太小，尚未写明安全策略，你应该写出来。

本书考虑的是应用程序的安全，而不是整个组织的安全。但是由于这种安全性存在于一个大的体系框架之中，你的应用程序安全性如何满足整个系统的安全性，是一个非常重要的问题。我们将涉及很多这样的由商业需求的影响带来的课题。

安全需求

在研究或写出了安全策略之后，你需要审查的是应用程序的安全需求。我们把这个题目分成两个方面：风险评估（risk assessment）和数据公开（data exposure）。当然，任何系统都建立在实践的基础之上，做些妥协是不可避免的。

风险评估

风险评估即对你的应用系统的安全风险进行描述。它能帮助你决定哪些资源需要被明确地保护起来，它们的价值有多大。你需要向自己提出以下几个问题：

- 如果数据被破坏，造成的损失有多大？
- 我的数据对其他人的价值有多大？
- 如果我的数据被破坏，对其他人的影响有多大？
- 应用程序的使用价值是多少？
- 非授权用户使用了应用程序使我们付出的代价有多大？
- 其他人通过何种途径了解到我们的应用程序？

一旦初步确定了资源的价值，你需要问自己：系统最薄弱的环节在哪里？谁最有可能

攻入系统？谁最有可能窃取你的数据，登录到你的应用程序，或破坏你的系统？以上行为的最大受益者将是谁？如果你实在找不出有谁，你还需要考虑其他的潜在破坏者、黑客等纯粹为了好玩才闯入你系统的人。他们有时候到最后才被想到。

数据公开

你需要考虑的下一个因素是数据公开的思想。你的数据何时最容易被访问到——在公共数据库里、在用户的计算机中，还是在备份磁带上？仔细检查这些位置，确认哪些地方的数据最容易被窃取或故意篡改。

然后，你需要仔细检查这些地方，确定哪一个最需要加强。要做到这一点，需要对你组织中最薄弱点进行彻底和客观的评估。系统最容易被攻击的有两个地方：人员和访问点（access points），我们将把焦点放在它们身上。

人员

不幸的是，由于比其他任何人更了解系统的工作方式，组织内部的人员负有更大的责任。人们往往有一个讨厌的习惯是把自己的口令粘贴在显示器上，或者把口令告诉其他同事。因为他们有对数据更大的访问权限，他们比网络外部的黑客或偷窥者有更大破坏能力。

正因为如此，一件极其重要的事情是：把你的系统分成多个子系统，并且把数据访问权限仅仅赋给需要访问它的人。

强调一下，需要花多少时间和金钱来划分和保护数据的安全，取决于数据的价值和它被非法访问的可能性大小。一个有用的策略是根据角色而不是人员来设计系统的安全性。换句话说，不是把访问资源X的权限给Jeannine，把访问资源Y的权限给Joe，而是把访问资源X的权限给所有的数据库管理员DBAs，把资源Y的访问权限给销售人员。这种做法将有助于从系统和数据的角度划分你的组织中不同人员担任的不同角色。

此外，在组织内部的人员加入、离职、调动等情况下，按角色赋予权限更为灵活。这种思维方式还有利于把安全系统从理论设计转换为实际实现。角色的概念很容易转化到公钥底层结构和加密策略（如认证链接，certificate chaining），本书后面有介绍。

容易被攻击的弱点

划分好内部人员的角色以后，下一个至关重要的事情是保护系统中的潜在安全弱点。不同系统的安全弱点各不相同，有些很明显，有些则很隐蔽。一些常见和明显的例子是：

- 系统中直接或间接连接到互联网的地方
- 物理位置处在公共或半公共场所的机器
- 提供远程服务的机器
- 存储敏感数据或宝贵数据的装置，如数据库

不明显的例子有：

- 备份数据，如磁带或CD
- 对机器的不显眼的物理访问点

尽管这些看上去都显而易见，但令人难以置信的是，究竟有多少人认为这些是理所当

然的呢？人们往往忘记，一个看上去是完全安全网络的，牢不可破的机器，如果允许任何一个未经授权的人走近来，从软盘启动系统，把硬盘格式化或获取root权限，这个系统的安全性将一钱不值。

另一个很重要的事情是：对外公开的任何机器必须使用最新版本的软件和应用程序。程序中任何时候都存在安全漏洞，如果不及时给系统打补丁，你可能在网络上留下一个安全漏洞。

在安全需求和可用性之间做平衡

正如我们已经提到，并且在本书中将反复阐述的那样，一个“安全”的系统是一系列事物妥协的结果，它要求对被保护的数据的内在价值、相关的人员、硬件和软件有深入的了解。

关键是作出正确的妥协。例如，在对系统完整性的保护和系统用户的使用方便性之间，我们需要做出平衡。这次我们又回到了人员这个要素。虽然有些人是恶意的威胁，但大多数人是由于粗心而违背安全原则。正因为如此，给用户提供一个易于使用的系统，将变得极其重要。一个好的系统将提供很多默认方式，甚至于可以让用户根本感觉不到安全系统的存在。

即使系统有复杂的口令系统，可以防止任何口令破解工具，如果用户把自己的口令粘贴在显示器上，密码系统也会毫无用处。如果系统的安全措施过于复杂，用户将为了方便而寻求越过安全措施的方法。在可用性和安全性之间找到一个成功的平衡，是创建安全系统的最大挑战。

由于不可能完全防止系统中违背安全性的行为，如果数据万一被破坏或篡改，应急处理计划将非常重要。当然，处理这个问题的最通用做法是经常地定期地对数据进行备份，如果需要还应该进行加密。当然对于数据被篡改的情况，这个方法没有任何帮助。

对个人数据完整性的破坏同样也是一个困难的问题。一般来说，我们应该采取使损失最小的策略。如果能够得到非法访问的时间和对数据的“每一次”访问的日志，我们就可能找出哪些数据被篡改，并且确保将来的数据不被破坏。日志技术是计算机安全的一个重要方面，尽管常常被忽视。

由于你运行的系统类型各不相同，这些措施有可能足够，也可能不够。有些系统只是简单地要求你改变口令，或用新的私钥加密数据。其他系统则可能要求在系统被攻击后不再留下新的漏洞或安全后门。

实现安全

一旦描述出哪些内容需要维护安全，需要维护到什么程度，下面要考虑的就是为它们分别选择工具。我们选择的是Java编程语言，因为我们相信它是维护安全的最好工具。本书就像一本介绍各类安全方法的食谱，教你如何使用密码学和Java安全模型来维护应用程序的安全。但它不是一本参考手册。Sun公司提供的Java文档是一个极好的信息源，它提供了API和开发包的各种信息。而我们将介绍的是这些库的高级应用。

我们将从密码学的基本元素，如消息摘要、密码算法等开始讲起，然后再介绍更高级的主题，如用SSL进行网络加密。最后，我们将写出一个适当完整的与安全相关的银行程序，阐述如何把不同的Java安全库和工具综合到一起使用。

显然，Java安全只是整个系统安全性的一个方面。所以，在讨论Java安全之前，我们先来浏览一下其他几个可以维护应用程序安全的技术。它们主要不是与Java竞争，而是有助于支持Java。

安全技术和工具

这里我们将快速地讲述如下内容：

- 操作系统
- IP安全
- 虚拟专用网（Virtual Private Networks）
- 防火墙
- 入侵检测工具

显然这里列出的不是安全范畴中其他领域的所有例子，但它们将有助于我们理解应用程序安全在整个计算机安全方案中的位置。这一节的目标不是提供网络安全的详细描述，而仅仅是指出你需要时刻意识到的一些问题。

我们先从操作系统说起。

操作系统

操作系统将极大地影响应用程序的安全性。如果你的操作系统有漏洞，攻击者可以改变组成应用程序的类文件，从而把应用程序替换掉。不论你把自己的应用程序的行为做得如何安全，如果别人能够改变它的工作方式，应用程序安全性就不存在。

某些操作系统很难保证安全，如Windows 95/98/Me和MacOS在X版以前版本。它们主要是面向消费者，开发时并没有时刻注意安全性。如果需要保证安全，你应该使用Linux，Solaris或Windows NT/2000等更为安全的操作系统。

安装了某个操作系统以后，经常给它打厂商提供的最新的补丁就非常重要了。另外，不要运行无关的网络服务，如FTP和telnet。文件权限也要仔细设置，用户不需要的文件就不要给访问权限。

某些操作系统可以对许可的连接进行配置。对于像数据库和RMI服务器这样的应用，这个特性极其有用，因为它使你能够配置谁可以对某个服务建立连接。如果和一个配置良好的java.policy文件联合使用的话，将大大减低对服务器的非授权使用。

关于操作系统安全的话题，讲一整本书也讲不完。如果可能的话，你应该安装一个经验丰富的系统管理程序，对你需要使用的任何操作系统进行配置。

IP安全

默认状态下，网络流量并没有被加密。当前处理网络流量的协议，IPv4并不支持加密。新的协议IPv6中有一个部分叫IPsec，它支持对IP层网络通讯的保护，与我们在第9章中要讲