万水网络与安全技术丛书

网络安全之道

被动侦查和间接攻击实用指南

SILENCE ON THE WIRE

a Field Guide to Passive Reconnaissance and Indirect Attacks

[美] Michal Zalewski 著

王兰波 李蕾 等译

中国水利水电出版社
www.waterpub.com.cn

万水网络与安全技术丛书

网络安全之道

——被动侦查和间接攻击实用指南

[美] Michal Zalewski 著

王兰波 李 蕾 等译

中国水利水电出版社

内 容 提 要

本书共分四个部分，第一部分从信息输入以及计算机存储和处理信息的角度开始分析，剖析了计算机通信初期阶段的安全问题及解决方法。第二部分介绍信息从本地计算机发送到因特网的过程中所面临的各类安全风险及对策。第三部分通过分析因特网各类协议及其漏洞，深入剖析了信息在因特网上传输的过程中可能受到的攻击和相应对策。最后一部分则从因特网整体视图的角度阐述了其他计算机安全相关问题。另外，每章所附的思考材料可激发读者对所讲内容进行进一步的思考。

本书适合计算机安全技术爱好者阅读，无论是安全专家、高级黑客，还是计算机专业的学生、普通网络爱好者，都可以从本书受益。

北京市版权局著作权合同登记号：图字 01-2005-4709

图书在版编目(CIP)数据

网络安全之道：被动侦查和间接攻击实用指南 /（美）扎勒维斯基（Zalewski,M.）著；王兰波等译. —北京：中国水利水电出版社，2007

（万水网络与安全技术丛书）

书名原文：Silence on the Wire

ISBN 978-7-5084-4151-1

Ⅰ. 网... Ⅱ. ①扎...②王... Ⅲ. 计算机网络－安全技术 Ⅳ. TP393.08

中国版本图书馆 CIP 数据核字（2006）第 126943 号

书　　名	网络安全之道——被动侦查和间接攻击实用指南
作　　者	[美] Michal Zalewski 著　王兰波　李　蕾　等译
出版发行	中国水利水电出版社（北京市三里河路 6 号　100044） 网址：www.waterpub.com.cn E-mail：mchannel@263.net（万水） sales@waterpub.com.cn 电话：（010）63202266（总机）、68331835（营销中心）、82562819（万水）
经　　售	全国各地新华书店和相关出版物销售网点
排　　版	北京万水电子信息有限公司
印　　刷	北京市天竺颖华印刷厂
规　　格	787mm×1092mm　16 开本　13 印张　310 千字
版　　次	2007 年 1 月第 1 版　2007 年 1 月第 1 次印刷
印　　数	0001—4000 册
定　　价	24.00 元

凡购买我社图书，如有缺页、倒页、脱页的，本社营销中心负责调换

序　　言

写一本关于计算机安全的长篇著作需要什么呢？或者说，写一本关于现代计算的长篇著作需要什么呢？

需要一位虽然年轻但是在很多领域都有着丰富经验和天赋的作者，这些领域包括计算、数学和电子（可能还是一个机器人技术爱好者），以及其他一些看起来毫不相干的兴趣（包括艺术摄影等），并且他真正具有写东西的天赋和欲望。

从前，在黑暗的原始森林里，树的魔力化学成分（脑细胞）产生了一比特信息，只是为了让他沿着急促的溪流航行，直到辽阔的大海（互联网），并最终找到它的新家、坟墓或是博物馆中的一个位置。

故事就这样开始了。不论我们的这比特信息是好还是坏，在年轻时他会到达溪流中，流淌到一个闪闪发光的、由白色的金属薄片做成的城堡（然而很多人认为它是一个黑匣子）中。他将通过入口并且到前台登记入住。如果他不是非常天真且短视，他可能会注意到一群看起来邪恶的比特从远处盯着柜台，记录比特们入住和离开的时间；然而，他没有其他选择，只能继续登记。

一旦歇下来，我们的英雄可能会被要求与其兄弟姐妹们组成一个团队或者加入另一些比特和女比特的团队中，他们紧紧挤在一艘不平静的船上。一个仔细的比特可能会注意到船上有垃圾比特（或者那是垃圾吗），他们可能是以前的团队留下来的。

观察交通灯并挤在拥塞的路途上，我们的比特们进入一个安全的港口并驶向码头。他们会被附近城堡和灯塔上的人看到吗？是否会有人记录交通灯的开关来确定我们的团队何时航行？是否会有人点亮码头上的灯并拍照？其他邪恶的比特会假扮成我们的比特并抢先出航吗？我们的比特不会知道。

这样，他们在码头换了船并驶向大海。我们宠爱的比特们继续他们的航行，但是前路布满荆棘。

不过，Michal 的书并非像我上面那样将技术细节隐藏在神话故事后面，而是在享受阅读的同时，提供了直接而迅速解答每一章开始提出的所有问题的事实。

《网络安全之道》在很多方面都是独特的，但在其中两个方面尤为突出：首先，它深入探讨了几乎关于数据处理所有关键阶段的问题——从击键到该击键动作所预期的最终结果，正是这些阶段才使得当今“网络互连”成为可能。其次，它概括了每个网络互连阶段及整个过程主要的、正在研究之中的和固有的安全问题。涉及的安全问题非常适合于从攻击者和防御者的角度演示弱点研究的艺术，并且鼓励部分读者进一步研究。

显然，计算机安全的书不可能通俗易懂。在本书中，Michal 争议性地选择了不探讨所有已知的高危险和广泛传播的弱点和攻击，这些问题是当前信息安全领域大多数人在讨论和研究的问题。他将教大家关于微妙的键击定时攻击的东西，但不会提醒关于“特洛伊木马”软件攻

击，这是一种带键击登录能力的软件，目前，它比以往的类似攻击更普遍和更易于使用。

为什么提到键击定时攻击但是却不提特洛伊木马呢？因为定时攻击远未得到充分认识，即使是信息安全专家都对它存在误解，而特洛伊木马则是一种众所周知且明显的攻击。定时攻击的弱点是与许多组件设计相关的属性，而植入特洛伊木马需要软件 bug 或终端用户错误。

类似地，在本书中也不会找到关于众所周知的软件 bug（或者甚至是像“缓存溢出”这样的通用软件 bug）的描述，一点也没有。如果您还不熟悉常见计算机安全威胁并想得到这方面的知识，在读本书时，就需要阅读一些能够从因特网和其他书本上找到的不足以引人入胜的材料，尤其是与所用的特定操作系统相关的材料。

为什么要研究让网络保持安静呢，您可能会疑惑——不是什么也没发生吗？是的，在某种意义上这是对的。从这个角度来说，“零”表示什么也没有。但是它也是一个数字，缺了它我们就不可能真正理解这个世界的某个概念。

享受这份安静吧——尽情享受本书！

Alexander Peslyak
Openwall, Inc.创始人和 CTO

Solar Designer
Openwall 项目负责人

前　言

我就是我

我似乎天生就是一个计算机怪物，但只是因为一次巧合才使我开始网络安全冒险的历程。我一直爱好动手做实验，探究新概念，并且解决那些看起来定义明确但仍然充满挑战的问题，这需要革新和创新的方法——即使在解决这些问题时总是遇到失败。年轻时，我把大部分时间花在探索化学、数学、电子、计算等领域，而不是成天骑着自行车在社区周围瞎转（我可能有一点夸大，但是我母亲看起来总是在担心），我的这些努力有时很大胆，但经常是愚蠢的。

在我涉足因特网之后不久（在 20 世纪 90 年代中期，大概是我在一台深爱的 8 位机器上完成第一次“Hello world”编码之后的 8 年），我收到了一个不寻常的请求：一份垃圾邮件，真是让人难以置信，要求我（以及其他数千人）加入一个地下组织，这个组织看起来是邪恶的黑帽黑客。我禁住了诱惑，没有加入这个地下组织（可能是因为我强烈的自我保护意识，这在某些圈子里可能被视为懦弱），但这为我提供了深入探索计算机安全的某种动力。在大量的业余编程之后，我发现从不同的角度研究代码以及尝试寻找比预定方法更有效的算法是一种很有吸引力的事情。因特网似乎是我所渴求的挑战的极好来源——一个庞大而复杂的系统，且只有一条指导原则：不能相信任何人。这样，一切就开始了。

我缺乏人们预期的普通计算机安全专家的背景，计算机安全专家是一个现在正变得越来越普遍的职业。我从未接受过任何计算机科学教育，也没有让人印象深刻的证书。计算机安全过去一直是我的一个主要爱好（现在则是我的生活来源）。我不是一个立体的计算机怪物——我确实曾经从另类的角度来看待我的工作，有时甚至完全远离计算机。

无论好坏，所有这一切都影响了本书的成形及其所传递的消息。我的目标是向其他人说明我是如何看待计算机安全的，而不是以通常形式讲解计算机安全。对我来说，安全不是要解决的单个问题，也不是要遵循的简单过程。它不是关于特定领域内的专门技术。它是了解整个生态系统和理解其组成部分之后的实践。

关于作者

Michal Zalewski 是一个自学成才的信息安全研究者，他的工作经历涉及从硬件和操作系统设计到网络的方方面面。自 20 世纪 90 年代中期以来，他就已经成为一个多产的 Bug 捕捉者，并且经常发布关于 Bug 的帖子。此外，他还创作了一些十分流行的安全工具，如 p0f，这是一个被动的操作系统跟踪程序。他还发表了一系列颇受好评的安全研究论文。Michal 曾在他的祖国波兰以及美国的一些著名公司担任安全专家，这其中包括两家大型电信公司。除了作为一名热心的研究者并偶尔编写程序之外，Michal 还涉足人工智能、应用数学和电子等领域，同时，他还是一名摄影爱好者。

关于本书

即使在监视器微弱的光线下，我们仍然只是人类。我们被教导要互相信任，而且不希望成为偏执狂。我们需要在安全和生产力之间找到明智的折衷，这样才能舒适地生存。

然而，因特网是一个与现实世界不同的社会。在这里遵循规则无法获得通常的好处，对于网上虚拟犯罪，也很少有人会自责。我们不能单纯地相信系统，试图提出一条解决所有问题的规则只会导致可悲的失败。我们本能地在“我们”和“他们”之间划出一条分隔线，并认为我们自己一方很安全。那么，我们留心地平线上的海盗船吧！很快，安全问题开始出现，像是可以轻易定义、诊断和解决的固定问题。从这个角度来看，攻击者像是受到某些明显动机的驱使，如果我们警惕的话，我们可以监视他们，并在他们出现时加以制止。

然而，虚拟世界与现实世界大相径庭：安全不仅仅是缺少 bug；安全也不再等于处于攻击者的范围之外。只要是与信息有关的任何过程都具备固有的安全含义，在我们看到过程试图完成的那一刻，就能够看到安全问题。理解安全的艺术就是能够跨越界限并从不同角度看问题的艺术。

这是一本非传统的书，或者说我是这么希望的。它不是问题的纲要，也不是保护系统的指南。本书试图从讲述一个信息故事开始，故事包括从手接触键盘的那一刻起，叙述了到达网络另一端远程方的各种方式。它介绍了相关的技术及其安全含义，重点讨论了那些不能视为 bug 的问题，其中没有攻击者，不分析和解决缺陷，甚至没有可检测到的攻击（至少没有我们可通过传统方式发现的攻击）。本书的目标是为了表明：理解因特网的惟一方法是要有勇气超越规范，或者理解规范字里行间的真实含义。

正如本书副标题所暗示的，本书主要探讨与日常通信和计算密切相关的隐私和安全问题。其中有些内容具有深刻的含义，而有些则只是出于个人兴趣和满足好奇心。其中没有任何一个会对环境产生直接危害或者破坏硬盘上的数据。本书内容对于以下 IT 专业人员和经验丰富的业余爱好者非常具有实用价值：他们渴望通过挑战以锻炼他们的思维，他们想要了解设计决策可能带来的其他结果。本书适用于想要了解如何使用本书中的技巧来操控其环境并从外界获得好处的读者。

本书分为四个部分：前三个部分涉及数据流阶段及相关技术，最后一部分重点探讨如何将网络视为一个整体。每一章介绍了每个阶段用于处理数据的相关技术组成、针对安全含义的讨论、副作用的描述、如何解决这些问题的建议（如果可能），以及如何进一步研究该主题的建议。我尽量避免使用规范的图、表和页面等（但是您会看到大量脚注）。由于能够轻松地找到大量很好的在线资源，所以本书的重点就是能够更轻松地供读者阅读。

让我们开始吧！

本书由王兰波、李蕾、谢琳翻译，张波审校。参与本书翻译工作的人员还有沈金河、谢君英、郭蓓、唐美艳、易磊等，在此一并表示感谢。由于本书作者知识渊博，翻译过程中存在一定的难度，如有不妥之处，望广大读者谅解，并批评指正。

目　　录

第二部分 安全的港湾

第三部分 因特网的问题

第四部分 总览全局

第一部分　问题的根源

源于人们通过网络发送任何信息之前可能出现的问题

第 1 章　监控键盘输入

本章探讨如何从很远的地方监控击键

从敲击键盘上第一个键开始，所发送的信息就开始了穿越虚拟世界的漫长旅途。在数据包沿着光纤链路飞速发送并从卫星收发器反射回来的数微秒内，一段信息就已经过令人惊奇的电路迷宫穿行很远了。在击键被操作系统以及可能运行的任何应用程序接收之前，与这些处理相关的许多精确且微妙的机制成为各种黑客感兴趣的对象，而且它们也被证明对于安全人员很有意义。在到达用户的路径上潜伏着很多惊奇。

本章就来关注移动数据的这些早期阶段，以及在自己的终端前舒适地工作时可能会给（可能调皮捣蛋的）同事所带来的很多机会。

与计算机处理输入方式相关的潜在信息泄漏的绝佳例子与下面这个主题相关（乍看来，好像毫不相关）：在机器上产生随机数这道难题，以一种可以预知的方式进行。很难想象两者之间的关联，然而我提到的问题千真万确，它可能使得鬼鬼祟祟的观察者推导出用户的大多数行为，包括正在输入的密码以及私人电子邮件等。

1.1　随机性需要

计算机是完全可预测的。它们处理数据的方式由定义良好的规则集合控制。工程师竭尽全力弥补与生产过程和电子部件本身属性（接口、热噪声等）相关的不足，这一切都是为了保证系统总是遵循相同的逻辑并正确工作。然而，随着时间和压力，这些部件拒绝按照预期工作时，我们就会认为计算机出错了。

机器达成这一级别一致性的能力，与它们非凡的计算能力结合在一起，使得计算机成为那些努力掌握和控制它们的人的强大工具。自然还要提一件事情：世上没有完美的东西，那些抱怨计算机不可靠的人并非完全是错误的。尽管这个装置运行正确，但是程序本身在不同情况下可能会出错。这是因为即使计算机硬件通常是一致而且可靠的，但是，即使假设可以得到一个详细的、十分严格且无错的和关于程序应该做什么的假定模型，也通常无法对十分复杂的计算机程序进行远期预测，更不要说互相依赖的程序所构成的复杂矩阵（如典型的操作系统）；这使得验证计算机程序变得非常困难。为什么呢？是这样，在 1936 年，现代计算之父 Alan Turing 通过反证法（转化成不合理的）证明不可能有通用方法在有限次数内来决定任何计算机程序和算法的结果（虽然可能对一些算法有某些特定方法）[1]。

这在实际中意味着虽然不可能期待操作系统或文本编辑器永远精确地按照您或作者想要的方法运转，但是仍可合理地期待在输入相同时，运行在相同硬件之上的系统中的文本编辑器的两个实例会产生一致且相同的结果（当然，除非其中一个实例损坏，或者受到其他极端外部事件的干扰）。这对于软件公司是很好的消息，但不管怎样，在某些情形下我们这些安全从业人员更倾向于计算机有点不确定。不一定是其行为如何，而是它可能得到的东西。

来看看数据加密，特别是公钥加密算法。这种新颖且优秀的加密方式最初是由 Whitfield Diffie 和 Martin Hellman 在 20 世纪 70 年代提出的，很快就由 Ron Rivest、Adi Shamir 和 Len Adleman 转换成为完整的加密系统，它基于简单的概念：有些事情比其他事情更加难。这看起来当然显而易见，却引出了几个高级概念，您要准备好打破常规。

传统的对称加密需要给秘密通信的所有参与者发布相同的共享“秘密”值（密钥）。密钥是必要且充分的，它用于加密传输的信息以及之后的解密，因此仅知道加密算法的第三方观察者仍然无法算出消息。由于需要使用共享的秘密，所以这种方法在计算机通信中并非总是很实际，这主要是因为参与方不得不在通信之前先建立安全的交换通道；通过不安全的流传输秘密会使得这种机制易于被解密。在计算机世界里，人们经常会与素未谋面的人或系统通信，而且和他们没有其他可行且安全的通信渠道。

另一方面，公钥加密算法不是基于共享的秘密。每一方拥有两个信息：一个（公钥）用于创建和加密消息，但不用于解密，另一个（私钥）用于解密先前加密过的消息。即便被偷听，参与方现在也可以通过不安全的通道来交换他们的公钥。他们互相向对方提供需要用于加密消息的信息（对观察者没有意义），但是他们保留了访问加密后数据所需的那部分私钥。突然，在全然陌生的人之间的安全通信（比如坐在自己公寓里沙发上的顾客与在线购物服务器之间）变得离现实更近了。

从根本上来说，最初的 RSA（Rivest、Shamir 和 Adleman）公钥加密系统基于以下结论：两个任意大数乘积的计算复杂度相当低，它直接与乘数的位数成比例。另一方面，寻找大数的因数（因数分解）的过程则相对要复杂得多，除非是对于那些为国家安全局工作的、神话般的密码天才。算法是：首先选择两个任意的、非常大的质数*，p 和 q，并求它们的乘积。然后将乘积和互质的+(p-1)(q-1)一起用来构造公钥。该密钥可用于加密信息，但是没有因数分解的话，它自己不足以解密这些信息。

得到的结果：对两个大质数的乘积进行因数分解通常是不切实际的，从而可防范这类攻击。在传统计算机上用最快的通用整数因数分解算法——通用数字域过滤（General Number Field Sieve，GNFS）以每秒 100 万次测试的测试速率，也需要超过 1000 年时间才能计算出像 1024 位整数的因数。从另一方面来说，找到两个生成很大乘积的质数，对于普通 PC 而言只需要几秒钟。

正如前面所指出的，在 RSA 中，除了公钥，还需要生成私钥。私钥携带的一段额外信息包括可用于解密的质数，它解密已经用该公钥加密过的任何信息。这种方法之所以可行要归功于中国余数定理、欧拉定理和其他一些可怕但是令人着魔的数学理论，对特别好奇的读者而言，他们可能希望自己去探索这些理论[2]。

随后又发明了依赖于其他数学难题的公钥加密系统（包括椭圆曲线加密系统等），但是所有的加密系统都共享同样的公钥和私钥基本理论。已经证明该方法对于保护电子邮件、Web 交易等非常实用，即使是双方从未通信过也是如此，而且在建立连接之前没有安全通道交换任

---

*:质数是只能被 1 和它自己整除的正整数。

+ :与 x 互质的数（也称为与 x 相对互质）是指除了 1 和-1 以外，与 x 没有公因数的数（它们的最大公因数为 1）。

何额外信息*。几乎我们当今使用的所有加密设计，从安全 shell（Secure Shell，SSH）和安全套接字层（Secure Sockets Layer，SSL）到数字签名更新或智能卡，都要归功于 Diffie、Hellman、Rivest、Shamir 和 Adleman。

使随机数生成自动化

现在只剩下一个问题：在某台确定的机器上实现 RSA 时，第一步是生成两个大质数 p 和 q。对计算机而言找到大质数很简单，但还是有一个小问题：这两个质数还必须让其他人难以猜到，而且它们不能在每台机器上相同（如果它们相同，那么对该算法的攻击就无需任何因数分解，任何拥有类似计算机的人都会知道 p 和 q）。

过去几年内人们已经开发了许多算法可快速找到质数候选者（伪质数）并执行初步的原始测试（用于验证伪质数[3]）。但是，要生成真正不可预测的质数，需要使用一定范围的熵或随机数，从而可在一个范围内盲目地选择一个质数，或者从一个随机的地方开始并选择遇到的第一个质数。

虽然在密钥生成时需要一些随机性是很关键的，但是要求不止如此。公钥加密依赖于相对复杂的计算，因此相当慢，特别是与传统对称密钥加密相比，后者使用短的共享密钥和一组人们所知执行得很快的操作。

为了实现像 SSH 这样的功能，人们期望这些功能具备合理性能，更为明智的做法是使用公钥算法建立初始通信和基本验证，这样就创建了一个安全通道。下一步是交换紧凑的、可能是 128 位的对称加密密钥，并转换回旧式的对称加密算法继续通信。通过以下方法可弥补对称加密算法的主要问题：创建初始（慢的）安全流来交换共享秘密，然后再转换到较快的算法，这样可使用户受益于不会牺牲安全性的较快算法。然而，在敏感的路线上使用对称加密时，仍需要使用一定数量的熵来为每次安全通信生成不可预测的对称会话密钥。

1.2 随机数生成器的安全

程序员们已经发明了许多方法让计算机生成看起来是随机的数；这些算法的通常名称是伪随机数生成器（Pseudorandom Number Generator，PRNG）。

PRNG 对于低价值的应用已经足够了，如为计算机游戏产生“随机”事件，或是为特别令人厌恶的垃圾邮件生成无意义的主题行。比如，线性一致（aka 幂剩余）生成器[4]就是这类算法的一个典型代表。抛开这个深奥的名称不谈，这个随机数生成器每次生成“随机”输出时都执行一系列简单运算（乘、加和模*）。生成器使用前一次的输出 r_t 来计算下一个输出值 r_{t+1}（其中 t 表示次数）：

$$r_{t+1}=(a\times r_t+c) \bmod M$$

---

* :为了完整性，除了其他一些问题，需要注意的是即兴公钥加密容易受到“中间人”攻击，在这类攻击中攻击者模仿其中一个端点并提供他自己的、假冒的公钥，从而能够截取通信。要预防这类攻击，必须设计额外的方法来校验密钥的真实性，或者通过安排安全的交换或建立认证中心来分发和证明密钥（Public Key Infrastructure，PKI）。

*模运算符返回两个数相除的整数余数。比如，7 除以 3 产生一个整数结果 2 和余数 1(7 = 2 ×3 + 1)；这样 7 模 3 得 1。